# 8th International Conference on Cognitive Modeling

July 26–29, 2007

University of Michigan

Ann Arbor, Michigan

*Edited by*

*Richard L. Lewis, Thad A. Polk, and John E. Laird*

# Contents

**About ICCM**

**Sponsors**

**Committees**

**Session 1: Basic Mechanisms**

1. Accounting for Subliminal Priming in ACT-R [pp. 1–6]
*Leendert van Maanen and Hedderik van Rijn*

2. How Readers Retrieve Referents for Nouns in Real Time: A Memory-based Model of Context Effects on Referent Accessibility [pp. 7–12]
*Aryn Pyke, Robert L. West, and Jo-Anne LeFevre*

3. A Model of Parallel Time Estimation [pp. 13–18]
*Hedderik van Rijn and Niels Taatgen*

4. A Control Perspective on Imaginal Perspective Taking [pp. 19–24]
*Holger Schultheis*

5. Towards Incorporating Visual Imagery into a Cognitive Architecture [pp. 25–30]
*Scott D. Lathrop and John E. Laird*

**Session 2: Methodology and Applications**

6. Evaluating the Performance of Optimizing Constraint Satisfaction Techniques for Cognitive Constraint Modeling [pp. 31–36]
*Alina Chu, Richard L. Lewis, and Andrew Howes*

7. Optimizing Knowledge Component Learning Using a Dynamic Structural Model of Practice [pp. 37–42]
*Philip I. Pavlik Jr., Nora Presson, and Kenneth Koedinger*

8. Creating Cognitive Models from Activity Analysis: A Knowledge Engineering Approach to Car Driver Modeling [pp. 43–48]
*Olivier Georgeon, Matthias J. Henning, Thierry Bellet, and Alain Mille*

9. Comparing Modeling Idioms in ACT-R and Soar [pp. 49–54]
*Randolph M. Jones, Christian Lebiere, and Jacob A. Crossman*

10. Dynamic Spatial Reasoning Capability in a Graphical Interface Evaluation Tool [pp. 55–60]
*Michael Matessa, Rick Archer, and Rebecca Mui*

**Session 3: Adaptive Control**

11. Learning to Control a Dynamic Task: A System Dynamics Cognitive Model of the Slope Effect [pp. 61–66]
*Cleotilde Gonzalez and Varun Dutt*

12. Instance-Based Decision Making Model of Repeated Binary Choice [pp. 67–72]
*Christian Lebiere, Cleotilde Gonzalez, and Michael Martin*

13. Modeling Control Strategies in the N-Back Task [pp. 73–78]
*Ion Juvina and Niels A. Taatgen*

14. ACT-R Models of Cognitive Control in the Abstract Decision Making Task [pp. 79–84]
*Daniel Dickison and Niels A. Taatgen*

15. The Importance of Action History in Decision Making and Reinforcement Learning [pp. 85–90]
*Yongjia Wang and John E. Laird*

16. Attentional Blink: An Internal Traffic Jam? [pp. 91–96]
*Niels A. Taatgen, Ion Juvina, Seth Herd, David Jilk, and Sander Martens*

**Session 4: Learning and Memory**

17. Category Development and Reorganization Using a Bidirectional Associative Memory-inspired Architecture [pp. 97–102]
*Gyslain Giguère, Sylvain Chartier, Robert Proulx, and Jean-Marc Lina*

18. The Emergence of Semantic Topography in a Neurally-Inspired Computational Model [pp. 103–108]
*Lee I. Newman and Thad A. Polk*

19. Simulating the Noun-Verb Asymmetry in the Productivity of Children’s Speech [pp. 109–114]
*Daniel Freudenthal, Julian M. Pine, and Fernand Gobet*

20. Structural Transfer of Cognitive Skills [pp. 115–120]
*Dongkyu Choi, Tolga Könik, Negin Nejati, Chunki Park, and Pat Langley*

**Session 5: Multitasking**

21. Dialing while Driving? A Bounded Rational Analysis of Concurrent Multi-task Behavior [pp. 121–126]
*Duncan P. Brumby, Dario D. Salvucci, and Andrew Howes*

22. From 1000ms to 650ms: Why Interleaving, Soft Constraints, and Milliseconds Matter [pp. 127–132]
*Bella Z. Veksler, Wayne D. Gray, and Michael J. Schoelles*

23. The Costs of Multitasking in Threaded Cognition [pp. 133–138]
*Jelmer Borst and Niels Taatgen*

24. Goal and Spatial Memory Following Interruption [pp. 139–144]
*Michel E. Brudzinski, Raj M. Ratwani, and J. Gregory Trafton*

**Session 6: Multiagent Interaction**

25. Learning Cognitive Load Models for Developing Team Shared Mental Models [pp. 145–150]
*Xiaocong Fan, Po-Chun Chen, and John Yen*

26. Effect of Communication on Belief Dynamics in Multi-Agent Systems [pp. 151–156]
*M. Afzal Upal and Ravikanth Sama*

27. Using Reflective Learning to Master Opponent Strategy in Competitive Environment [pp. 157–162]
*Mark A. Cohen, Frank E. Ritter, and Steven R. Haynes*

**Posters**

28. Toward a Large-Scale Model of Language Comprehension in ACT-R 6 [pp. 163–168]
*Jerry Ball, Andrea Heiberg, and Ronnie Silber*

29. Attention and Association Explain the Emergence of Reasoning About False Belief in Young Children [pp. 169–174]
*Paul Bello, Perrin Bignoli, and Nicholas Cassimatis*

30. A 3-node Queuing Network Template of Cognitive and Neural Differences As Induced by Gray and White Matter Changes [pp. 175–180]
*Marc G. Berman, Yili Liu, and Changxu Wu*

31. Integrating Rational Choice and Subjective Biological and Psychological Factors in Criminal Behaviour Models [pp. 181–186]
*Tibor Bosse, Charlotte Gerritsen, and Jan Treur*

32. A Dynamical System Modelling Approach to Gross' Model of Emotion Regulation [pp. 187–192]
*Tibor Bosse, Matthijs Pontier, and Jan Treur*

33. Modelling Animal Behaviour Based on Interpretation of Another Animal's Behaviour [pp. 193–198]
*Tibor Bosse, Zulfiqar A. Memon, and Jan Treur*

34. From a Formal Cognitive Task Model to an Implemented ACT-R Model [pp. 199–204]
*Fiemke Both and Annerieke Heuvelink*

35. A Qualitative GOMS Approach to Evaluating Diagrammatic Interfaces [pp. 205–210]
*B. Chandrasekaran and Tsviatko Yovtchev*

36. The First Second of Symmetry: Towards a Model of Visual Search during Symmetry Verification [pp. 211–216]
*Kenneth Czechowski, Ronald W. Ferguson, and Rudolph L. Mappus IV*

37. Diagrammatic Reasoning: Route Planning on Maps with ACT-RH [pp. 217–218]
*H. A. Dye*

38. Meter based Omission of Function Words in MOSAIC [pp. 219–224]
*Daniel Freudenthal, Julian Pine, and Fernand Gobet*

39. Storm: A Framework for Biologically-Inspired Cognitive Architecture Research [pp. 225–230]
*Douglas Pearson, Nicholas A. Gorski, Richard L. Lewis, and John E. Laird*

40. Vector Generation of an Explicitly-defined Multidimensional Semantic Space [pp. 231–232]
*Alex Grintsvayg, Vladislav D. Veksler, Robert Lindsey, and Wayne D. Gray*

41. Dynamic Visualization of ACT-R Declarative Memory Structure [pp. 233–234]
*Andrea Heiberg, Jack Harris, and Jerry T. Ball*

42. A Belief Framework for Modeling Cognitive Agents [pp. 235–240]
*Annerieke Heuvelink*

43. A Formal Empirical Analysis Method for Human Reasoning and Interpretation [pp. 241–246]
*Tibor Bosse, Mark Hoogendoorn, Catholijn M. Jonker, and Jan Treur*

44. Towards Human-Like Robustness in an Intelligent Tutoring System [pp. 247–252]
*Hameedullah Kazi, Peter Haddawy, and Siriwan Suebnukarn*

45. Using Simulations to Model Shared Mental Models [pp. 253–254]
*William G. Kennedy and J. Gregory Trafton*

46. Investigation of Procedural Skills Degradation from Different Modalities [pp. 255–260]
*Jong W. Kim, Richard J. Koubek, and Frank E. Ritter*

47. Information Seeking in Complex Problem Solving [pp. 261–266]
*Xiaohui Kong and Christian D. Schunn*

48. Modeling Memories of Large-scale Space Using a Bimodal Cognitive Architecture [pp. 267–272]
*Unmesh Kurup and B. Chandrasekaran*

49. Learning and Decision Model Selection for a Class of Complex Adaptive Systems [pp. 273–278]
*Tei Laine*

50. Be Wary of What Your Computer Reads: The Effects of Corpus Selection on Measuring Semantic Relatedness [pp. 279–284]
*Robert Lindsey, Vladislav D. Veksler, Alex Grintsvayg, and Wayne D. Gray*

51. Queueing Network Modeling of Mental Architecture, Response Time, and Response Accuracy: Reflected Multidimensional Diffusions [pp. 285–290]
*Yili Liu*

52. Are Simpler Strategies Less Error-prone in Inference from Memory? [pp. 291–292]
*Rui Mata*

53. The RecMap Model of Active Recognition Based on Analogical Mapping [pp. 293–298]
*Georgi Petkov and Luiza Shahbazyan*

54. Modeling the Range of Performance on the Serial Subtraction Task [pp. 299–304]
*Frank E. Ritter, Michael Schoelles, Laura Cousino Klein, and Sue E. Kase*

55. Interactions Between Data Labeling and Ratio of Hebbian to Error-Driven Learning in Mixed-Model Networks [pp. 305–306]
*Rebecca J. Robare*

56. Prototypical Relations for Cortex-Inspired Semantic Representations [pp. 307–312]
*Florian Röhrbein, Julian Eggert, and Edgar Körner*

57. Cognitive Redeployment in ACT-R: Salience, Vision, and Memory [pp. 313–318]
*Terrence C. Stewart and Robert L. West*

58. Fast Learning in a Simple Probabilistic Visual Environment: A Comparison of ACT-R's Old PG-C and New Reinforcement Learning Algorithms [pp. 319–324]
*Franklin P. Tamborello II and Michael D. Byrne*

59. Towards a Complete, Multi-level Cognitive Architecture [pp. 325–330]
*Robert Wray, Christian Lebiere, Peter Weinstein, Krishna Jha, Jonathan Springer, Ted Belding, Bradley Best, and Van Parunak*

60. Scheduling of Eye Movements and Manual Responses in Performing a Sequence of Choice Responses: Empirical Data and Model [pp. 331–334]
*Shu-Chieh Wu, Roger W. Remington, and Harold Pashler*

**Author index**

## About ICCM

ICCM is the premier international conference for research on computational models and computation-based theories of human behavior. ICCM is a forum for presenting, discussing, and evaluating the complete spectrum of cognitive models, including connectionism, symbolic modeling, dynamical systems, Bayesian modeling, and cognitive architectures. ICCM includes basic and applied research, across a wide variety of domains, ranging from low-level perception and attention to higher-level problem-solving and learning.

This year’s conference features invited talks by Neil Burgess, Marcel Just, and Walt Schneider, peer-reviewed research talks and posters, a panel discussion on how brain imaging and cognitive modeling can better inform each other, tutorials on four different approaches to cognitive modeling, and a doctoral consortium for students working on dissertations in the field of cognitive modeling.

Please visit iccm2007.org for more information about the ICCM 2007 conference program.

## Sponsors

### Platinum Sponsors

- University of Michigan, Center for Cognitive Architecture
- National Science Foundation
- Soar Technology

### Gold Sponsors

- Air Force Office Scientific Research
- Army Research Lab
- Office of Naval Research
- Lockheed Martin Advanced Technology Laboratories

### Silver Sponsors

- Institute of Creative Technology, University of Southern California
- Charles River Analytics

## Conference Organizing & Program Committees:

**General Chair:**
John Laird

**Program Chairs:**
Richard Lewis
Thad Polk

**Local Arrangements:**
David Kieras

**Tutorials:**
Frank Ritter

**Conference Secretary:**
Karen Alexa: ICCM2007@eecs.umich.edu

**Program Committee:**

- Erik Altmann, MSU
- Matt Botvinick, Penn
- Mike Byrne, Rice University, Houston
- Balakrishnan Castelfranchi, National Research Council (CNR) Roma, Italy
- Balakrishnan Chandrasekaran, Ohio State Computer Science
- Richard Cooper, Birkbeck University of London
- Gary Cottrell, UCSD
- Matt Crocker, University of Saarland
- Fabio Del Missier, University of Trieste
- Gary Dell, Illinois
- Danilo Fum, University of Trieste
- Kevin Gluck, Air Force Research Laboratory, Mesa, AZ
- Jonathan Gratch, USC ISI
- Glenn Gunzzelmann, Air Force Research Laboratory, Mesa, AZ
- John Hale, Michigan State University
- Andrew Howes, Manchester University
- John Hummel, Illinois
- Bonnie John, Carnegie Mellon
- Matt Jones, University of Texas
- Randy Jones, Soar Tech
- Boicho Kokinov, New Bulgarian Univ. , Sophia, Bulgaria
- Adam Krawitz, University of Michigan
- Frank Lee, Drexel University , Philadelphia, PA
- Shu-Chen Li, Max Planck Institute for Human Development,Berling
- Yili Liu, UM Industrial Engineering
- Deryle Lonsdale, Brigam Young University
- Marsha Lovett, Carnegie Mellon
- Mike Matessa, NASA Ames Research Center
- Shane Mueller, Indiana University

- Roger Remington, Australia
- Frank Ritter , Penn State University
- Dario Salvucci, Drexel University , Philadelphia, PA
- Lael Schooler, Max Planck Institute, Berlin
- Chris Schunn, University of Pittsburgh
- Travis Seymour, Santa Cruz
- Patrick Simen, Princeton
- Satinder Singh, UM Computer Science
- Andrea Stocco, Carnegie Mellon
- Ron Sun, University of Missouri-Columbia
- Niels Taatgen, Carnegie Mellon
- Hedderik van Rijn, University of Groningen , The Netherlands
- Shravan Vasishth, Potsdam University
- Boris Velichkovsky, Dresden Univ. of Technology, Dresden ,Germany
- Alonso Vera, NASA Ames Research Center
- Tor Wager, Columbia Psychology
- Amy Weinberg, Maryland
- Janet Wiles, University of Queensland
- Richard Young, UCL Interaction Centre

# Accounting for subliminal priming in ACT-R

**Leendert van Maanen (leendert@ai.rug.nl)**
**Hedderik van Rijn (D.H.van.Rijn@rug.nl)**
Artificial Intelligence, University of Groningen
Grote Kruisstraat 2/1, 9712 TS Groningen, the Netherlands

## Abstract

This paper presents a cognitive model of a subliminal priming task, using Retrieval by ACcumulating Evidence (RACE), a model of declarative memory retrieval. RACE is implemented as an extension to the ACT-R architecture of cognition. First, we will discuss the exact implementation of RACE within the constraints imposed by ACT-R. Second, we will discuss the subliminal priming task that we modeled and present a cognitive model of this task that incorporates RACE.

## Introduction

Successful behavior depends for a large part on having declarative knowledge available at the right time. Humans are therefore continuously retrieving declarative facts from long-term memory storage, based on their continuously updated perception of the environment. The continuous character of perception is reflected in the memory retrieval process, as can for instance be observed in the retrieval latencies of psychonomic experiments in which stimuli are asynchronously presented (e.g., picture-word interference, Glaser & Düngelhoff, 1984) or in experiments in which the presentation durations of stimuli are manipulated (e.g., subliminal priming, Marcel, 1983). A cognitive model of declarative memory retrieval should also reflect the continuous character of the input on which memory retrievals are based. However, current cognitive architectures such as ACT-R (Anderson, Bothell, Byrne, Douglass, Lebiere, & Qin, 2004) or Soar (Newell, 1990) cannot satisfactorily account for this (Van Maanen & Van Rijn, 2006;  in press). Retrieval by ACcumulating Evidence (RACE) is a model that does describe the *process* of retrieving one or more chunks of information from memory. In RACE, memory retrieval is not considered ballistic, but is rather thought of as a process in which the likelihood that a piece of information will be needed for successful behavior is continuously estimated. Therefore, the likelihood estimate can be continuously adapted to the changing environment.

RACE can be perceived as an interaction of ideas from cognitive architectures that rely on symbol manipulation (Anderson et al., 2004; Newell, 1990) and ideas from sequential sampling models (Ratcliff & Smith, 2004; Usher & McClelland, 2001). The architectural nature is clear from the cognitive constraints imposed on RACE. In the current implementation of the theory, we constrained RACE by adopting the rational approach that is intrinsic to the ACT-R cognitive architecture (Anderson et al., 2004). However, the *subsymbolic* computations that drive declarative memory retrieval are rooted in sequential sampling.

This paper will describe how RACE is implemented in the ACT-R architecture of cognition and will present a RACE model of a subliminal priming task. We will discuss which features of RACE naturally align with ACT-R, and which features of RACE seem to contrast with ACT-R. We chose to implement RACE as an extension to ACT-R because of ACT-R's widespread use in the cognitive modeling world (see for instance the web site of the ACT-R community: http://actr.psy.cmu.edu). More importantly however, adopting an existing general approach towards cognition will reduce the proliferation of different cognitive theories (Newell, 1990), and will constrain theorizing about RACE. A third reason for choosing ACT-R as a modeling framework is that the way ACT-R defines retrieval latency has difficulties with modeling semantic interference (Van Maanen & Van Rijn, 2006;  in press). Extending ACT-R with RACE might solve this issue.

## ACT-R

A prominent theory that explains behavior at the symbol manipulation level is the ACT-R architecture of cognition (Anderson et al., 2004). Because RACE is implemented as an extension to ACT-R, we will give a very short overview of the architecture, concentrating on these aspects of the theory that relate to declarative memory retrieval.

ACT-R is a cognitive theory in which production rules operate on declarative memory and the environment. Production rules are conditions-actions pairs whose actions are executed if their conditions are met. To determine which production rule's actions will be executed, ACT-R contains a set of *buffers* of which the content is matched against the conditions of each production rule. If multiple production rules are applicable – meaning that, given the buffer contents, multiple sets of actions may be performed – the production rule with the highest utility will be selected, a process called *conflict resolution*. By default, the buffers represent the current goal of the system, the current perceptual state, and a declarative fact that is currently in the focus of attention, that is, that is recently retrieved from long-term memory. Other buffers may be defined if necessary for the task at hand (as has for instance been done for prospective time interval estimation, Taatgen, Van Rijn, & Anderson, in press). The content of a buffer is a chunk: a symbolic unit that represents a simple fact, such as *The capital of Canada is Ottawa*, or *The object I am attending is green and spherical*. Both these example chunks are declarative facts, but the first example can typically be found in the *retrieval buffer*, and represents a fact that has been retrieved from long-term memory, whereas the second example represents a visually observable fact of the world, and might be present in the *visual buffer*. In the context of this paper, we are primarily interested in the way ACT-R incorporates retrieval of chunks from long-term memory, although we not necessarily want to constrain RACE to declarative memory retrieval.

All chunks have an activation level that represents the likelihood that a chunk will be needed in the near future. The likelihood is in part determined by a component describing the history of usage of a chunk called the *base-level activation* ($B_i$ in Equation 1).

$$B_i = \ln\left(\sum_{j=1}^{n} t_j^{-d}\right) \tag{1}$$

In this equation, $t_j$ represents the time since the *j*th presentation of a memory chunk and *d* is the parameter that controls decay, which in most ACT-R models is fixed at 0.5 (Anderson et al., 2004). The idea is that the activation of a chunk decays over time unless attention is shifted to that chunk and its activation is increased. This way, the base-level activation can be used to model both forgetting and learning effects (Anderson & Schooler, 1991).

The total activation is the sum of the base-level activation and another component describing the influence of the current context (*spreading activation*, Equation 2). The spreading activation component is the sum of strengths of association from chunks *j* to chunk *i*, weighed by $W_{kj}$, representing the importance of various buffers (*k*) and of associated chunks (*j*).

$$A_i = B_i + \sum_k \sum_j W_{kj} S_{ji} \tag{2}$$

A more detailed description of the ACT-R cognitive architecture is provided in (Anderson & Lebiere, 1998; Anderson et al., 2004).

## RACE model of memory retrieval

RACE is a proposal for a new retrieval mechanism in ACT-R. In RACE, retrieval of a chunk is thought of as a process in which the likelihood that a chunk will be needed given the current context is continuously estimated. This is different from ACT-R, were the context can influence the retrieval of a chunk only at the onset of a particular retrieval request. Note that the continuous aspects of ACT-R's base-level learning equation (Equation 1) are retained in RACE. The continuous updating of context-based activation is similar to the account presented in the leaky competitive accumulator model described by Usher and McClelland (2001).

Also similar to ACT-R, the accumulation process in RACE is influenced by various sources of evidence. Increases in activation may be caused by the current context, which may be formed by the current buffer contents, or other chunks that are currently active. Via a spreading activation mechanism these chunks provide evidence for the likelihood that other chunks will be needed. That is, they increase the activation of these chunks.

Another source of evidence for the likelihood that a chunk will be needed is the history of usage of that chunk. Frequently or recently used chunks are more likely to be used again in the near future. In RACE, this is reflected by the starting point of the accumulation process. The level of activation at which accumulation starts is determined by the base-level activation of ACT-R, which reflects the frequency and recency of the usage of a chunk (Anderson & Schooler, 1991).

To preserve the temporal nature of the evidence for a chunk, the accumulated RACE activation is subject to continuous decay. Activation of a chunk thus decreases if not enough evidence for that chunk is present. Since the context may change over time, the accumulation process is not determined when a retrieval process is initiated (the retrieval onset), but may also change. Therefore, incoming information or the removal of information from the buffers may influence which chunk will be retrieved.

Activation values represent the *relative* likelihood that a chunk may be needed (Anderson & Lebiere, 1998), which means that the level of activation at which a chunk has been retrieved should also be defined *relative* to the activation of other chunks. Therefore, RACE uses a *retrieval ratio* that determines how much the activation of a particular chunk must stand out against the total activation of all competing chunks. This is analogous to the relative stopping rule described by Ratcliff and Smith (2004; cf., ACT-R's former competitive latency mechanism, discussed in Van Rijn & Anderson, 2003). If multiple chunks match the criteria of the retrieval request, the chunk that reaches the retrieval ratio first will be retrieved. In these cases, the eligible chunks compete for retrieval. If the activation levels of multiple chunks increase, the total activation of the system also increases, making it more difficult for a chunk to reach the retrieval ratio. This feature of RACE will prove to be important in explaining differences in retrieval latency, for example in the model of subliminal priming explained later in this paper.

So far, we described the general idea of the RACE model of memory retrieval. In this section, the exact implementation of RACE will be presented and how RACE relates to the ACT-R architecture.

The accumulated activation component of RACE is described by the following equation:

$$C_i(t+\Delta t) = d^{\mathrm{acc}} C_i(t) + \beta \sum_{j \in k} C_j(t) S_{ji} \tag{3}$$

This equation reflects the idea that the accumulated activation of a chunk at a certain moment in time ($C_i(t+\Delta t)$) is determined by the level of accumulated activation one time step ago ($C_i(t)$), summed with spreading activation from other chunks; that is, the accumulated activation of other chunks ($C_j(t)$) weighed by strengths of association between these chunks and the chunk $i$ ($S_{ji}$). At retrieval onset, accumulation starts with the history-based evidence, which is the current base-level activation. Thus

$$C_i(\text{retrieval onset}) = B_i(\text{retrieval onset}) \tag{4}$$

Accumulated activation decays away, the speed of which is controlled by the parameter $d^{\text{acc}}$. A smaller value of $d^{\text{acc}}$ results in faster decay. The parameter $\beta$ in Equation 3 controls the amount of influence of the context. Although in ACT-R activation can have a negative value, we have chosen in our current implementation to ignore the spreading activation from very small – that is, negative – activation values for reasons of computational efficiency.

By continuously updating spreading activation towards a chunk, the chunk may reach a level of activation at which retrieval can take place. The time at which retrieval takes place is the first moment after the start of accumulation at which the following inequality holds:

$$\frac{e^{A_i}}{\sum_j e^{A_j}} \geq \theta \tag{5}$$

This means that for a chunk to be retrieved ($i$ in Inequality 5) the activation should be high with respect to all competing chunks ($j$). Because ACT-R activation values represent the *relative* likelihood that a chunk will be needed, an exponential scaling is applied to eliminate effects from possible negative values, as is common in ACT-R equations.

Perhaps a clarification is needed on the notions base-level activation ($B_i$, defined in Equations 1 and 4) and accumulated activation ($C_i$ in Equation 3). To incorporate frequency and recency effects in the retrieval process, the accumulation of activation starts at the current level of base-level activation (Equation 4). During a retrieval process however, activation is estimated according to Equation 3. At retrieval, the base-level activation of the retrieved chunk is also increased to account for the recent encounter with the retrieved chunk, because at the next retrieval attempt the base-level activation is again used as the starting value of the accumulation process.

The question arises which of the two activations ($B_i$ or $C_i$) is a better predictor of the likelihood that a chunk will be needed. We believe that at very short time intervals – such as the SOAs from the subliminal priming experiment discussed below – accumulated activation better aligns with the empirical data. However, at longer time intervals, base-level activation has been shown to give good predictions (e.g., Anderson, Bothell, Lebiere, & Matessa, 1998; Anderson & Schooler, 1991). Because in the subliminal priming task and model described below prime and target are retrieved in a very small time window, focusing on accumulated activation only will suffice to model the priming effects. Therefore, for this model the base-level activation values were kept constant over all chunks.

## Subliminal priming

In this section, we will discuss the task we modeled using RACE: a subliminal priming study by Marcel (1983). Also, we will discuss why this particular task is interesting given the specific nature of RACE. In subliminal priming tasks, primes are presented that are not consciously perceived by the participant. Usually, primes are presented for a very short period and are followed by a visual mask, so that participants can not discriminate between the presence and absence of a prime (Marcel, 1983; Merikle, Smilek, & Eastwood, 2001). Marcel (1983) showed that under these circumstances priming effects persisted. His Experiment 3 describes a Stroop-task in which words are presented as primes, and color patches are presented as cues. Participants had to respond to the color patches by pressing a button associated to one of the colors. He found the same kind of interference and facilitation as usual in the Stroop paradigm, but a smaller effect for the subliminal primes than for the consciously perceived primes (Figure 3 presents the latencies that Marcel observed). Marcel concluded that subliminal primes have an effect on latency, even though participants are not aware of their presence.

Four prime conditions were tested by Marcel (1983, Experiment 3): Color congruent, color incongruent, neutral, and no-word. In the congruent condition, the prime was the name of the target color, whereas in the incongruent condition the prime was the name of another color. In the neutral condition, the prime was a non-color word that was also unrelated to colors. The no-word condition presented the mask only. Thus, no prime was presented. The condition in which the prime was subliminal was called the unaware condition. In the aware condition, by contrast, the presentation duration was 400ms. Both prime and cue were presented at the same time.[1]

From a symbolic perspective, stimuli have to be considered as symbols in order to engage in cognitive processing. In ACT-R, this means that a stimulus has to be present in a buffer. However, stimuli that are presented for such short durations as common in subliminal priming paradigms do not reach the visual buffer. ACT-R assumes an attention shift to the stimulus before an object can be encoded as a symbolic chunk, which takes a certain amount of time, estimated at 185ms (Anderson, Matessa, & Lebiere, 1998). This exceeds the presentation duration

[1] In the original experiment, Marcel included also another condition with a Stimulus Onset Asynchrony between prime and cue. This condition is similar to the picture-word interference study by Glaser and Düngelhoff (1984), which has previously been discussed by Van Maanen and Van Rijn (2006; in press). Therefore, it is not included here.

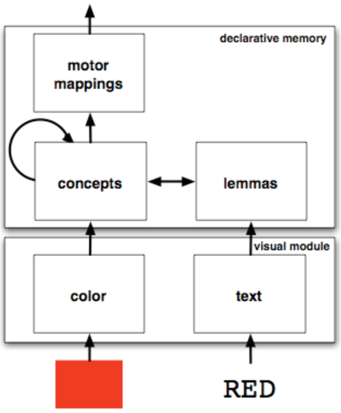


Figure 1: The flow of activation in the congruent condition of the subliminal priming model.

of the prime in the unaware conditions (which is 80ms at maximum, Marcel, 1983). In ACT-R models, stimuli that are presented for less than the time it takes to shift attention can therefore not influence central cognition. The way ACT-R deals with stimulus durations is all or none. Either the stimulus has been presented not long enough, and the stimulus is not perceived at all, or it is fully is perceived. Consequently, symbolic theories of cognition cannot account for subliminal priming data. The next section will show how RACE deals with the short presentation durations typical in subliminal priming tasks.

## Subliminal priming model

The subliminal priming model comprises three chunk types, as outlined in Figure 1: Lemmas, concepts, and motor mappings. The concept chunks can be regarded as representations of semantic properties. Chunks of the lemma type can be regarded as sets of orthographic and syntactic properties of a word. The motor mapping chunks represent the information which button to press for which color.

Now, for example in a no-word condition, the cue (being a color patch) spreads activation to its associated concept, which spreads activation to the associated motor mapping resulting in a button press. A similar *flow of activation* will occur in the other conditions, albeit that because of the presentation of a prime word, lemma chunks will also be activated. The activation of multiple motor mapping chunks causes competition in RACE, because the retrieval ratio is harder to reach with multiple accumulating chunks.

Before the experiment, Marcel determined for each participant the critical presentation duration for which participants could not discriminate between presence and absence of a prime (see Marcel, 1983 for details of the procedure). The presentation durations he found ranged from 30 to 80 ms. We used the presentation duration as an extra parameter in fitting the model to the data, with the constraints that its value should be in the range that Marcel found and that the activation of the prime chunk would not exceed the retrieval ratio (Inequality 5). Because the primes in the original experiment were visually masked, we assume that the presentation duration is equal to the time that the prime is available to the visual system.

Table 1: Estimated parameter values for the subliminal priming model.

| Parameter | Value |
|---|---|
| $A_{\text{color}}$ | 1.8 |
| $A_{\text{text}}$ | 1.5 |
| $\boldsymbol{\beta}$ | **.255** |
| $\boldsymbol{d^{acc}}$ | **.72** |
| $\boldsymbol{\Delta t}$ | **5 ms** |
| $\boldsymbol{\theta}$ | **.81** |
| *aware presentation duration* | 400 ms |
| *unaware presentation duration* | 70 ms |

Table 1 presents all relevant parameters for the subliminal priming model. The presentation duration of primes in the aware condition is 400ms, as in the original experiment. The unaware presentation duration was estimated at 70ms, serving as the model's critical presentation duration. This duration depends on the RACE parameters presented in bold-face in Table 1. These parameters were not estimated for this experiment, but rather copied from a RACE model of picture-word interference (an updated version of Van Maanen & Van Rijn, 2006;  in press). Hence, the only parameters presented here that were estimated for this model were the activation of the words ($A_{text}$) and of the color ($A_{color}$).  The association values ($S_{ji}$) between chunks are presented in Figure 2.

## Results

The results of the subliminal priming model are presented in Figure 3. We present here differences in latency relative to the no-word condition as this model only captures the memory retrieval process, which comprises the time course from the start of retrieval of a chunk until the retrieval of the motor mapping chunk. The model

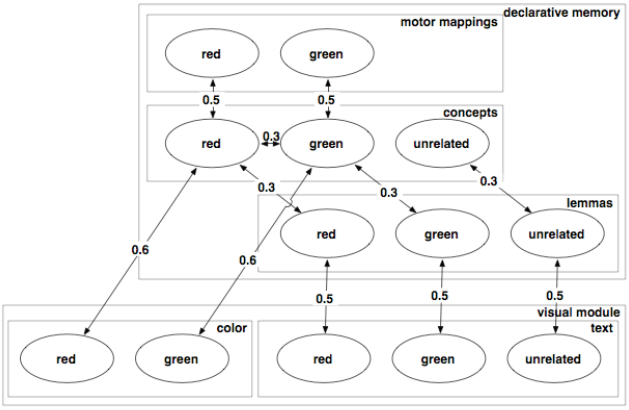


Figure 2: Associative values between different chunks in the subliminal priming model.

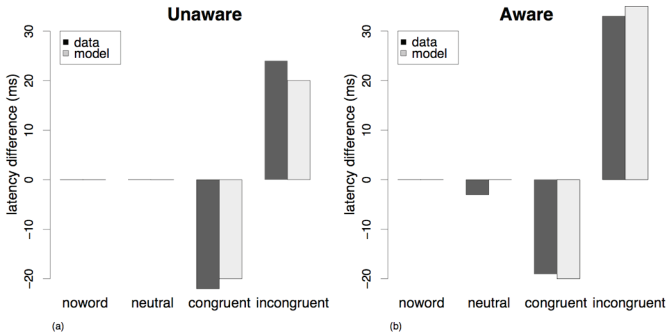


Figure 3: Comparison of the latencies found by Marcel (1983) (a) and the latencies predicted by the subliminal priming model (b). Shown here are the latency differences relative to the no-word condition.

captures quite nicely the effects observed in the data[2] by Marcel (1983) ($r^2 = 0.987$).

In the unaware conditions, only the target chunks reached the retrieval ratio, and no other chunks. Therefore, these chunks are the only ones that are consciously perceived by the model. The model thus remained unaware of all other chunks, as is required for these conditions.

An explication of how the activation flows through the model will be insightful. We split this up in four sections, each describing one condition.

### Neutral

In the neutral condition, there is no competition between motor mappings, because there is no button associated with the neutral word. The activation cascades through the network similarly to the no-word condition, because no association exists between the neutral word and the target color, both at the lemma level and at the concept level. Therefore, the activation of the motor mapping associated with the target color increases similarly to the no-word condition because the activation of *all* the motor mapping chunks increases as in the no-word condition.

As an example, Figure 4 gives the activation accumulation in the neutral unaware condition. The activation of the neutral word lemma increases, but, due to the short presentation duration, it does not reach the retrieval ratio. This indicates that the neutral word does not reach awareness.

### No-word

The no-word condition is similar to the Neutral condition, because no distractor stimulus is present, resulting in the same behavior of the model as in the Neutral condition. Because in the no-word condition, there is no distractor, there is no difference between aware and unaware.

### Congruent

Both target and distractor stimuli activate the same concept: the color chunk directly, the text chunk mediated via the lemma chunk. Spreading activation towards the associated motor mapping chunk is therefore higher than in the Neutral and No-word conditions, resulting in faster retrieval.

### Incongruent

Because both target stimulus and distractor activate a motor mapping, competition for retrieval takes place at the motor-mapping level. Higher activation for competing chunks means that it is more difficult to cross the retrieval ratio, leading to longer retrieval latencies. The effect is strongest in the aware condition, representing the longer presentation duration of the prime, and thus the longer accumulation of activation of prime-related chunks.

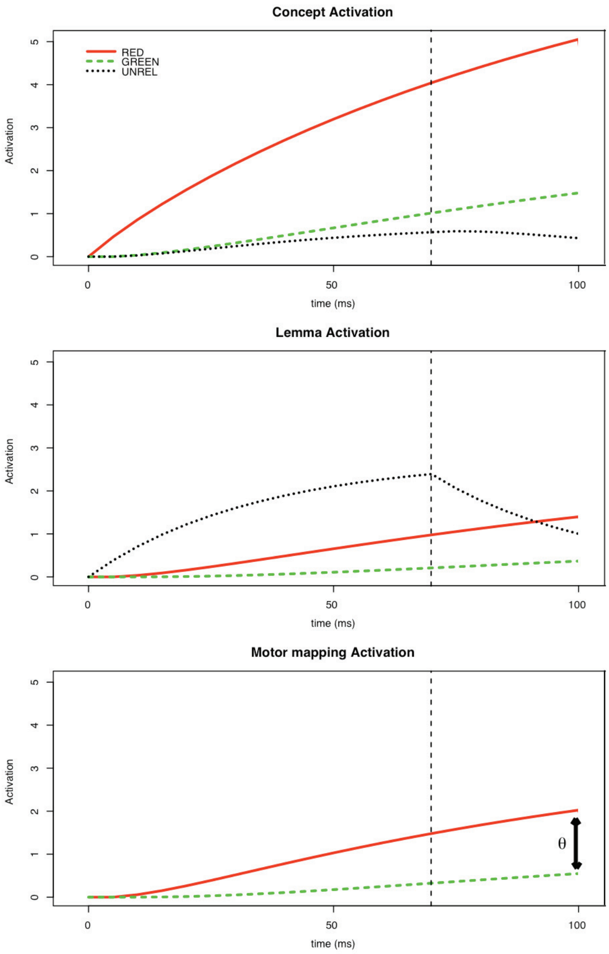


Figure 4: Activation accumulation in the Neutral Unaware condition. The stimuli are a red color patch and an unrelated prime word. The vertical dotted line indicates when the presentation of the prime ends. At this point, the activation of the related lemma has not crossed the retrieval ratio. After 100ms the red motor mapping chunk does cross the retrieval ratio.

[2] As the variance in the original data cannot be deduced from the published results, a sensible formal comparison is not possible.

## Discussion

A difficult question when modeling cognitive tasks that deal with awareness is how awareness is defined within the model. We chose to set a strict boundary for awareness, the retrieval ratio. When a chunk reaches the retrieval ratio, it becomes available inside the buffers. We assume that people are aware of chunks that are currently in the buffers (Taatgen, submitted), and not aware of chunks that have not yet reached the activation needed to enter the buffers.

RACE involves a direct connection between information in the external world (that is, the visual module) and the activation values of declarative chunks in declarative memory. In this respect, RACE deviates from ACT-R, in which all visual information must be mediated by the visual buffer. However, since the visual buffer is associated with awareness as chunks appearing in the visual buffer enter the declarative system, another pathway must be present to account for the subliminal priming data modeled in this paper. We hypothesize that the connections in RACE from the visual module to the declarative memory module may represent part of the ventral visual pathway, that is known to involve connections from striate cortex (associated with ACT-R's visual module) to temporal brain regions (associated with the declarative memory module, Anderson et al., 2004).

The model of subliminal priming discussed in this paper demonstrates that RACE can account for the retrieval latencies observed by Marcel (1983, Experiment 3). By using standard RACE parameter values, the fit of our model to the data set of Marcel was quite good. In combination with previous models of declarative memory retrieval that use RACE (Borst & Van Rijn, 2006; Van Maanen & Van Rijn, 2006;  in press), this suggests that RACE might be regarded as a general model of declarative memory retrieval. The added value of the RACE model is that it gives a rational account of how the process of declarative memory retrieval develops. Even the effects on declarative memory retrievals of changes in the world that last only milliseconds can now be taken into account.

## References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111*(4), 1036-1060.

Anderson, J. R., Bothell, D., Lebiere, C., & Matessa, M. (1998). An integrated theory of list memory. *Journal of Memory and Language, 38*(4), 341-380.

Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.

Anderson, J. R., Matessa, M., & Lebiere, C. (1998). The visual interface. In J. R. Anderson & C. Lebiere (Eds.), *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.

Anderson, J. R., & Milson, R. (1989). Human memory: An adaptive perspective. *Psychological Review, 96*(4), 703-719.

Anderson, J. R., & Schooler, L. J. (1991). Reflections of the environment in memory. *Psychological Science, 2*(6), 396-408.

Borst, J. P., & Van Rijn, H. (2006). Memory decay problems in a level-repetition switch task model. In D. Fum, F. Del Missier & A. Stocco (Eds.), *Proceedings of the Seventh International Conference on Cognitive Modeling* (pp. 351-352). Trieste, ITA.

Glaser, W. R., & Düngelhoff, F. J. (1984). The time course of picture-word interference. *Journal of Experimental Psychology: Human Perception and Performance, 10*(5), 640-654.

Marcel, A. J. (1983). Conscious and unconscious perception: Experiments on visual masking and word recognition. *Cognitive Psychology, 15*(2), 197-237.

Merikle, P. M., Smilek, D., & Eastwood, J. D. (2001). Perception without awareness: Perspectives from cognitive psychology. *Cognition, 79*(1-2), 115-134.

Newell, A. (1990). *Unified theories of cognition*. Cambridge, MA: Harvard UP.

Ratcliff, R., & Smith, P. L. (2004). A comparison of sequential sampling models for two-choice reaction time. *Psychological Review, 111*(2), 333-367.

Taatgen, N. (submitted). Consciousness in the ACT-R architecture. To appear in *Oxford companion to consciousness*: Oxford UP.

Taatgen, N., Van Rijn, H., & Anderson, J. R. (in press). An integrated theory of prospective time interval estimation: The role of cognition, attention and learning. *Psychological Review*.

Usher, M., & McClelland, J. L. (2001). The time course of perceptual choice: The leaky, competing accumulator model. *Psychological Review, 108*(3), 550-592.

Van Maanen, L., & Van Rijn, H. (2006). An accumulator model account of semantic interference in memory retrieval. In D. Fum, F. Del Missier & A. Stocco (Eds.), *Proceedings of the Seventh International Conference on Cognitive Modeling* (pp. 322-327). Trieste, ITA.

Van Maanen, L., & Van Rijn, H. (in press). An accumulator model of semantic interference. *Cognitive Systems Research*.

Van Rijn, H., & Anderson, J. R. (2003). Modeling lexical decision as ordinary retrieval. In F. Detje, D. Dörner & H. Schaub (Eds.), *Proceedings of the Fifth International Conference on Cognitive Modeling* (pp. 207-212). Bamberg. GER.

# How Readers Retrieve Referents for Nouns in Real Time: A Memory-based Model of Context Effects on Referent Accessibility

**Aryn Pyke** (aryn.pyke@gmail.com)
Institute of Cognitive Science, Carleton University,1125 Colonel By Drive
Ottawa, ON K1S 5B6 Canada

**Robert L. West** (rlwest@connect.carleton.ca)
Institute of Cognitive Science, 1125 Colonel By Drive
Ottawa, ON K1S 5B6 Canada

**Jo-Anne LeFevre** (jlefevre@connect.carleton.ca)
Department of Psychology, 1125 Colonel By Drive
Ottawa, ON K1S 5B6 Canada

## Abstract

When a reader encounters a noun, she tends to rapidly associate the noun with a mental referent (representation of entity in question). Our computational model confirms that a memory-based account is sufficient to account for a high rate of success at preliminary referent retrieval. Definite noun phrases ("The dog") can be used *anaphorically* to refer to referents already mentioned in the text, but they also frequently introduce a new referent into the mix (Poesio & Vieira,1998). An adequate model must account for how a reader makes an explicit or implicit decision about each noun's anaphoric status. We suggest that LTM contains both generic referent types and specific referent tokens, which simultaneously compete for retrieval via resonance. Our ACT-R simulation operationalizes the memory-based view to model the pre and post-noun activations of referents in memory. It predicts which referent will be retrieved (i.e. the most active), and consequently whether an anaphor will be initially treated as a new referent. The influence of anaphor word choice is explained, and encompasses metaphoric anaphors. Simulations results are congruent with human performance in our eye-tracked reading study, in which regressions to reanalyze an anaphor are indicative of the incidence of preliminary error.

**Keywords:** noun anaphora; memory-based text processing; resonance; reference assignment; cognitive modeling; ACT-R

## Introduction

Evidence suggests that readers interpret incoming sentences incrementally, as the words become serially available (e.g., Sevidy, et al., 1999). In particular, human interpreters make a rapid preliminary association of a noun phrase (e.g., "the fruit) with a referent almost immediately after they encounter the noun (Just & Carpenter, 1980; see also Dell, McKoon, & Ratcliff, 1983, Sanford & Garrod, 1989). The term ‘referent’ is used here in the cognitive sense (as in Gundel, Hedberg, & Zacharski, 2001) to mean a mental representation of the person or object in question. At the time a noun is encountered, the processing system is not privy to information in the sentence that occurs after said noun. Thus, a reader's preliminary referent assignment is influenced by the preceding context and the noun itself. Our objective was to model this preliminary, on-line referent assignment for definite noun phrases. Such preliminary referent assignments may in turn be subject to subsequent adjustment. However, the present focus is on the nature and accuracy of the preliminary referent assignment process itself. The performance of the model will be compared with the accuracy of human data.

## The Cognitive Task: Referent Assignment

To motivate the model, we first discuss the cognitive task (problem space) in more detail. In general, a noun can be used to introduce a new referent into the discourse (introductory use), or to refer to a referent that was previously discussed (anaphoric use). To enable us to process these two uses, we have two types of referents in memory: representations of specific people and objects that the cognitive agent is already familiar with (e.g., the particular apple in your bag), and more generic/prototypical representations (e.g. a generic apple). The latter, generic referents are appropriate to retrieve when the noun is used in an introductory capacity. For example, in (2) “fruit" is used anaphorically to denote the referent introduced by the antecedent “apple” in (1). Note that an anaphoric noun need not match the label that was previously used for that referent, and the same noun "apple" can be used to refer to different referents in (1) and (3).

(1) John bought an apple.
(2) John bit into the fruit.
(3) The apple Mary bought was green.

A more fundamental issue is that a definite noun, "The apple" bears no explicit indication of whether it used in an introductory or anaphoric capacity. The definite article “the” was often assumed to indicate that the intended referent is already familiar to the reader (e.g., Clark & Sengul, 1979; Garnham, 1989; Just & Carpenter, 1987). However, as exemplified in (3), corpus analyses have established that definite noun phrases are equally likely to introduce new referents into the discourse as to denote referents that have already been mentioned (Gundel et al., 2001; Poesio & Vieira, 1998). Thus, readers are not privy to the anaphoric status of a definite noun a priori. Consequently, during preliminary assignment, readers may sometimes misinterpret an anaphoric noun as if it is introducing a new referent (or an introductory noun as anaphoric). In the literature, the reader’s task upon encountering a definite noun phrase is sometimes dubbed *anaphor resolution*, but this terminology clearly under-represent the full scope of the reader's (and modeler's) task. Thus, the job of the present model is to describe the means in which a referent for a noun (in a given context) is rapidly selected on-encounter. And in so doing, the model should serve to predict the likelihood that the selected referent will be the correct one (i.e., will it be generic-new or specific).

The present treatment applies to the preliminary assignment of referents to definite *nouns*. As Gernsbacher (1989) suggests, that this memory-based process may apply to other types of referring expressions like names (John) and pronouns, (he). However, this issue goes beyond our present data. Furthermore, in contrast to nouns, which can be introductory or anaphoric, pronouns are almost exclusively anaphoric (Kintsch, 1998), and are subject to more syntactic constraints, so there is reason to believe that the problem space and process may be somewhat different. Consequently, appropriate referents for non-noun referring expressions are effectively hard-coded in our simulation.

## Criteria for a Cognitive Referent Retrieval Model

Reference assignment is addressed by some non-cognitive Natural Language Processing algorithms (e.g., Bean & Riloff, 1999; Vieira & Poesio, 2000), however, they involve multiple passes forward and back through the text, and do not directly speak to the development of a *cognitive* model of *on-line* processing. Thus, to set the stage for the proposed model, we layout the general criteria for a psychologically plausible model of referent retrieval for noun phrases.

1. **Incremental Processing:** The system should be able to make use of information in each incoming sentence incrementally, roughly word-by-word.
2. **Appropriate Representational Units**: In line criteria 1, the unit of analysis for the processing system must be smaller than a complete sentence. The cognitive task being modeled is to associate a definite noun phrase with a mental representation of its referent, so the system must, minimally, have representations (though possibly atomic ones) for individual nouns and potential referents, in order to model the task of associating the former to the latter.
3. **Context Sensitivity:** The model should allow and account for the influence of: (i) preceding context sentences, and (ii) preceding parts of the current sentence, and (iii) the current noun itself, on the processing of the current noun. To account for the processing of a particular anaphor, the model should also, as a pre-requisite, model the processing of the prior context, to get the memory system into the appropriate state (so relative accessibilities of specific and generic referents reflect the influence of prior context).
4. **Real-Time Simulation**: Ideally, the reading process should be simulated in real-time units (e.g., ms vs. 'cycles'). Cognitive effects like priming (spreading activation) are time-sensitive and subject to decay. The system should ideally model such memory effects (fluctuations in activation over time), and should simulate processing of the text at a representative reading rate (e.g., 150 ms/word).
5. **Appropriate Problem Space**: Models necessarily abstract away from some level of detail, but care must be taken to ensure that the problem space in the model is not artificially skewed or trivially sparse. The characterization of the task and the representation of the problem space (e.g., range of possible referent choices) should be sufficiently rich to reflect the interpretation challenge facing a real reader, and so permit key types of possible errors. Since readers do not know a priori whether a current definite noun phrase is anaphoric or introductory, the explicit or implicit determination of this property is part of the referent assignment process. Thus, the model should ideally be able to operate on and discriminate between (though not always accurately) both anaphoric and non-anaphoric definite noun phrases. In this vein, the memory system must be populated with not only the correct referent, but also the other referents in the discourse, and generic referent prototypes.

We are unaware of a previous model that fulfills all of these criteria. Other models (e.g. Budiu & Anderson, 2004, Lemaire & Bianco, 2003) fulfill some but not others, notably criteria 2 and 5. In view of these criteria, the model was implemented in ACT-R (discussed later). Next, we outline the theoretical framework underlying our model.

## The Framework: Memory-based Processing

Our model is inspired by the memory-based view of text-processing (see Gerrig & O'Brien, 2005 for a review). In particular, we propose that preliminary referent retrieval is driven by general-purpose memory mechanisms. In particular, under the resonance model (e.g., Gernsbacher, 1989; Myers & O'Brien, 1998) current information in working memory (i.e., the anaphoric noun) serves as a cue that automatically boosts activation of other entities throughout long-term memory -- including, ideally, the intended referent -- in accord with their conceptual overlap with the cue. Thus, at the time the anaphor 'fruit' in (2) is encountered, the apple referent can be automatically re-activated via resonance in virtue of its conceptual overlap with the anaphor (a pre-existing conceptual association). In our model (see also see also Budiu & Anderson, 2004), the strengths of conceptual associations were estimated using Latent Semantic Analysis (LSA) values (lsa.colorado.edu; see Landauer, Foltz, & Laham, 1998 for a review), which give a maximum similarity of 1 (i.e., similarity of a concept to itself). So had the anaphor used been "apple" it would have provided an even larger activation boost to the intended referent [R1:apple], than was provided by the anaphor "fruit" (LSAs:<apple,fruit>=.47,<apple,apple> =1). Thus choice of noun itself exerts an immediate impact on the relative accessibilities of referents via resonance.

This memory-based account can be contrasted with a more special-purpose process of referent retrieval. Some suggest that when a reader encounters an anaphoric noun, he/she undertakes a proactive search for a referent mentioned within the current text (e.g., Clark & Sengul, 1979; Kintsch & Vandijk, 1978; O'Brien, Plewes, & Albrecht, 1990). The discourse might be mentally represented as a series or network of propositions, and the reader might systematically troll backwards through it in search of a referent that (according to some criterion) could constitute a match to the current anaphor term. However, since many definite nouns not anaphors such a proactive, process-of-elimination search would often be a waste of time. Further, if general memory mechanisms are often sufficient to automatically bring a referent to mind, parsimony argues against the proposal of a proactive special-purpose search process. The model in this paper confirms that memory-based accounts are sufficient to account for a high rate of success at preliminary referent retrieval for anaphoric nouns. The theoretical and empirical arguments against a special-purpose search account are outlined more fully in Pyke, West and LeFevre (2007).

In our model, generic referents and specific discourse referents simultaneously compete for retrieval. The most active referent is retrieved. Thus,it is *not* the failure to find/retrieve a referent that then, serially, leads to treating a noun as a new referent. Rather, retrieval, per se, typically always succeeds. Whether it is a specific or generic referent that is retrieved determines whether the noun as treated as anaphoric or introductory in preliminary analysis.

### Factors Affecting the Activation Levels of Referents

Each referent's accessibility during preliminary noun processing owes to two components: i) the activation boost/spread from the noun term currently being processed; and ii) its 'context dependent' pre-noun activation level. Such context factors are outlined below.

1.**Spread of Activation from Pre-Anaphor Words**. Just as the anaphor resonates with, or spreads activation to, referents, our model assumes that such activation spread generally occurs as each content word in the discourse is encountered. The activation boosts received by referents may persist (as do lexical priming effects, e.g., Collins & Loftus, 1975) even when the reader progresses on to the next word. While such effects decay they may exert a cumulative effect on a referent's activation.

3.**Recency and Frequency of Use of the Referent**. These general factors affect any mental representation's accessibility. Evidence indicates that the further back that an antecedent is (in sentences, and consequently in time), the more challenging it is to process the anaphor (e.g., Clark et al., 1979; Duffy & Rayner, 1990; Levine, Guzman, & Klin, 2000). A referent referred to many times in a text, and/or referred to in the sentence preceding the critical anaphor should be more active, ceteris paribus, than a referent mentioned only once several sentences back.

4.**Sentence Wrap-Up Effects**. Just and Carpenter (1980) suggested that integrative processes occur at sentence end, which is why readers spend tend to spend relatively longer on the final word in each sentence. These wrap-up processes may result in sentence-end activation effects (Balogh, Zurif, Prather, Swinney, & Finkel, 1998). Probe studies suggest that a referent mentioned early in a sentence may also produce facilitation effects at sentence end (e.g., Dell et al., 1983; McKoon & Ratcliff, 1980). In our model, the processes at sentence end result in an activation boost of the specific referents mentioned in the sentence.

5. **Discourse Dependent Associations**. In addition to pre-existing associations like those we are modeling with LSA, discourse dependent associations may be formed in memory. Spread of activation through such associations may produce intermittent (yet cumulatively significant) activation contributions to an intended referent during pre-anaphor processing. For example, each sentence (and proposition) in a discourse may contain several referents. In Dell et al. (1983, see also McKoon & Ratcliff, 1980), two referents which have appeared in a common sentence are called *companions*. The comprehension process appears to forge a lasting association between companion referents in memory, possibly during sentence wrap-up, such that when a referent is subsequently encountered, its companion(s) from prior sentences also become re-activated right away, and to a comparable degree (Dell et al., 1983). Thus, in our model, whenever a referent is mentioned, its companions are also boosted in activation, thereby making them more accessible as referent candidates for up-coming nouns.

Our treatment extends the prior treatments in the literature in that it provides a more explicit, comprehensive, quantitative and real-time operationalization of such contextual influences on the LTM referents' pre-anaphor activation levels. Furthermore, most discussions of noun anaphor processing in the literature (c.f., Garrod, Freudenthal & Boyle, 1994, Levine et al, 2000), fail to address the fact that the reader does not know that a noun is an anaphor a priori (The classification problem, see also Pyke, West & LeFevre, 2007), so the incidence of new-referent errors during preliminary assignment has been largely unexplored and, in our view, underestimated.

## The Data: Human Performance

How likely are readers to associate an anaphoric noun to a new referent on encounter? It was originally believed that people made almost no preliminary errors in assigning a referent to noun anaphors (Sanford & Garrod, 1989), however subsequent evidence has established that such errors do indeed occur (e.g., Levine, Guzman & Klin, 2000, and the present research). Such errors were elicited by manipulating factors related to the accessibility of the intended referent (e.g., the referent had not been mentioned for several sentences).

Because a reader's preliminary referent assignment for a noun occurs on-line, (mid-sentence), and is not directly observable by an experimenter, information about the nature (accuracy) of such preliminary assignments is somewhat challenging to empirically obtain. Note that the reader's final interpretation at sentence end may reflect the influence of subsequent processes and information. We observe that a reader's eye-movements can provide a useful indication of the effectiveness of preliminary referent assignment. In the course of processing the remainder of the sentence after the anaphor, errors in preliminary assignment are often detected. In general, when readers detect that they have made earlier errors in interpretation, they often regress their eyes back to the site of the initial misinterpretation to do an *overt reanalysis* (e.g., Altmann, Garnham, & Dennis, 1992; Meseguer, Carreiras, & Clifton, 2002). We conducted an eye-tracked reading study to determine the relative likelihood of regressions to reanalyze the anaphor, when we manipulated the choice of anaphoric noun.

## Experiment

The stimuli were 42 stories that were each 4 sentences long. The fourth, critical sentence commenced with a definite noun phrase: "The (noun)", and this noun was intended anaphoricallly to denote a target referent introduced in the first or second sentence. For example:

Mary had a pet terrier.
It was white and shaggy.
She took it to the beach.
The terrier/dog/mop barked at the birds.

Anaphor word choice was manipulated so each of the 42 stories had three different versions: i) antecedent-match (terrier-terrier), – anaphor was identical to the noun which originally introduced the referent ii) category (terrier-dog); and iii) metaphoric (terrier-mop). Recent research suggests that the same general mechanisms apply to the processing of literal and figurative content (Budiu & Anderson, 2004, Giora, 2002; Glucksberg, 2003; Kintsch, 1998, chap. 5.3). The memory-based model is compatible with this claim. Activation spreads automatically in accord with similarity, so just as activation spreads from the anaphor dog to the referent [R1:white, shaggy, terrier], activation should also spread, from the anaphor mop to the mental representation of the white, shaggy terrier. The latter case provides less spread of activation (LSAs: <dog, white-shaggy-terrier>= .18, <mop, white-shaggy-terrier>=.07), so readers should be more prone to make a preliminary new-referent assignment errors for the metaphoric anaphors.

***Procedure.*** Stories were presented line-by-line to participants (N=24), while an EyeGaze™ System tracked their right eye. Story versions were counterbalanced, so each participant saw only one version of each story, and thus saw 14 stories of each anaphor type.

***Results***. Eye-movements were analyzed for the critical sentence. First-pass reading time for the anaphor did not vary for the different noun versions ($p$>0.05, word length and frequency as co-variates), however, the likelihood of later regressing the eye back to the anaphor differed, $F$(3,71.9) = 7.51, $MSE$ = 1.60, $p$ = .000. Readers made a regression back to the anaphor on 52% of metaphoric anaphors, 36% of category anaphors and 41% of antecedent-match anaphors (see also Figure 2).

*Discussion*: Regressions to the antecedent-match and the category anaphors were comparable, though, somewhat surprisingly, more regressions were made to the antecedent-match anaphors. Readers regressed most to the metaphorically intended anaphors, presumably because they had made many preliminary assignment errors and treated them as new referents. This explanation was confirmed in a follow-up cloze study. Readers were presented with the first 3 sentences and the critical noun phrase (e.g., The mop_____________), and created their own completion of the sentence. The completions revealed whether the reader had associated the noun with the intended referent or had treated it as a new referent. In 48% of the trials, readers associated the metaphoric anaphor with a new referent.

# The Model

In the model, the referent retrieval process in play while reading the anaphoric noun is basic, blind and strictly memory-based: When 'reading' the noun, activation automatically spreads from the noun to all referents in memory (both generic and specific), and then the most active referent from memory is retrieved. The impact of prior discourse processing will be entirely mediated by its lingering effect on the activations of the various referents in LTM. Thus, to model the preliminary referent assignment for a particular noun, we must also simulate the processing of the preceding discourse, but only so far as is necessary to approximate its effect on the immediately pre-anaphor activation levels of the various referents in LTM.

The 3 versions of each story are identical up until the anaphor. Thus the pre-anaphor activation levels of the intended referent and other specific referents in the story are the same across versions (say, [R1:Mary]: 2.4, [R2:terrier]:1.4, [R3:beach]:2.8). What about generic referents? The anaphor term determines the relevant (most competitive) generic referent in play. If the anaphor is "mop", the intended referent [R2:terrier] competes not just with the other story referents but also with the generic referent [G1::mop]. For the version of the story with the anaphor dog, the relevant generic competitor is [G2:dog]. The activation levels of the generic competitor will depend on the spread of activation it receives from the pre-anaphor words and the anaphor itself. In contrast, specific referents in the story also get activation boosts from sentence-wrap up and companion spreading.

## Overview of Operation

0. LTM is seeded with generic referents for various discourse concepts, including, importantly the antecedent concept and also (if different) the anaphor concept. For the "mop" version of our example story, memory will be seed with [R1:Mary], [G1:terrier], [G2:Beach], [G3:birds] and [G1:mop]. If, upon reading

1. Words of a story are then processed serially.

2. Each content word (e.g., noun, adjective, verb) automatically spreads activation to both specific and generic referents in memory according to the LSA similarity between the word and the referent.

3. If the current word is a referring term (noun, name, pronoun), a referent is retrieved from memory. The referent retrieved will be the most active one, be it specific or generic. In the latter case, the generic referent is used to create a new specific referent to associate with the noun.

The assigned referent is automatically boosted in activation (in virtue of it's current use), and activation also spreads from it to its companion referents from previous sentences. A fan-effect applied, so if the current referent had n companions in the previous sentence, the weight factor for activation spread to each companion will be (1/n).

4. At the end of a sentence during wrap-up, the sentence's referents become reactivated and mutually associated.

Besides the original descriptor used to introduce a referent (e.g., "terrier"), other properties can also be explicitly mentioned in a text (e.g., white & shaggy). In step 2 above, activation from incoming words spreads to each *explicit* attribute of each referent [R1:terrier,white,shaggy]. Each attribute then spreads activation to the referent representation as a whole. The more active (recently mentioned or primed) an attribute is, the stronger its relative contribution (weight = attribute's activation level * LSA<current word, attribute>).

## Implementation Architecture: ACT-R

The criteria outlined previously motivated the choice of ACT-R (Anderson et al., 2004) as a suitable cognitive modeling platform for our model. In particular, we used the python extension of ACT-R (Stewart & West, in press).

The ACT-R architecture is predicated on a Unified Theory of Cognition (Newell, 1987) - that is, on the belief that our performance on a vast range of tasks can be accounted for parsimoniously by a common cognitive system operating with general-purpose mechanisms and principles. As such, ACT-R is spiritually compatible with the memory-based framework in which referent retrieval is attributed to general-purpose memory mechanisms.

Computer implementations do not inherently impose any psychological constraints on the character of the model, nor do they necessarily simulate the process in real-time. Consequently, the cognitive plausibility of one-off task-specific models is sometimes open to question. ACT-R has in-built psychologically motivated constraints, though it does have some 'arbitrarily' adjustable parameters. However, the architecture has proved conducive to modeling a vast range of cognitive tasks, and the data accrued has provided theoretically and empirically motivated constraints for the values/ranges for its key parameters. In our model, key parameters are set to recommended 'universal' defaults (e.g., noise=0.3, decay=0.5, production time=50 ms).

ACT-R supports two concurrent levels of functionality:

(i) a production system that carries out sequences of situation-driven productions (i.e. if-then rules) that serve as the procedural building blocks (steps) for various tasks.

(ii) a dynamic memory system which contains the various respresentational units (called chunks) upon which the productions act. The memory system can simulate the real-time fluctuation of activation of a representational unit. For example, the effect of recency and frequency of use on the activation level of representation $R_i$ is quantified by $B_{Ri}$, where tj is the time since use j, and d is a constant whose default value is .5 (Anderson et al., 2004).

$$B_{Ri} = \ln \sum_{j=1}^{N} \text{tj\^{}-d}$$

Thus, ACT-R affords sufficient functionality to fulfill almost all of the desired criteria: i) incremental processing: cognitive operations (productions) can be executed at a rate of one per 50ms, so a model can perform several operations (e.g. recognize word, spread activation, retrieve referent) during the typical reading time for each incoming word (150 ms); ii) representational units – the modeler can specify the types and numbers of representations in memory, for example, noun-chunks, generic referent chunks, and specific referent chunks, iv) problem space – the architecture allows the modeler to define the task (set of productions) and populate memory (set of chunks) as appropriate, iii) real-time simulation – the rate of productions is paced to reflect the time for the mind to perform a single simple operation, and the memory system can simulate the fluctuations in activation of the chunks (referents) over time.

The content that is currently in the system's focus (e.g., the noun-chunk), can spread activation to other chunks (e.g., referent-chunks) in memory. However, in traditional ACT-R the boost in activation received by a chunk (referent) from a stimulus (word) is removed as soon as the stimulus word is no longer in focus. This precludes any priming effects of spreading activation from prior words. To allow for such persistent effects in our model, we introduced this functionality into python ACT-R. In so doing, we feel we've augmented rather than circumvented the psychological plausibility of the architecture.

## Simulation Results

The model was run 100 times on each version (match, category, metaphor) of each of 42 stimuli stories. Figure 1 depicts mean pre and post anaphor activation levels of the intended referent (R) and the relevant generic competitor (G). Match anaphors produced the highest, absolute post-spread R-activations. The simulation revealed considerable variation in pre-anaphor accessibility of the target referent from story to story. However, the pre-anaphor R-activation is the same for all 3 versions of a given story. In general, the greater the activation ‘head-start’ a specific intended referent has due to pre-anaphor context influences, the greater the latitude in anaphor word choice.

Figure 2 indicates how frequently the simulation retrieved the correct referent. These results are correlated with the likelihood of regression for the 42*3 items in the human data (r=-.333, p=.000). For match anaphors, humans regressed on more trials than would be expected in light of the minimal number of preliminary referent retrieval errors predicted by the model (<10%). Regressions in the antecedent-match case may result not only from preliminary errors but may be inflated due to a pragmatic "repeated name effect" (Kennison & Gordon, 1997). When a reader does regress, they may be re-engaging the memory-based referent retrieval process, at a time when the referent's accessibility may be boosted due to priming from post-anaphor words. We will extend our simulation to test this.

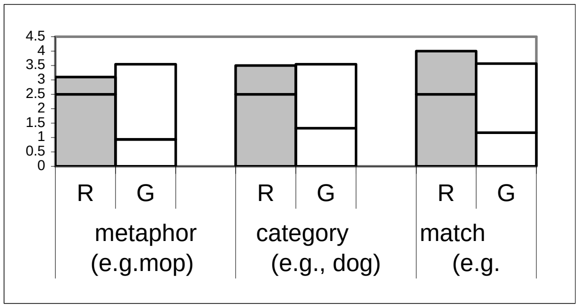


**Figure 1**: Pre(lower half) & Post-spread activation level for the intended referent (R) and its generic competitor (G).

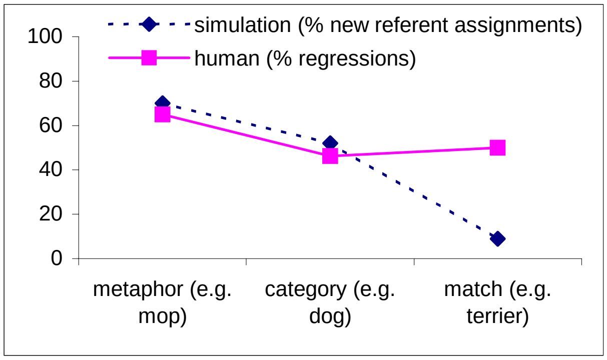


**Figure 2**:Comparison of Simulation & Human Performance

## Concluding Remarks

Our model operationalizes the memory-based view to estimate the (pre & post-noun) accessibilities of referents in memory, and thus predicts when a particular anaphor will be initially misinterpreted as a new referent. Such a simulation tool could have a practical application to assess how comprehensible (each referring expression in a) text is. And in psycholinguistics research to check whether accessibility levels are controlled/comparable across stimuli. Although readers can often later correct preliminary errors, their final representation of the text may be degraded since vestiges of misinterpretations persist in memory (Johnson & Seifert, 1998). Further research is necessary to explore such potential long-term 'costs' of using anaphors that are difficult to resolve during preliminary analysis.

## Acknowledgments

We express our thanks to Terry Stewart and NSERC.

## References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. L. (2004). An integrated theory of the mind. *Psychological Review, 111,* 1036-1060.

Balogh, J., Zurif, E., Prather, P., Swinney, D., & Finkel, L. (1998). Gap-filling and end-of-sentence effects in real-time language processing. *Brain and Language, 61,* 169-182.

Bean, D. L. & Riloff, E. (1999). Corpus-based identification of non-anaphoric noun phrases. *In Proceedings of the 37th Annual Meeting of the Assoc. for Computational Linguistics.,* 373-380.

Budiu, R. & Anderson, J. R. (2004). Interpretation-based processing: a unified theory of semantic sentence comprehension. *Cognitive Science, 28,* 1-44.

Clark, H. H. & Sengul, C. J. (1979). In search of referents for nouns and pronouns. *Memory & Cognition, 7,* 35-41.

Dell, G., McKoon, G., & Ratcliff, R. (1983). The activation of antecedent information during the processing of anaphoric reference.*.J. of Verb. Learning and Verb. Behav 22,*121-132

Duffy, S. A. & Rayner, K. (1990). Eye-movements and anaphor resolution - Effects of antecedent typicality and distance. *Language and Speech, 33,* 103-119.

Garrod, S., Freudenthal, D., & Boyle, E. (1994). The role of different types of anaphor in the online resolution of sentences in a discourse. *Journal of Memory and Language,* 33, 39-68.

Gerrig, R., & O'Brien, E. (2005). The scope of memory-based text processing. *Discourse Processes,* 39(2&3), 225-242.

Gernsbacher, M. A. (1989). Mechanisms that improve referential access. *Cognition, 32,* 99-156.

Gundel, J., Hedberg, N., & Zacharski, R. (2001). Definite descriptions and cognitive status in English. *Journal of English Language and Linguistics, 5,* 273-295.

Just, M. A. & Carpenter, P. A. (1987). *The psychology of reading and language comprehension.* Boston: Allyn and Bacon, Inc.

Kennison, S. & Gordon, P. (1997). Comprehending Referential Expressions During Reading. *Discourse Processes, 24,* 229-252

Kintsch, W. (1998). *Comprehension: A Paradigm for Cognition.* New York: Cambridge University Press

Kintsch, W. & Vandijk, T. A. (1978). Toward a model of text comprehension. *Psychological Review, 85,* 363-394.

Landauer, T.,Foltz, P.& Laham,D.(1998).An introduction to latent semantic analysis. *Discourse Processes,25,* 259-284

Lemaire, B. & Bianco, M. (2003). Context effects on metaphor comprehension: Experiment and simulation. In *Proceedings of 5th Int.Conf. on Cognitive Modelling (ICCM),* 153-158

Levine, W. H., Guzman, A. E., & Klin, C. M. (2000). When anaphor resolution fails. *J. of Mem. and Language, 43,* 594-617.

Myers, J. L. & O'Brien, E. J. (1998). Accessing the discourse representation during reading. *Discourse Processes,26,* 131-157.

O'Brien, E. J., Plewes, P. S., & Albrecht, J. E. (1990). Antecedent retrieval-processes. *JEP: LMC, 16,* 241-249.

Poesio, M. & Vieira, R. (1998). A corpus-based investigation of definite description use. *Computational Linguistics, 24,*183-216.

Pyke, A., West, R. L., & LeFevre, J. (2007). On-line reference assignment for anaphoric and non-anaphoric nouns. In *Proc. of* the 29th Annual Meeting of the Cognitive Science Society.

Sanford, A. J. & Garrod, S. C. (1989). What, when, and how?: Questions of immediacy in anaphoric reference resolution. Language and Cognitive Processes, 4, 235-262.

Sedivy, J., Tanenhaus, M., Chambers, C., & Carlson, G. (1999). Achieving incremental semantic interpretation through contextual representation. *Cognition, 71,* 109-147.

Stewart, T.C., West, R.L. (in press). Deconstructing and Reconstructing ACT-R: Exploring the Architectural Space. *Cognitive Systems Research.*

# A model of parallel time estimation

**Hedderik van Rijn[1] and Niels Taatgen[1,2]**

[1]Department of Artificial Intelligence, University of Groningen
Grote Kruisstraat 2/1, 9712 TS Groningen

[2]Department of Psychology, Carnegie Mellon University
5000 Forbes Av., Pittsburgh PA 15212

## Abstract

In earlier work, Taatgen, Van Rijn and Anderson (in press) have shown that embedding a simple module that generates temporal information in a more general cognitive architecture explains timing phenomena that were earlier attributed to a hypothesized more complex temporal system. However, the embedded temporal module does not support parallel time estimations, of, for example, two concurrent estimations. Explaining the human capacity of doing multiple time estimations requires either adding additional temporal modules, or assuming higher level processing to strategically use a single timer for parallel timing. This paper presents an experiment and a computational model that show that the latter approach is more plausible for human parallel time estimation.

**Keywords:** parallel time estimation; temporal processing; temporal arithmetic; ACT-R; cognitive modeling.

## Introduction

Timing is an essential aspect of human behavior, both inside and outside psychological laboratories. How long does one wait before pressing the back-button in an Internet browser when the requested page does not appear? How does one know how time passes during a lexical-decision experiment in which one has to be as fast and accurate as possible? Several theories have tried to describe the nature of time estimation both in terms of brain areas and processes (e.g., Buhusi & Meck, 2005), and in terms of behavior (e.g., Gibbon, 1977). Although accounts differ on the exact mechanisms, all assume time is perceived on a logarithmic scale, which means that two longer intervals that differ in duration by a certain amount of time are considered to be more similar than two shorter intervals differing by the same amount of time. In other words, perceived differences grow smaller as the duration increases. Because a logarithmic scale needs a starting point, a start signal is needed to indicate the beginning of an interval. The logarithmic scale is reflected in empirical observations such as the scalar property (the variance of a time-estimation distribution is linearly related to the duration of the estimated time, Gibbon, 1977) and bisection phenomena (a duration exactly in between a long and a short interval is more often considered long than short, Allan & Gibbon, 1991).

The requirement of a start-signal and a logarithmic scale raises the question whether and how multiple overlapping time intervals can be measured. Several studies have investigated this question, and have concluded that both animals and humans are capable of estimating multiple intervals (Meck & Church, 1984; Ivry & Richardson, 2002; Brown & West, 1990; Penney, Gibbon & Meck, 2000). These studies either seem to suggest or are the basis for claims that there are multiple clocks that can operate in parallel. However, an alternative explanation is that there is only a single clock that is intelligently used by the cognitive system to estimate multiple intervals in parallel. To test this hypothesis, we designed the experiment presented below. In this experiment, participants have to produce two time intervals that overlap partially. They receive a start signal for one of the intervals, and after a SOA of 500-1500ms the start signal for the second interval. They then have to respond to each of the intervals at the appropriate moment. The random interval between the two start signals prevents fixed timing strategies, and produces a variable overlap between the two intervals (Figure 1).

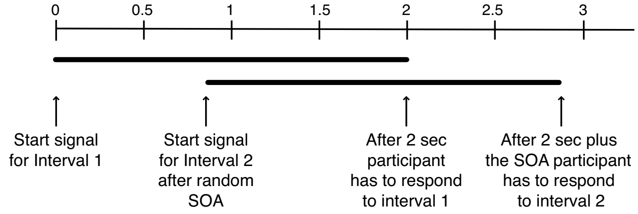

0
0.5
1
1.5
2
2.5
3
Start signal for Interval 1
Start signal for Interval 2 after random SOA
After 2 sec participant has to respond to interval 1
After 2 sec plus the SOA participant has to respond to interval 2

Figure 1: Experimental paradigm used in the experiment

Based on our approach to time perception, which we will discuss in more detail later, two accounts of multiple, parallel time estimation can be proposed. One proposal is to have multiple timers that can be used independently[1], an account similar to that suggested by Meck and Church (1984). If humans can recruit multiple parallel timers, the timers themselves should not produce any decrease in performance when multiple parallel intervals have to be estimated. A decrease in performance, however, can be due to other factors like attention, dual-tasking costs, the need to learn multiple different intervals, etc. These factors are either independent of the overlap between the two intervals, or, in the case of dual-tasking costs, increase with the

[1] Note that this does not necessarily imply multiple neural clocks. It might be that a single time source, for example an oscillating group of neurons, is driving multiple estimations.

amount of overlap. The general finding is that multi-tasking increases the length of time estimations (Block & Zakay, 1997). The multiple timer account would therefore predict either complete independence of the estimates, or an increase of the estimates with an increased amount of overlap between the intervals.

A second proposal is a single source of time information that can be used strategically by the general cognitive architecture to estimate multiple parallel time intervals sequentially. To produce the two intervals in Figure 1, the SOA between the two start signals has to be remembered during the production of the first interval. After the response on the first interval has been made, one has to wait for a duration equal to the remembered SOA before making the second response. The consequence of this method is that estimates are no longer independent. For example, if the first estimate is too long we also expect the second estimate to be too long. A second consequence of serialization is that the logarithmic time scale will bias the second estimate. The 800ms between the onset of stimulus one and two is internally represented on the logarithmic scale, resulting in, for example, an internal length of 5. When this internal representation is added to the first interval to estimate the second interval, this length of 5 represents a longer time than was earlier perceived. This temporal discounting results in overestimates of the second interval, which becomes larger as the SOA increases. A larger SOA leads to a smaller overlap between intervals, so where the single-timer account predicts longer estimates for larger SOAs (because of the logarithmic scale), the multiple-timer account predicts either no effect (no multitask penalty) or shorter estimates for large SOAs than for short SOAs (assuming a multitask penalty).

# Experiment

## Method

**Subjects** Twenty-six students from Carnegie Mellon University were paid $8 for participation in the experiment. Five participants were excluded from further analysis because of not adhering to the instructions.

**Design** The experiment consists of two blocks. The purpose of the first block was to have the participants learn a solid and correct representation of the to be estimated time. Hereto, no parallel timing was necessary in this block: participants were required to estimate just a single interval per trial. For each trial, correctness feedback was given. In the second block, this learned duration had to be reproduced twice per trial as illustrated in Figure 1. The main manipulation in this second block was the stimulus onset asynchrony.

**Stimuli & Procedure** In the first block of 46 trials, participants were asked to estimate an unspecified interval. Each trial started with a colored circle being shown on either the left or the right side of the screen. The participants were instructed to press a key when the presented stimulus was on the screen for the to be estimated time. For the left, green stimulus the “z” had to be pressed, for the right, blue stimulus the “/”. During 6 startup trials, their response was plotted on a timeline where an area was marked as correct. This way, participants could infer whether they had to lengthen or shorten their estimated durations. The marked region ranged from 1750 to 2250ms, making a response of 2000ms optimal. After the six startup trials with timeline feedback, feedback was only given in terms of “correct”, “too fast” or “too slow”. In the second, experimental block of 120 trials, participants had to respond to both left and right stimuli in each trial. Either stimulus appeared first in half of the trials. The stimulus onset asynchrony (SOA) was randomly sampled from the interval a=<500, 900> or b=<1100, 1500>. For both stimuli, feedback was given as “correct”, “too fast” or “too slow”.

## Results and Discussion

The first block of trials was presented to internalize the to be estimated duration and to asses single estimation performance. The average estimated duration in the last ten trials of the first block was 1930ms (SD=383). Note that given the positive skew in the distribution of time estimations, an estimated time shorter than 2000ms is optimal. The proportion of correct responses is .583, which is close to the expected proportion given the used correctness-range.

The proportion of correct responses in the second, experimental block was .448 for the first estimation and .435 for the second estimation. This difference is not significant (*t*(20)=0.59). When compared with Block 1, both first and second estimations of Block 2 show worse performance than during the last phase of Block 1 (*t*(20)=2.46), *p* = 0.023 and *t*(20)=2.72, *p*=0.013 respectively). At first sight, this seems to be in line with a multiple independent clocks account with a dual-tasking penalty. However, another prediction of this account is that the amount of overlap – and therefore the SOA – should influence the accuracy. For short SOAs, associated with a longer overlap and therefore a higher dual-tasking penalty, the accuracy should be lower than for long SOAs, which result in a shorter overlap. To test this, we compared two linear mixed effect (LME) models (Bates, 2005, and see for a non-technical introduction, Baayen et al, submitted). The first model, predicting accuracy of the second estimate using a binomial distribution, contains trial number (to account for learning or fatigue effects), starting side (to account for possible effects on left versus right initial presentation), SOA and first estimated duration, and a random effect for participants. The second model is identical apart from having SOA removed, representing an account in which SOA, and therefore multitasking-penalty, does not influence performance. A model comparison shows that the first model with SOA has indeed a significantly better fit to the data ( $\chi^2_{(1)}$=5.61,p=0.018), indicating that SOA does have an effect on proportion of correct responses. The direction of the estimated effect ($\hat{\beta}$=-.0003, z=-2.34, p=0.019) implies that longer SOAs are associated with a lower accuracy on the second estimate, which is consistent with the single timer account, but in the *opposite direction* as predicted by a dual-tasking penalty account.

To test whether the first estimate has an influence on the second estimate, we again compared two LME models. The first model contains the second estimated duration as a function of the fixed effects of trial number, starting side, SOA and first estimated duration, and a random effect for subjects. The second model has the first estimated duration removed, but is otherwise identical. The fit of the second model is significantly worse ($\chi^2_{(1)}$=316,p<0.001), indicating a significant contribution of the first estimated duration for the prediction of the second estimated duration. The estimated effect ($\hat{\beta}$=.284, t(3335)=18.21, p<0.001) indicates that longer first estimations yield longer second estimations. This is obviously in line with a single timer account, as the second estimation is dependent on the first. However, this could also be explained in the context of multiple timers as the amount of overlap influences both first and second estimations. This assumes a negative contribution of SOA on the second estimation: the shorter the SOA, the longer the overlap, and therefore the longer the estimated durations will be.

To assess the contribution of SOA on the second estimation, we constructed a similar model to the best fitting model described above, but in which SOA was removed. The fit of the reduced model is significantly worse ($\chi^2_{(1)}$=347, p<0.001), indicating a significant contribution of SOA in the prediction of the second estimate. However, the estimated effect of SOA ($\hat{\beta}$=.494, *t*(3335)=19.12, *p*<0.001) indicates that longer SOAs yield longer second estimations (see Figure 5). Again, this estimated effect is in the *opposite* direction as predicted by the multiple-timers account, but is in line with the single timer with temporal discounting account.

### Summary

As the estimate of the first duration contributes significantly to the estimated second duration, and the estimated SOA effects are both significant and in the same direction as predicted by the single timer account, the conclusion from this experiment is that it is more probable that humans have only a single timer. However, this single timer can be used relatively efficiently to estimate multiple intervals, although performance of the second estimation is negatively influenced by the logarithmic scale of the time generating mechanism.

An effect unexplained by the single timer account is the lower level of performance on the first estimation. According to the single timer account, the proportion of correct *first* estimations in the two-intervals phase should be similar to proportion of correct estimations in the training phase. As discussed above, the observed proportion was significantly lower during the two intervals phase than during training. This effect cannot be explained on the basis of timing alone. However, embedding a timing mechanism in a general cognitive architecture explains this effect elegantly, as we will discuss in the next section.

## Model

In Taatgen, Van Rijn and Anderson (in press), we have proposed a time mechanism that is embedded in the ACT-R general cognitive architecture (Anderson, 2007). We have shown that some of the phenomena traditionally described as pure timing phenomena, for example the effects of attention on cognitive timing, can better be explained as effects of the cognitive architecture or context on the task. The system contains a simple time generating system that consists of a single internal clock that generates logarithmically scaled pulses. These pulses can be read out by the cognitive architecture and stored in a declarative memory trace for later reuse. This approach explains, when embedded in an architecture that accounts for the retrieval of stored durations, both the basic findings associated with timing and more complex attention-demand related phenomena. Although ACT-R is a fairly complex theory, only a few components are crucial in understanding how the model can fit the experimental data presented here.

### Time estimation

The temporal module of ACT-R measure time in units that start at 100ms, but become gradually longer, creating a logarithmic representation of time, as illustrated in Figure 3 (see Taatgen, van Rijn & Anderson, in press, for details). This means that 2 seconds corresponds to 17 units or pulses in the temporal module, but 4 seconds only to 29 pulses instead of 34. Based on the initial single presentation of the intervals we assume participants have arrived at a reasonably stable internal representation of 2 seconds (17 pulses) at the start of the overlapping presentations. When the start signal for the first interval is given, the timer is started. After the SOA, the start signal for the second interval is given, prompting the model to store the value of the timer at that moment (in the examples 5 pulses for a 0.6 sec SOA and 13 pulses for a 1.5 sec SOA). When the timer reaches the 17 pulses, the value that corresponds to 2 seconds, the model will make the first response. It then adds the stored SOA value to 17, and waits until the timer reaches that value (i.e., either 17 + 5 = 22 or 17 + 13 = 30 pulses) to make the second response. As Figure 3 illustrates, the logarithmic scale introduces a bias in the second response that becomes larger with longer SOAs: The bias for the 0.6 sec SOA trial is 2.73 - 2 - .6 = .13 sec, for the 1.5 sec SOA trial, the bias is 4.06 - 2 - 1.5 = .56 sec.

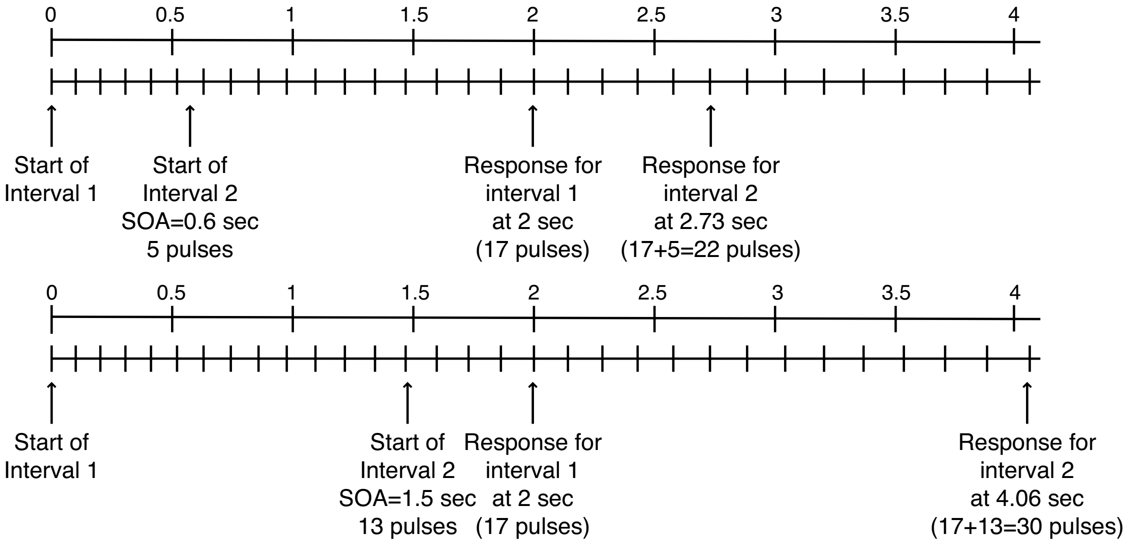

0
0.5
1
1.5
2
2.5
3
3.5
4
Start of Interval 1
Start of Interval 2 SOA=0.6 sec 5 pulses
Response for interval 1 at 2 sec (17 pulses)
Response for interval 2 at 2.73 sec (17+5=22 pulses)
Start of Interval 1
Start of Interval 2 SOA=1.5 sec 13 pulses
Response for interval 1 at 2 sec (17 pulses)
Response for interval 2 at 4.06 sec (17+13=30 pulses)

Figure 3. Illustration of the model for a short (0.6 sec) and a long (1.5 sec) SOA. The top line shows real time on a scale of seconds, while the lower line shows the subjective time in terms of logarithmically scaled pulses of the temporal module.

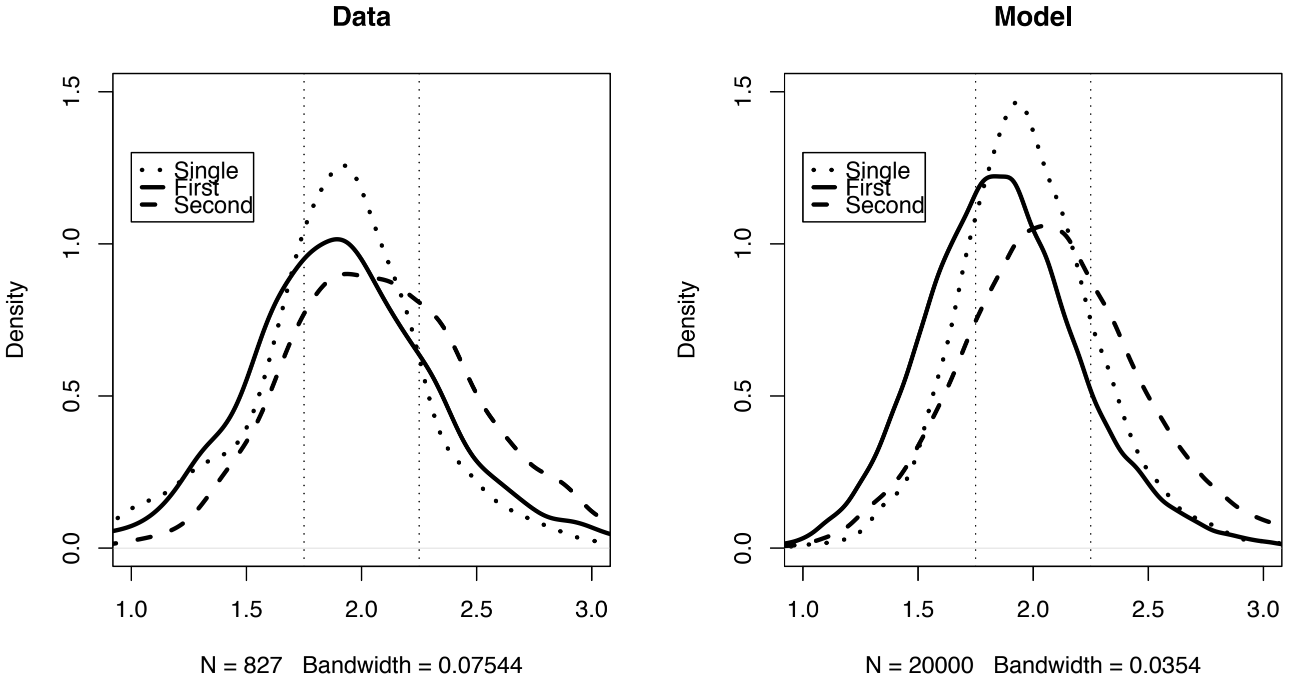


Figure 4. Distributions of estimations in the single task block, the first estimate in the dual task block and the second estimate in the dual task block. The vertical lines indicate the region of correct responses (1.75-2.25 sec).

## Representation of the time interval

The model maintains a representation of the time interval in its declarative memory. This representation is based on instance theory (Logan, 1988), which assumes that each experience creates an example in memory. Each time the model produces an interval and receives positive feedback, it will create an instance in declarative memory for that interval. If the model receives "too late" as feedback, it will not store the instance, and will bias its next response by subtracting one or two pulses (randomly with equal probability) from the next instance it retrieves The reverse is true if the model receives "too early" as feedback.

During the single task block, the instances will average around 17 pulses, which corresponds to 2 seconds. However, during the dual-task block its response on the second estimate will often be too late because of the addition of the perceived SOA, leading to feedback that prompts the model to shorten its representation of the interval. This shorting will continue until accuracy on both intervals is approximately equal and the too earlier responses for the first estimation cancel out modification based on the too late second responses. However, performance will never be completely stable, because the SOA introduces extra variability in the second estimate that the model cannot fully compensate for.

## Model results

Figure 4 shows the distributions of time estimates for the first block in which only a single estimate had to be made, and the first and second response in the second block. It shows that second responses are generally later than first responses, which is due to the bias of the logarithmic scale. It also makes evident that the distribution of the first response is pushed somewhat to the left, corresponding to around 16 pulses, compared to 17 pulses in the single task case to compensate for the second response. This explains the decrease in accuracy on the first interval. Figure 5 shows the accuracies for both the model and the data. Consistent with Figure 4, the model performs slightly better than the participants, but shows the same overall effects, including an almost identical accuracy on the first and the second estimate in the dual-task condition.

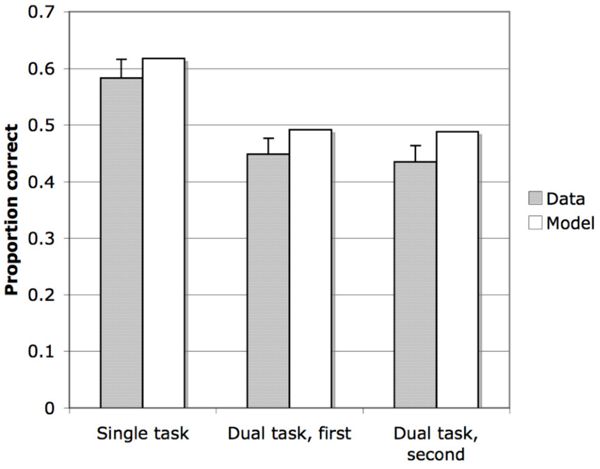


Figure 5. Comparison of accuracies for data and model

Figure 6 shows the effect of the SOA on the second response, and illustrates the effect of the logarithmic scale as the longer SOAs result in an overestimation of the second interval's duration.

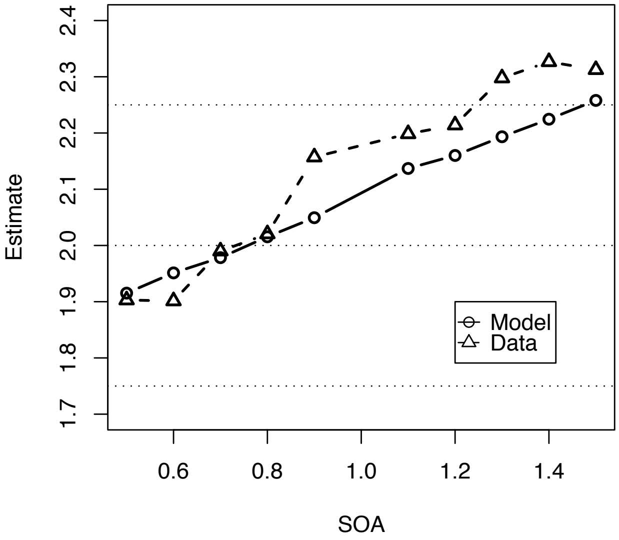


Figure 6. Effect of SOA on second estimate (model and empirical data)

## Discussion & Conclusion

Do multiple independent time generators drive human parallel time estimation, or do we strategically use the output of a single time mechanism for parallel time estimations? In this paper we presented an experiment that provides evidence for the latter account. Although all discussed analyses favor a single timer account, the most striking result is the positive relation between length of SOA and the estimated duration of the second interval. As the multiple-timers account predicts a negative effect, this positive relation can only be explained by strategically using the output of a single internal time generator.

However, the lower accuracy of the first estimate in the dual-timing phase compared to the single-timing phase cannot be explained by a theoretical analysis of the single-timer account. According to the single-timer account, performance of the first estimate in dual timing conditions should be equal to performance in single timing conditions, as processing the second estimation takes place after the first estimation is given. However, the presented computational model gives an elegant explanation of this effect. Because the two time intervals are estimated on the basis of a single main-estimation (resulting in the response for the first estimate), the feedback given for both estimations cannot be attributed to two different estimations. Thus, the model uses the feedback in a more general way, resulting in a shift towards earlier responses for both second and first estimations when the second estimation was too late (and vice versa). Because the logarithmic scale biases responses towards late responses, both estimations will shorten, yielding a lower accuracy for the first estimate in dual-time estimations than during single-time estimation.

An interesting aspect of the model is that it assumes that some form of temporal arithmetic is possible, at least for very simple but non-trivial additions. We are currently setting up a new experiment to test the implications of this assumption. Important to note is that, although the model internally represents the time in terms of a number of pulses, the value of this number is assumed to be meaningless with respect to inspectability: being able to time a certain interval correctly does not imply that one can state how many pulses are associated with that interval.

Concluding, we have, as in Taatgen, Van Rijn, and Anderson (in press), showed in this paper that adding temporal processing capacities to ACT-R facilitates more precise explanations of what is necessary to keep the time.

## Acknowledgments

This work was supported by Office of Naval Research grant N00014-06-1-005.

## References

Allan, L. G., & Gibbon, J. (1991). Human bisection at the geometric mean. *Learning and Motivation, 22*, 39-58.

Anderson, J. R. (2007). *How can the human mind occur in the physical universe?* New York: Oxford university press.

Block, R. A., & Zakay, D. (1997). Prospective and retrospective duration judgments: A meta-analytic review. *Psychonomic Bulletin & Review, 42*(2), 184-197.

Baayen, R.H., Davidson, D.J., & Bates, D.M. (submitted). Mixed-effects modeling with crossed random effects for subjects and items.

Bates, D. M. (2005). Fitting linear mixed models in R, R News, 5, 27–30.

Brown, S. W., & West, A. N. (1990). Multiple timing and the allocation of attention. *Acta Psychologica,* 75(2), 103-121.

Buhusi, C. V., & Meck, W. H. (2005). What makes us tick? Functional and neural mechanisms of interval timing. *Nature Reviews Neuroscience, 6*, 755-765.

Gibbon, J. (1977). Scalar expectancy theory and Weber's Law in animal timing. *Psychological Review, 84*, 279-325.

Ivry, R. B. and Richardson, T. C. (2002). Temporal control and coordination: The multiple timer model. *Brain and Cognition*, 48(1):117–132.

Meck, W. H. and Church, R. M. (1984). Simultaneous temporal processing. *Journal of Experimental Psychology: Animal Behavior Processes*, 10:1–29.

Penney, T. B., Gibbon, J., & Meck, W. H. (2000). Differential effects of auditory and visual signals on clock speed and temporal memory. *Journal of Experimental Psychology: Human Perception and Performance, 26*(6), 1770-1787.

Taatgen, N. A., Rijn, H. v., & Anderson, J. R. (in press). An Integrated Theory of Prospective Time Interval Estimation: The Role of Cognition, Attention and Learning. *Psychological Review.*

# A Control Perspective on Imaginal Perspective Taking

**Holger Schultheis (schulth@sfbtr8.uni-bremen.de)**
SFB/TR 8 Spatial Cognition, Universität Bremen, Enrique-Schmidt-Str. 5, 28359 Bremen, Germany

## Abstract

This contribution presents a computational cognitive model of imaginal perspective taking (IPT). The model is shown to account for major effects observed in IPT as well as concrete human data from an IPT experiment. Model development was based on the observation that control mechanisms play a central role in IPT performance. In taking this approach, model development and the model itself reveal similarities between tasks which so far have been considered only in isolation, suggesting a common basis for these tasks in terms of computational (control) mechanisms.

## Introduction

*Imaginal perspective taking* (*IPT*, see May, 2004) refers to the ability of humans to judge spatial relations between objects of a previously seen configuration without having sensory access to the configuration at the time of judgment. Although numerous studies (e.g., Hodgson & Waller, 2006; Mou, McNamara, Valiquette, & Rump, 2004; Sholl, 2001; Wang, 2004) have investigated human behavior during IPT, one important aspect of IPT, namely control, has received virtually no attention so far. Since, as shown by Schultheis (2007), control mechanisms can be assumed to be crucial for IPT, neglecting control seems to be a serious shortcoming. The aim of this contribution is to close this gap.

By analyzing available empirical data on IPT, we identified and implemented the control mechanisms involved in IPT. More precisely, we developed a computational cognitive model of IPT which is based solely on control mechanisms. This model is able to account for most of the variance in the human data for a typical IPT task. As a result, the developed model not only stresses the relevance of control to the understanding of IPT, in particular, and spatial cognition, in general, but also constitutes a first computational model for IPT. Moreover, the analysis yielding the employed control mechanisms reveals similarities between the mechanisms underlying (a) IPT and other spatial cognition tasks as well as (b) IPT and task switching. These similarities are considered a particular strength of the model, as they ground the model in independently motivated theoretical principles and at the same time render our model more parsimonious than any model explaining IPT in terms of IPT-specific mechanisms.

In the following we will first give a brief description of IPT, task switching, and the analogies between them (see also Schultheis, 2007). Subsequently, we will detail the theoretical rationale and control mechanisms underlying the developed model. This will be followed by a short exposition of how the model explains the main empirical effects and an application of the model to one particular IPT experiment. Finally, we will discuss related approaches before concluding with issues for future work.

## Imaginal Perspective Taking & Task Switching

### Imaginal Perspective Taking

In a typical IPT experiment, participants have to memorize a configuration of objects in a certain environment (most often a room). Then participants are blindfolded and often also deprived of any auditory input apart from the experimenter's instructions. In this state the participants are asked to point to objects of the learned configuration from (imaginal) perspectives which differ from the perspective defined by their bodily orientation and position. The to be taken perspective can differ from the bodily one only in orientation, called (imaginal) *rotation*, only in position, called (imaginal) *translation*, or in both. For example, in the case of an imaginal rotation the participant would be asked to indicate the direction to one object $x$ as if facing another object $y$. In imaginal translation the participant needs to indicate the direction to $x$ as if standing at $y$. The object to point to is often termed *target*. Usually measured are the time it takes participants to give the pointing response and the pointing error, that is, the deviation of the indicated and the actual direction to the target.

The following main effects on these two measures have been observed in IPT studies: Pointing from an imaginal perspective is more difficult (i.e., takes more time and leads to larger pointing errors) than pointing from the bodily perspective (e.g., Farrell & Robertson, 1998). Rotations are more difficult than translations (e.g., Sholl, 2001). The difficulty increases with increasing angular disparity between the pointing direction from the bodily perspective and the pointing direction from the imaginal perspective (May, 2004). Giving the participants time to prepare for the imaginal perspective before presenting the target reduces the difficulty, but pointing is still more difficult from the imaginal than from the bodily perspective (e.g., Sohn & Carlson, 2003).

### Task Switching

The ability of humans to change the task they are currently working on has been investigated intensely in the *task switching* paradigm (see Monsell, 2003). In this paradigm participants have to work on a succession of comparatively simple tasks such as adding or multiplying two digits. The defining characteristic of task switching studies is that the task the participants have to work on repeatedly and frequently changes during the experiment. Furthermore, in most of the task switching studies the stimuli and responses are *bivalent*, that is, the same for the different tasks. For example, in the experiment by Sohn and Anderson (2001) participants had to work on pairs of digits and letters with two possible tasks: Either judge whether the digit is odd or even or judge whether the letter is a consonant or vowel. The judgment had to be indicated for both tasks by either pressing the "z" or the "/" key on a computer keyboard. As in IPT, reaction time and errors are the main focus of analyses in task switching studies.

Generally, the following effects have been observed in task switching experiments (see Monsell, 2003, but also, e.g., Meiran, 2000; Sohn & Anderson, 2001): First, task execution is more difficult (i.e., slower and more error prone) just after a task switch. This decrement is called *switch cost*. Second, switch cost can be reduced if the participants are allowed to prepare for a change of task(s). Third, preparation for a switch does not eliminate switch cost completely. The cost remaining after preparation has been termed *residual cost*.

### Analogies

Several analogies hold between task switching and IPT. These analogies become apparent when assuming that different perspectives correspond to different tasks and the bodily perspective corresponds to the previously executed task. Framing IPT this way reveals that IPT—as task switching—uses bivalent stimuli, because targets (the objects to point to) and responses (pointing to the target) are the same for all perspectives. In addition, IPT exhibits a "switch cost" for taking a perspective different from the bodily one and preparation can reduce but not completely eliminate this switch cost.

This similarity of IPT to task switching suggests that a considerable part of IPT performance can be assumed to arise from the working of control mechanisms, since task switching is generally agreed to depend heavily on control facilities. Furthermore, the control involved in IPT might bear resemblance to the control involved in task switching.

Based on these considerations we developed a computational model of IPT in terms of control. This model will be detailed in the subsequent section.

## Computational Analysis and Model

### IPT as Task Switching

Despite considerable differences regarding the particulars (computational) models of task switching such as the models developed by Meiran (2000), Sohn and Anderson (2001), and Rubinstein, Meyer, and Evans (2001) generally assume that the observed task switching performance is the result of the combined effect of two distinct mechanisms. Furthermore, these models all suggest that one of the mechanisms is responsible for that part of the switch cost which can be eliminated during preparation, whereas the other mechanism is responsible for the residual cost. Although both mechanisms have been termed and implemented differently in the models, the gist of these terms and implementations is the same across models and associated with stimulus ambivalence and response selection. More precisely, the mechanism underlying the reducible cost seems to be related to stimulus disambiguation, that is, to determine how the stimulus is to be interpreted or which part of the stimulus to attend to. The idea is that a certain way of perceiving / interpreting the stimulus is associated with each task and the current mode of perception / interpretation is in accord with the previously executed task. In case of a task switch this mode has to be changed to conform to the new task. The mechanism underlying the residual cost, on the other hand, seem to be related to the influence the previously executed task has on the inclination to select certain responses. Thus, task switching can be viewed as consisting of two components, where the first is associated with stimulus disambiguation and the second is associated with response selection.

Based on the analogy of task switching and IPT a similar two component structure can be assumed to underlie performance in IPT. This is not to say that the processes of disambiguation and response selection involved in IPT are identical to those in typical task switching settings. For example, whereas switch costs in task switching are on the order of 50 - 100 ms, "switching" from the bodily to an imaginal perspective might take several seconds. Consequently, the components of disambiguation and selection in IPT seem to be realized differently than in task switching. How these two components might be conceived of in the scope of IPT and what the underlying mechanisms are, will be discussed in the next two sections.

**Stimulus Disambiguation: Reference Frame Selection** To be able to point to the target in the IPT task the symbolic target description given by the experimenter (e.g., "point to the phone") has to be re-represented as an egocentric pointing direction. Since this direction is dependent on the perspective taken, different perspectives afford different target representations and, thus, the stimuli employed in IPT are ambiguous. Moreover, disambiguation is a prerequisite for accurately performing on the IPT task. Without successful disambiguation the symbolic target description would in most cases be re-represented incorrectly leading to an incorrect pointing response. Consequently, stimulus disambiguation seems to be a crucial mechanism for successful IPT.

We propose that it is reference frame selection which underlies stimulus disambiguation in IPT. To see this, consider a typical IPT trial where a person has to point to a target from an imaginal perspective. The only source of information for determining this direction is the person's memory of the object configuration, because she has no sensory access to the configuration during IPT. In accord with a number of studies (Mou et al., 2004; Sholl, 2001; Waller, Montello, Richardson, & Hegarty, 2002), we assume that the enduring memory representation which is used during IPT consists of a network of nodes (i.e., object representations), where the links between the nodes represent directions between the objects represented by the nodes. To make use of such a representation, a reference frame is needed which defines a reference direction with respect to which the inter-object relations can be interpreted and experimental evidence (e.g., Hodgson & Waller, 2006) suggests that any such memory representation stores a certain reference direction in addition to the node network. Since the employed reference direction determines all inter-object relations, selecting a reference frame amounts to choosing a certain target direction interpretation, that is, by selecting a reference frame the stimulus can be disambiguated in IPT. Put in terms of the analogy of task switching and IPT, the reference frame is the mode of perception / interpretation which is associated with each task (i.e., each perspective). Accordingly, in a typical IPT trial the mode has to be switched / shifted from the reference frame of the bodily perspective (the previous task is always the bodily perspective; see above) to the reference frame of the imaginal perspective.

Viewing stimulus disambiguation as reference frame selection reveals similarities of IPT to other spatial cognition tasks. In particular, the use of spatial terms such as "above" or "right" has been shown to involve reference frame selection as one important step (Carlson, 1999). Based on this strand of research Schultheis (to appear) has developed a computational model of reference frame selection. This connectionist model consists of a number of units representing different reference frames or, more precisely, reference frame characteristics such as direction or orientation. Via bottom-up and top-down input connections these units are activated to a certain level. Subsequently, all activated units indirectly compete via shunting models until the mutual relation between the competing unit's activation reaches a certain threshold. This threshold is a free parameter of the model and determines how exclusively the reference frame characteristic represented by the unit with the highest activation is retained in the output of the model (see Schultheis, to appear, for model details).

Since IPT seems to rely on reference frame selection, we employed the just described model as the mechanism underlying stimulus disambiguation in IPT. For any IPT trial, stimulus disambiguation by reference frame selection takes place as follows: The reference frame of the bodily perspective and the reference frame of the imaginal perspective will activate certain units in the model. If the imaginal perspective is different from the bodily perspective they will activate different units which then compete until the criterion is reached. As an additional factor the reference frame stored with the node network (see above) will also activate one of the competing units sometimes facilitating and sometimes hampering the selection process. The number of iterations until competition terminates is assumed to be proportional to the time participants need to disambiguate the stimulus.

**Response Selection: Response Priming** The residual costs in task switching are generally thought to arise from priming. In all of the discussed models of task switching (Meiran, 2000; Rubinstein et al., 2001; Sohn & Anderson, 2001) it is assumed that executing a certain task in one trial will lead—in some form or the other—to the priming of stimulus response mappings relevant to this task. More precisely, all of the models assume that stimulus response mappings for tasks are held in working memory and primed to be more selectable on subsequent trials.

Again drawing on the analogy of task switching and IPT we propose that the second component of IPT is also arising from priming effects. However, due to the results of Wang (2004) it did not seem justified to adopt the priming mechanisms proposed in task switching models as they stand for IPT. Wang (2004) showed that the response effects in IPT seem to arise from rather low-level motor mechanisms. This suggests that (a) priming of working memory representations in the case of IPT is inappropriate and (b) not stimulus response mappings but only the motor responses are primed.

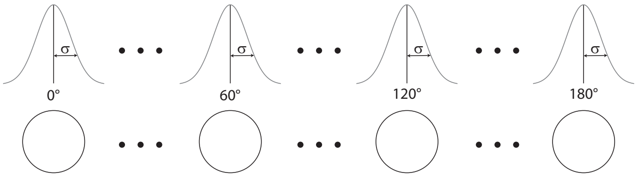


Figure 1: Array of motor units (four examples shown) with their tuning curves. The numbers between each curve and unit signify this unit's preferred direction in degrees.

Based on these assumptions we developed a new computational model. This model better corresponds to the experimental results of Wang (2004), while at the same time keeping with the idea of priming as the main mechanism. Our model consists of an array of units which are thought to represent cells in the motor cortex. These units are sensitive to certain directions, such that (a) each unit is maximally activated by one particular direction, the *preferred direction* and (b) activated to a lesser extent by similar directions. More precisely, each unit is thought to respond according to a Gaussian tuning curve which is centered at a cell's preferred direction. The variance of this tuning curve $\sigma$ is assumed to be a free parameter but the same for all units (see Figure 1).

If in the scope of IPT the pointing direction has been determined, this direction will activate the corresponding motor unit. Once any motor unit is activated to its maximum which is defined as the density of the Gaussian tuning curve at the preferred direction, the corresponding pointing response will be selected and executed. It is further assumed that the time it takes to activate any motor unit to its maximum (i.e., the time to select the corresponding pointing response) is proportional to the relation of a motor unit's preactivation and maximum activation. The lower the preactivation of a motor unit, the longer it will take to select and execute the corresponding response. It is by preactivation of the motor units that priming exerts its influence on the time necessary to perform IPT.

As in task switching, priming is thought to arise from the previously executed task. Since in IPT the previously executed task corresponds to pointing from the bodily perspective (see above), the pointing directions from the bodily perspective should somehow prime the motor units when pointing from an imaginal perspective. We propose that such priming stems from the working of a special representational system for space which has been termed *perceptual-motor system* (e.g., Sholl, 2001). This system continuously keeps track of the directions from a human to different objects in the environment, that is, on a motor level, the direction to different objects is continuously available. Thus, if the target in IPT is presented, this target can activate—on a motor level—the motor unit corresponding to the direction to this target. However, this unit will only be partially activated, because it is not the goal of the participant to point in this bodily defined direction. Furthermore, not only this motor unit, but also motor units with a similar preferred direction will be partially activated. More precisely, due to the tuning curves the activation of a motor unit will be proportional to the similarity of its own preferred direction and the bodily direction to the target.

To sum up, the two proposed components of our IPT model interact in the following way: After the to be taken perspective has been presented, stimulus disambiguation in the form of reference frame selection takes place. When selection is finished and the target has been presented (a) the target direction from the imaginal perspective is determined using the selected frame and the existing memory representation, (b) the corresponding motor unit will be activated, and (c) response priming will add to the activation of the relevant motor unit. Once any motor unit is activated maximally the corresponding pointing response is selected and executed.

To prove that this model constitutes a satisfactory account of IPT, we will both in the following section explain how the model accounts for the main IPT effects and in the section after next show the model's ability to simulate human IPT behavior by applying it to one concrete IPT experiment. In thus evaluating the model we will concentrate on reaction times, since these have been the focus of the presented analysis. However, as the section "Challenges" will explicate, the mechanisms in the model easily allow extending the modeled domain to the errors made by participants in IPT.

## Explained Effects

**Difficulty of IPT** According to the model, longer response time for pointing to a target from an imaginal perspective than for pointing from the bodily perspective has two causes. The first is stimulus disambiguation. If the target direction has to be judged from the bodily perspective, the bodily direction and the imaginal direction coincide and thus will activate the same unit in the reference frame selection mechanism. If the two directions do not coincide, they activate different units and, thus, both the desired direction will be activated less and an undesired direction will be activated more than in pointing from the bodily perspective. As a result, the initial difference between the unit activations will be less when bodily and imaginal direction do not coincide and therefore the competition will take more iterations (i.e., more time) to terminate. The second cause is response priming. As detailed above, motor units will be primed by the perceptual-motor system. This priming will be—due to the tuning curves—be strongest for that motor unit which has a preferred direction identical to the direction of the target from the bodily perspective. As a result it will take more time to activate the response from the imaginal perspective than from the bodily perspective adding to the slower response time in this condition.

**Difficulty of Rotations** One important aspect of imaginal translations is that the reference direction for the resulting imaginal perspective is the same as for the bodily perspective. Accordingly, during stimulus disambiguation the bodily and imaginal direction activate the same unit in the reference frame selection mechanism which results in lesser iterations until competition termination and, thus, imaginal translation are faster than imaginal rotations.

**Difficulty Increases with Disparity** This effect is owed mainly to the response priming mechanism. As already said, the response direction from the bodily perspective will be primed most. In particular, due to the tuning curves of the units in the response selection mechanism, any response unit will be so much more activated—by virtue of priming—the closer the unit's preferred direction is to the target direction from the bodily response. Thus, with increasing disparity between the bodily and imaginal target direction the priming activation for the imaginal target direction will decrease. Since lower activations entail more time to fully activate a unit (see above), response time will increase with increasing disparity.

**Difficulty Can Partly be Reduced by Preparation** The information processed during stimulus disambiguation in IPT is (a) the bodily orientation, (b) the direction stored in memory, and (c) the imaginal orientation. Since (a) and (b) are available anyway, stimulus disambiguation can start as soon as the imaginal orientation is known. Moreover, the availability of the necessary direction for disambiguation (namely the imaginal orientation) does not depend on the availability of the target. Given advance information on the imaginal perspective to be taken, identification of the reference direction can start and proceed prior to target presentation. The effect of response priming, on the other hand, can only take effect after the target is known. Without knowing the target the pointing direction cannot be computed and, consequently, the motor units cannot be activated. Thus, difficulty can partly (due to disambiguation), but not completely (due to response selection) be reduced by preparation.

## Model Application

In further evaluating the model we applied it to the data from experiment 3 of May (2004). This experiment seemed to be especially suited, because it is one of the few studies which systematically varied preparation time as well as angular disparity of pointing responses in both imaginal translations and imaginal rotations. In this experiment participants were allowed to learn and memorize an object configuration for 10 min. After learning they had to point to different targets from different imaginal perspectives while standing in the center of the objects. Participants had no sensory access to the configuration during pointing. The to be taken imaginal perspectives were either translations or rotations resulting in an angular disparity of 22.5°, 67.5°, 112.5°, or 157.5°between the target pointing directions from the bodily and the imaginal perspective. Furthermore, each IPT trial presented first the to be taken perspective and then, after a variable SOA of 1, 3, or 5 sec, the target. This design resulted in 24 different conditions for each of which pointing latency and pointing error were measured. As already said, we will concentrate on the latency data in modeling human behavior in this experiment. Accordingly, in modeling we will assume that both stimulus disambiguation and response selection will yield correct results. For stimulus disambiguation this means that in each trial the imaginal orientation will be selected as the reference direction, that is, in each trial the unit representing the imaginal orientation will initially be higher activated than any other of the competing reference direction units.

To model the data from May's experiment we employed the reference frame selection mechanism from Schultheis (to appear) nearly unchanged. In particular, we adopted the same maximum overall activation of the competing units, that is, the sum of the activation of all competing units was restricted to be not above 10. The only thing we changed was the gating criterion which was set to 10.

Given this setup, the model had three free parameters: First, the amount of activation initially received by the unit representing the imaginal reference frame direction. Second,

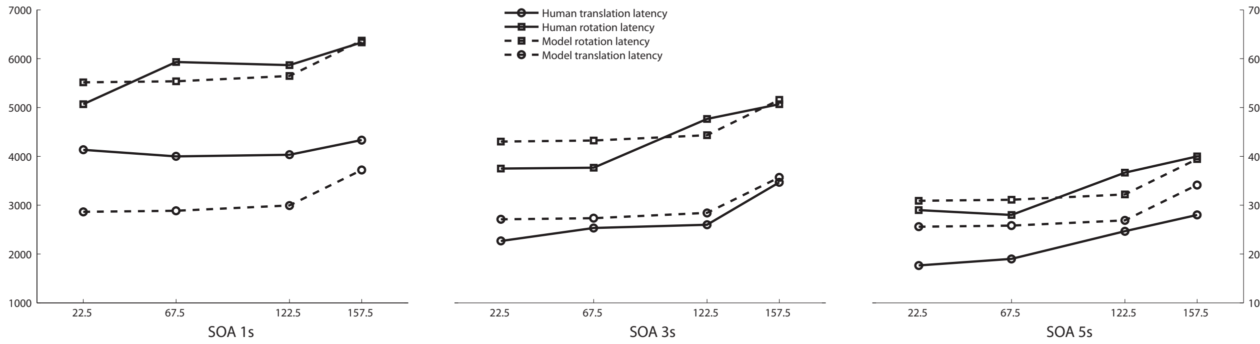


Figure 2: Empirical (May, 2004) and model latencies in ms. The x axis shows angular disparity in degree.

the amount of activation initially received by the unit representing the bodily reference frame direction. Due to the restriction of a maximal activation of 10, the initial activation for the reference frame direction stored with the memory representation was automatically determined by estimating the other two initial activations. Third, the variance of the tuning curves in the response selection mechanism.

The first two were estimated from the latency data of experiments 1 and 2 of Mou et al. (2004) and the third parameter was estimated from the to be modeled data of May (2004). We chose to estimate the first two parameters from a different data set, since this constituted a more rigorous test of the model: With only one parameter to fit 24 data points, a good model fit lends strong support to the validity of the employed mechanisms. The reason to utilize the particular data set of Mou et al. (2004) is that it is one of the few studies which systematically investigated the differential influence of the bodily reference direction and the reference direction stored in memory on IPT latencies. The initial imaginal, bodily, and memory reference frame direction activation was estimated to be 5.7, 3.182, and 1.118, respectively, resulting in a correlation of $r = 0.89$ between human and model data for experiments 1 and 2 of Mou et al. (2004). These parameter values indicate that the influence of the bodily direction is larger than the influence of the direction stored in memory.

Utilizing these activation values we estimated the variance of the tuning curves of the motor units from the data of May (2004). The variance was found to be 1.04. The modeling results using this variance and the corresponding human data for all 24 conditions of the experiment are displayed in Figure 2. The figure shows the reaction times for the different experimental conditions in milliseconds. Model reaction times have been determined by a linear regression of the human data on the raw model data (i.e., iterations resulting from the stimulus disambiguation mechanism plus the relation of the preactivation and the maximum activation of the motor unit; see above). As can be seen from the figure the model data corresponds quite nicely to the human data resulting in a correlation of $r = 0.89$. Note again that this fit of 24 data points was achieved by estimating only one parameter which indicates the appropriateness of the employed mechanisms.

## Challenges

Although the model can reasonably account for main behavioral effects found in IPT, there are several aspects of the model which require further considerations.

For example, modeling has so far concentrated on latencies disregarding the effects IPT has on pointing error. Yet, the mechanisms employed in the model seem to be suitable to also account for accuracy data. According to the model pointing error can arise either from stimulus disambiguation or response priming. Regarding the former, depending on the height of the gating criterion, the output of the selection mechanism need not correspond to one of the directions which have competed, but to a weighted combination of these. The lower the criterion the stronger the influence of all competing directions on the final output will be. In terms of IPT this means that with a low(er) criterion the selected direction will be the imaginal direction shifted towards the bodily direction. Since the pointing direction is based on the selected reference direction, a shifted reference direction will lead to an erroneous pointing direction. Regarding response priming, the pointing response based on the bodily direction will have activated other motor units representing a similar response. A response from the imaginal perspective will activate another motor unit and with it also units between the bodily and the imaginal pointing direction. Assuming noisy activation processes, one of these intermediate motor units may reach its maximum first and thus an erroneous response might be given by the participant. Both mechanisms predict errors to occur in the form of a shift of the imaginal response to the bodily response which is in accord with the observation of May (2004) that this kind of error is the most frequent one.

A second aspect is that the response selection mechanism in its current form, explains selection effects by positive priming. Thus, bodily based responses can influence IPT performance only by facilitating but not by hampering imaginal based responses. This seems to run counter to experimental results indicating that the bodily perspective can both facilitate and interfere with the imaginal perspective (Waller et al., 2002). On the basis of the presented analysis it is not clear whether the proposed mechanisms are able to account for the relevant empirical effects. Consequently, one major point of consideration in the further development of the model will have to be the clarification of this matter.

# Related Work

Other computational accounts related to human perspective taking have recently been proposed. Hiatt, Trafton, Harrison, and Schultz (2004), for instance, have developed a system which allows a robot to disambiguate a spatial utterance by taking the perspective of a human speaker. Although the task of the robot is similar to IPT, the developed computational account does not seem to be a cognitive model in the strong sense. Instead of trying to most accurately account for human behavior, the focus has been more on building a technical system which is able to conveniently interact with a human. In contrast, Gunzelmann and colleagues (see, e.g., Gunzelmann, Anderson, & Douglas, 2004) have developed detailed and accurate computational models of human behavior, but the task they used (and modeled) differed in important aspects from IPT. For instance, a map-like view of the object configuration was available to the participants during the whole task.

As a result, our model differs from these two approaches in both considering IPT and trying to accurately account for human behavior and, thus, our model constitutes a first computational model of human cognition in IPT.

## Conclusion

In this contribution we presented a computational model of human cognition and behavior in imaginal perspective taking. Model design was motivated and governed by a previously observed analogy of task switching and IPT. Not only did this analogy indicate the importance of control for IPT, but also suggested that stimulus disambiguation and response selection are two important components in IPT. Both components have been realized by separate mechanisms in the model. In particular, the mechanism used for stimulus disambiguation—namely reference frame selection—was not developed specifically for modeling IPT, but has been drawn from research on the use of spatial terms. The resulting model has been shown to be able to account for main effects observed in IPT as well as concrete human data from an IPT experiment.

Besides constituting a first computational account of IPT, the model points out similarities between IPT and task switching as well as IPT and spatial term use. As a result, one main contribution of the model is to highlight commonalities of seemingly different tasks, that is, revealing possible fundamental mechanisms governing human cognition, in general, and spatial cognition, in particular.

Future work will concentrate on extending and refining the proposed model with respect to, for example, error modeling and interference in response priming. Furthermore, we plan to explore the suitability of the reference frame selection mechanism for other spatial cognition tasks.

## Acknowledgments

In this paper work done in the project R1-[ImageSpace] of the Transregional Collaborative Research Center SFB/TR 8 Spatial Cognition is presented. Funding by the German Research Foundation (DFG) is gratefully acknowledged. We also thank the reviewers for their valuable suggestions.

## References

Carlson, L. A. (1999). Selecting a reference frame. *Spatial Cognition and Computation*, *1*(4), 365 - 379.

Farrell, M. J., & Robertson, I. H. (1998). Mental rotation and the automatic updating of body-centered spatial relationships. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *24*(1), 227 - 233.

Gunzelmann, G., Anderson, J. R., & Douglas, S. (2004). Orientation tasks with multiple views of space: strategies and performance. *Spatial Cognition and Computation*, *4*(3).

Hiatt, L., Trafton, J., Harrison, A., & Schultz, A. (2004). A cognitive model for spatial perspective taking. In M. Lovett, C. Schunn, C. Lebiere, & P. Munro (Eds.), *Proceedings of the 6th ICCM.* Mahwah, NJ: LEA.

Hodgson, E., & Waller, D. (2006). Lack of set size effects in spatial updating: Evidence for offline updating. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *32*(4), 854 - 866.

May, M. (2004). Imaginal perspective switches in remembered environments: transformation versus interference accounts. *Cognitive Psychology*, *48*, 163-206.

Meiran, N. (2000). Modeling cognitive control in task switching. *Psychological Research*, *63*, 234 - 249.

Monsell, S. (2003). Task switching. *TRENDS in Cognitive Sciences*, *7*(3), 134 - 140.

Mou, W., McNamara, T. P., Valiquette, C. M., & Rump, B. (2004). Allocentric and egocentric updating of spatial memories. *Journal of experimental psychology: Learning, Memory, and Cognition*, *30*(1), 142 - 157.

Rubinstein, J. S., Meyer, D. E., & Evans, J. E. (2001). Executive control of cognitive processes in task switching. *Journal of Experimental Psychology: Human Perception and Performance*, *27*(4), 763 - 797.

Schultheis, H. (2007). Advancing the understanding of spatial cognition by considering control. In *Proceedings of EuroCogSci'07: The European Cognitive Science Conference* (p. 746 - 751). Lawrence Erlbaum Associates.

Schultheis, H. (to appear). A computational model of control mechanisms in spatial term use. In *Proceedings of the 29th annual meeting of the Cognitive Science Society.*

Sholl, M. J. (2001). The role of a self-reference system in spatial navigation. In D. R. Montello (Ed.), *Proceedings of COSIT 2001.* Berlin: Springer.

Sohn, M.-H., & Anderson, J. R. (2001). Task preparation and task repetition: Two-component model of task switching. *Journal of Experimental Psychology: General*, *130*, 764–778.

Sohn, M.-H., & Carlson, R. A. (2003). Viewpoint alignment and response conflict during spatial judgement. *Psychonomic Bulletin & Review*, *10*(4), 907 - 916.

Waller, D., Montello, D. R., Richardson, A. E., & Hegarty, M. (2002). Orientation specificity and spatial updating of memories for layouts. *Journal of experimental psychology: Learning, Memory, and Cognition*, *28*(6), 1051 - 1063.

Wang, R. F. (2004). Action, verbal response and spatial reasoning. *Cognition*, *94*(2), 185 - 192.

# Towards Incorporating Visual Imagery into a Cognitive Architecture

**Scott D. Lathrop (slathrop@umich.edu)**
Computer Science and Engineering, 2260 Hayward Street
Ann Arbor, MI 48109-2121 USA

**John E. Laird (laird@umich.edu)**
Computer Science and Engineering, 2260 Hayward Street
Ann Arbor, MI 48109-2121 USA

## Abstract

This paper presents a synthesis of cognitive architecture and visual imagery. Visual imagery is a mental process that relies both on cognitive and perceptual mechanisms and is useful for tasks requiring visual-feature and visual-spatial reasoning. Using visual imagery as motivation, we have extended the Soar cognitive architecture to support the construction, transformation, generation, and inspection of visual representations for general problem solving. This paper presents the high-level architectural design and discusses initial results from two domains.

**Keywords:** Cognitive architecture; visual imagery; multi-representational reasoning.

## Introduction

Cognitive architecture research focuses primarily on abstract, symbolic representations and computations. Non-symbolic representations are used, but for control, and not for representing or manipulating task knowledge. There is, however, significant evidence that visual imagery plays an important role in many cognitive tasks (Kosslyn, et al., 2006; Barsalou, 1999). Our work seeks to investigate the synthesis of and interactions between cognition and mental imagery by extending the Soar cognitive architecture with visual imagery. In addition to Soar's native symbolic representation, visual imagery in our architecture uses a *depictive* representation as well as an intermediate, *quantitative* representation for images.

Our major result is a computational implementation of visual imagery and integration within a cognitive architecture. Functionally, this provides a computational advantage and additional capability for visual-feature and visual-spatial reasoning. Although our design is based on psychological and biological constraints, at this point, visual processing algorithms are ad hoc, and do not model the details of human performance. Our results illustrate the functional value of visual imagery and the challenges of creating complete models of such complex processes.

## Related Work

Two of the most prominent cognitive architectures, EPIC (Kieras & Meyer, 1997) and ACT-R (Anderson et al., 2004), incorporate models of human perceptual and motor systems. However, rather than specifying and implementing the low-level details of perception and motor processing, (e.g. edge detection, joint coordinates), these systems focus on the timing and resource constraints between perception, cognition, and motor processing. Moreover, neither system has a long-term perceptual memory, which is necessary to gain access to a remembered object's visual features (i.e. shape representation). Neither system has any mechanism to support visual imagery.

Previous efforts to build computational models of imagery have not included the constraints that arise in integration with a general cognitive architecture. Kosslyn composed a detailed mental imagery model and created a computational implementation to simulate and test his ideas (1980). Glasgow and her colleagues built a computational model of imagery for a molecular scene analysis application (Glasgow & Papadias, 1992). While Glasgow incorporated psychological constraints in her model, such as the inclusion of three separate representations (descriptive, spatial, and visual), their implementation is application specific.

The CaMeRa model of Tabachneck-Schijf's et al. (1997) uses multiple representations and simulates the cognitive and visual perceptual processes of an economics expert teaching the laws of supply and demand. Their system includes both visual short-term and long-term memories that complement verbal memories, but the generality of the overall architecture is unclear. Visual STM includes a quantitative (node-link structure) and a depictive (bitmap) representation that is similar in design, although not in implementation, to our representations. Their shape representation is limited to algebraic (i.e. lines and curves) shapes and their spatial structure only models an object's location while ignoring orientation and size.

Barkowsky (in press) proposes that any model of mental imagery must include the following:

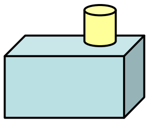
(1) Hybrid representational formats to include propositional and visual structures involving shape.
(2) Coupling between imagery and visual perception.
(3) Construction of images from pieces of knowledge.
(4) Processing with or without external stimuli.
(5) Multi-directional distributed processing and control.

Our architecture addresses (1) – (3) and our future plans include incorporating visual imagery processing in the presence of perceptual stimulus (4). Our control structure initiates and controls imagery processes in a top-down manner while perceptual mechanisms process results in a bottom-up fashion. In Soar, the contents of working memory determine which memories and processes are active without any centralized control (5). We also propose that the architecture must support transformation and generation of a depictive representation. The following sections discuss our initial implementation.

## Visual Representations

We assume visual imagery uses three distinct visual representations to include (1) an abstract symbolic representation, (2) a hybrid symbolic and quantitative representation, and (3) a depictive representation (Table 1). Each visual representation becomes more specific and committal as you move down the hierarchy.

Table 1: Visual Representations

| Representation | Uses | Example |
|---|---|---|
| Abstract symbols | General, qualitative visual-feature and visual-spatial reasoning | object (can)<br>object (box)<br>color (can, yellow)<br>color (box, blue)<br>on (can, box) |
| Hybrid abstract and quantitative symbols | Quantitative visual-spatial reasoning | can<br>height 5<br>radius 1<br>location <2,1,2><br>box<br>length 10<br>width 6<br>height 4<br>location <0,0,0> |
| Depictive symbols | Visual-feature recognition<br>Quantitative visual-spatial reasoning |  |

The abstract symbolic visual representation is the neutral, stable medium useful for general reasoning (Newell, 1990). Symbols denote an object, some visual properties of that object, and qualitative spatial relationships between objects. The meaning of the symbols is dependent on their context and interpretation rather than how the symbols are spatially arranged. The symbols are composable using universal and existential quantification, conjunction, disjunction, negation, and other predicate symbols.

The hybrid, intermediate representation labels objects with abstract symbols and denotes each object's location, orientation, and size with quantitative, vector-based values. The computational processes that infer information from this representation are sentential, algebraic equations.

The intermediate representation does not receive much attention in the imagery representational debate (Kosslyn, et al., 2006; Pylyshyn, 2002). However, it is important for the following reasons. First, neurological evidence shows that during visual-spatial imagery tasks, the visual cortex, or depictive representation, is not active (Mellet et al., 2000). However, the parietal cortex is active signifying a visual format distinct from the depictive representation.

Second, Marr stresses that bottom-up visual processing uses incremental, increasingly abstract levels of representations (Marr, 1982). This rational is also pertinent to visual imagery but in the "opposite" direction. Visual imagery cannot generate a depictive representation directly from qualitative, abstract symbols without first specifying metric properties, such as location, orientation, and size. Finally, from a computational perspective, there are some spatial reasoning tasks where reverting from qualitative symbolic representations to quantitative information is necessary for either efficiency or simply to infer new information (Forbus, Neilsen, & Faltings*, 1991).*

The depictive representation is useful for detecting object features (e.g. "does the letter 'A' have an enclosed space?") and spatial properties where the objects' topographical structure is relevant (e.g. "which is wider in the center, Michigan's lower peninsula or the state of Ohio?"). Space implies spatial extent within and between objects in a visual scene. Each point in the representation can have variable color and intensity, and the spatial arrangement of the points resembles the object(s) specific shape. Computationally, the depiction is a pixel-based data structure and the algorithmic processes are either algebraic or ordinal algorithms that take advantage of the topological structure.

## Architecture

There are two software components in our architecture, (1) Soar and (2) Soar Visual Imagery (SVI). Soar provides the underlying control (via its procedural production memory and its decision procedure) and state representation (via its symbolic memories). SVI encompasses both visual perception and visual imagery mechanisms. Figure 1 shows the architecture with Soar (not to scale) across the top and the visual mechanisms inherent to SVI underneath. We will refer to this figure as we explain the architecture and elaborate on the specific visual imagery processes not shown in it. The architecture makes a distinction between memories (rectangles) and processes (rounded rectangles). The terminology is either Kosslyn's et al. (2006) or our own. We will start by explaining the memories and processes associated with visual perception working from the bottom to the top of Figure 1. Then we will discuss visual imagery from a top-down perspective.

### Visual Perception

The *Visual Buffer* is the SVI short-term memory associated with the visual cortex. It maintains the depictive representation (Kosslyn, et al., 2006). A *Refresher* process activates the depiction based on information received from visual perception. Two sets of processes in SVI correspond to the ventral or "what" pathway and the dorsal or "where" pathway that extend from the visual cortex (Ungerleider & Mishkin, 1982). The *"What" Inspectors* are responsible for extracting object features, shape, and color from the Visual Buffer. They store each object's shape and color in a *Visual long-term memory* (LTM), neurologically believed to be in the region of the inferior temporal lobe. SVI stores the shapes as a mesh topology in the Euclidean space, $R^3$.

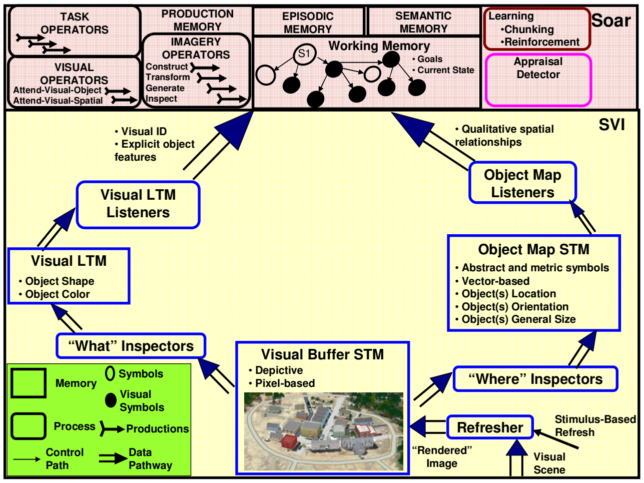


Figure 1: Architecture overview

The *"Where" inspectors* extract the location, orientation, and size of the objects in the Visual Buffer and store this information in the *Object Map* short-term memory. The Object Map roughly corresponds with the posterior parietal cortex and maintains the quantitative visual representation from Table 1. SVI implements this representation with a scene-graph data structure.

The *VisualLTMListeners* and the *ObjectMapListeners* consolidate the inspectors' results and create an abstract symbolic format for Soar's working memory. The Visual LTM Listeners provide an object's qualitative features along with a symbol (*visual-id)* denoting the object's shape and color in Visual LTM. Likewise, the Object Map Listeners create the qualitative spatial relationships between objects in the Object Map. Visual operators in Soar's production memory attend to the listeners input and associate it with existing knowledge.

## Visual Imagery

For illustration, consider a Soar robot setting the table for dinner. Its current goal is to set one place setting, and in order to accomplish the goal it has to set each individual object (napkin, fork, plate, etc). It prefers to set the center object (i.e. plate) first so it can place the other objects relative to the center. The robot's working memory contains the symbolic representation of the place setting (Figure 2).

Each object's symbol structure is associated with the current state in Soar's working memory via a *visual-object* attribute. The place setting structure includes the primitive visual objects napkin, fork, plate, knife (not shown), and spoon (not shown) objects. Primitive visual objects have a visual-id attribute. Composite visual objects (i.e. place setting) denote an object containing other visual objects. Composites are augmented with *has-a* and *spatial-relationship* attributes defining how the object is composed. Spatial relationships indicate an object's location and topology in relation to other objects. For example, the fork is above (location) and connected (topology) to the napkin and left-of and disconnected from the plate. A viewpoint attribute specifies the spatial relationship perspective. Note that primitive objects may be associated with many composite objects and task knowledge may rearrange the spatial relationships or even synthesize composite objects to enable the creation of novel visual images.

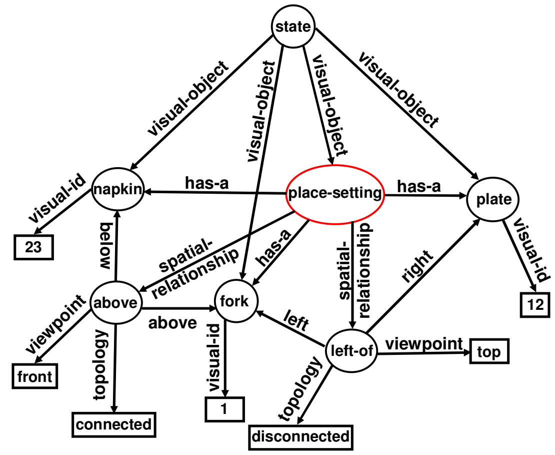


Figure 2: Soar working memory visual representation

Although the symbol structure in Figure 2 encapsulates a lot of information, it does not indicate the place setting's center object—either directly or through inference. When there is a lack of visual-feature or visual-spatial knowledge, an impasse occurs and Soar creates a special, visual imagery state. The state's initial knowledge consists of the symbolic representation of the object in question and the goal is to determine the desired information.

As a first step, visual imagery, processing has to re-encode Soar's symbolic representation into the intermediate, quantitative representation. To support this, general-purpose operators for *constructing* the metric representation (Figure 1) are encoded in Soar's production memory. Construction derives from a commonly demonstrated phenomenon in behavioral imagery experiments showing the time to generate a visual image is linearly dependent on the number of parts in the visual representation (Kosslyn, et al., 2006).

Within SVI, there are functional processes specific to imagery. The *Imager* receives the operator's command and symbolic information from Soar, interprets it, and passes the required information to a *Constructor* process (Figure 3). The Constructor builds the quantitative representation in the Object Map by combining each object's general shape information from Visual LTM with its qualitative spatial knowledge from Soar's working memory. For example, to build the place setting, visual imagery may first compose the fork and the plate by locating the fork to the left of the plate. In a similar fashion, processing adds the other objects to complete the quantitative representation.

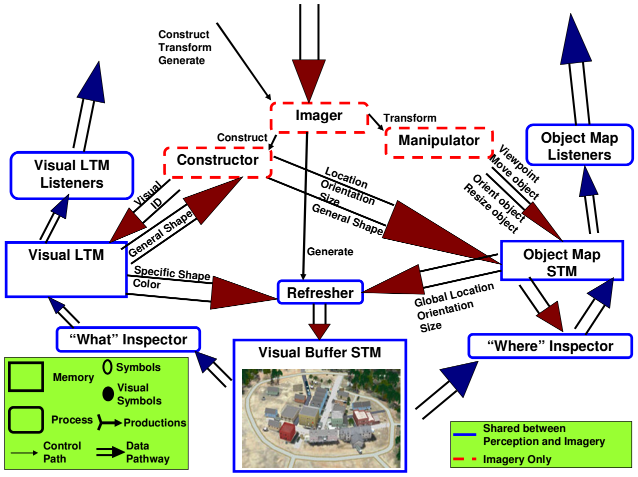


Figure 3: Construction, transformation, generation

The *transformation* operator (Figure 1) and the *Manipulator* process (Figure 3) emerge from another common behavioral phenomenon, made famous by Shepard's and Metzler's "mental rotation" experiment (1971). The operator changes the location, orientation, or size of a specific object or the perspective of the scene.

If the original query refers to an object's spatial orientation or relative size then the metric representation is sufficient. In the case of inferring the place setting's center object, this is the case. However, if the robot finishes setting the plate and is ready to pick up the napkin, it may want to know the relative difference in width between the plate and napkin. In this case, a depictive representation with each objects' specific shape is required. The *generation* operator initiates processing, and the Imager interprets the command and invokes the Refresher (Figure 3). The Refresher combines each object's specific shape and color from Visual LTM with the Object Map information and generates the depictive representation in the Visual Buffer.

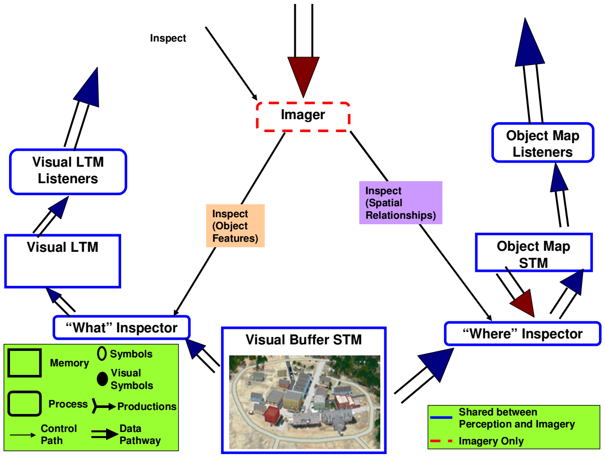


Figure 4: Inspection

After the system has constructed, transformed, and, if necessary, generated the depictive representation, the conditions are set for the inspection process (Figure 4). The *inspect* operator provides the Imager with the query. For example, "what is the center object of the scene?" or "which object is wider?" The Imager then activates the "What" and/or "Where" processes. These processes function as previously discussed with the exception that in visual imagery the Imager may direct the "where" inspectors to focus on the Object Map if the depiction is not required. The agent may iteratively add more detail to its visual representation and inspect it to refine its search.

## Results

The results illustrate the functional and computational value of visual imagery in two distinct domains. The first domain derives from Larkin & Simon's work demonstrating the computational advantage of diagrams (1987). In the problem they investigate (Figure 5), the model must locate object features, (e.g. vertices, line segments, triangles) and infer relationships (e.g. angles, congruency) that initial task knowledge does not specify.

Although we doubt a human could solve the problem without an external diagram, we chose this task because it stresses the construction and inspection of a quantitative representation. The task does not require a depiction as initial knowledge specifies the main feature (lines) from which other features can be inferred. As either symbolic or metric representations are sufficient, we can compare them and determine computational and functional differences.

The second domain derives from Kosslyn and Thompson (2007). In this experiment, the subjects hear a letter from the English alphabet and the experimenters ask them to visualize it in its uppercase format. Next, the subjects hear a cue, such as "curve", "enclosed-space", or "symmetry" and indicate whether the letter has the particular feature. For example, the letter 'A' has an enclosed space and vertical symmetry while 'U' has a curve. The Soar model also "hears" a question, visualizes the letter, searches for the desired feature, and then "verbally" responds

We chose this visual-feature task because it involves all imagery processes and representations. Unlike the geometry domain, symbolic or quantitative representations cannot solve this task without explicitly encoding every feature. The task also includes an external environment that emphasizes the interaction of visual imagery and cognition.

Although our initial goals are functionality driven, we also make comparisons with human data and discuss the shortcomings. Two reasons for these shortcomings include our uncertainty of the types of algorithms humans use to recognize features, and our architecture's lack of "image maintenance" that occurs when the image's vividness decays and must be refreshed (Kosslyn et al. 2006).

### Geometry Problem

The problem states that there are four lines (A, B, C, D). Line A is parallel to line B and line C intersects line A. Line D bisects the line segment formed by the intersection of line C with lines A and B (Figure 5). The goal is to show that the two triangles formed are congruent. To prove congruency, the model must employ a basic geometry rule, such as the angle-side-angle (ASA) rule. The ASA rule states if two angles and the included side of a triangle are congruent to two angles and the included side of another triangle, then the two triangles are congruent. In Figure 5, the model must show E1=E2, e1=e2, and c=b.

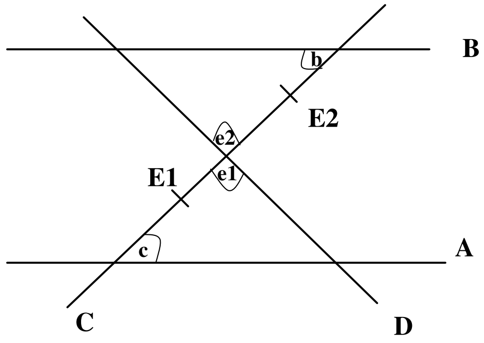


Figure 5: Geometry Problem

We compared two models. The first uses only symbolic representations (Soar Only) and has operators to create and process geometric objects and relationships. For example, "if two lines intersect, then create a vertex". The model creates these features until it can show the triangles are congruent. The second model (SVI) constructs a metric representation from the original description. It then inspects it for the desired features and relationships and uses the information along with the ASA rule to prove congruency.

The SVI model requires less real and simulated time (Figure 6). "Soar Only" spent much of its time considering objects and relationships that were not required to solve the problem. The SVI model also requires less task knowledge (Figure 7). The "Soar Only" model requires knowledge about geometric structures inherent to SVI's imagery operations. Functionally, this suggests that SVI decreases the amount of knowledge required to learn such a task.

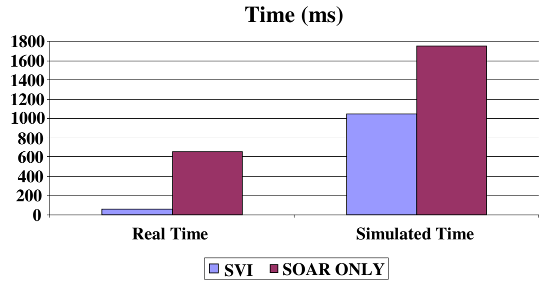


Figure 6: Time for each agent.
Simulated time is decision cycles x 50 ms.

The model is not psychologically plausible because of its unrealistic ability to maintain arbitrary amounts of information in its visual buffer. We expect humans would require an external diagram and thus require more time to solve the problem. However, the task demonstrates imagery's computational advantages and added capability.

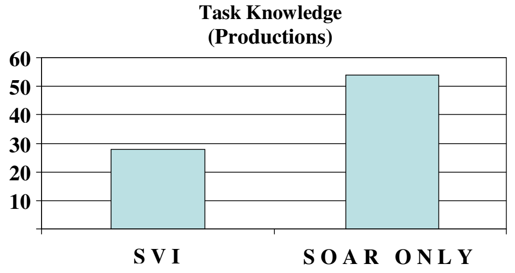


Figure 7: Initial task knowledge for each agent

## Alphabet Experiment

Our evaluation for this experiment focuses on three areas. First, the requirement for generating and transforming depictive representations forced us to reconsider the design. Our previous discussion reflects this evaluation. Second, we make a subjective comparison between the feature detection algorithms and note that even though the representation is depictive, the processing may not. For example, to detect curves we use a variation of the Hough transform (Mat Jafri & Deravi 1994). The algorithm maps edge pixels onto a parameter space and uses a "voting" algorithm to determine the parameters that indicate a curve. Although the algorithm has interesting perceptual characteristics in that it is parallelizable, it uses sentential, algebraic computations. For detecting enclosed spaces, we employ an algorithm using pixel rewrites to take advantage of the topological space and locality of neighboring pixels that is clearly more "depictive" (Furnas, et al., 2000).

Finally, we compare the model's response time[1] (RT) with human data from Kosslyn's experiment (2007). Figures 8–9 show the comparison with the letters along the x-axis sorted from left to right according to human response time. Both humans and Soar show variability in the time to detect enclosed spaces[2], but the average time is almost identical (Figure 8). In the case of symmetry, however, Soar shows little variability while humans show a lot (Figure 9).

Again, we make no claim that the algorithms are similar to how humans recognize these features. Since the architecture does not incorporate image maintenance, the time required to recognize symmetry dominates the results. Our algorithm determines symmetry by transforming the original depiction around the axis of symmetry and comparing it with the original orientation. Rather than performing this operation in a single step, we hypothesize that humans must continuously rotate and regenerate the letter. This demonstrates that even if the overall architecture

[1] Based on average CPU time over 30 trials and scaled for comparison with human data.

[2] Curves and enclosed spaces show a similar graph with the exception that the range of response times were spread out more for both the human (~600ms) and Soar agent (~500ms) data.

is correct (our hypothesis), the devil of modeling human behavior is in the details of low-level visual processing.

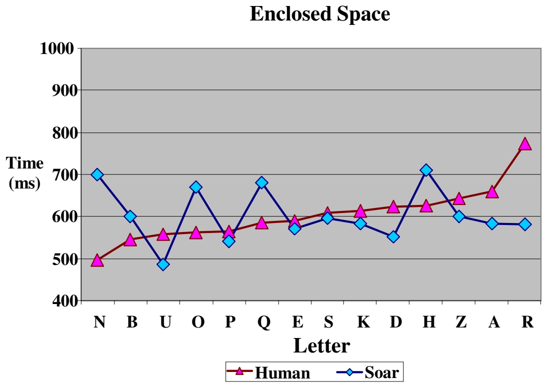


Figure 8: Enclosed space response time comparison Human $\mu: 604, \sigma: 65$, Soar $\mu: 595, \sigma: 14$

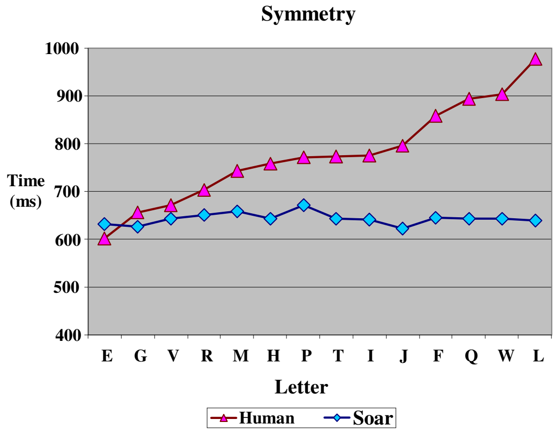


Figure 9: Symmetry response time comparison Human $\mu: 778, \sigma: 104$ Soar $\mu: 643, \sigma: 12$

## Conclusion

We have demonstrated that it is possible to extend a general cognitive architecture with a comprehensive model of imagery that includes using multiple visual representations; sharing mechanisms with vision; and incorporating construction, transformation, generation, and inspection. It also expands architectures by linking perceptual-based thought and cognition. This union provides new capabilities and computational efficiency for visual-feature and visual-spatial reasoning. As we move forward, we desire to expand the inspection processes and evaluate the architecture in an environment where perception and imagery interact, spatial and depictive forms of imagery are necessary, and the overall task is not to answer a question but involves making decisions and executing them in a rich environment.

## Acknowledgments

The authors would like to thank Stephen Kosslyn and William Thompson for the human data and collaboration.

## References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An Integrated Theory of the Mind. *Psychological Review, 111*(4), 1036-1060.

Barkowsky, T. (in press). Modeling mental spatial knowledge processing: An AI perspective. In: F. Mast and L. Jaenke (Eds.), Spatial processing in navigation, imagery, and perception. Berlin: Springer.

Barsalou, L. W. (1999). Perceptual symbol systems. *Behavioral and Brain Sciences, 22*, 577-660.

Forbus, K. D., Neilsen, P., & Faltings, B. (1991). Qualitative spatial reasoning: the clock project. *Artificial Intelligence, 51*(1-3), 417–471.

Furnas, G., Qu, Y., Shrivastava, S., & Peters, G. (2000). *The Use of Intermediate Graphical Constructions in Problem Solving with Dynamic, Pixel-Level Diagrams* (Vol. 1889).

Glasgow, J., & Papadias, D. (1992). Computational Imagery. *Cognitive Science, 16*, 355-394.

Kieras, D. E., & Meyer, D. E. (1997). An Overview of the EPIC Architecture for Cognition and Performance with Application to Human-Computer Interaction. *Human-Computer Interaction, 12*, 391-483.

Kosslyn, S. M. (1980). *Image and Mind*. Cambridge: Harvard University Press.

Kosslyn, S. M., & Thompson, W. L. (2007). *Can people "see" implicit properties as easily in imagery and perception? (In preparation).*Unpublished manuscript.

Kosslyn, S. M., Thompson, W. L., & Ganis, G. (2006). *The Case for Mental Imagery*. New York, New York: Oxford University Press.

Larkin, J. H., & Simon, H. A. (1987). Why a Diagram is (Sometimes) Worth Ten Thousand Words. *Cognitive Science, 11*, 65-99.

Marr, D. (1982). *Vision*. San Francisco: Freeman.

Mat Jafri, M. Z., & Deravi, F. (1994, 2 November 1994). *Efficient algorithm for the detection of parabolic curves.* Paper presented at the Proceedings of SPIE - Vision Geometry III, Boston, MA, USA

Mellet, E., Bricogne, S., Tzourio-Mazoyer, N., Ghaem, O., Petit, L., Zago, L., et al. (2000). Neural Correlates of Topographic Mental Exploration: The Impact of Route versus Survey Perspective Learning. *NeuroImage, 12*, 588-600.

Newell, A. (1990). *Unified Theories of Cognition*. Cambridge, Massachusetts: Harvard University Press.

Pylyshyn, Z. (2002). Mental Imagery: In search of a theory. *Behavioral and Brain Sciences, 25*, 157-238.

Shepard, R. N., & Metzler, J. (1971). Mental rotation of three-dimensional objects. *Science, 171*, 701-703.

Tabachneck-Schijf, H. J. M., Leonardo, A. M., & Simon, H. A. (1997). CaMeRa: A computational model of multiple representations. *Cognitive Science*, 21(3), 305-350.

Ungerleider, L. G., & Mishkin, M. (1982). Two cortical visual systems. In D. J. Ingle, G. M.A. & R. J. W. Mansfield (Eds.), *Analysis of visual behavior* (pp. 549-586). Cambridge, MA: MIT Press.

# Evaluating the Performance of Optimizing Constraint Satisfaction Techniques for Cognitive Constraint Modeling

**Alina Chu (achu@umich.edu)**
Department of Computer Science and Engineering; University of Michigan
Ann Arbor, MI 48109

**Richard L. Lewis (rickl@umich.edu)**
Department of Psychology; University of Michigan
Ann Arbor, MI 48109

**Andrew Howes (howesa@manchester.ac.uk)**
Manchester Business School; University of Manchester
Manchester, UK.

## Abstract

Cognitive Constraint Modeling is an emerging modeling framework that allows a modeler to derive predictions of asymptotic performance from (a) a specification of architectural constraints; (b) a specification of a space of possible task strategies; and (c) an explicit objective (payoff) function. This approach has distinct advantages over traditional approaches in that it reduces the degrees of freedom inherent in specifying the details of particular strategies when explaining behavior; instead these specific strategies are selected on the basis of their maximum payoff given the architectural constraints. However, this approach potentially suffers from high computational cost and intractability on some problems because it is grounded in optimizing constraint-satisfaction techniques. To understand and address this problem, we have built a cognitive constraint problem generator which stochastically generates large populations of parametrically controlled problem instances on which we can test CCM performance. Initial results from our use of this tool have identified at least one clear property of problems-*order strength*-that has a significant and non-linear effect on performance time. These results, when extended, will pave the way to efficient constraint modeling.

## Introduction

It is a commonplace of psychology that human behavior is adaptive, but research over the last 10-15 years has revealed the astonishing degree to which strategic adaptation and variability manifests itself across behavioral timescales-even at the lowest levels of extremely rapid behavior (e.g., [Meyer and Kieras, 1997]). Cognitive architectures naturally admit of such strategic variation because they are programmable. Unfortunately, this general theoretical virtue becomes explanatory vice in trying to explain specific behaviors, because the strategies and knowledge supplied to the architecture become degrees of freedom in accounting for data [Kieras and Meyer, 1999, Newell, 1990]. This leads to the following question: How can we account for the complex details of adaptive interactive behavior-without effectively building into the model what we are trying to explain?

*Cognitive Constraint Modeling* (CCM) is an emerging framework for modeling that attempts to address this issue by allowing modelers to work in terms of formally defined *spaces* of strategies and formally defined architectural theories [Howes et al., 2004]. This is an advantage in that points in the strategy space are then evaluated in terms of explicit theorist-defined objective functions (such as “go as fast as possible” or “minimize working memory load”) rather than being selected because of data fit.

The approach is realized computationally in a system called *CORE* (Constraint-based Optimality Reasoning Engine) [Howes et al., 2004]. Computational modeling in CORE has several distinguishing characteristics, but we focus on two here. First, descriptions of behavior are derived via constraint satisfaction over explicitly declared architectural, task, and strategy constraints, rather than running a simulation. Second, the complex details of behavioral control emerge in part from optimizing behavior with respect to explicit payoff functions ascribed to the human. Predictions of asymptotic performance are generated from architectural theory while minimizing specific assumptions about the details of the strategies. CORE has been used to model a range of different domains, including driving behavior [Brumby and Salvucci, 2006], a call center [Howes et al., 2005], interactions with an Automated Teller Machine task [Lewis et al., 2004], and interruptions during procedural cockpit tasks [Eng et al., 2006].

### The computational problem

Generating predictions from CORE can be computationally expensive: each CORE model constitutes a constraint satisfaction problem defining a potentially large search space of possible behaviors. Our practical experience with CORE suggests that even with relatively short stretches of behavior (under 20 seconds), it may be possible to create modeling problems that takes hours or days of CPU time to compute, as well as problems that are essentially intractable.

We do not believe this is a problem specific to our particular implementation of CORE. Rather, the computational complexity of these models is a fundamental problem that must be addressed for the general class of modeling methods in which the modeler is effectively specifying and exploring large spaces of possible strategies rather than specific (and possibly arbitrary) points in an implicit strategy space. This paper reports our first steps at understanding and addressing this computational problem.[1]

[1] This is not the first explicit attempt to systematically and quantitatively analyze the computational performance of a symbolic cognitive modeling technique; many current

## Our approach

Our approach to understanding and addressing this computational problem is to develop a *cognitive constraint model problem generator*, which allows us to stochastically generate large populations of parametrically controlled problem instances on which we can test the performance of CORE. Initial results from our use of this tool have identified at least one clear property of problems—*order strength*—that has a significant and non-linear effect on performance time. As we discuss below, this property has been previously identified in the literature on constraint satisfaction techniques.

In the remainder of the paper, we first summarize the basic structure of the computational problem as it arises in CORE, showing how CORE cognitive models are reduced to resource-bounded constraint satisfaction problems (RCSPs). We then briefly describe a particular class of algorithms for solving RCSPs and the time complexity of RCSPs. Next, we provide an overview of our problem generator framework, and then report on our initial results of experiments with this system. We conclude with the first set of general lessons we draw from these experiments, and suggest practical steps to take to make constraint modeling more efficient.

## From cognitive constraint models to scheduling problems

CORE takes in a specification of constraints and outputs a behavioral prediction. The architectural constraints are described in terms of cognitive, perceptual, and motor processes which use a set of processors/resources with set capabilities (following Model Human Processor [Card et al., 1983] and CPM-GOMS [John and Gray, 1995]) and high-level task grammars that define a space of possible task strategies. CORE transforms the architectural constraints and the task grammar specification into a constraint satisfaction problem, described as a set of variables, domains, and constraints. This formulation allows for the treatment of the models as scheduling problems, in particular, a Resource-Constrained Scheduling Problem (RCSP). CORE uses an off-the-shelf constraint solver: the Constraint Logic Programming over Finite Domains (CLPFD) Sicstus Prolog solver, to solve these constraint satisfaction problems. This solver utilizes the standard Branch and Bound algorithm for optimized solution search.

### Formulation as Resource-Constrained Scheduling Problem

A RCSP consists of a number of different activities, where each of the activities has a duration, and a maximum time domain in which it must be scheduled. Each activity also has a resource requirement which it uses during its processing time. The activities must be scheduled within the total time bound (or optimized with respect to time), as well as stay within the total resource capacity at any given time. A solution schedule is an assignment of activities such that all specifications described are satisfied. There may be more than one solution per problem, depending on the tightness of the specifications.

A CORE cognitive constraint model can be interpreted as an RCSP. The variables consist of the processes, the domains of the variables consist of the duration distributions of the processes, and the constraints consist of (1) the task constraints denoting the flow of information required, (2) the capacity of the resources and the processes' allowed resource usage, (3) the architectural constraints such as the resources' capacities and amount of processing allowed at any given time, and (4) the minimization of time (or some other objective function). The space of solutions consists of the possible sets of domain values assigned to processes such that all the constraints are satisfied.

This interpretation is useful because it allows us to make contact with the large body of research already in place for RCSPs. In particular, parameters which characterize RCSPs and measures used to evaluate the performance of RCSP-solving algorithms have been developed, tested, and widely used [Kolisch et al., 1995, Kolisch et al., 1998, Patterson, 1976]. This means that input task models, the space of solutions (behavior prediction), and the performance of the constraint solver itself all may be evaluated in terms of these established RCSP metrics.

### Complexity, phrase transitions, and control parameters

RCSP itself is an NP-complete problem [Herroelen and De Reyck, 1999]. However, there are properties of NP-complete scheduling problems which can be exploited to provide a characterization of the complexity of the problem space. A particularly useful one is that many NP-complete problems exhibit *phase transitions*: sudden changes in computational complexity. Phase transitions are products of the way *control parameters*, parameters that characterize the problems to be solved, affect the hardness of the problem. Hard to solve instances occur around a critical value of a control parameter, and are responsible for the phase transition. [Herroelen and De Reyck, 1999]. Therefore one of the goals of the present work is to search for those control parameters that predict phase transitions in cognitive constraint models.

For the RCSP, most control parameters fall into these categories: (1) size of the problem, (2) topological structure (morphology) of the problem, or (3) availability of different resource types in the problem [Herroelen and De Reyck, 1999]. A widely-used example of these is *order strength*, which falls into the topological structure category. Order strength is intuitively the strength of the partial ordering, or the density of the network.

production system architectures such as Soar and EPIC [Meyer and Kieras, 1997] were made viable by early foundational work on the production rule match which led to efficient new algorithms [Forgy et al., 1984, Tambe et al., 1990].

**Definition 1** Order strength *is the number of precedence relations divided by the theoretical maximum of such precedence relations: $T/U$, where $U = n(n-1)/2$, and $n$ is the number of activities.*

In terms of the task model RCSPs, activities are processes, precedence relations are the cascades denoting the flow of information between processes, and the network is the schedule itself, with the processes as nodes and the precedence relations as topological constraints between them. Order strength has been shown to successfully characterize phase transitions for many of the main RCSP-solving algorithms such as Branch-and-Bound [Herroelen and De Reyck, 1999]. There are many other RCSP control parameters; below we present empirical evidence that order strength is significant for cognitive constraint models.

## Cognitive Constraint Problem Generator (CCPG) Framework

The motivation for the CCPG is to develop a tool which can be used to systematically test the performance of CORE on large sets of model instances that we parametrically define. This will allow us to identify characteristics of problems which may increase or decrease efficiency. In general, the CCPG should be able to create, run, and evaluate scheduling problems of varying complexity with respect to some controlled measure. Here we present results with respect to the controlled measure of order strength (refer to Definition 1). The requirements for this tool include:

1. The ability to generate scheduling problems that have similar structure to the cognitive modeling problems to which CORE is intended to be applied.

2. The ability to provide the modeler with enough input flexibility to control some aspects of the scheduling problems generated without losing the advantage of the large variability between types of problems. For example, one input parameter may be the number of processes in the generated problem: using this, the modeler can relatively control the size of the problem, however there is a huge amount of variability in the types of problems that can be generated with a certain number of processes.

3. The ability to guarantee that each generated problem in fact has a solution.

Given these requirements, we made the design choice of creating a problem generator rather than using an off-the-shelf RCSP generator, because existing generators (e.g., [Kolisch et al., 1998, Schwindt, 1995]) do not satisfy 1 and do not always satisfy 2 and 3 in tandem.

### CCPG structure and implementation

The CCPG is implemented in three main stages: the first stage generates fully constrained schedules, the second stage removes constraints systematically, creating under-constrained scheduling problems, and the third stage executes and evaluates the performance of these problems in CORE. Refer to figure 1.

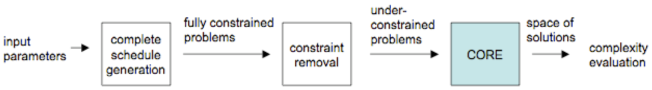

input parameters
complete schedule generation
fully constrained problems
constraint removal
under-constrained problems
CORE
space of solutions
complexity evaluation

Figure 1: Stages of the cognitive constraint problem generator.

**Stochastic generation of fully-constrained schedules** The first stage receives input parameters from the modeler and outputs a number of fully-constrained schedules generated from those input parameters and some randomization. The two types of input parameters are the characteristic and shaping parameters. The characteristic parameters (number of processes, number of resources, etc.) describe the basic structure of the schedules to be generated, characterizing the bounds of the schedule space to be explored. The shaping parameters control the general shape of the schedule, approximating the location of this schedule in the schedule space. The algorithm grows the schedules in a tree-like implementation, where the precedence relations between nodes of the tree correspond to the flow of information between processes. The scheduling of each of the processes is dependent on probability distributions which are affected by the shaping parameters. However, these parameter constraints are not tight enough to limit the number of schedules possible to generate.

The stochastic method allows for the generation of a huge range of similarly-structured schedules, supporting the requirement for variability in 2. To test if the generator gives sufficient parametric control so that schedules produced are similar to cognitive problems of interest (requirement 1), an assessment was carried out using a regression analysis and common metrics used to describe directed acyclic graphs. The analysis showed that the parametric control is significant in shaping the schedule structure and provided guidelines for the choosing of parameters to manipulate schedule formation. This analysis also showed that solution schedule variation is extremely large despite the parametric control (requirement 2) and that the space of possible generated solutions is much larger than the current scope of existing models so absolute verification of a mapping from the space of possible generated task models to the space of human task models is not feasible, at least at this time (requirement 1).

**Turning schedules into problems via constraint removal** In this stage of the CCPG, constraints are removed systematically from the fully constrained schedules to create under-constrained scheduling problems. This systematic removal guarantees that every problem generated is solvable (but does not guarantee what the solution is) satisfying requirement 3, and allows the varying of complexity to be controlled for evaluation with respect to the experimenter-chosen controlled measure. In this implementation, an RCSP control parameter, order strength, is chosen. To systematically vary the order strength, the type of constraints removed must be ordering (flow of information) constraints (Definition 1 precedence relations). This investigates the strength of the partial ordering as a property of scheduling problems which may help characterize the space of solvable task models for CORE.

## An Empirical Investigation of the Effects of Order Strength

We used the problem generator to empirically investigate the effects of order strength variation on the performance of the branch-and-bound algorithm. In what follows, we first describe the structure of the experiment and the parameters varied, and then report the results in terms of execution time and time-out rates as a function of order strength.

In light of this order-strength specific evaluation, we have found it useful to describe the results in two stages. First we establish the low run time model trends, using a low time-out value meaning the problems which take more than a certain CPU time value to run will be stopped and remain unfinished. This is a user-specified quit mechanism in Sictus Prolog.) Then we address the question of fitting higher run time models to these low run time curves, evaluating the ability of the low run time curves to predict the behavior of higher run-time models.

### Structure of the experiment

We embedded the generator in nested loops of execution which ran large numbers of models systematically through CORE under the same machine conditions and recorded the CPU time per run.

Our models include multiple fully-constrained schedules using the same exact input parameters, and also multiple fully-constrained schedules using different numbers of processes but otherwise unchanged input parameters. Each schedule was the starting point for multiple sequences of constraints removed (the sequences ranging from no constraints to all constraints removed), each additional removal amounting to a separate model. This allows for time complexity to be analyzed across (but not limited to) numbers of processes, number of constraints removed (NCRs), and order strength.

### Low run time models

Figure 2 shows the results of the analysis characterizing the behavior of low run time models: their relationship between order strength and execution time. The data in this plot is the result of a range of 5-30 process problems, each with 10 different fully-constrained schedules with 10 different sequences of constraints removed each. This is a total of 52,500 models. Each line is a local polynomial regression fitting of the data points for the models of one number of processes, and each line is labeled with that number of processes. The "Average CPU time per trial" is the average search time the constraint solver needs to find a solution for that model. The time-out is 100ms. (The data plotted here is only for the models that finished running and did not time-out.) The reason for this tight time-out value is the result of a search through ranges of time-outs which exhibited an interesting processing time duality: their run time either took hours or was under a minute. With low run time models it is computationally feasible to run enough models to find a trend if one exists, and if that trend can predict the complexity of the high run time models, then the control measures used are good predictors of the the task models' run time complexity.

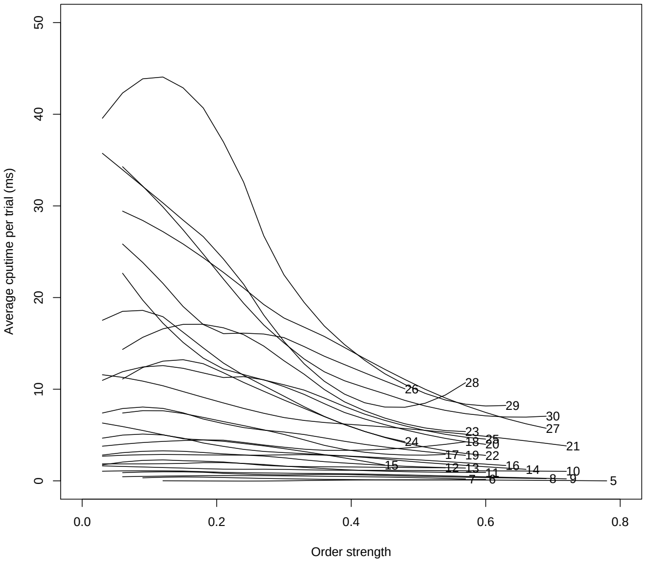


Figure 2: Time complexity of models of different numbers of processes (labeled numbers in the plot) according to order strength. For low run time models.

The trends in Figure 2 curve downward, and more sharply with increasing numbers of processes. This is consistent with the expectation that increasing order strength (increasing strength of the partial ordering of a problem) will decrease the run time. The stronger the structure of the problem, the less time it will take to find a solution because the search space is smaller. With models of smaller numbers of processes, the search space is small to begin with, and so the run time is still low regardless of order strength and the lines are rather low and flat.

A plausible explanation for the downward turn (near 0 order strength) at the peaks of some of the lines is an effect of the order strength formulation: as the order strength approaches 0, the equation $T/U$ ensures that the ratio of existing precedence relations with respect to the theoretical maximum of all precedence relations is getting smaller. For models with large numbers of processes, the ratio can approach 0 fairly quickly without the need for very small numbers of existing precedence relations because of the unstructured nature of the problem. The point at 0 can only occur when there are no existing precedence relations, which is fairly easy (small run time) to schedule a problem with no constraints.

### Higher run time models

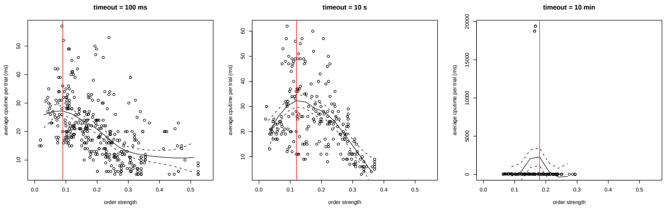


Figure 3: Time complexity across increasing run time models with respect to order strength. Raw data is behind the local polynomial regression fitted line. Dotted lines represent error within 95% confidence interval.

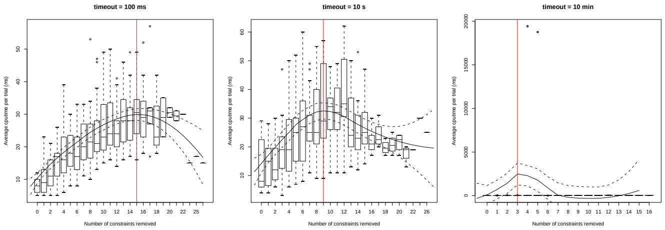


Figure 4: Time complexity across increasing run time models with respect to number of constraints removed. Raw data in boxplot form behind the local polynomial regression fitted line. Dotted lines represent error within 95% confidence interval.

Figures 3 and 4 are examples of the results comparing the behavior of the previous low run-time models to two other sets of higher run time models. The changes across the systematically varied run times was gradual; these particular sets were chosen to illustrate the differences between the higher and lower run time models. (These examples also come from a range of 5-30 process models, same as the low run time models). The trends shown by these examples are representative of the results gathered; they are shown by example to facilitate viewing of the data.

Figure 3 shows the relationship between order strength and execution time across increasing run time models. The vertical line notates the order strength at which the maximum execution time of the smoothed fit occurs, used to compare the hill-shaped trends observed in the lower run time models with the higher run-time models. The top of the hills shift between the different time-out plots, however there is a set range between 0.1 and 0.2 order strength within which the peak of the hill (across all models) lies.

The processing time duality referring to the sudden increase in computational complexity of certain models can be seen here in the 10min time-out plot. The majority of the models have an execution time similar to the 100ms and 10s models, however there are a few models which have a very sudden increase in complexity. These models *cause* the smoothed hill in that plot to be peaked at that order strength, however, when compared to the lower time-out plots, the peak is very close to the prediction of the lower run time models. Because these trends were observed across the total data set, it suggests that order strength may be able to identify those problematic models that have run time complexity many orders of magnitude above its similar but low run time neighbor models.

However, order strength does not explain the reason for the jump in complexity. We are currently examining "minimal pairs" of scheduling problems that are identical in number of processes and resources and differ only in the removal of a single constraint—but which show radically different complexities in that one of the problems takes at least 3 orders of magnitude longer to solve than the other. Such minimal pairs will help us to identify perhaps a different set of control parameters underlying the phase transitions.

These particulars make it worthwhile to compare the same run time data against the number of constraints removed (NCR) from the fully-constrained schedule. The difference between NCR and order strength is subtle: the order strength measures a structural aspect: the density of the problem regardless of the size, and the particular constraint removed does not matter to the NCR measure, however it does to order strength because of the transitive relations involved. Figure 4 consists of the same data as Figure 3. The same basic trends can be recognized (as expected, because of the correlation between the number of constraints removed and order strength). However the shift in the peak of the hill covers a wide range of the NCR metric and does not seem to be bounded well. This suggests that order strength might be a better indicator of those highly computationally complex run time models.

## Discussion

We have examined the computational nature of task models used in Cognitive Constraint Modeling. CCM is an approach to computational modeling which is unique in that it formulates cognitive models as optimized constraint satisfaction problems. Although it may be argued that all cognitive architectures treat modeling as a type of constraint satisfaction problem with resource constraints, it has not been as precisely formulated as it is here. CCM models may be interpreted as resource-constrained scheduling problems, allowing us to take advantage of the complexity and control measures that the RCSP literature has to offer. We have created a problem generator framework to allow for time complexity experimentation with the Branch-and-Bound algorithm used by CORE—a standard approach to algorithm investigation in the planning and constraint satisfaction fields—and we have exploited the advantages of RCSPs both in the framework and in the choice of analyses. This has allowed us to take a very close look at the structure of the problems themselves, and their effect on the computational feasibility of CORE modeling. Our initial use of this tool has yielded promising results. First, it has confirmed that order strength is a predictor of problem complexity for the subclass of RCSPs that are generated in cognitive constraint modeling. Second, it has created a large set of minimal pairs of problems whose constraint removal straddles phase transitions, which should yield insight into the precise characteristics that give rise to such transitions.

The practical implications of this work are significant: If we can precisely characterize the space of the computational complexity of CORE models, we can use the information to make constraint modeling not only more efficient, but in some cases, possible at all. These changes to the constraint modeling may include changing the model specifications by adding or subtracting constraints in a way that does not compromise the theoretical goals of the modeler; use different specialized algorithms on models of high complexity (some algorithms may be better than others at particular kinds of hardness; e.g.[Patterson, 1976]); and also providing the modeler with characterization of the expected complexity of models *before* they are executed.

## Acknowledgments

This work was supported by ONR grant N00014-03-1-0087 and NASA grants NNA05CS65A and NNA04CK14A.

## References

[Brumby and Salvucci, 2006] Brumby, D. P. and Salvucci, D. D. (2006). Towards a constraint analysis of human multitasking. Poster presented at the 7th International Conference on Cognitive Modeling.

[Card et al., 1983] Card, K., S., Moran, P., T., and Newell, A. (1983). *The Psychology of Human-Computer Interaction.* Lawrence Erlbaum, Hillsdale, NJ.

[Eng et al., 2006] Eng, K., Lewis, R. L., Tollinger, I., Chu, A., Howes, A., and Vera, A. (2006). Generating automated predictions of behavior strategically adapted to specific performance objectives. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI)*, Montreal, Canada.

[Forgy et al., 1984] Forgy, C., Gupta, A., Newell, A., and Wedig, R. (1984). Initial assessment of architecture for production systems. In *Proceedings of the National Conference for Artificial Intelligence (AAAI)*, pages 116–120, Austin, TX.

[Herroelen and De Reyck, 1999] Herroelen, W. and De Reyck, B. (1999). Phase transitions in project scheduling. *Operational Research Society*, 50:148–156.

[Howes et al., 2005] Howes, A., Lewis, R. L., Vera, A. H., and Richardson, J. (2005). Information-requirements grammar: A theory of the structure of competence for interaction. In *Proceedings of the 27th Annual Meeting of the Cognitive Science Society*, Stresa, Italy.

[Howes et al., 2004] Howes, A., Vera, A. H., Lewis, R. L., and McCurdy, M. (2004). Cognitive constraint modeling: A formal approach to supporting reasoning about behavior. In *Proceedings of the Cognitive Science Society*, Chicago.

[John and Gray, 1995] John, B. E. and Gray, W. D. (1995). Cpm-goms: an analysis method for tasks with parallel activities. In Katz, I., Mack, R., and Marks, L., editors, *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, Denver, Colorado. ACM Press, New York, NY.

[Kieras and Meyer, 1999] Kieras and Meyer, D. (1999). The role of cognitive task analysis in the application of predictive models of human performance. In Schraagen, J. M., Chipman, S. F., and Shalin, V. L., editors, *Cognitive Task Analysis*, pages 237–260. Lawrence Erlbaum Associates, Mahwah, NJ.

[Kolisch et al., 1998] Kolisch, R., Schwindt, C., and Sprecher, A. (1998). Benchmark instances for project scheduling problems. In Weglarz, J., editor, *Handbook on Recent Advance in Project Scheduling.* Kluwer, Norwell, MA.

[Kolisch et al., 1995] Kolisch, R., Sprecher, A., and Drexl, A. (1995). Characterization and generation of a general class of resource-constrained project scheduling problems. *Management Science*, 41:1693–1703.

[Lewis et al., 2004] Lewis, R. L., Vera, A. H., and Howes, A. (2004). A constraint-based approach to understanding the composition of skill. In *Proceedings of the International Conference on Cognitive Modeling (ICCM)*, Pittsburgh.

[Meyer and Kieras, 1997] Meyer, D. E. and Kieras, D. E. (1997). A computational theory of executive cognitive processes and multiple-task performance: Part 1. Basic mechanisms. *Psychological Review*, 104:3–65.

[Newell, 1990] Newell, A. (1990). *Unified Theories of Cognition.* Harvard University Press, Cambridge, MA.

[Patterson, 1976] Patterson, J. H. (1976). Project scheduling: The effects of problem sturcture on heuristic performance. *Naval Research Logistics Quarterly*, 23:95–123.

[Schwindt, 1995] Schwindt, C. (1995). Progen/max: a new problem generator for different resource constrainted project scheduling problems with minimal and maximal time lags. Research report wior-449, Universität Karlsruhe.

[Tambe et al., 1990] Tambe, M., Newell, A., Rosenbloom, and S., P. (1990). The problem of expensive chunks and its solution by restricting expressiveness. *Machine Learning*, 5:299–348.

# Optimizing Knowledge Component Learning Using a Dynamic Structural Model of Practice

**Philip I Pavlik Jr. (ppavlik@cs.cmu.edu)**, Human Computer Interaction Institute
**Nora Presson (presson@cmu.edu)**, Psychology Department
**Kenneth Koedinger (koedinger@cmu.edu)**, Human Computer Interaction Institute
Pittsburgh Science of Learning Center
Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213 USA

## Abstract

This paper presents a generalized scheme for modeling learning in simple and more complex tasks, and shows how such a model can be applied to optimizing conditions of practice to maximize some desired performance. To enable this optimal allocation of lesson time, this paper describes how to quantify the preferences of students using utility functions that can be maximized. This conventional game theoretic approach is enabled by specifying a mathematical model that allows us to compute expected utility of various student choices to choose the choice with maximal expected utility. This method is applied to several educational decisions that can benefit from optimization.

**Keywords:** Memory; Economics; Practice; Computer-Aided Instruction.

## Introduction

This paper describes a method for applying economic principles in order to allocate the scarce resource of learning time toward satisfying the unlimited need for education. To do this, we describe a model that decomposes learning into individual knowledge components (KCs) that possess some degree of independence from other skills (a "knowledge component" is any proficiency that can be learned). By assuming this independence, the model accounts for the unique effects of practice on specific KCs, with the goal of optimizing the benefit of practice.

We do not argue that the model is a precise representation of all the processes involved in learning, but rather that it provides a heuristic tool to track observed strengths of KCs as a general function of practice, so that improvement over time and across KCs can be optimized. The model we will present, like similar models, is effective in capturing practice effects (Cen, Koedinger, & Junker, 2006). Further, it is interesting to note that the dynamic practice model presented here (based on the ACT-R computational model of declarative memory, Anderson & Lebiere, 1998) might be substituted with another model of cognition with only minimal modification to the approach.

Although the model is a simplification of learning processes in most cases, this simplicity provides an important advantage in application. It allows closed form predictions of which learning events (LEs) might be assigned at what times to maximize learning (a "learning event" is any discrete interval over which a learned proficiency increases). Ultimately, it is explaining this collection of closed form predictions and recommendations that is the goal of this paper.

To explain these concepts this paper has three parts. The first section on the dynamic practice model is largely a review of the ACT-R model of declarative memory. This section serves to orient the reader on the output functions (probability and latency of recall) that will be used later. The second section on structural models details how compound events can be modeled using the dynamic practice model. Compound events are important to consider when responses are not independent and are especially relevant for certain kinds of optimization situations (i.e. part-task to whole-task transfer of performance). The final section shows several ways the model built in the first half of the paper can be applied to optimizing knowledge component learning.

## Dynamic Practice Model

To understand the quantitative model that will be used to predict and optimize learning, we will begin with the equations that predict probability of correct performance and latency of correct performance as a function of the activation strength of a KC.

**Probability Correct**. The first dependent measure of KC performance is probability of correct response. Equation 1 shows the standard Boltzmann equation (similar to the Rasch model used in item response theory), a logistic function that characterizes the threshold of correct performance (the level of activation at which performance is correct greater than 50% of the time) and distributional noise as $\tau$ and $s$ respectively. Equation 1 describes a model of the probability of giving a correct response ($p$) for a given KC activation strength value ($m$) and the parameters described above.

$$p_m = \frac{1}{1 + e^{\frac{\tau - m}{s}}} \qquad \text{Equation 1}$$

**Latency.** A second dependent measure used to track KC performance is latency (labeled $q$ in our model). Various sources suggest modeling latency with a Weibull distribution (Anderson & Lebiere, 1998; Logan, 1995). Such a Weibull distribution can be produced by using Equation 2 to represent latency as a function of $F$ (which scales latency magnitude), $m$ (memory strength) and a fixed cost (which is determined from data and captures the minimum time necessary for perceptual and motor costs of responding). Logistic noise on *m* determines the shape of the aggregate Weibull function for a population of response latencies.

$$q_m = Fe^{-m} + fixedtimecost \qquad \text{Equation 2}$$

## Knowledge Component Strength Function

Given these two output functions, which correspond to two important ways of measuring KC performance, we can now elaborate how current *m* is computed as a function of the history of a student's practice of a KC practice item across *n* prior LEs. Equation 3 shows this KC strength function. The history term, the final portion of Equation 3, is essentially described by three values, *t, d* and *b,* for each LE. The values for *t* represent the times since each past LE (the ages of each LE effect). The *d* values are the power law decay values for each LE. The *b* values scales the effect of each LE depending on the amount of learning for the LE (i.e. longer duration LEs and successful test LEs result in higher *b*s). To model history, the $bt^{-d}$ quantity is summed for each of the *n* learning events (LEs). The logarithm serves to scale the quantity from $-\infty$ to $\infty$. This power law decay formulation was first explored by Anderson and Schooler (1991), who showed that it results in patterns of forgetting that match the relative need for performance in the environment. The $\beta$ parameters, the first portion of Equation 3, capture naturally occurring error when the model is fit to data from multiple students or multiple KCs. $\beta_s$ is the parameter that captures consistent error across KCs as a function of student. $\beta_i$ captures consistent error across students as a function of KC (i stands for item). Finally $\beta_{si}$ captures the residual error for a specific KC and a specific student over multiple LEs.

$$m_n = \beta_s + \beta_i + \beta_{si} + \ln\left(\sum_{k=1}^{n} b_k t_k^{-d_k}\right) \qquad \text{Equation 3}$$

$$b_{study} = g(1 - e^{-v \cdot studyduration}) \qquad \text{Equation 4}$$

Equation 4 shows how *b* can be computed as a function of the duration of a study LE (where *v* and *g* represent a growth constant and the maximum possible encoding respectively). This captures the notion that continuous time spent on a single KC has a diminishing effect on learning (Metcalfe & Kornell, 2003). Recent work by Pavlik (in press) has shown how this *b* scalar can be used to capture the learning difference between active correct responding and passive study. In such work, $b_{successfulretrieval}$ is typically set at a constant, whereas $b_{study}$ varies as described in equation 4. This supposes two canonical forms of the LE: the "study LE," which comes from unassessed study over some fixed period of time of a stimulus representing a KC, and the "test LE," which comes from a variable-duration assessment of learning (test LEs are often followed by study opportunity and then are called "drill LEs"). Test LEs are interesting not only because they tend to lead to more learning than passive study (for correct responses), but also because they provide information about the current state of learning that can be used to implement knowledge tracing. Such knowledge tracing algorithms have changed form over different applications of the model. In the original version (Pavlik Jr., 2005), the distribution of residual $\beta_{si}$ variance is used as the initial Bayesian prior for item strength and numerical integration is used to adjust this value after each practice by integrating the logistic distribution for correctness given the response of the student. In the more recent version, we have found that a more computationally inexpensive model that allows the simpler $b_{successfulretrieval}$ parameter to capture the $\beta_{si}$ variance works well in practice (Pavlik Jr. et al., 2007). Further, the latest version also uses a $b_{latency}$ parameter multiplied by each $b_{successfulretrieval}$ parameter for each successful test. This $b_{latency}$ parameter is a natural log transform (with a scalar parameter) of the difference between $q_m$ (the predicted latency) and the latency data from the student. This creates a knowledge tracing model that assumes that faster responding means more learning has occurred.

Equation 5 shows a more recent modification of the ACT-R equations to capture the spacing effect, the spacing-by-practice interaction, and the spacing-by-retention interval interaction (Pavlik Jr. & Anderson, 2005). This change says that the forgetting rate from any LE depends on the level of activation at the time of the LE. As modeled in Equation 5, when spacing between trials gets wider, activation decreases between presentations; decay is therefore less for each new presentation, and long-term probability of correct performance does not decrease as much.

In Equation 5, the decay rate $d_k$ is calculated for the *k*th presentation of a KC item as a function of the activation $m_{k-1}$ at the time the presentation occurred (e.g., the decay rate for the 7th LE ($t_7$) depends on the activation at the time of the 7th LE, which is a function of the time from last exposure of the prior 6 LEs and their decay rates. It is important to note that since $t_k$s are ages (or differences between the current time and the time of the past trial), activation and decay depend on the current time as well as the number of LEs).

$$d_{m_{k-1}} = ce^{m_{k-1}} + a \qquad \text{Equation 5}$$

Anderson, Fincham, and Douglass (1997) found that Equation 3 could account for practice and forgetting during an experiment, but it could not fit retention data over long intervals. Because of this, they concluded that between sessions, the presence of intervening events erodes KCs more slowly than during an experimental session. This slower forgetting was modeled by scaling time as if it were slower outside the experiment. Forgetting is therefore dependent on the "psychological time" between presentations, rather than the true intersession interval. This factor is implemented by multiplying the portion of time that occurs between sessions by *h* (a scalar parameter for time) when calculating recall. This is done by subtracting *h*total intersession time* from each age ($t_k$) in Equation 11 (Pavlik Jr., 2005; Pavlik Jr. & Anderson, 2005). Because of this mechanism, time in the model is essentially a measure of destructive interfering events. The decay rate, therefore, is a measure of "fragility" of memories to the corrosive effect of these other events.

This model has the flexibility to capture many varieties of learning and practice effects. To further understand this flexibility, consider the issue of more implicit production rule (procedural) learning in contrast to explicit factual (declarative) learning. This distinction is supported by research from widely distinct theoretical perspectives such as ACT-R and connectionism and is supported by dissociable neural mechanisms (McClelland, McNaughton, & O'Reilly, 1995). We might wonder whether the equations just presented are adequate to capture both knowledge (and KC) types. Specifically, that work implies that declarative learning is both faster (reflected by a larger *b* parameter) and more easily forgotten (reflected by a larger *d* parameter) than procedural learning, and our model can clearly characterize these differences.

## Structural Model

The structural level model assumes that few domains are made up of entirely independent KCs, as seems to be implied in the model we just presented. The word "structural" refers to the fact that, because of this lack of independence, the modeler must be concerned with the structure that links the multiple KCs and their association. In many domains, predictions of the probability of correct response and latency are derived from the strength of more than one underlying KC. For example, in studies of Chinese vocabulary learning, stimuli can be presented in one of four modes (Hanzi character, pinyin text, sound file, and English text). This results in 6 possible test LE types, two of which are English→pinyin and Hanzi→pinyin drill LEs [(stimulus)→(response)]. In both of these cases, drill success depends not only on the strength of the link between the stimulus and response, but also on the ability to recall and produce the pinyin response. Because of this, performance for these pairs cannot be independent.

Similarly, in work with a French gender identification task, words fall into gender categories based on spelling and semantic cues. For instance, words that end in –age are most often masculine in French, as in *le fromage*. Although each of these words might yield a correct response independent of the general rule (through recall), it is also obvious that all rule exemplars share a KC that can be used to respond to any items in a cue category (and in fact, it is this generalized responding, rather than exemplar-based recall, that we want to optimize).

To deal with the fact that multiple KCs are required for these single skills, we will propose two basic structural models that account for this, each of which fits some possible learning tasks: the conjunctive structure and the disjunctive structure.

### Conjunctive Model

In a conjunctive model, all component KCs must be active to produce a correct response. For instance, in the Chinese vocabulary work, probability of correct performance for each trial is captured by the probability of correct recall for both the response and the link between the stimulus and the response. Given this model, probability correct depends on both the strength of the link and the strength of the response in a conjunctive function: p(link) * p(response), such that both elements are necessary for a correct response. The more general form for the conjunction of 2 KCs is shown in Equation 6. Latency, on the other hand, is handled as the sum of the perceptual motor costs, the cost for recall of the link KC, and the cost for recall of the response KC. Not only does this structural model handle the pinyin response example above, but it also captures data showing that responding with a word in the native language should be easier than the recently learned foreign equivalent (e.g. Schneider, Healy, & Bourne, 2002).

$$\mathrm{p}(KC_1 and KC_2) = \mathrm{p}(KC_1)\,\mathrm{p}(KC_2) \qquad \text{Equation 6}$$

### Disjunctive Model

The disjunctive model, in contrast, assumes that a trial can yield a correct response due to performance of any one of the two or more independent KCs. Often disjunctive models apply in a generalization situation where the domain contains specific KCs that apply for individual stimuli and general KCs that each apply to a group of stimuli, as in the French gender case. In this example, we can imagine that general group KCs control performance for "clusters," the members of which can also be learned by rote. Given the example of a general (rule-based) and specific (rote) component controlling each performance, probability of correct skill performance depends on the strength of both general and specific components in a disjunctive function, p(general) + p(specific) * (1-p(general)), such that (for example) a student could classify a novel word on the sole basis of the general KC. The general form of this model is shown in Equation 7.

$$\mathrm{p}(KC_1 or KC_2) = \mathrm{p}(KC_1) + \mathrm{p}(KC_2)(1 - \mathrm{p}(KC_1))$$

Equation 7

## Optimizing Learning

The following procedures describe how one can use the model to compute optimal practice schedules. Usually, we assume that what is being optimized is gain in some long-term measure of learning for a KC or multiple KCs. Although using long-term probability correct as a dependent measure works when we focus on optimizing some global aggregate task (like the optimal total number of practices for an item), we need a different utility function for more dynamic local scheduling (such as picking an item to practice next), in order to formalize preferences for the learning gains from different LE schedules.

### Utility Optimizations

We propose to use Equation 8 as the utility function for a LE (where *b* controls the weight of the LE, *t* is the desired retention interval of the LE, and decay (*d*) is a function of the activation (*m*) at the time of practice). Most importantly, Equation 8 does not have the all-or-none property of probability correct (because probability correct is a sigmoid function, it usually approaches 0 or 1). If we tried to use long-term probability correct as our measure of local utility, it would value practice most heavily when it comes near the transition from mostly incorrect performance to mostly correct performance across a sequence of test LEs (those LEs that fall on the intermediate part of the curve). This bias distorts the fact that we are ultimately more concerned with the minimum number of practice trials required to reach a certain long-term retention, not scheduling each practice trial so that it increases percent correct maximally. These goals are actually quite different since long-term percent correct gain from the next practice depends on nearness to long-term floor or ceiling performance, while utility gain is not affected by these bounds. Thus, our utility function maximizes the overall goal by valuing LEs independently of the order they occurred, considering only their unique contributions (a function of strength of encoding, recency, and the decay rate) to the long-term KC strength.

$$u = bt^{-d_m} \qquad \text{Equation 8}$$

We will use Equation 8 as a cardinal utility function: e.g., a .2 increase in strength is half as good as a .4 increase in strength. One reason why this assumption is reasonable is because LEs contribute to KC strength in small increments and these increments are interchangeable, as illustrated in Equation 3. Using a cardinal utility function allows us to directly compare different possible spacings and KC presentation orders, to determine when learning is maximal, given learning history. Further, we assume that this utility equation satisfies the von Neumann and Morgenstern game theoretic axioms of completeness, transitivity, continuity and independence required for comparing expected utility lotteries (Von Neumann & Morgenstern, 1944).

**Practice Spacing Optimization (PSO).** For each KC and each student, it is useful to decide when it would optimal to repeat a drill LE of that KC. Therefore, we are trying to schedule the LEs under conditions of allocative efficiency. In economics, allocative efficiency is a condition where costs (time spent learning) are allocated in a way that maximizes gains (increases in utility). Taking this parallel to learning theory, we search for the retention interval (for each KC) at which the expected rate of learning utility gain is maximal given a new LE. This is expressed in Equation 9, which calculates the maximum utility gain for a KC as a function of *m* (activation of that KC) and *t* (the target retention interval needed to compute *g* in Equation 8). All the other values are fixed parameters ($b_s$ = success LE weight from Equation 8 if the test LE is successful, $b_f$ = failure LE weight from Equation 8 for the study LE given as review, *-d* computed from the current *m* (needed with *t*, $b_f$ and $b_s$ to compute *u* values), $p_m$ and $q_m$ estimated for the test LE from Equations 1 and 2, and failure costs estimated from prior data). Because *t* and *m* are the only values that vary in finding the optimum spacing, we can solve for the optimal level of the one given the other. For example, if we know the desired retention interval, we can solve for the max of Equation 9 to solve for the optimal level of activation at

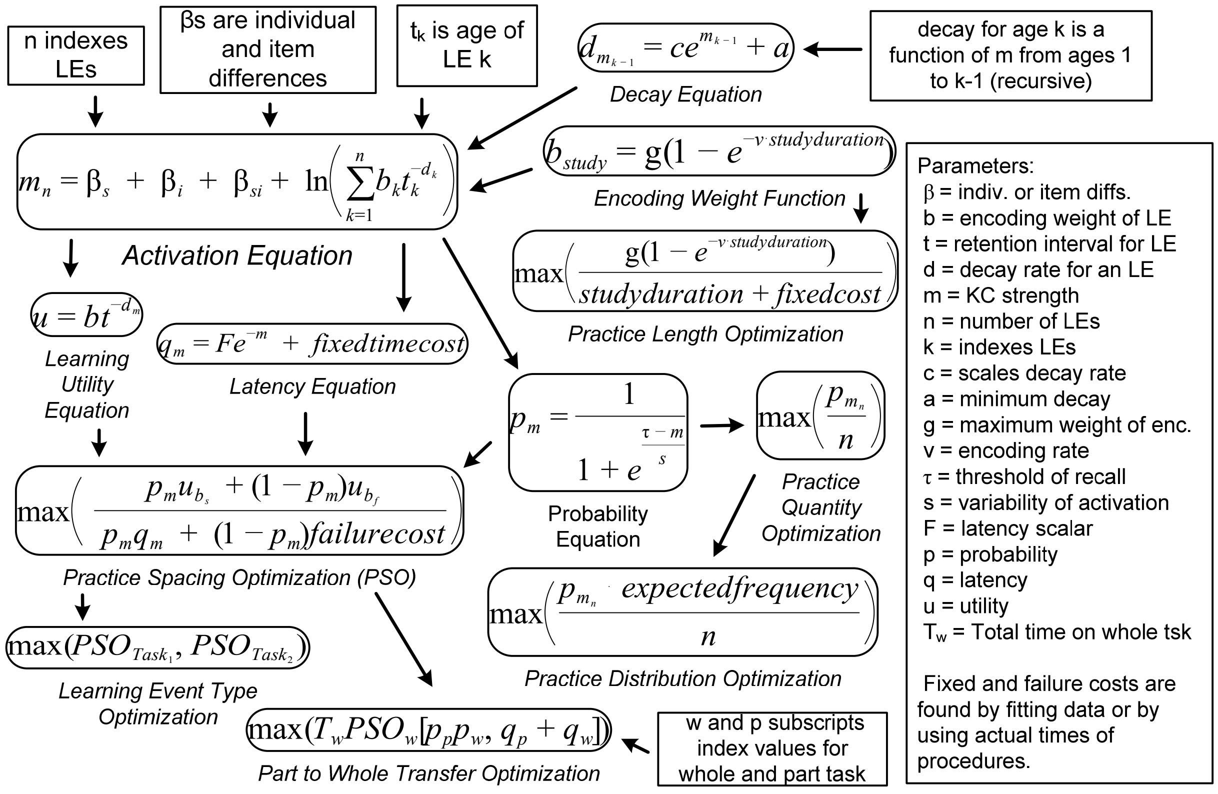

n indexes LEs
βs are individual and item differences
$t_k$ is age of LE k
$d_{m_{k-1}} = ce^{m_{k-1}} + a$
Decay Equation
decay for age k is a function of m from ages 1 to k-1 (recursive)
$m_n = \beta_s + \beta_i + \beta_{si} + \ln\left(\sum_{k=1}^{n} b_k t_k^{-d_k}\right)$
Activation Equation
$b_{study} = g(1 - e^{-v \cdot studyduration})$
Encoding Weight Function
$\max\left(\frac{g(1 - e^{-v \cdot studyduration})}{studyduration + fixedcost}\right)$
Practice Length Optimization
$u = bt^{-d_m}$
Learning Utility Equation
$q_m = Fe^{-m} + fixedtimecost$
Latency Equation
$p_m = \frac{1}{1 + e^{\frac{\tau - m}{s}}}$
Probability Equation
$\max\left(\frac{p_{m_n}}{n}\right)$
Practice Quantity Optimization
$\max\left(\frac{p_m u_{b_s} + (1 - p_m) u_{b_f}}{p_m q_m + (1 - p_m) failurecost}\right)$
Practice Spacing Optimization (PSO)
$\max\left(\frac{p_{m_n} \cdot expectedfrequency}{n}\right)$
Practice Distribution Optimization
$\max(PSO_{Task_1}, PSO_{Task_2})$
Learning Event Type Optimization
$\max(T_w PSO_w [p_p p_w, q_p + q_w])$
Part to Whole Transfer Optimization
w and p subscripts index values for whole and part task
Parameters:
β = indiv. or item diffs.
b = encoding weight of LE
t = retention interval for LE
d = decay rate for an LE
m = KC strength
n = number of LEs
k = indexes LEs
c = scales decay rate
a = minimum decay
g = maximum weight of enc.
v = encoding rate
τ = threshold of recall
s = variability of activation
F = latency scalar
p = probability
q = latency
u = utility
$T_w$ = Total time on whole tsk
Fixed and failure costs are found by fitting data or by using actual times of procedures.

Figure 1. Organizing diagram of the mathematical relationships in this paper.

which practice should occur.

In practice, Equation 9 tends to suggest (for a drill procedure) that when failure costs for errors and error feedback are high, or success gains from correct responding are much greater than failure gains from feedback study, long-term gains in utility per second of practice will be highest when repetitions are scheduled so that test LE performance is maintained at a high probability. However, because the decay parameter can be large for an LE after a short spacing, some spacing is always preferred.

$$\max\left( \frac{p_m u_{b_s} + (1 - p_m) u_{b_f}}{p_m q_m + (1 - p_m) failurecost} \right)$$

Equation 9

**Learning Event Type Optimization.** The above discussion assumes a single task (drill) which can be selected for each item. However, we can also propose other types of LEs and then compare them with the drill trial. For example, we could decide whether it was better to give a study LE alone or to give a drill LE( a test LE followed by a study LE when the test fails). To do this, Equation 10 shows how we can compare the learning rates for each trial type to determine the optimal next trial type for the student. This principle can be extended to compare any two tasks (e.g., tutored problem solving vs. untutored problem solving). This is typically used in combination with dynamic PSO calculations (when the *PSO*s in Equation 10 are computed as a function of the current time) to pick the optimal time for the optimal task.

$$\max(PSO_{Task_1}, PSO_{Task_2}) \qquad \text{Equation 10}$$

**Part- to Whole-Task Transfer Optimization.** For this optimization, the question is whether to practice only single KC components of a whole skill (a conjunctive skill containing at least 2 KCs), only the whole skill or some mixture of the two types of practice. Imagine, for example, practicing simple algebra, and consider that a component of the whole task may be knowing the times tables (the low level component). In this case, the question is how much practice should be allocated to times tables practice before doing algebra practice (the high level component). We might expect that either spending no time on times tables or no time on algebra would likely result in poorer algebra performance than some mixture of these extremes, and that an optimal mixture would allow for the best possible algebra performance. Part to whole transfer optimization allows us to determine this optimal mixture.

To compute this optimal mixture, we model the effect of the low level component LEs on the high level component learning rate. To do this, we must create an equation expressing whole task learning as a function of part task learning. Equation 11 (where subscripts *w* and *p* refer to whole and part task respectively) captures the notion that we are looking to maximize whole task time ($T_w$) * learning rate from an optimally spaced LE, which equals the total learning (this method assumes that all practice occurs at the *PSO* optimal point). Here we specify that *PSO* for a conjunctive task is a function of the strength of the whole (dependent) KC and the probability and latency estimates for the part task. By doing this, we have created a new version of the PSO, $PSO_w$, that depends on the strength of both the part and whole task KCs. At the same time, we are only concerned with the learning of the whole task, so in practice, the *t* (retention interval) and *g* (utility gain) terms are not changed from the original PSO. This provides a mechanism whereby the higher probability and lower latency for a practiced part task increases the expected strength of the $PSO_w$.

$$\max(T_w PSO_w[p_p p_w, q_p + q_w]) \qquad \text{Equation 11}$$

Having this mechanism, we can compute the time needed to train the part task to maximize its effects on whole task learning. In this case, it can be noted that $totaltime\text{-}T_w$ is spent on the part task, with a learning rate of $PSO_p$; these values, therefore, control $p_p$ (probability correct) and $q_p$ (latency). This allows us to construct Equation 11, which represents total learning as a function of time spent on the whole task, multiplied by the learning rate for the whole task (which, because of the conjunctive response functions in the $PSO_w$, is itself a function of time spent on the part task multiplied by the part task learning rate). Equation 11 can then be solved for $T_w$ where $T_w \geq 0$ and $T_w \leq totaltime.$

## Practice Length Optimization

Practice length optimization determines the optimal duration of a given LE. PLO relies on the fact that KC study for each LE has diminishing marginal returns as a function of time as shown in various studies (Metcalfe & Kornell, 2003; Pavlik Jr., in press). Equation 12 shows how this optimal study duration is found when the total LE weight score (from Equation 4) divided by the time spent studying is maximized. (Equation 12 assumes some minimum study duration greater than 0 to account for fixed costs.)

$$\max\left( \frac{\mathrm{g}(1 - e^{-v\, studyduration})}{studyduration + fixedcost} \right) \qquad \text{Equation 12}$$

## Practice Quantity Optimization

Practice quantity optimization uses probability correct for long-term practice as a utility measure, then determines how many optimally spaced repetitions it takes to reach the point where probability gain per LE is maximal (the practice quantity optimization point is the $p_m$ value when Equation 13 is maximized) for each item being learned of a set of items.

$$\max\left( \frac{p_{m_n}}{n} \right) \qquad \text{Equation 13}$$

Figure 2 graphs Equation 13 for the parameter set in Pavlik Jr. (2005, Experiment 4) where it was found that 11 practices would have been optimal for each KC, as the maximum value of the probability correct/practices curve occurs at 11 repetitions. It is useful to note that the utility function should reflect the nature of our preferences for target knowledge. For example, if the need for one KC is higher than others, then getting it correct has a higher utility.

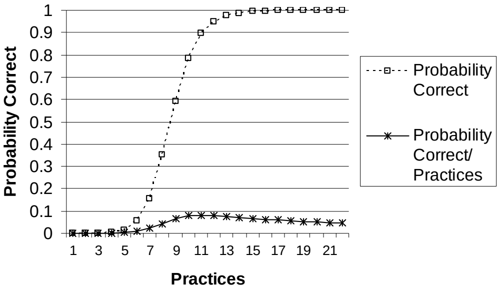


Figure 2. Practice quantity optimization.

To implement this in the model, for instance, we can weight the utility function by the expected frequency of the item we are interested in. This captures the notion that it is twice as important to know a word when that word is used twice as frequently. Having weighted the utility functions, we could then determine a cutoff word frequency below which we will not be concerned with learning the word (this fixes the total amount of time we will need to spend learning the corpus in question).

$$\max\left(\frac{p_{m_n} \cdot expectedfrequency}{n}\right) \qquad \text{Equation 14}$$

Because the weights represent our preferences, other ways of weighting the relative values of different distributions of practice amongst items might further improve the usefulness of such procedures in implementation. For example, items could also be weighted based on the consequences for slow or incorrect performance with the item.

## Conclusion

This paper was about a general “microeconomic” method of using a computational model of cognition to compute the efficiency of various decisions that occur during practice. This work is relevant to education because it shows a new approach to understanding how to improve education by considering learning by the student as the measure of profit. In this new approach, the learning of sets of skills can be optimized to maximize output given input.

While we tied this method to an ACT-R cognitive model, there seems no reason why this method could not be used to optimize learning using another computational model. The elegance of the method explained here is that it is theory neutral (given a particular model) and so results in predictions that must be true given the limits of the particular model and the accuracy of the utility function used to capture preferences. In practice, however, the potential of this method can be limited in domains where the complexity of the KC or LEs prevents the clear specification of a utility function to optimize.

## Acknowledgments

This research was supported in part by grant from Ronald Zdrojkowski for educational research; the Pittsburgh Science of Learning Center which is funded by the National Science Foundation award number SBE-0354420, and a Graduate Training Grant awarded to Carnegie Mellon University by the Dept. of Education (#R305B040063).

## References

Anderson, J. R., Fincham, J. M., & Douglass, S. (1997). The role of examples and rules in the acquisition of a cognitive skill. Journal of Experimental Psychology: Learning, Memory, & Cognition, 23(4), 932-945.

Anderson, J. R., & Lebiere, C. (1998). The atomic components of thought. Mahwah, NJ, US: Lawrence Erlbaum Associates Publishers.

Anderson, J. R., & Schooler, L. J. (1991). Reflections of the environment in memory. Psychological Science, 2(6), 396-408.

Cen, H., Koedinger, K. R., & Junker, B. (2006). Learning factors analysis - A general method for cognitive model evaluation and improvement. In T.-W. Chan (Ed.), Lecture Notes in Computer Science Intelligent Tutoring Systems (Vol. 4053, pp. 164-175): Springer.

Logan, G. D. (1995). The Weibull distribution, the power law, and the instance theory of automaticity. Psychological Review, 102(4), 751-756.

McClelland, J. L., McNaughton, B. L., & O'Reilly, R. C. (1995). Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. Psychological Review, 102(3), 419-437.

Metcalfe, J., & Kornell, N. (2003). The dynamics of learning and allocation of study time to a region of proximal learning. Journal of Experimental Psychology: General, 132(4), 530-542.

Pavlik Jr., P. I. (2005). The microeconomics of learning: Optimizing paired-associate memory. Dissertation Abstracts International: Section B: The Sciences and Engineering, 66(10-B), 5704.

Pavlik Jr., P. I. (in press). Understanding and applying the dynamics of test practice and study practice. Instructional Science.

Pavlik Jr., P. I., & Anderson, J. R. (2005). Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect. Cognitive Science, 29(4), 559-586.

Pavlik Jr., P. I., Presson, N., Dozzi, G., Wu, S.-m., MacWhinney, B., & Koedinger, K. R. (2007). The FaCT (Fact and Concept Training) System: A new tool linking cognitive science with educators. In D. S. McNamara & J. G. Trafton (Eds.). Mahwah, NJ: Lawrence Erlbaum.

Schneider, V. I., Healy, A. F., & Bourne, L. E., Jr. (2002). What is learned under difficult conditions is hard to forget: Contextual interference effects in foreign vocabulary acquisition, retention, and transfer. Journal of Memory and Language, 46(2), 419-440.

Von Neumann, J., & Morgenstern, O. (1944). Theory of games and economic behavior. Princeton,: Princeton university press.

# Creating Cognitive Models from Activity Analysis: A Knowledge Engineering Approach to Car Driver Modeling

**Olivier Georgeon (olivier.georgeon@inrets.fr)**
Institut National de Recherche sur les Transports et leur Sécurité,
25 avenue François Mitterrand, 69675 Bron, France.

**Matthias J. Henning (matthias.henning@phil.tu-chemnitz.de)**
Chemnitz University of Technology, Psychological Institute, 09107 Chemnitz, Germany.

**Thierry Bellet (thiery.bellet@inrets.fr)**
Institut National de Recherche sur les Transports et leur Sécurité.

**Alain Mille (alain.mille@liris.cnrs.fr)**
Université Claude Bernard Lyon 1 - LIRIS, 8, Boulevard Niels Bohr, 69622 Villeurbanne, France.

## Abstract

A cognitive modeling approach coming from ergonomics is presented. It is based on an activity analysis that starts from a "trace of activity" made up of data recorded with sensors. Knowledge engineering software was developed to transform this trace and map it with explicative concepts. These concepts are defined by cognitive ergonomists to explain the activity by the mental states and processes of the operator. These concepts are produced on a pragmatic and evolutionist basis with the interactive help of the software, and are confronted with the operator's own assessment of his subjectivity. The approach is illustrated by its application for car driver cognitive modeling.

## Introduction

The increase of professional activities having a "mental dimension" has encouraged the development of a cognitive ergonomics for several decades. Wilson and Corlett (2005) defined ergonomics as the theoretical and fundamental understanding of human behavior and performance in purposeful interacting systems, and its application to design interactions in the context of real settings. More specifically, cognitive ergonomics rests overall on the statement that this kind of activity cannot be understood without any reference to the operator's "subjectivity", including perceptions, intentions, feelings, expectations, knowledge, mental representation, etc. The main investigating method for completing this type of understanding is "activity analysis". It starts from an observation of the activity in a situation as natural as possible, then it consists of an analysis for creating models of activity, and eventually cognitive models.

Our work focuses on the first half of Wilson's definition: understanding for creating models that could explain and predict the operator's behavior. However, we take the term understanding in a pragmatic sense, which considers any meaning as grounded by a final purpose. We present here a methodology and a software tool to help ergonomists build cognitive models of an operator involved in an activity. We illustrate it by the creation of models of the car driver, within the global purpose of assessing or designing intelligent driving assistance systems.

## Cognitive Modeling of the Car Driver

Most of the cognitive user models come from Human Computer Interaction studies. As Salvucci and Lee (2003) introduced it, they have typically addressed behavior on one of two levels of abstraction. The higher level modeling frameworks, e.g. GOMS, represents behavior as basic user actions, such as moving a mouse or pressing a key. They mostly represent the user's knowledge and representations under complex data structures such as "scripts" or "frames" (Minsky, 1975). The lower level frameworks, e.g. ACT-R or EPIC, describe "atomic components" of behavior with rules triggering over cognitive steps of roughly 50 milliseconds. Here the user's knowledge is represented under declarative "chunks" and procedural "production rules".

There have been many cognitive models of the car driver at both of these levels. At the higher level, Bellet and Tattegrain-Veste (1999) implemented Cosmodrive, a specific modeling framework of the car driver. At the lower level, ACT-R was used by Salvucci, Boer, and Liu (2001) to predict behavior during lane changes on a motorway, or by Ritter, Van Rooy, St. Amant, and Simpson (2006) to model the driver of a driving simulator.

The modeling task is facilitated by these modeling frameworks that increasingly embed general scientific knowledge about human cognition. The existence of these frameworks thus raises the question of how to feed them with a realistic coding of the specific cognitive activity which is modeled. The specification of these "frames", "scripts", "chunks" and "production rules", for creating a specific model is indeed more than a simple programming task. It is a task of modeling the activity from a cognitive point of view. It involves an expertise about the activity itself, and also about how the operator builds and processes the information needed to complete it. This expertise is acquired by ergonomists through their professional practice. What interests us is to support the development of this expertise, formalize it and capitalize it in a knowledge engineering tool for activity analysis and modeling.

## Explaining Behavior by Mental Activity

A car passenger who sees something dangerous and that the driver does not react to could obviously infer that the driver is not aware of the danger and warn him. Inferring others' mental state is something we are always doing in our everyday life. This fact of granting other people with having a subjectivity can be taken in our approach either as an ethic injunction, or as a pragmatic viewpoint: the intentional stance (Dennett, 1987). Our purpose is to make this assessment of the subjectivity more rational, from the observation of behavior and situation as it can be recorded by sensors.

We rely on concepts of cognitive psychology such as "mental representation", or "Situation Awareness" (Endsley, 1995). The latter simultaneously covers the perception, the comprehension and the anticipation for describing how the elements of the situation are "mentally" taken into account by the operators. However, if these concepts seem clear for explaining behavior which results from a symbolic reasoning, it becomes less clear when considering an activity in which the subject is physically involved. Here, their subjective experience "accompanies" their actions and can be seen as much as a cause than as a consequence of their activity. In this case, behavior can be explained by "operational schemas", "implicit knowledge" or automatic processing (Schneider & Shiffrin, 1977). Finally the relation between different levels of control can be studied: skills, rules, and knowledge: (Rasmussen, 1983), or implicit versus explicit control.

### The Objectivity of Mental Activity

There could be a vicious circle in explaining the operators' behavior by their mental activity that is itself inferred from their behavior. This raises the question of how to take other's subjectivity as an objective reality, which can be scientifically studied. Psychologists have been discussing this issue since the beginning of psychology. In this study we retain three elements of response:

(a) A pragmatic epistemology should be adopted, where the concepts created for explaining activity make sense in virtue of their usage (Wittgenstein, 1953). The approach should support a short testing of the usability of the concepts and an interactive negotiation of their meaning.

(b) The approach should also conform to an evolutionist epistemology (Popper, 1972). An explanation cannot be proved to be true in an absolute sense, but can be retained as long as it has not been falsified. The approach should facilitate the production of explanations and their possible falsification in a short loop.

(c) The explanations should be confronted with the subjective assessment made by the subjects themselves. These assessments should be reasonably trusted and analyzed as formal input data (Ericsson & Simon, 1993).

To address these requirements, we need some facilities to easily define and manipulate explicative concepts. The data collected from a recording of the activity should be easily linked to these concepts.

## Methodology and Tool

This issue of making sense of collected data addresses the research area of "knowledge discovery" as defined by Fayyad (1996): the overall process of discovering potentially useful and previously unknown information or knowledge from a database. In our case, the data is the recording of the activity collected by sensors, and the expected knowledge consists of models explaining the activity in terms of the cognition of the subject. To Fayyad, this problem is one of mapping low-level data into other forms that might be more compact, more abstract, or more useful. A knowledge discovery system should enable a user to drive a cyclical process of abstracting and understanding data. The knowledge is finally built "in the analyst's mind" with the interactive use of the system. Progressively, with this interaction, the knowledge is however capitalized in the software under the form of a step by step improvement of the abstraction and of the visualization facilities.

To Miles and Huberman (1994), analysis involves a classification of datum that could be theory-driven, data-driven (evolving a "grounded theory"), or a compromise between both, which he calls "ontological coding". The latter is what we are proposing here, to define people-centered categories such as context, intention, state, or patterns of relationship between elements.

### A Trace-Based System

The specificity of our data is that it is sequential, i.e. every item of data is associated to a time-stamp and related to a particular moment of the activity. Sanderson and Fisher (1994) have set the bases of what they called Exploratory Sequential Data Analysis (ESDA) in this founding paper. They define it as an exploratory process that aims at "Looking at data to see what it seems to say". Hilbert and Redmiles (2000) propose a survey of such tools dedicated to the analysis of human computer interaction. Namely, MacShapa (Sanderson, McNeese & Zaff, 1994) can be highlighted for its covering facilities of visualization, searching, filtering, abstracting and computing statistics. Hawk (Gusdial, Santos & Badre, 1994) is also noticeable for its facilities of abstraction based on a specific programming language. But these tools are dedicated to usage modeling and not to cognitive modeling. The need remains for facilities which could help model the mapping between low level events and higher level explicative concepts, and help investigate the meaning of these concepts.

We suggest addressing this need by modeling the sequential data as a "trace", as defined by Laflaquière, Sofiane-Settouti, Prié, and Mille (2006). For them, a trace is a graph where nodes are facts or events and arcs are relations between them. A trace is associated with a "trace model” that consists of an ontology where the semantic of facts, events, and relations is defined. On this basis, transformation and mapping of traces can be modeled and performed by what they call a “Trace-Based System”, which is a Knowledge-Based System dedicated to traces.

### The ABSTRACT Software Tool

Our implementation of a Trace-Based System for cognitive activity analysis is named ABSTRACT (Analysis of Behavior and Situation for menTal Representation Assessment and Cognitive acTivity modeling). It is described from a computer science point of view in (Georgeon, Mille & Bellet, 2006). ABSTRACT relies on several basic standards and tools to represent and manipulate the traces. The traces themselves are encoded under the RDF format (Resource Description Framework), a standard for encoding graphs. The trace models are encoded under the RDFS format (Resource Description Framework Schema), a standard for encoding ontologies. The inference rules for abstracting the traces are written in SPARQL, a graph query language for RDF. The graphical visualization is made in the SVG format (Scalable Vector Graphic). It is a vector graphic standard which enables interaction.

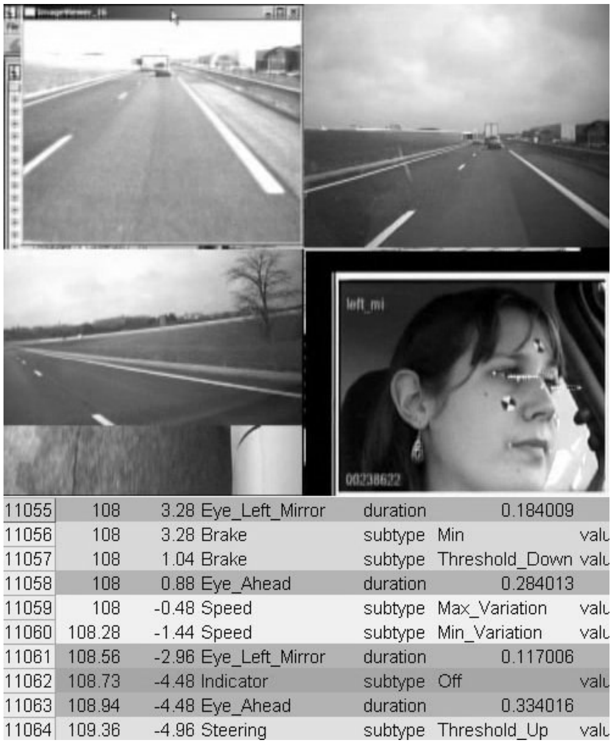
The most challenging issue concerns interactivity, since the whole process rests on the identification, the understanding and the definition of patterns of interest by the ergonomist. We address this issue by providing facilities to easily browse and transform the visual plots, define new explicative symbols and specify inference rules for producing new instances of these symbols. So far, the specification of these inference rules requires the ergonomist to have a basic understanding of the SPARQL langage. He creates them in a semi-graphic way, i.e. he interactively produces skeletons of queries, and then he manually completes them. We are still working on improving the functionality of creating these queries towards a full graphical interface.

## The Modeling Process with ABSTRACT

### The Experimental Data

In our example, the data is collected with an instrumented car during driving experimentations on an open road. Figure 1 shows an example of the video recording. The top left image is the video output of the obstacle detection system, the top right is the front view, the lower left - upper part is the rear view, the lower left - bottom part is the lane view, and the lower right is the video output of the eye tracker. In addition, data is collected from sensors: speed, steering angle, pedal use, distance ahead, obstacle detection from telemeters, and GPS positioning. Situations of interest are marked by the experimenter in the car by pressing a button. Subjective evaluation is obtained from the driver during a post-experiment “self-confrontation” interview. The driver reviews the video with the ergonomist and is asked to assess specific situations by placing cursors on linear scales for each criteria: difficult, critical, dangerous, stressful, responsibility, surprise, fear, and performance.

The experimentation collects a set of time-stamped data. Its choice and its form is not neutral; it is based on the initial knowledge of the activity, and on the research goals. This choice constitutes a form of “anticipatory data reduction”. This data is also possibly pre-processed with algorithms of noise reduction or of sensor calibration.



| | | | | | | |
|---|---|---|---|---|---|---|
| 11055 | 108 | 3.28 | Eye_Left_Mirror | duration | 0.184009 | |
| 11056 | 108 | 3.28 | Brake | subtype | Min | valu |
| 11057 | 108 | 1.04 | Brake | subtype | Threshold_Down | valu |
| 11058 | 108 | 0.88 | Eye_Ahead | duration | 0.284013 | |
| 11059 | 108 | -0.48 | Speed | subtype | Max_Variation | valu |
| 11060 | 108.28 | -1.44 | Speed | subtype | Min_Variation | valu |
| 11061 | 108.56 | -2.96 | Eye_Left_Mirror | duration | 0.117006 | |
| 11062 | 108.73 | -4.48 | Indicator | subtype | Off | valu |
| 11063 | 108.94 | -4.48 | Eye_Ahead | duration | 0.334016 | |
| 11064 | 109.36 | -4.96 | Steering | subtype | Threshold_Up | valu |

Figure 1: Confronting the collected trace to the video.

### The Collected Trace

The collected trace is the first level of trace in ABSTRACT. It is obtained by converting the collected data into a succession of low level events precisely defined in an ontology. For instance, Figure 2 shows how points of interest of the analogical curves are extracted and merged into the trace: Thresholds, local Minimums and Maximums, inflection points, etc.

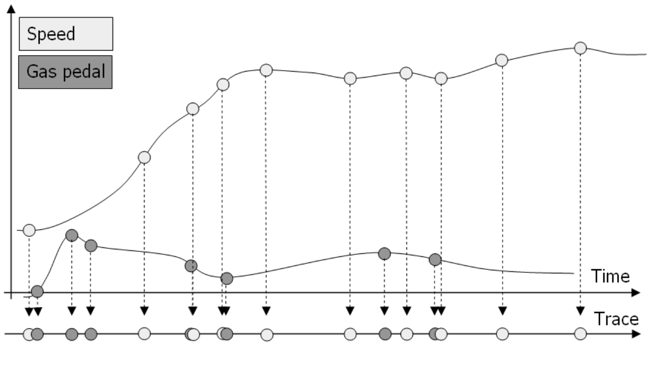


Figure 2: Discretization of the analogical data.

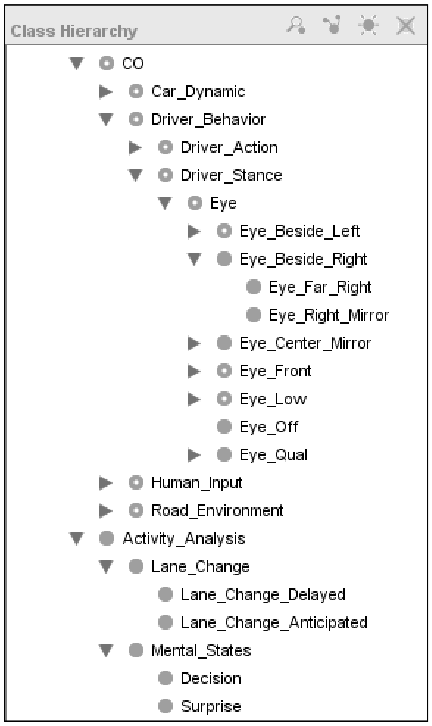
The line at the bottom of the figure represents the collected trace. The circles represent the events from different sources. Numerical or textual properties are attached to these events as needed: their time-code, their source, their type, their value, their duration, the variation rate of their value at this point, etc. This collected trace requires a validation by the ergonomist to ensure that these events can respond to his scientific issues. The adjustment of the various parameters used for producing these events is a delicate but important step. This validation is made with the help of a facility provided by ABSTRACT for playing the video in synchronization with the trace (Figure 1). The collected trace is represented in a spreadsheet with a line for each event and their properties in columns. A color code makes it easier to read. The spreadsheet is automatically scrolled up or down to follow the video when it is played forwards or backwards. The first event of Figure 1 represents a triggering event of the left mirror area of the eye tracker. It lasts 184 ms and happened at a speed of 108 km/h with a steering wheel angle of 3.28°. The last line of the figure is a threshold crossing event upwards of the steering wheel angle. The threshold value of 5° was chosen after different trials and checks because it proved to be a meaningful/useful threshold for studying lane changes on motorways. The video is not entirely encoded into symbols and remains exploited by the ergonomist in parallel with the symbolic traces.

## The Analyzed Traces

The collected trace is then enriched by more abstract symbols to produce traces as shown in Figures 4 and 5. In these displays, the time goes from left to right and the "x" position of the events is given by their time-code. The ergonomist is free to define the shapes of symbols, their colors and their "y" position in the display. He can click on symbols to see their properties. These figures use a display model where the lower level symbols are the dots at the bottom. The middle level symbols are placed around the middle horizontal axis. What concerns the left of the vehicle is placed above this axis and what concerns the right is placed below it. Triangles oriented to the right concern something "frontward" (e.g. "Look ahead"), triangles oriented to the left concern something "backward" (e.g. "Left mirror glance"). The lines are the relations of inference from lower level symbols to higher level symbols. The upper part of the display is used for the highest level symbols which are not concerned by this left/right rule (e.g. "Decision"). The colors in the real display are used to indicate the type of data (i.e., blinker in orange), it makes it much easier to read than in print. When applied, an inference rule adds a new instance of a symbol in the trace everywhere the specified pattern is found. Filters can mask the undesired symbols.

## The Ontology

Parallel to the creation of the inference rules, the ergonomist defines the class of the symbols in an ontology. ABSTRACT provides them with an access to the "Protégé" ontology editor to let them define the semantic and visual properties of each class of symbol. This ontology is exploited by the SPARQL inference engine and by the display functions.



- CO
  - Car_Dynamic
  - Driver_Behavior
    - Driver_Action
    - Driver_Stance
      - Eye
        - Eye_Beside_Left
        - Eye_Beside_Right
          - Eye_Far_Right
          - Eye_Right_Mirror
        - Eye_Center_Mirror
        - Eye_Front
        - Eye_Low
        - Eye_Off
        - Eye_Qual
  - Human_Input
  - Road_Environment
- Activity_Analysis
  - Lane_Change
    - Lane_Change_Delayed
    - Lane_Change_Anticipated
  - Mental_States
    - Decision
    - Surprise

Figure 3: The ontology edited with Protégé.

Figure 3 shows an extract of the top classes of the current ontology. The CO class defines all the Collected Objects existing in the collected trace. One branch of the eye tracker data is expanded to show how the area of interest can be organized. Heritage is supported so that inference rules and visualization settings can be defined at any level of the hierarchy.

The Activity_Analysis class concerns concepts that are inferred from CO. For instance, two types of lane changes could be modeled and can be recognized from specific patterns in the data: lane-change-anticipated and lane-change-delayed, published in (Henning, Georgeon & Krems, 2007). All these concepts are defined on a pragmatic and evolutionist basis as said above. They come from a trade-off between, on one hand, the possibility of inferring them from the data and on the other hand from the ergonomist's quest for explanation. Moreover they can be searched in collaboration between the ergonomist and the subject himself during the self-confrontation interview.

## Examples of Modeling

Figure 4 shows a typical example of a lane-change-delayed schema. In this figure, the "Start-thinking" symbol corresponds to the moment when the driver declares that he starts considering changing lane in the self-confrontation interview. The "Button" is a signal from the experimenter recorded during the course for indexing every lane change, and the "Lane-crossing" is the moment where the left front wheel crosses the lane, manually encoded from the video. All other symbols are automatically inferred from the sensor data.

In this kind of situation the driver is impeded by a vehicle slower than his desired speed. He may check his left mirror several times, then, when he decides to overtake, he accelerates while checking his mirror; he switches his blinker on, starts steering, and crosses the line. In this situation, the decision to perform the lane change can be inferred from the conjunction of both the acceleration and the left mirror glance within a certain lap of time ("Decision" symbol). This symbol can be useful as a predictor of the lane change in this situation, we call it "Decision" within the explicative framework of "operational schemas". The inference is made by a query expressing: "add a new node of type "decision", linked through edges of type "inferred", to pairs of nodes matching the condition: one is of type "accelerate", other is of type "Left mirror glance" and their "time-stamp" properties are comprised within one second. Once defined, this query can be applied to the whole trace for statistical assessment of true detection and false alarm. This driver made 11 lane changes, 5 could be categorized as "delayed" and 2 as "anticipated". The decision is detected before the blinker in the 5 lane-change-delayed, and there is one false detection.

Another example of analysis is shown in figure 5. It is a case study where the driver attempts to change lane but did not see a car arriving from the rear on the left lane. He initiated the maneuver but aborted it at the last moment. The figure shows that the driver did not look at his rear mirror: he was looking at his speedometer instead. He stabilized his speed at a higher value than the car ahead. The "Distance ahead" event at a relative speed of 3.5km/h comes from the front telemeter. From these two pieces of information we can already make the diagnosis that he was intending to overtake. Then he looked at his left mirror almost at the same time as switching his indicator on. At this moment he saw the car coming on the left lane from behind and he aborted the lane change. He made an abrupt steering back to the right, switched off his indicator, and pressed the brake pedal to avoid bumping into the car he was intended to overtake. Finally he started looking again at his left mirror and accelerated again to perform the lane change when the other car had passed him.

The subjective evaluation from the interview indicates that he was very surprised: (90% of the scale), stressed (75%) but the difficulty was not so high: 30%.

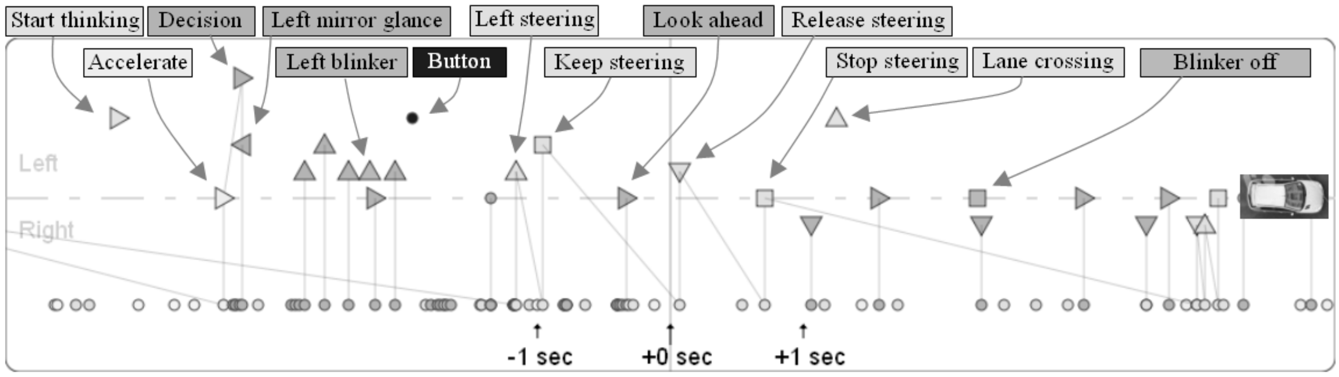


Figure 4: Prototype of a "lane-change-delayed" schema.

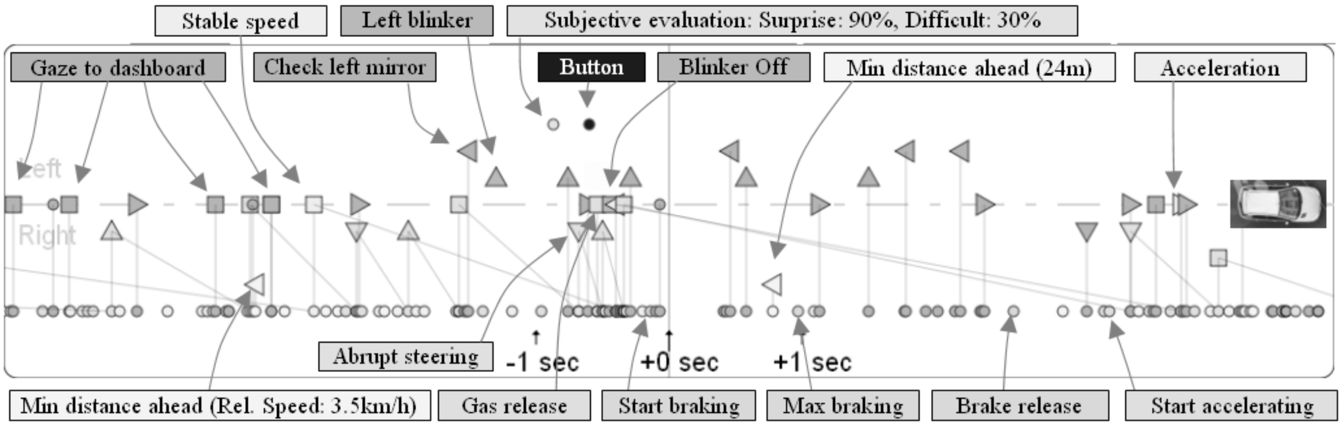


Figure 5: Lane change attempt with mistake of situation awareness.

This example shows how we can explain a mistake by an error of "situation awareness". It also let us propose a threshold value of the variation rate of the steering wheel that can indicate the subjective surprise of the driver. Finally, it shows that more than one second before this mistake, we could automatically make the diagnosis that the driver was intending to make an error from the objective data. Based on this diagnosis, a driving assistance system could warn him.

## Conclusion

A methodology and tool were presented to support the activity analysis and the cognitive modeling of an operator. They are intended to formalize the process of explaining activity by the mental states and processes of the operator. The tool can capitalize on the expertise of the ergonomist through inference rules, ontologies, and visualization settings. The resulting models consist of explicative concepts defined in the ontology and of abstract traces describing patterns of behavior with these concepts. The modeling process lets categories emerge from situations in parallel with their description and provides means to validate them by statistics. The analysis is made from data collected in the field of the studied activity. It consists of a step-by-step process of abstraction driven by the ergonomist in interaction with a software tool. It is shown how this analysis can help design assistance systems for dealing with specific situations.

The obtained models could also be exploited for programming simulations in cognitive simulation frameworks, which would confront them to theoretical constraints on human cognition.

## Acknowledgments

This project was supported by the Humanist Network of Excellence funded by the European Commission. We appreciate the assistance of B. Mathern for software implementation and of F. Ritter for his useful comments on this paper.

## References

Bellet, T. & Tattegrain-Veste, H. (1999). A framework for representing driving knowledge. *International journal of cognitive ergonomics*, 3(1), 37-49.

Dennett, D. C. (1987). *The intentional stance*. Cambridge, MA: MIT Press.

Endsley, M. R. (1995). Toward a theory of situation awareness in dynamic systems. *Human Factors*, 37(1), 32-64.

Ericsson, K. A. & Simon, H. A. (1993). *Protocol analysis: Verbal reports as data*. Cambridge, MA: MIT Press.

Fayyad, U. (1996). From Data Mining to Knowledge Discovery in Database. *AI Magazine*, 17(3), 37-54.

Georgeon, O., Mille, A. & Bellet, T. (2006). Analyzing behavioral data for refining cognitive models of operator. *Philosophies and Methodologies for Knowledge Discovery, Seventeenth international Workshop on Database and Expert Systems Applications* (pp. 588-592). Krakow, Poland: IEEE Computer Society Press.

Gusdial, M., Santos, P. & Badre, A. (1994). Analyzing and visualizing log files: A computational science of usability. *HCI Consortium Workshop*.

Henning, M. J., Georgeon, O. & Krems, J. (2007). The quality of behavioral and environmental indicators used to infer the intention to change lanes. *4th International Driving Symposium on Human Factors in Driver Assessment*. Stevenson, Washington USA.

Hilbert, D. M. & Redmiles, D. F. (2000). Extracting Usability Information from User Interface Events. *ACM Computing Surveys (CSUR)*, 32(4), 384–421.

Laflaquière, J., Sofiane Settouti, L., Prié, Y. & Mille, A. (2006). A trace-based System Framework for Experience Management and Engineering. *Second International Workshop on Experience Management and Engineering (EME 2006)*. Bournemouth, UK.

Miles, M. B. & Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook*. London: Sage Publications.

Minsky, M. (1975). A Framework for Representing Knowledge. The Psychology of Computer Vision. P. H. Winston. New York: Mc Graw-Hill.

Popper, K. (1972). *Objective Knowledge*: Oxford University Press.

Rasmussen, J. (1983). Skills, Rules and knowledge; Signals, Signs and Symbols and other Distinctions in Human Performance Models. *IEEE Transaction on Systems, Man and Cybernetics*, 13(3), 257-266.

Ritter, F. E., Van Rooy, D., Saint Amant, R. & Simpson, K. (2006). Providing user models direct access to interfaces: An exploratory study of a simple interface with implications for HRI and HCI. *IEEE transactions on systems, man, and cybernetics, part a: Systems and Humans*, 36(3), 592-601.

Salvucci, D. D., Boer, E. R. & Liu, A. (2001). Toward an integrated model of driver behavior in a cognitive architecture. *Transportation Research Record*, 1779, 9-16.

Salvucci, D. D. & Lee, F. J. (2003). Simple Cognitive Modeling in a Complex Cognitive Architecture. *Human Factors in Computing Systems*. New York: ACM Press.

Sanderson, P. M. & Fisher, C. A. (1994). Exploratory sequential data analysis: Foundations. *Human -Computer Interaction*, 9, 251-317.

Sanderson, P. M., McNeese, M. D. & Zaff, B. S. (1994). Handling complex real-word data with two cognitive engineering tools: COGENT and MacSHAPA. *Behavior Research Methods, Instruments, and Computers*, 26, 117-124.

Schneider, W. & Shiffrin, R. M. (1977). Controlled and automatic human information processing: I. Detection, search and attention. *Psychological Review*, 84, pp 1-66.

Wilson, J. R. & Corlett, E. N. (2005). *Evaluation of Human Work*: CRC Press.

Wittgenstein, L. (1953). *Philosophical Investigations*.

# Comparing Modeling Idioms in ACT-R and Soar

**Randolph M. Jones (rjones@soartech.com)**
Soar Technology, 44 Burleigh Street
Waterville, ME 04901 USA

**Christian Lebiere (cl@cmu.edu)**
Psychology Department, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213 USA

**Jacob A. Crossman (jcrossman@soartech.com)**
Soar Technology, 3600 Green Court Suite 600
Ann Arbor, MI 48105 USA

## Abstract

This paper examines some of the constraints on cognition assumed and imposed by the ACT-R and Soar cognitive architectures. In particular, we study how these constraints either encourage or require particular types of "modeling idioms" in the form of programming patterns that commonly appear in implemented models. Because of the nature of the mapping of the architectures to human cognition, each modeling idiom translates relatively directly into changes in model behavior data, such as decision timing, memory access, and error rates. Our analysis notes that both architectures have sometimes adopted extreme and opposed constraints, where the human architecture most likely relies on some mixed or more moderate set of constraints.

## Implications of Architectural Constraints on Cognitive Modeling

Experienced cognitive modelers are well aware that "the devil is in the details," particularly when it comes to fine-grained models of deliberative behavior. Changes in the particular reasoning path chosen to model a task can manifest themselves as differences in task timing, type and rates of errors, and overall strategy differences. Cognitive architectures such as Soar (Laird, Rosenbloom, & Newell, 1987) and ACT-R (Anderson & Lebiere, 1998) implement constraining assumptions that encourage and sometimes require particular types of modeling idioms or patterns that in turn impact the data the model produces.

This paper compares some of the modeling idioms (perhaps alternatively described as programming patterns) that commonly appear in Soar and ACT-R models of decision making. We have come across many of these comparisons while developing HLSR, a language for building models that can compile both to ACT-R and to Soar (Jones et al., 2006). It is interesting that, although ACT-R and Soar are in some ways "close cousins", there are significant differences in how some types of low-level reasoning tasks must be modeled, although these differences are not necessarily obvious without getting into model details. In many cases, each architecture has adopted constraints that are diametrically opposed to each other, where an alternative architecture might encourage a more moderate or mixed approach.

Constraints in cognitive architectures often manifest themselves as computational bottlenecks that are inspired by assumed limitations on human processing. In ACT-R 6.0 (Anderson et al, 2004), there is a "cognitive bottleneck" that allows only one production rule instantiation to fire at a time (even if multiple rule instantiations currently match), and there are information bottlenecks that allow only one chunk per architectural module to be accessible for production matching through that module's buffer. In particular, that means that only one goal can be active at any time and that only one chunk can be retrieved from long-term declarative memory at a time. These limitations imply that complex logical decisions must be implemented via sequences of retrievals and actions, which in turn impacts the timing of retrieval/decision sequences.

In Soar 8.6, multiple rule instantiations can fire at once, and access to declarative memory is essentially unlimited. However, Soar imposes a cognitive bottleneck by allowing only one "operator object" to be selected at a time. Additionally, all operator selections must occur through Soar's preference/decision mechanism. Finally, in order to maintain logical self-consistency, operator objects can become automatically unselected if their logical preconditions become unmatched. This latter effect implies that individual Soar operators can usually not implement long sequences of actions. Such sequences must instead be implemented by series of operators, which can have an impact on the timing and granularity of decision sequences.

These combinations of imposed constraints and bottlenecks dictate some of the types of modeling idioms that programmers typically use when implementing decision-making processes in each architecture. The remainder of this paper provides examples contrasting four of these types of idioms.

## Sequences of Decisions and Actions

ACT-R production rules are allowed to execute multiple actions at a time, but in a limited fashion. They can make changes to the contents of each architectural buffer (for this paper's purposes, we will concern ourselves only with the goal and retrieval buffers). Consider the following example rule from a Towers-of-Hanoi model, which makes a change to the chunk in the goal buffer while simultaneously initiating a new retrieval to the retrieval buffer.

```
(p find-next-tower
   =goal>
      isa move-tower
      disk =disk
      peg =peg
      state nil
==>
 !output! "Retrieving disk smaller than ~S" =disk
   +retrieval>
      isa next-smallest-disk
      disk =disk
   =goal>
      state next)
```

Individual Soar rules can also implement multiple actions simultaneously. However, Soar rules are allowed to test complex logical patterns with more flexibility than in ACT-R, and multiple Soar rule instantiations can fire at the same time. As a result, a common Soar modeling idiom is to tease apart individual types of actions into separate rules. This allows the development of more adaptive code that does not introduce "artificial" conjunctions of conditions just because the modeler wants multiple things to happen at once. For the above example, the Soar idiom would typically divide into two separate rules, as shown below. These rules access a "current-retrieval" object that mimics ACT-R's retrieval buffer (there is no architectural requirement for such a buffer in Soar models). The first rule initiates the retrieval, while the second makes the change to the goal state. Notice that the first rule can fire even if some other set of conditions want to change the goal state. The second rule can fire whenever the appropriate information is in the retrieval buffer, regardless of which process might have initiated that retrieval.

```
sp {find-next-tower*apply*retrieve
   (state <s> ^operator <o> ^current-goal <g>
              ^next-smallest-disk <nsd>)
   (<o> ^name find-next-tower
        ^goal <g> ^disk <disk>)
   (<nsd> ^disk <disk>)
  -(<s> ^current-retrieval <nsd>)
   (<disk> ^name <dname>)
-->
   (write (crlf) |Retrieving disk smaller than |
                 <dname>)
   (<s> ^change-value <cv>)
   (<cv> ^id <s> ^att current-retrieval
         ^value <nsd>)}

sp {find-next-tower*apply*change-state
   (state <s> ^operator <o> ^current-goal <g>
              ^next-smallest-disk <nsd>)
   (<o> ^name find-next-tower
        ^goal <g> ^disk <disk>)
   (<nsd> ^disk <disk>)
   (<s> ^current-retrieval <nsd>)
  -(<g> ^state next)
-->
   (write (crlf) |Moving to state "next"|)
   (<s> ^change-value <cv>)
   (<cv> ^id <g> ^att state ^value next)}
```

In any rule-based system, combinations of conditional actions must either be implemented by a combinatorial number of rules covering the space of possible condition combinations or by a set of rules that reason through the combination of conditions. One important difference is that Soar models can sometimes execute such rule combinations in parallel where ACT-R must execute them in sequence. Either choice has an impact on the timing of decision making, as well as the types of errors and adaptivity that the model might produce.

The standard Soar idiom for implementing multiple actions can also encounter problems that impact timing, errors, and adaptivity. The typical approach in Soar would have a single operator object that has associated with it multiple rules that implement the conditional logic for various combinations of actions. However, different sequences of action may require rule-firing sequences of different lengths, some of which can cause the operator object to be deselected automatically (this is in fact the case in the above example, where there are additional rules that match against the "change-value" pattern). This can introduce "race conditions" where one stream of decision making does not get a chance to complete because another stream has deselected the operator. Consider the following, slightly more complicated, ACT-R rule, which implements three separate actions simultaneously.

```
(p clear-disk
   =goal>
      isa move-disk
      disk =disk
      peg =peg
      state peg
   =retrieval>
      isa disk-on-peg
      disk =disk
      peg =on
    - peg =peg
==>
   !output! "Subgoaling clear-disk with disk ~S on
peg ~S to peg ~S parent ~S" =disk =on =peg =goal
   +goal>
      isa clear-disk
      disk =disk
      current =on
      peg =peg
      parent =goal
   +retrieval>
      isa next-smallest-disk
      disk =disk
   =goal>
      state =retrieval)
```

Attempting to implement this with a single Soar operator would almost certainly lead to race conditions that would cause the model to break. The standard Soar idiom to respond to such a situation is to break these simultaneous conditional actions into individual operators, so they cannot race with each other. But because Soar only allows one operator at a time, this imposes sequential processing, where the initial desire was to implement a set of parallel actions. Again, a combination of constraints within the architecture directly leads to meaningful changes in the data that models will produce.

## Sequential vs. Parallel Memory Retrieval

In ACT-R, the combined bottlenecks for individual rule firing and memory access through a retrieval buffer produce a common idiom for accessing and processing elements from long-term declarative memory. Before any memory object can be accessed, it must first be fetched into the retrieval buffer. Thus, the idiom is to include one rule (or more) to initiate the retrieval, and one rule (or more) to "harvest" the retrieved item, processing it in the desired way. Below are two example rules, again from a Towers-of-Hanoi model. These rules process a "clear-disk" goal by creating a subgoal to move the "next smaller tower" off of the current disk. In order to accomplish this, the ACT-R model must first find a peg to move the subgoal tower to. This is accomplished by searching for a spare peg and fetching it into the retrieval buffer, where it then becomes available to provide information for the new subgoal.

```
(p find-spare-peg
   =goal>
      isa clear-disk
      disk =disk
      current =on
      peg =peg
      state nil
   =retrieval>
      isa next-smallest-disk disk =disk next =next
==>
   !output! "Next smaller disk to ~S is ~S and
retrieving peg other than ~s and ~S" =disk =next
=on =peg
   =goal>
      disk =next
      state other
   +retrieval>
      isa spare-peg
      current =on
      destination =peg)

(p clear-tower
   =goal>
      isa clear-disk
      disk =disk
      current =on
      peg =peg
      state other
      parent =parent
   =retrieval>
      isa spare-peg
      current =on
      destination =peg
      other =other
==>
   !output! "Subgoaling move-tower with disk ~S
peg ~S parent ~S" =disk =peg =parent
   +goal>
      isa move-tower
      disk =disk
      peg =other
      parent =parent)
```

As in our first example above, a Soar model could be built similarly by mimicking the retrieval buffer within Soar's working memory. However, the more typical idiom in Soar would take advantage of Soar's unfettered access to all elements in declarative memory. In such a Soar model, a single rule can perform a complex conditional query and use the information to create the desired subgoal, without requiring the extra step of going through a retrieval buffer.

```
sp {clear-disk*propose*create-subgoal*move-tower
  (state <s> ^current-goal <g> ^disk <disk>
             ^next-smallest-disk <nsd>
             ^spare-peg <sp>)
  (<g> ^name clear-disk ^disk <disk>
       ^current <on> ^peg <peg> ^parent <parent>)
  (<nsd> ^disk <disk> ^next <next>)
  (<sp> ^current <on> ^destination <peg>
        ^other <other>)
  (<next> ^name <dname>)
  (<peg> ^name <pname>)
  (<other> ^name <oname>)
-->
   (write (crlf) |Create new subgoal move-tower
disk | <dname> | to peg | <oname> | to replace
clear-disk from peg | <pname>)
  (<s> ^operator <o>)
  (<o> ^name create-subgoal ^goal <ng>)
  (<ng> ^name move-tower ^disk <next> ^peg <other>
        ^parent <parent> ^clear-parent *yes*)}
```

In general, the lack of a retrieval buffer in Soar allows Soar models to be written in a more compact way with more opportunities for the reuse of individual operators and rules. The primary potential downside is that many Soar models do not take the memory-retrieval bottleneck seriously, as ACT-R models must. It is possible to find Soar models that have literally hundreds of accessible items in their declarative memory at one time, although this is generally truer for "applied" Soar systems than it is for serious cognitive models built in Soar. There are a number of Soar-based cognitive models that self-impose more declarative-memory constraints than the architecture itself requires (e.g., Wray & Chong, 2005; Young & Lewis, 1999). It is also worth noting that Soar models with large declarative memories are usually compensating for the fact that they do not use Soar's built-in learning mechanism. Models that use learning usually use the learned rules for declarative access, rather than relying on huge declarative memories. The situation is similar in ACT-R, except that ACT-R's constraints are more forceful in the sense that it is more difficult to "cheat" in the ways that you sometimes can when using Soar.

There are some senses in which loosely limited declarative memory access may be plausible, but other senses in which it certainly is not. On the other hand, the restriction in ACT-R to have a single retrieval buffer that can hold only a single chunk is probably overly restrictive in some cases. In the example above, it would seem reasonable that a model of even a slightly experienced Towers-of-Hanoi practitioner should just "know" what the third peg is. However, under the current architectural constraints, that is only possible by encoding in the production rules all the combinatorial possibilities of origin and destination pegs (admittedly a limited number with only three pegs, but still too large to be considered elegant or even plausible). It would seem plausible to have a small number of frequently and/or recently used chunks directly accessible from some sort of working memory, but that is currently only possible by having the modeler pack a given buffer with the content of those chunks, a practice that often leads to brittle and/or implausible models. Both assumptions lead to interesting models that are qualitatively different, but perhaps plausible and implausible in their own ways.

The main reason for the differing idioms in this case is that Soar implements its "retrieval process" through rules and rule conditions that can encode arbitrarily complex conjunctions of declarative memory elements. Retrieval in ACT-R is instead a sequential process that takes a set of cues as input and returns a single set of elements to fill the retrieval buffer. Both of these approaches to memory access manifest themselves in modeling idioms that predict different types of behavior. In this case, it is interesting to note that each architecture adopts a rather extreme approach to memory access, where a more accurate model of the human architecture would probably be somewhere in between the two. It seems unlikely that human memory is limited to holding accessible a single chunk at a time (e.g. Miller, 1956), but equally unlikely that human memory is capable of unfettered retrieval of arbitrarily complex conjunctions.

## Partial Matching vs. Preferences for Conflict Resolution

One of the more unique aspects of the Soar architecture involves its mechanisms for supporting symbolic rule-based preferences for conflict resolution. In Soar, all conflict resolution centers around deciding which operator object to select next, and this is generally accomplished by preference rules that propose binary comparisons between the various candidates (O1 is better than O2, O2 is just as good as O3, etc.). The rule-based preference mechanism is necessary because there is no architectural conflict resolution mechanism (other than the architectural component that makes a selection based on the symbolic preferences).

In ACT-R, conflict resolution centers around two types of choices: which rule instantiation should fire next and which chunk should be retrieved from declarative memory into the retrieval buffer. ACT-R includes architectural mechanisms to support both of these modes of conflict resolution. Both mechanisms are similar, being grounded in subsymbolic concepts (utility and activation, respectively) and including similar restrictions such as learning constraints. Thus, the idiom in ACT-R modeling is to create numerically oriented "preferences" that are assumed to reflect some sort of learning from prior experience. The Soar idiom is to encode the preferences as (sometimes complex) sets of logical ordering constraints (which are also assumed to be learned). The result is that we see some significant differences between ACT-R and Soar in conflict-resolution modeling, depending on the type of model. For purely symbolic models, ACT-R must include rule conditions that encode the combinations of constraints that could be represented as individual preference rules in a Soar model. However, ACT-R also provides a subsymbolic partial-matching idiom that is not directly available to Soar modelers. Similarly, the most recent versions of Soar have introduced the ability to specify numeric and probabilistic preferences, so there are some new opportunities to explore non-symbolic preference idioms in Soar, as well.

Following is a simple example of the relatively compact representation of preferences that can be encoded into a Soar model. In this example, the model is to select either an "eat" operator or a "drink" operator, but it prefers to eat before drinking.

```
sp {eat*propose
   (state <s> ^agent <a>)
   (<a> ^hungry yes)
-->
   (<s> ^operator <o> + =)
   (<o> ^name eat ^agent <a>)}

sp {drink*propose
   (state <s> ^agent <a>)
   (<a> ^thirsty yes)
-->
   (<s> ^operator <o> + =)
   (<o> ^name drink ^agent <a>)}

sp {prefer*eat*over*drink
   (state <s> ^operator <o1> + <o2> +)
   (<o1> ^name eat)
   (<o2> ^name drink)
-->
   (<s> ^operator <o1> > <o2>)}
```

Note that, if Soar did not include its preference-based conflict-resolution mechanism, a modeler would be forced to encode the semantics of the various preferences into the operator proposal rules themselves. For example, in the above code, we would have to change the drink proposal rule to the following:

```
sp {drink*propose
   (state <s> ^agent <a>)
   (<a> ^thirsty yes -^hungry yes)
-->
   (<s> ^operator <o> + =)
   (<o> ^name drink ^agent <a>)}
```

A potential problem with this approach to conflict resolution is that it will lead to a combinatorial explosion of conditions for complex preferences between multiple potential choices. In a purely symbolic ACT-R model, the approach would be similar, but with an added constraint. Because only one item can be in the retrieval buffer at a time, an ACT-R model must test the different logical conditions sequentially and either make each test depend on the results of the previous one(s) or accumulate the results in the goal (or some other) buffer for some final decision. In contrast, Soar proposals can each check their combinations of conditions with less restricted access to declarative memory. Thus the symbolic ACT-R approach might look as follows:

```
(p check-hungry
   =goal>
     isa agent
     name =name
     state nil
==>
   +retrieval>
     isa property
     agent =name
     attribute hungry
     value yes
   =goal>
      state hungry)

(p check-thirsty
   =goal>
      isa agent
      name =name
      state hungry
   =retrieval>
      isa error
==>
   +retrieval>
      isa property
      agent =name
      attribute thirsty
      value yes
   =goal>
      state thirsty)
```

In the above example, the first rule's retrieval will succeed if and only if there is a "hungry" property with a value of "yes" in declarative memory. If that retrieval fails, the check-thirsty rule will look for a "thirsty" property with a value of "yes". However, ACT-R modelers are not restricted to doing symbolic conflict resolution. For choices like this, ACT-R also supports similarity-based partial matching for retrieval. It is possible to define a "similarity relationship" between different attribute values, which will in turn influence how the retrieval process executes. Using ACT-R's partial-matching mechanism, we can rëimplement the above example as follows:

```
(setsimilarities (hungry thirsty -0.5))

(p choose-action
   =goal>
      isa agent
      name =name
      state nil
==>
   +retrieval>
      isa property
      agent =name
      attribute hungry
      value yes
   =goal>
      state unknown)
```

In this case, the attribute values "hungry" and "thirsty" are set to be relatively dissimilar to each other. But the fact that they are defined with any similarity measure at all indicates that they are candidates to be substituted for each other in any partial-matching retrieval. Thus, the choose-action rule initiates a search for "hungry yes", and it will retrieve a perfectly matching chunk if one exists in declarative memory. But if there is no perfectly matching chunk, the retrieval process will instead look for the closest partial match. In this case, a chunk representing "thirsty yes" would be the next best match. Based on whichever chunk happens to get retrieved, the program can then choose to "eat" or "drink", as appropriate. However, if the set of options is so complex or heterogeneous that checking the options cannot be reduced to a single retrieval, then an outer loop must be explicitly maintained to access the various options sequentially, where in Soar they could be combined into a single complex conditional rule. The problem in ACT-R is that if each option involves checking some additional condition (such as perceptual or memory information), then the utility preferences are not helpful because they would attempt to check the same condition over and over again. Either an explicit round robin check of the various conditions has to be set up symbolically in the production conditions or learning of the utilities can be used to iterate through the options by having the failure of each option temporarily depress the utility of the production selecting that option (Lebiere et al., in press).

## Exhaustive Processing and Search

The final pattern we investigate involves performing exhaustive iterative actions on a set of similar object or chunk types. For example, imagine that declarative memory contains a number of message objects, each with a text attribute. We would like to build a model that iterates through all of the messages and prints out the text value of each one. In a Soar program this can be done relatively simply because an individual operator application rule can match against multiple objects at a time, and each matching instantiation will execute simultaneously. For example, the following Soar rule simultaneously finds all "unhandled" message objects in declarative memory, prints their messages, and marks the message objects as "handled".

```
sp {handle-messages*apply
  (state <s> ^operator <o> ^message <m>)
  (<o> ^name handle-messages)
  (<m> ^text <t> ^message-handled false)
-->
  (write (crlf) | Message is: | <t>)
  (<m> ^message-handled false - true +)}
```

In contrast, ACT-R is restricted to matching one object at a time through the retrieval buffer. In older versions of ACT-R, this would be accomplished by iterating over a sequence of retrievals and harvests, tagging each chunk as it is processed. This approach also requires an additional rule that detects when the retrieval process has failed to find any further matching candidates for processing.

```
(p find-message-to-handle
   =goal>
      isa handle-message
      state nil
==>
   =goal>
      state harvest
   +retrieval>
      isa message
      handled false)

(p handle-message
   =goal>
      isa handle-message
      state harvest
   =retrieval>
      isa message
      text =text
      handled false
==>
  !output! "~S" =text
   =goal>
      state nil
   =retrieval>
      handled true)

(p finish-handle-message
   =goal>
      isa handle-message
      state harvest
   =retrieval>
      isa ERROR
      condition Failure
==>
   !output! "Done handling messages"
   =goal>
      state finished)
```

However, the most recent version of ACT-R does not allow non-monotonic changes (such as tagging) to chunks in the retrieval buffer, so new idioms are developing that rely on the subsymbolic processing of the retrieval mechanism.

These new idioms encounter additional confounding factors. A major problem is that the dynamics of the activation calculus, and in particular the learning of the base level to reflect frequency and recency of access, conspire against that iterative process. Recently accessed chunks become more active while chunks that have not been accessed decay and become less active, leading to the opposite dynamics of the iteration desired, namely a winner-take-all tendency to retrieve the same candidate(s) again and again. One typical idiom to get around this problem is to alter subsymbolic processing parameters such as noise, in order to "break out" of bad retrieval sequences. However, this is often only partially successful in moving the iteration along.

Another example of iteration comes again from the Towers of Hanoi. In this problem, it is useful to compute which disk is currently at the top of a particular peg. In a Soar model, the encoded logic is along the lines of "find a disk on the peg for which all other disks on the peg have a lower position". Although this gets a bit messy, the logic can be encoded in the conditions of a single Soar rule. In contrast, an ACT-R model must implement this logic using a sequential loop or by clever configuration of the partial-matching mechanism. Although the sequential iteration can be implemented in a relatively straightforward fashion, it again runs into the stumbling block that ACT-R prefers to retrieve the same disk repeatedly, instead of iterating through all of the disks on the peg.

Note that it is also possible to implement sequential iteration using operators in Soar. Soar does not include the restriction against altering declarative memory items, so the typical Soar idiom in such situations is to tag each object as it is processed in sequence. However, depending on the situation, the alternative idiom in Soar is to use a single rule to process everything at once. It is certainly a valid question, however, whether Soar *ought* to make it so easy to do this type of computation. It could be argued persuasively that humans in general cannot perform this type of exhaustive, instantaneous, massively parallel processing, and so it is a mistake for Soar to allow and even encourage this type of solution. On the other hand, there are certainly some types of massively parallel processing occurring in the human architecture. So once again, we are faced with two architectures that embody extreme constraints, where the truth is probably a combination or compromise.

It should also be noted that there are particular problems of this type that also *require* a sequential solution approach in Soar. For example, although a Soar program can easily use one rule to operate on a whole set of objects simultaneously, it currently has no way to *count* the number of objects in that set. For the task of counting the number of elements in a set, both ACT-R and Soar demand sequentially implemented solutions.

## Conclusion

We have examined four classes of modeling idioms that arise relatively directly from the combination of assumed constraints on cognitive processing imposed by the ACT-R and Soar cognitive architectures. We hope that these examples provide a more detailed feeling to the modeling community about what some of the differences and similarities are between the architectures, particularly when it gets to the nitty-gritty of building detailed models. From a cognitive modeling perspective, this is not just an exercise in examining computationally equivalent modeling approaches. Each of the idioms implies measurable differences in the type of data the models will produce. We have also observed that the constraints and bottlenecks assumed by each architecture tend be rather extreme and often opposed to each other. We join others in recommending future work that includes finding more intermediate constraints on the cognitive architecture, which should translate to some variation in the common modeling idioms, and in turn to cognitive models that produce better matches to human data.

## References

Anderson, J., & Lebiere, C. (1998). *The Atomic Components of Thought*. Mahwah, NJ: Lawrence Erlbaum.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y . (2004). An integrated theory of the mind. *Psychological Review 111,* (4). 1036-1060.

Jones, R. M., Crossman, J. A., Lebiere, C., & Best, B. J. (2006). An abstract language for cognitive modeling. *Proceedings of the Seventh International Conference on Cognitive Modeling*. Trieste, Italy.

Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence* 33(1): 1-64.

Lebiere, C., Archer, R., Best, B., & Schunk, D. (in press). Modeling pilot performance with an integrated task network and cognitive architecture approach. In Foyle, D. & Hooey, B. (Eds.) *Human Performance Modeling in Aviation.* Mahwah, NJ: Lawrence Erlbaum.

Miller, G. A. (1956). The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. *Psychological Review, 63*, 81-97.

Wray, R., & Chong, R., (2005). Comparing cognitive models and human behavior representations: Computational tools for expressing human behavior. *Proceedings of the Infotech@Aerospace 2005 Conference*, Arlington, VA. American Institute of Aeronautics and Astronautics.

Young, R. M., & Lewis, R. L. (1999). The Soar cognitive architecture and human working memory (1999). In A. Miyake & P. Shah (Eds.), *Models of Working Memory: Mechanisms of Active Maintenance and Executive Control,* 224-256. Cambridge University Press.

## Acknowledgments

This work was supported in part by contract N00014-05-C-0245 from the Office of Naval Research. Many thanks to Bob Wray for his helpful comments on an earlier draft.

# Dynamic Spatial Reasoning Capability in a Graphical Interface Evaluation Tool

**Michael Matessa (mmatessa@alionscience.com)**
**Rick Archer (rarcher@alionscience.com)**
**Rebecca Mui (rmui@alionscience.com)**
Alion Science and Technology
Micro Analysis & Design Operation
1789 South Braddock Avenue, Suite 400
Pittsburgh, PA 15218 USA

## Abstract

This paper describes dynamic and spatial reasoning enhancements to the Graph-Based Interface Language tool (GRBIL), which creates ACT-R models by demonstration. A new ability for users to create monitors  enables procedures to be dynamically triggered. A new integration of ACT-R with a diagrammatic reasoning theory allows ACT-R to perform spatial reasoning. Capabilities of the tool are demonstrated in a robotic control task.

## Introduction

Cognitive modeling can add value to interface evaluation, but it is currently not very practical in terms of amount of expertise and time. Recent efforts to allow easier construction of cognitive models have utilized high level languages (Howes, Lewis, Vera, & Richardson, 2005; Salvucci & Lee, 2003; St. Amant & Ritter, 2005) and modeling by demonstration (Archer, Lebiere, & Warwick, 2005; John, Prevas, Salvucci, & Koedinger, 2004). Modeling by demonstration is an easy way to create a sequence of procedural steps, but with dynamic interfaces, an additional method is needed to designate the condition in which the procedure should be applied. In addition, many real-world tasks require aspects of spatial reasoning as well as dynamic interaction. Examples of interfaces with these features include radar operator interfaces and interfaces for controlling robotic vehicles. This paper describes dynamic and spatial reasoning enhancements to the GRaph-Based Interface Language tool (GRBIL – Archer, Lebiere, & Warwick, 2005), which creates ACT-R models by demonstration.  A new ability for users to create monitors enables procedures to be dynamically triggered. A new integration of ACT-R with a diagrammatic reasoning theory allows ACT-R to perform spatial reasoning.

## ACT-R

The ACT-R cognitive architecture has shown an increasing ability to account for human visual information processing. Early ACT-R accounts of visual processing made a distinction between pre-attentive features available to vision and objects available after a shift of attention (Anderson, Matessa, & Lebiere, 1997). More recent work uses brain imaging as evidence for an Imaginal module where information can be visualized and manipulated (Anderson et al., 2004). However, current ACT-R theory is limited in the visual objects that can be recognized (text, lines, rectangles, and ovals) and does not provide primitive operators for getting attribute information such as length, relational information such as what objects are inside or next to other objects, inferred information such as projected intersections, and transformations on objects such as rotation. One solution to these limitations is to integrate a spatial reasoning theory into ACT-R. For interface evaluation, diagrams are a useful level of representation for reasoning. Larkin and Simon (1987) point out that diagrams automatically support a large number of perceptual inferences which are extremely easy for humans but that "...diagrams are useful only to those who know the appropriate computational processes for taking advantage of them."

## DRS

Chandrasekaran et al. (2004) developed the diagrammatic reasoning theory DRS, which consists of a basic set of primitive objects, information gathering capabilities called Perceptual Routines, and creation/modification operations called Action Routines. A diagrammatic object can be one of three types: point, curve, and region. Figure 1 shows examples of object types and how they can be composed hierarchically. Perceptual Routines can be qualitative (e.g., LeftOf, On, InsideOf), quantitative (e.g., Distance, Angle, Length), or related to object recognition (e.g., ScanPath, Intersect). Action Routines can create or modify objects (e.g., Translate, Rotate, PathFinder).

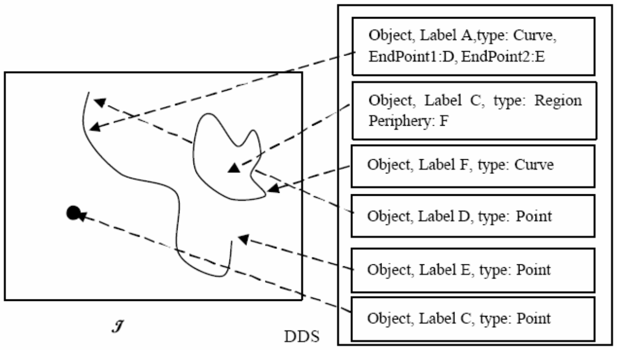

Object, Label A,type: Curve, EndPoint1:D, EndPoint2:E
Object, Label C, type: Region Periphery: F
Object, Label F, type: Curve
Object, Label D, type: Point
Object, Label E, type: Point
Object, Label C, type: Point
𝒥
DDS

Figure 1: DRS object types

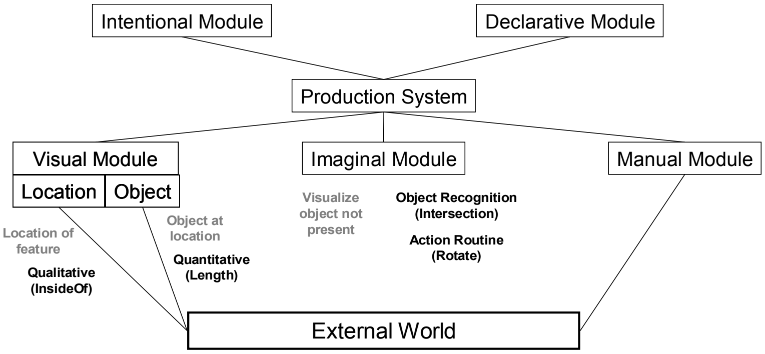


Figure 2: DRS routines in the ACT-R architecture

Chandrasekaran and Kurup (2007) incorporated DRS into Soar and used its chunking mechanism to learn efficient abstractions of observed visual material. The abstractions led to beneficial simplification in the case of memory of a specific path in a complex environment, and also led to incorrect but psychologically realistic reasoning in the case of determining whether Reno is east or west of San Diego (it is actually west). Lathrop and Laird (2006) incorporated a diagrammatic reasoning theory motivated by DRS into the Soar cognitive architecture and found an increase in functional capability and increase in efficiency of code as measured by number of decision cycles.

## Combining DRS and ACT-R

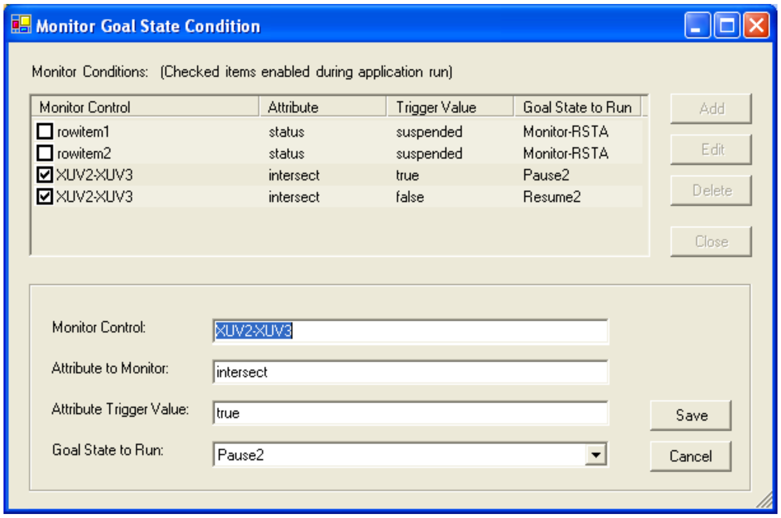
Matessa and Brockett (2007) describe how the perceptual capabilities of ACT-R can be enhanced by the addition of DRS. Figure 2 shows where DRS routine types can naturally fit in the ACT-R architecture. One unnatural fit would be for qualitative routines to put all relational information (e.g., Region4 is above Region5) in the declarative module, which would cause a combinatorial explosion of facts. More naturally, qualitative routines can constrain locations returned by the visual module. For example, DRS could return the location of some object InsideOf a particular object. Quantitative routines can associate information with visual objects returned by the visual module. For example, DRS could associate the Length of a line with the returned visual object. Object recognition routines can return objects that are either literally in the display or implied by the display. For example, DRS could infer the projected line from a specified object to the nearest Intersecting object and return it to the imaginal module. Action routines can create or modify objects stored in the imaginal module. For example, DRS could Rotate an object in the imaginal module 45 degrees clockwise. To demonstrate these concepts, a model of maze navigation was created where the decision to move ahead or turn was based on the length of a projected line to the nearest intersecting wall. Without the integrated DRS, an ACT-R model would be forced to do maze navigation using mental arithmetic on coordinates instead of using more natural representations of intersection and line length.

In order to use DRS-enhanced ACT-R models to evaluate dynamic interfaces, the code from Matessa and Brockett (2007) was integrated into the GRBIL evaluation tool.

## GRBIL

The goal of developing the GRaph-Based Interface Language tool (GRBIL) is to allow developers to easily design and evaluate system interfaces. The system allows the user to graphically define a system interface, demonstrate a set of procedures for using the interface, and automatically generate an ACT-R model of an interface operator, providing a time-stamped series of events and potential errors. In addition, the system allows the incorporation of dynamic models of the external world using a task network modeling environment named IMPRINT (IMProved Research INTegration Tool – Archer & Adkins, 1999). This enables the evaluation of interfaces that involve continually changing environments. Multiple IMPRINT tasks can run independently, and with multiple machines, multiple ACT-R agents can interact with IMPRINT in a common environment.

The first step in constructing an interface in GRBIL is to design the physical layout of the interface. This involves choosing the windows, subwindows, and interface controls that comprise the interface. This is done in a similar fashion to many modern interface layout applications using WYSIWYG drag and drop functionality. Once a control is added to a window, attributes such as size and background image may then be customized further from a menu. The second step in designing an interface is to provide a description of what actions each control will be capable of and what the desired effect of each action will be. This is done for each control in GRBIL via an "Event Actions" menu for each control. Using this process of adding interface windows, placing controls on those interfaces and then describing the effects of using those controls on the state of the interface, a GRBIL user can describe the functionality of an entire user interface.



Monitor Conditions: (Checked items enabled during application run)

| Monitor Control | Attribute | Trigger Value | Goal State to Run |
|---|---|---|---|
| ☐ rowitem1 | status | suspended | Monitor-RSTA |
| ☐ rowitem2 | status | suspended | Monitor-RSTA |
| ☑ XUV2-XUV3 | intersect | true | Pause2 |
| ☑ XUV2-XUV3 | intersect | false | Resume2 |

Monitor Control: XUV2-XUV3

Attribute to Monitor: intersect

Attribute Trigger Value: true

Goal State to Run: Pause2

Figure 3: GRBIL monitor used to define a condition of procedure execution

The next step in setting up an interface for evaluation is to define the procedures that a user of the system might wish to perform. This is accomplished by demonstrating a series of actions (e.g., button clicks, object movement). The recorded series of actions is referred to as a "goal state" procedure in the GRBIL terminology. Any number of goal state procedures can be demonstrated and saved for later execution by an ACT-R model. In the model, action steps are represented as a list that is retrieved from declarative memory, and so can be used to predict errors that are dependent on sequence length and positional confusion (cf. Anderson & Matessa, 1997; Matessa & Polson, 2006). Model tracing is used so that potential errors are only noted in the output while the correct retrievals are actually made so the model runs correctly.

Once goal state procedures are defined, monitors can be created that designate the condition in which the procedure should apply. There are two types of monitors: object monitors that check the status of an interface object and spatial monitors that use DRS to check the status of graphical objects. Figure 3 shows the interface used to create monitors. A monitor control is either the name of an interface object, name of a graphical object or names o multiple graphical-objects. An attribute to monitor is an attribute of an interface object defined in GRBIL or a DRS routine for a graphical object. An attribute trigger value must match the value of an attribute before the given goal state procedure can be executed. Any number of monitors can be demonstrated and saved for later execution by an ACT-R model

At run time, goal state procedures and monitors are chosen and ordered. An ACT-R model is created that performs executes the procedures in the given order. The action steps of a goal state procedure are performed without interruption. Between goal state procedures, the model evaluates the next unmatched monitor, using necessary shifts of attention. If the monitor matches, the goal state procedure for that monitor is executed.

Output from the model run includes a time-stamped series of events and possible memory errors (failures to retrieve or the retrieval of similar but incorrect chunks).

## Robotic Interface Test Case

In order to test the capabilities of ACT-R models generated with GRBIL, an interface for an unmanned vehicle Operator Control Unit (OCU) was selected (Figure 4). The OCU is used by operators to control unmanned vehicles in the field. Operators set up plans for the vehicles which are then executed independently. Operators are also responsible for monitoring the status of the vehicles. The interface for the OCU is quite complex, with several modes of operation and control menus. In our implementation, ACT-R performed the actions of operator agents and IMPRINT performed the actions of unmanned vehicle agents. Multiple IMPRINT vehicle agents can run independently, and with multiple machines, multiple ACT-R operator agents can interact with the IMPRINT vehicle agents in a common environment. The ACT-R operators are able to monitor specific attribute values in the environment (such as text describing the status of a vehicle) and use DRS spatial reasoning to detect more general conditions such as projected vehicle intersection (Figure 5). For a particular set of procedures, interacting models of multiple operators (one setting up and initiating vehicles, one monitoring) predict improved performance over a model of a single operator. This is a result of the ability of the purely monitoring operator to react at the same time the single operator would be busy with a procedure.

In order to validate the model's predictions of errors, latencies, and other performance measures, the project team has begun to collect performance data from human operators performing interface tasks such as mission planning and execution on the actual Robotic OCU. Fleetwood et al. (2006) report that timing predictions of tasks not involving spatial reasoning generally match preliminary data.

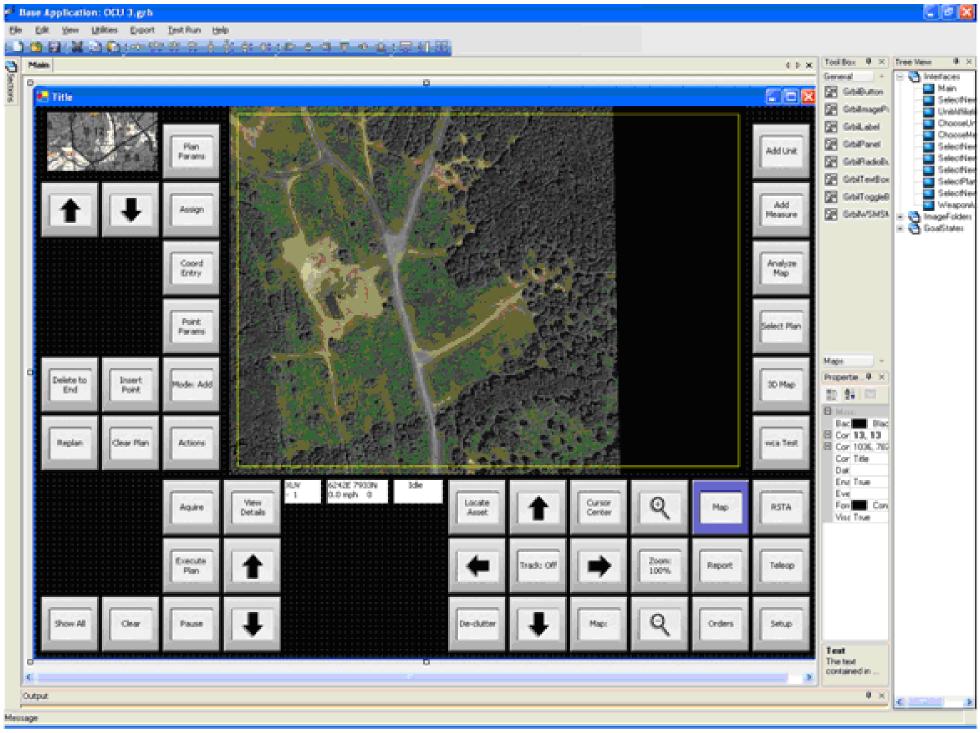


Figure 4: GRBIL tool viewing the Operator Control Unit interface

## Related Work

Tools that enable modeling by demonstration, such as CogTool (John et al., 2004), solve some affordability problems. However, they offer no standard solution for spatial reasoning or integration with dynamic environments. As of this writing, visual processing in CogTool is limited to scripted attention movement to interface objects. The environment can not be monitored to detect a change. CogTool simulates the environment with a storyboard of static frames and transitions between these frames triggered by user actions, not a dynamic environment. Under some circumstances, CogTool storyboards could be combined with dynamic simulations (e.g., driving) with some hand-coded modifications.

In addition, CogTool currently produces Keystroke-Level Models (Card, Moran, & Newell, 1983) implemented in ACT-R, while GRBL produces more direct ACT-R code. An example of this difference is that CogTool represents cognitive activity associated with memory retrieval with a 1.2 second mental operator, while GRBIL uses the retrieval mechanism of ACT-R to determine the timing of retrievals.

## Conclusion

The project has so far demonstrated that even a complex interface such as an OCU for robotic vehicles can be duplicated by the GRBIL system without writing a line of code. The creation of this interface requires only drag and drop placement of controls and a menu-driven description of functionality. The project has also shown the ability to automatically generate cognitive models that perform dynamic spatial reasoning. The creation of these models requires only the demonstration of procedures and the creation of monitors with a simple interface.

## Future Work

The next step in our research is to validate the enhanced GRBIL tool by collecting data on OCU tasks involving spatial reasoning such as projected vehicle intersection. The DRS theory implemented in ACT-R currently uses the default ACT-R timing for movement of attention and object recognition. The validation studies should indicate if further visual processing time is needed to accomplish the DRS routines.

Also, since GRBIL models retrieve procedure steps from memory, it will be possible to use ACT-R's theory of production compilation to predict the trajectory of learning from error-prone retrieval to efficiently compressed performance.

## Acknowledgments

We would like to thank the Army Research Laboratory for their funding of this effort and Bonnie John for her comments.

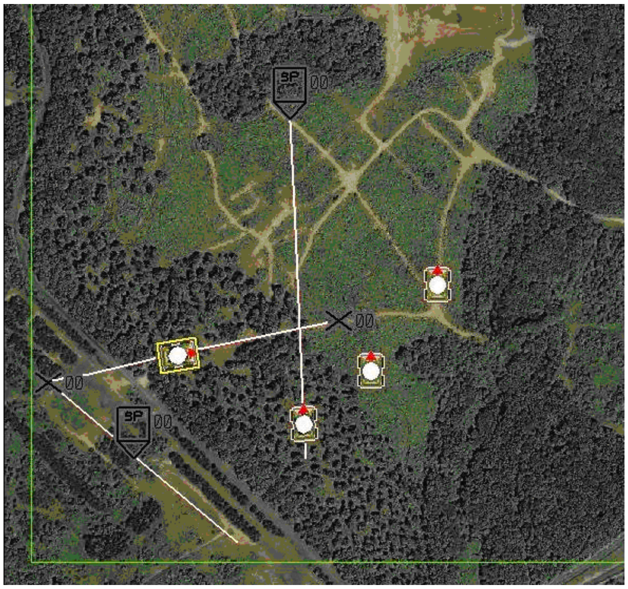


Figure 5. Vehicle path intersection detectable by DRS-enhanced models

## References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. Psychological Review 111, (4). 1036-1060.

Anderson, J. R. & Matessa, M. (1997). A production system theory of serial memory. Psychological Review, 104 (4), 728-748.

Anderson, J. R., Matessa, M., & Lebiere, C. (1997). ACT-R: A theory of higher level cognition and its relation to visual attention. Human Computer Interaction, 12(4), 439-462.

Archer, S. G. & Adkins, R. "IMPRINT User's Guide" prepared for US Army Research Laboratory, Human Research and Engineering Directorate, April 1999.

Archer, R. D., Lebiere, C., Warwick, W. (2005). Design and Evaluation of Interfaces Using the GRaph-Based Interface Language (GRBIL) Tool. ANSE Human Systems Integration Symposium on "Enhancing Combat Effectiveness Through Warfighter Performance", June 20-22, 2005, Arlington VA.

Byrne, M. D., (2001). ACT-R/PM and menu selection: Applying a cognitive architecture to HCI. International Journal of Human-Computer Studies, 55, 41-84.

Card, S. K., Moran, T. P., & Newell, A. (1983). The psychology of human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum Associates.

Chandrasekaran, B., & Kurup, U. (2007). Bimodal Cognitive Architectures: Learning and Memory in Spatial Reasoning. ARL Advanced Decision Architectures RMB Meeting, February 21-23. Westminster, CO.

Chandrasekaran, B., Kurup, Banerjee, Josephson, & Winkler (2004). An Architecture for Problem Solving with Diagrams. In Diagrammatic Representation and Inference, A. Blackwell, K. Marriott and A. Shomojima, Eds., Lecture Notes in Artificial Intelligence 2980, Berlin: Springer-Verlag, pp. 151-165.

Fleetwood, M., Lebiere, C., Archer, R., Mui, R., & Gosakan, M. (2006). Putting the Brain in the Box for Human-System Interface Evaluation. Proceedings of the 50th Annual Human Factors and Ergonomics Society Meeting. Santa Monica, CA

Howes, A., Lewis, R. L., Vera, A., & Richardson, J. (2005). Information-Requirements Grammar: A theory of the structure of competence for interaction. In Proceedings of the 27th Annual Meeting of the Cognitive Science Society, 977-983. Hillsdale, NJ: Lawrence Erlbaum.

John, B. E. & Salvucci, D. D. (2005) Multi-Purpose Prototypes for Assessing User Interfaces in Pervasive Computing Systems. IEEE Pervasive Computing 4(4), 27-34.

John, B., Prevas, K., Salvucci, D., & Koedinger, K. (2004) Predictive Human Performance Modeling Made Easy. Proceedings of CHI, 2004 (Vienna, Austria, April 24-29, 2004) ACM, New York.

Larkin, J.H. & Simon, H.A. (1987). Why a diagram is (sometimes) worth ten thousand words. Cognitive Science, 11, 65-99.

Lathrop, S., and Laird, J.E. (2006). Incorporating Visual Imagery into a Cognitive Architecture: An Initial Theory, Design and Implementation. University of Michigan Technical Report CCA-TR-2006-01.

Matessa, M. & Brockett, A. (2007). Using a Diagram Reasoning System with ACT-R. 16th Conference on Behavior Representation in Modeling and Simulation.

Matessa, M., & Polson, P. (2006). List Models of Procedure Learning. In Proceedings of the International Conference on Human-Computer Interaction in Aeronautics (HCI-Aero), San Francisco, CA.

Salvucci, D.D., & Lee, F. J. (2003). Simple cognitive modeling in a complex cognitive architecture. In Human Factors in Computing Systems: CHI 2003 Conference Proceedings (pp. 265-272). New York: ACM Press.

St. Amant, R., & Ritter, F. E. (2005). Specifying ACT-R models of user interaction with a GOMS language. Cognitive Systems Research. 6(1) 71-88.

# Learning to Control a Dynamic Task: A System Dynamics Cognitive Model of the Slope Effect

**Cleotilde Gonzalez (coty@cmu.edu) and Varun Dutt (varundutt@cmu.edu)**
Dynamic Decision Making Laboratory
Social and Decision Sciences Department
Carnegie Mellon University, 5000 Forbes Avenue
Pittsburgh, PA 15213 USA

## Abstract

We developed a system dynamics model for a simple, but important stock and flows task where the objective was to control the water level in a tank within an acceptable range of the goal, over a number of time periods, in the presence of an unknown environmental inflow and outflow. We also report how this model accounts for human behavior, using behavioral data we collected from human subjects in the task. This exercise helped us understand the strategy and mechanisms our participants used in the simple stock and flows task and develop a model on the task. The model provides an integrated explanation on how the variation in the parameters of the model affects the performance and learning for the participant's task. Finally, we present the model's validity and predictions derived by looking into how the human data fits different learning conditions.

**Keywords:** Learning; Dynamic decision making; System dynamics model; Stock; Flows.

## Introduction

An understanding of the building blocks of dynamic systems, such as stocks (accumulation), flows (rates of incomes and outcomes) and feedback (cause-effect relationships), is essential for dealing with realistic dynamic problems. For example, firms must manage their cash flows to maintain adequate stocks of working capital, and production must be adjusted as sales vary to maintain sufficient inventory. Other examples of these dynamic decision making problems include global warming (Sterman & Sweeney, 2002), factory production, demands and prices of goods (Forrester, 1961), and extinction of natural resources (Moxnes, 2003).

At the individual level, each of us also faces similar stock management challenges: we manage our bank accounts (stock of funds) to maintain a reasonable balance as our incomes (inflows) and expenses (outflows) vary, and we struggle to maintain a healthy weight by managing the inflow and outflow of calories through diet and exercise.

Accumulation is a pervasive process in everyday life, and arises at every temporal, spatial and organizational scale (Cronin, Gonzalez, & Sterman, under review). All stock-flow problems share the same underlying structure: a resource level (stock) accumulates its inflows less its outflows over time.

Unfortunately, there is strong and increasing evidence of poor human understanding of these basic concepts of dynamic systems. For example, Sweeney and Sterman (2000) presented MIT graduate students with a paper problem concerning accumulation of water in a bathtub and asked them to sketch the path for the quantity of water in the bathtub over time, given the patterns for inflow and outflow of water. Despite the apparent simplicity of this task (due to the presence of linearity in the inflow and outflow), they found that only 36% of the students answered correctly. More recently, researchers have found that this misunderstanding of the concepts of flow and accumulation is more fundamental, a phenomenon that has been termed *stock-flow failure* (Cronin & Gonzalez, 2007; Cronin et al., under review). Poor performance in the interpretation of very simple stock and flow problems cannot be attributed to an inability to interpret graphs, contextual knowledge, motivation, or cognitive capacity (Cronin et al., under review). Rather, stock-flow failure is a robust phenomenon that appears to be difficult to overcome.

In past research the stock-flow failure has been investigated through the perception and judgment of static graphs representing flows (Cronin et al., under review; Sterman, 2000). Guided by the tradition of research in dynamic decision making (DDM), we believe that an understanding of the causes and cure for the stock-flow failure will arise through the research on human learning, where individuals can actually experience the flows, influence the stock with their decisions, and have an extended opportunity to control the dynamic system.

With this goal in mind, we constructed a tool, "Dynamic Stock and Flows" (DSF) (Gonzalez & Dutt, submitted), that represents the simplest possible dynamic system containing its most essential elements: a single *stock* that represents accumulation (i.e., water) over time; *inflows,* which increase the level of the stock; and *outflows*, which decrease the level of the stock.  We have conducted several human experiments using DSF, including the investigation of environmental functions, the effects of feedback delays, and the timing of decisions, among others.

In this paper we present the results from an initial empirical study aimed at determining the effects of the slope of an inflow on dynamic control of DSF (Dutt & Gonzalez, 2007). Then, we present a system dynamics model that we created to study the cognitive processes involved in dynamic decision making with DSF.  We validate the model's results against human data, and present some interesting predictions that emerged from this model. The implications and use of system dynamics modeling of cognitive phenomena are discussed.

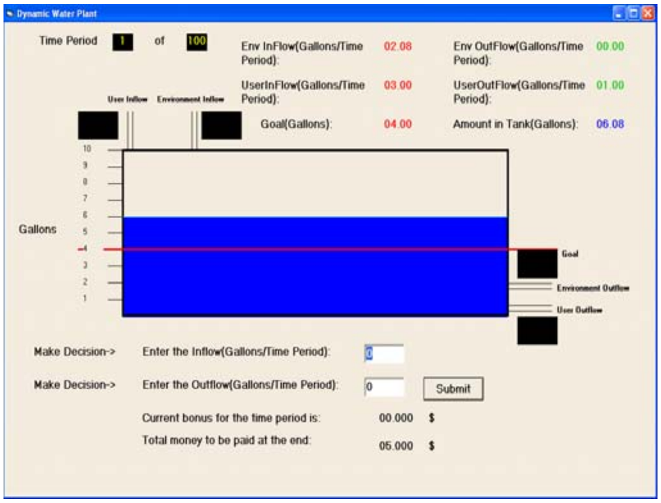
## The Dynamic Stock and Flows Task

DSF is a generic dynamic control task that we designed to help understand human dynamic decision making, and more concretely for this paper to understand the stock-flow failure.

The full capabilities of DSF are described elsewhere (Gonzalez & Dutt, submitted), and .here we will only give a brief overview of the DSF capabilities relevant for the empirical data and modeling reported in this paper.

The goal in DSF is to reach and maintain the level of water in a tank at a target level over a number of time periods. The level of water in the tank is the stock that increases with the inflows and decreases with the outflows. There are two types of inflows and outflows in this task: those that are exogenous (outside of the decision maker's control) and those endogenous (under the decision maker's control). The exogenous flows are called Environmental Inflow (that increases the level of the stock without the user's control) and the Environmental Outflow (that decreases the level of stock without user's control). The endogenous flows are User's Inflow and Outflow. These amounts are the main decisions made by the user in each time period that increase (user inflow) or decrease (user outflow) the level of the stock.

Figure 1 presents the graphical user interface of DSF. At each time period users see the values of Environment Inflow and Outflow, values of User Inflow and Outflow, the amount of water in the tank (stock) and the goal level. At each time period, users can submit two values (including zero): User Inflow and User Outflow, and click in the submit button. Users may also receive a 'bonus' performance monetary incentive in each time period in which they were close enough to the target level.



Time Period 1 of 100

Env InFlow(Gallons/Time Period): 02.08

Env OutFlow(Gallons/Time Period): 00.00

UserInFlow(Gallons/Time Period): 03.00

UserOutFlow(Gallons/Time Period): 01.00

Goal(Gallons): 04.00

Amount in Tank(Gallons): 06.08

Make Decision-> Enter the Inflow(Gallons/Time Period): 0

Make Decision-> Enter the Outflow(Gallons/Time Period): 0

Current bonus for the time period is: 00.000 $

Total money to be paid at the end: 05.000 $

Figure 1: A screenshot of the Dynamic Stock and Flows (DSF) task.

## The Slope Effect

In dynamic decision making, it has been observed that people may detect linear, positive correlations given enough trials with outcome feedback. However, people have difficulty when there is random error or non-linearity and negative correlations (Brehmer, 1980). As part of the stock-flow failure studies we have also observed that people have difficulty understanding the effects that increasing or decreasing trends of inflow and outflow have on controlling a stock (Gonzalez and Vanyukov, in preparation).

In a laboratory study we investigated how individuals controlled DSF over 100 time periods of practice when the environmental inflow increased (positive slope) or decreased (negative slope) as a function of time period (Dutt & Gonzalez, submitted).

Participants played DSF for 100 time periods with the objective of maintaining the tank's water level at the 4 gallon goal line (within +/- 0.1 gallons). In experiment 1, we used an Environment Inflow function that was either increasing: 0.08 * (*timeperiod*) + 2 or decreasing: (-7.92/99) * (*timeperiod-1*) + 10. Environment Outflow was constant and set at 0 gallons/time period during all 100 time periods. Both the increasing and decreasing functions resulted in an equal amount of environmental net flow into the tank over the course of 100 time periods (604 gallons).

Results showed that the stock was higher for the decreasing function condition (M = 5.909, SE = .205) than the increasing function condition (M = 4.297, SE = .027) (F(1,31)=12.71, p<.001). The analyses also indicated that the participants' inflow, outflow and stock diminished significantly over time in both conditions (i.e., subjects learned to control the system) (F(1,31)=9.894, p<.001); and the decrease of the participants' outflow and stock interacted with the slope of the Environmental Inflow function (F(1,31)=7.031, p<.001) (Figure 2 illustrates the interaction on the stock measure).

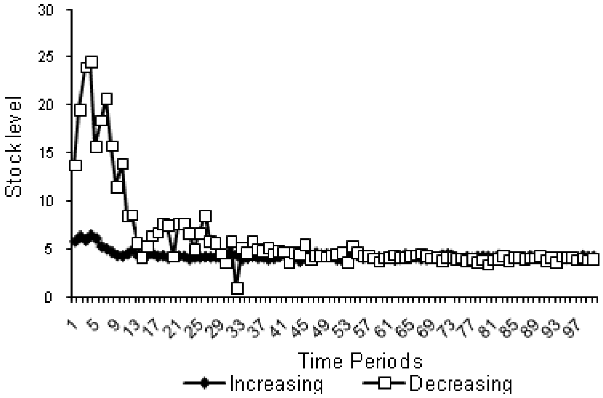


Figure 2: The stock for increasing and decreasing linear Environment Inflow curve conditions over 100 time periods.

In experiment 2, we used a non-linear environment inflow function that was again either increasing: 5*LOG (timeperiod) or decreasing: 5*LOG (101- timeperiod). Outflow was constant and set at 0 gallons/time period during all 100 time periods. Both the increasing and decreasing functions resulted in an equal amount of environmental net flow into the tank over the course of 100 time periods (831 gallons).

Results again showed that the user outflows follow the Environment Inflow functions quite closely. The user inflow and stock indicate that most variability occurred during the first half of the trials, where the decreasing (negative slope) function result in higher inflow and higher stock levels than the increasing (positive slope) function. No difference between the increasing and decreasing functions is observed for the last 50 trials of the experiment in the user inflow and stock results. An analysis of the first 50 trials indicated that the stock was on average higher in the decreasing (M = 7.938, SE = .419) than in the increasing condition (M = 4.757, SE = .143) (F(1,30) = 6.49, p<.05). The same analyses for the last 50 trials of the experiment did not show a difference between the increasing and decreasing functions for inflow and stock variables. A significant difference was found only for the user outflow (F(1,31) = 85.334, p<.001) where the outflow was higher for the increasing (M = 9.543, SE = .081) than the decreasing (M =7.067, SE = .195) function. Once again, the decrease of the user outflow and stock interacted with the slope of the environmental inflow function; (Figure 3 illustrates the interaction on the stock measure).

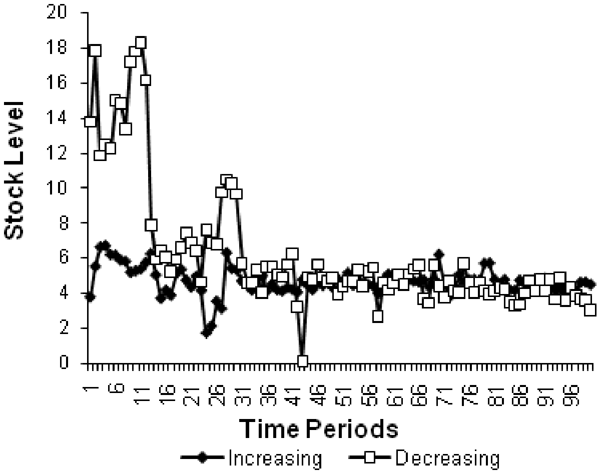


Figure 3: The nature of stock for increasing and decreasing Environment Inflow in non-linear curve slopes over 100 time periods.

## System Dynamics Model

System dynamics (SD) is a field that was created by Jay W. Forrester at MIT in the late 1950s and it involves a modeling approach using computer simulations (see Forrester, 1990 for an historical view of the field; Lane, 2000 for a discussion on the modeling approach).

A model in SD involves at its essence the concept of a feedback loop: the collection of information about the system state followed by an action that changes the state of the system (Lane, 2000). These causal links involve delays and non-linearities as well as processes of accumulation (stocks) and flows.

SD modeling has largely focused on the representation of social systems and their evolution over time. In fact, it has been argued that SD is concerned with aggregate social phenomena, not with individual meaningful actions (Lane, 2000).. In this paper we use SD modeling to represent and reproduce the dynamics of individual human behavior found in DSF. We also construct a cognitive interpretation of the SD model, something uncommon in the SD field.

A SD model was developed using Vensim®, an open source modeling software by Ventana Systems Inc. The software has a flexible GUI that provides easy capability to the modeler to represent stocks, define the stocks' inflows and outflows and define their causal relationships. Although the conventions for representing stocks and flows followed in Vensim® are well known and documented in the SD literature (Forrester, 1961; Sterman, 2000) we discuss only some of the many software features that we used for our model of learning and the slope effect. These features are fixed time delay and smoothing.

The function defined in Vensim® as DELAY FIXED (X, T, I) creates a delay of T time periods in an input X with the initial value I of the variable used on the left hand side of the function. In our model we use DELAY FIXED to create a unit time delay in the environment inflow at each time period. This is because participants in our DSF task are aware of the environment inflow value for a time period only at the end of that time period.

Smoothing is defined in Vensim® as SMOOTH (X, T) and SMOOTHI (X, T, I) and creates an exponential smoothing of T time periods in an input X with I as the initial value of the variable used on the left hand side of the function. If X is a step function which jumps to a new value X' at a time instance *t*, then the SMOOTH of X will start from X and approach the value X' over a long range of time periods. The greater the value of T the more time SMOOTH of X takes to approach X'. This smoothing effect of time averages to represent expectations is similar to blending parameters used in learning models of dynamic decision making under the ACT-R cognitive modeling approach (Gonzalez, Lerch, & Lebiere, 2003). In our SD model, we used a smoothing effect to account for the gradual correction of a discrepancy made by participants.

## A System Dynamics Model of the Slope Effect in DSF

To help develop this model, we used our observations from verbal protocols collected from four participants (Dutt & Gonzalez, 2007). We also used human data analyses of inflow and outflow decisions and their resulting stock; the averages of individuals' decisions for each of the conditions; and comparisons of the participants' inflow and outflow decisions to the stock and environmental flow values.

Based on these observations and empirical data analyses, we developed the SD model shown in Figure 4. The system essentially consists of 2 inputs (User Inflow and Environmental Inflow) that increase the stock and 2 outputs (User Outflow and Environmental Outflow) that decrease the stock.

The behavior is represented by causal loops described in the model. The Environmental Inflow and Outflow are perceived by the participants. The perception may be different from the reality, as determined by the Environment perception (EP) parameter. Then, the perceived environment netflow is used to forecast the future flow under a Forecast Horizon (FH). This forecast together with the perceived current discrepancy between the stock and the goal are used to determine the netflow correction. This correction is to account for the increase in discrepancy over the perceived time for correction (PTC) smoothed over the memory of discrepancy (MD). Then, according to the determined user netflow correction the user inflow and outflow are entered by the user. Over the course of practice, Users modify the weight they put to inflows and outflows and in general, the empirical data demonstrated that individuals end up realizing that they only need to enter the User Outflow to control for the Environmental Inflow (W=1).

The user net flow correction variable in our model (Figure 4) serves as the main decision function for user inflows and outflows and consists of two parts, discrepancy and forecast of flow. The user net flow correction is given by the procedure: If the discrepancy is beyond an acceptable threshold (.1 above or below the goal), then, attempt to correct for such discrepancy little by little, by smoothing the discrepancy over the perceived time for correction (PTC), and according to the memory of discrepancy (MD). Then, add to this smoothed discrepancy value, the value of the forecasted flow.

Discrepancy is the difference between the goal and stock. The forecast of flow is defined as per the formula: SMOOTH (Perceived Environment Net flow, "Forecast Horizon (FH)") where the Perceived Environment Net flow is a fixed unit delay function of environment inflows and outflows (Vensim® formula: DELAY FIXED ("Environment's Perception (EP)"*(Env Inflow – Env Ouflow), 1, 0).

Hence, the discrepancy over PTC is smoothed by MD and only adds to user net flow correction if the discrepancy is above or below the acceptable range of stock in the tank, else its affect to user net flow correction is zero. The perceived environment net flow is smoothed by parameter FH to makeup the forecast of flow. The environment inflow and outflow act on the stock each time period, with a subject becoming aware of actual values only at the end of that time period. This requirement is realized in our model by using a fixed unit delay as shown in the formula of perceived environment net flow. Participants' accuracy to perceive the environment's inflow and outflow affect is represented by the multiplier EP also present in the delay formula.

As experiment 1 consisted of a linear environment function and experiment 2 consisted of a non-linear environment function we simulated our Vensim® model twice over a course of 100 time periods, once for getting model data for fit to experiment 1's subject data and a second time for getting model data for fit to experiment 2's subject data. During the course of each of the two simulation runs, we varied the values of FH, PTC, MD and W with an increment of 0.05 each. Although each of these parameters could take infinitely many values, their chosen values were inspired from results on experiments 1 and 2 and the availability of human data from both experiments.

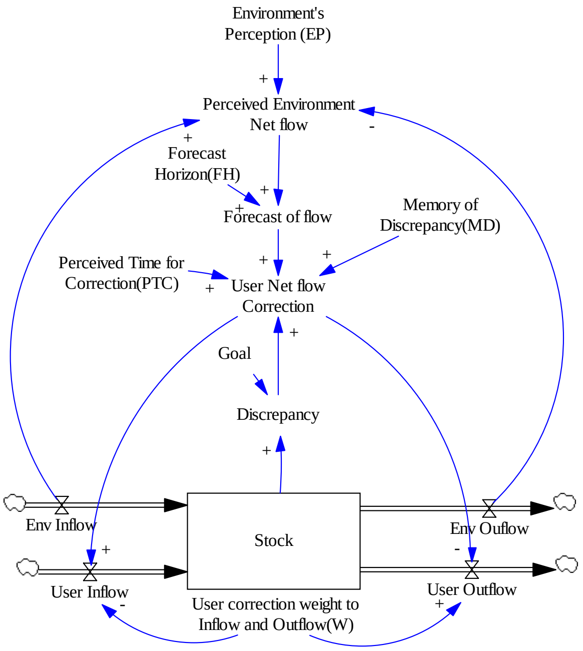


Figure 4: Stock and Flows Model of DSF task.

## Fit of Model and Human Data

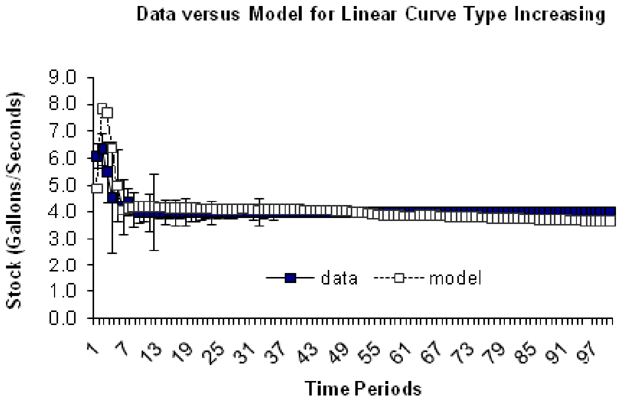
Based on the results from both experiments, we expected that the parameters of the model, the environment perception (EP), the forecast horizon (FH), the perceived time for correction (PTC), the memory for discrepancy (MD), and the correction for inflow-outflow (W), would be different for the positive and the negative slope conditions.

We went through a process of parameter tuning and sensitivity analysis and were guided by our results and expectations, while measuring the fit of the model's data to the humans' mean and median values over the course of 100 trials, using the r and RMSD statistics (Schunn & Wallach, 2001). The resulting parameter values and data fit statistics for our data from experiments 1 and 2 are summarized in Table 1. The same model depicted in Figure 4 was used to fit the data of the four different data groups from experiments 1 and 2. The value of the parameters summarized in Table 1 help provide a coherent explanation for the different human behavior found between the positive and negative slopes.

Table 1: The value of model parameters and the resulting fit to human data (measured against both, the median and the mean) for each of the 4 groups in the 2 experiments (Linear positive and negative; Non-linear positive and negative) for the stock as dependent variable.

| *Condition* | *Parameters* | | | | | *Fit Measures* | | | |
|---|---|---|---|---|---|---|---|---|---|
| | **EP** | **FH** | **MD** | **PTC** | **W** | **Median** | | **Mean** | |
| | | | | | | **r** | **RMSSD** | **r** | **RMSSD** |
| *Stock* | | | | | | | | | |
| **Linear Positive L+** | 1.03 | 1 | 1.1 | 2.5 | 1 | .82 | 1.94 | .79 | 2.34 |
| **Linear Negative L-** | 1.0 | 1.5 | 1.0 | 4.3 | 1 | .91 | 3.9 | .88 | 4.15 |
| **Non Linear Positive NL+** | 1.03 | 1 | 1 | 2.0 | 1 | .95 | 1.85 | .51 | 2.62 |
| **Non Linear Negative NL-** | 1.07 | 1.5 | 1 | 4.5 | 1 | .91 | 7.54 | .63 | 8.00 |

Some general observations from results shown in Table 1 are: PTC and FH parameter values are higher in the negative than in the positive functions; the model fits the linear functions better than the non-linear functions; and the model data fit the median of the human data better than the mean.



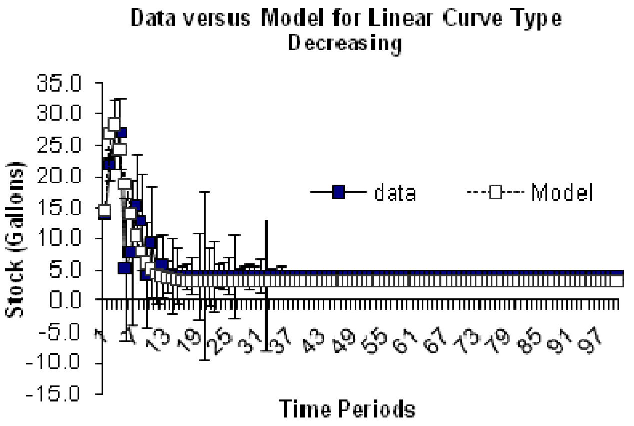


Figure 5: Graphs and parameter values for model's fit to stock.

Figure 5 is a graphical example of the fitting of stock data for the linear and non-linear, positive and negative slopes' functions as a result of our analysis and parameter tuning.

Greater FH value means that participants takes more time to forecast the environment inflow value, and this extra time causes the stock to increase due to the environment inflow, which drives the stock away in each time period. Similarly, greater PTC value means that participants take more time to perceive the discrepancy happening in the tank and hence also take more time to correct the discrepancy; this extra time causes the stock to rise again due to environment inflow into the stock with each elapsing time period. To meet the higher stocks, participants order higher user inflows and outflows, causing user inflow and user outflow values to rise as well. When we fit our model to the negative-sloped environment inflow cases, we see that increasing FH and PTC generates a good fit for the human data. This increase in FH and PTC in our model causes the stock to rise higher (due to the environment inflow action and slower corrective action), causing the discrepancy to increase further and hence, the user net flow correction to increase further. The user net flow correction is further increased by the forecast for net flow, which now happens over a larger time period range (due to FH increase, where there are increased environment inflows over this larger time range). The increase in user net flow correction as described above causes higher user inflow and user outflow in our models, where the user inflow and user outflow are derived from the user net flow correction and weighted by the user correction weight to inflows and outflows (W). The fact that the model's data fits the median of human data more closely than the mean is a reflection of the deterministic nature of SD models in general that fail to account for the variability of human performance, as they are deterministic models.

## Model Predictions

From the results on linear and non-linear environmental curves, we find that a negative slope in the environment inflow produces higher stock, user inflow and user outflow as well as higher variability in stock, user inflow and user outflow, particularly in the first 50 time periods (this is also the time when subjects are learning to control the stock in DSF).

If the parameters that we have proposed in our model and their fit to human data can have a cognitive interpretation as we presented in the previous section, then there are some interesting questions we could answer by looking into model fitting more deeply.

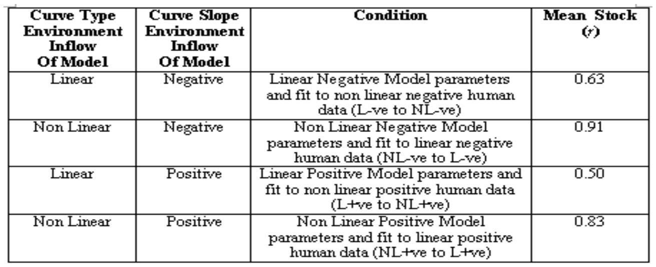
Specifically, we are interested in determining how human data from each of the collected groups: Linear Positive (L+), Linear Negative (L-), Non-linear Positive (NL +), and Non-linear Negative (NL -) would fit to the model's data from the other groups. For example, we test how human data in L+ fits to the model's data from L-, NL+, and NL- groups by looking at the difference between the r values (measures of fit) between equivalent groups from the model's data and the comparison group.

For this fit exercise, we found the mean and median of our human data for both experiments 1 and 2 under different conditions, L+, L-, NL+ and NL-. We already have our fits and model parameters calculated as a result of parameter tuning and sensitivity analysis to the mean and median of human data under L+, L-, NL+ and NL- conditions as mentioned in Table 1. We took the model parameters under the NL- condition and fit the model data due to these parameters to the mean of human data under the L- condition. Similarly we took the model parameters under L- condition and fit the model data due to these parameters to the mean of human data under the NL- condition. We did a similar exercise for L+ to NL+ and NL+ to L+. This process helps us to foresee the mapping of our model's cognitive parameters to different experimental conditions as measured by model fits, i.e. how different values of cognitive parameters EP, FH, PTC, MD and W perform under different experimental conditions. This mapping exercise can help us predict how the model experiencing a linear inflow would behave when put into a non-linear inflow and vice-versa. For example, from a managerial perspective a firm may suddenly face non-linear changes in demand after operating under a constant (linear) demand (Paich & Sterman, 1993). In addition, the knowledge gained from this cross fitting exercise helps us understand the nature of underlying task situations involved, task complexity as it would possibly be experienced by the decision makers.

The results from fitting model's parameters on linear environment inflow to non-linear environment inflow and vice-versa are tabulated in Table 2. The *r* values given in Tables 1 and 2 provide results for the mean stock in DSF.

The results show that *r*(NL- to L-) > *r*(L-), *r*(L- to NL-) < *r*(NL-) and *r*(NL- to L-) > *r*(L- to NL-). Also from Tables 1 and 2, similar results hold for the positive slope environment inflow cases for both the linear and non-linear curve types. This means *r*(NL+ to L+) > *r*(L+), *r*(L+ to NL+) < NL+ and *r*(NL+ to L+ > L+ to NL+.

Table 2: Values of correlation coefficients for model predictions on linear and non-linear positive and negative sloped environment inflow for stock mean.



| Curve Type Environment Inflow Of Model | Curve Slope Environment Inflow Of Model | Condition | Mean Stock (*r*) |
|---|---|---|---|
| Linear | Negative | Linear Negative Model parameters and fit to non linear negative human data (L-ve to NL-ve) | 0.63 |
| Non Linear | Negative | Non Linear Negative Model parameters and fit to linear negative human data (NL-ve to L-ve) | 0.91 |
| Linear | Positive | Linear Positive Model parameters and fit to non linear positive human data (L+ve to NL+ve) | 0.50 |
| Non Linear | Positive | Non Linear Positive Model parameters and fit to linear positive human data (NL+ve to L+ve) | 0.83 |

These cross fit results indicate the *r* value diminishes from linear to non-linear curves but increases from non-linear to linear inflows. This is, the non-linear conditions are more difficult to fit than the linear condition.

Similar results were reported on the human data collected from both experiments 1 and 2 earlier, where subjects' performance was poorer in the non-linear environment inflow DSF task when compared to performance in the linear environment inflow DSF task. These similarities between the nature of DSF model and the DSF human data also further support the choice of the model's parameters.

## Conclusions

*Stock-flow failure* is a phenomenon representing the poor interpretation of very simple problems involving accumulation over time by flows (Cronin & Gonzalez, 2007). In this paper we investigated one possible explanation for the stock-flow failure and that is the increased difficulty for controlling systems with decreasing more so than increasing inflows. We investigated this simple failure in a Dynamic Stock and Flows (DSF) task. We found that participants yielded greater quantity of stock, inflows and outflows and more variability in them for negative slope (decreasing) environment inflow conditions when compared with the quantity and the variability in stock inflows and outflows for the positive slope (increasing) environment inflow conditions.

We explain these results through a system dynamics model that helps derive differences in human behavior due to slopes of environment inflow. The constructed model and its fit revealed minimize of a number of human cognitive parameters in the dynamic task which makes us think that similar cognitive parameters would constitute many such simple dynamic stocks and flows tasks which are important in our day to day lives (our bank accounts to our weight gain and loss processes to name a few) where the understanding of such parameters would be helpful in overcoming the difficulties that most of us face while encountering them.

## Acknowledgments

This research was partially supported by the National Science Foundation (Human and Social Dynamics: Decision, Risk, and Uncertainty, Award number: 0624228) and by the Army Research Laboratory (DAAD19-01-2-0009) awards to Cleotilde Gonzalez.

## References

Brehmer, B. (1980). In one word: Not from experience. *Acta Psychologica, 45*, 223-241.

Cronin, M., & Gonzalez, C. (in press). Understanding the building blocks of system dynamics. *System Dynamics Review*.

Cronin, M., Gonzalez, C., & Sterman, J. D. (under review). Why don't well-educated adults understand accumulation? A challenge to researchers, educators and citizens.

Dutt, V., & Gonzalez, C. (2007). *Slope of inflow impacts dynamic decision making*.

Forrester, J. W. (1961). *Industrial dynamics*. Waltham, MA: Pegasus Communications.

Forrester, J. W. (1990). *The Beginning of System Dynamics*. Boston, MA: The Sloan School of Management, MIT.

Gonzalez, C., & Dutt, V. (submitted). A generic dynamic control system for management education.

Gonzalez, C., Lerch, J. F., & Lebiere, C. (2003). Instance-based learning in dynamic decision making. *Cognitive Science*, *27*(4), 591-635.

Lane, D. C. (2000). Should system dynamics be described as a 'hard' or 'deterministic' systems approach? *Systems Research and Behavioral Science, 17,* 3-22.

Moxnes, E. (2003). Misperceptions of basic dynamics: The case of renewable resource management. *System Dynamics Review, 20,* 139-162.

Paich, M., & Sterman, J. D. (1993). Boom, bust and failures to learn in experimental markets. *Management Science, 39*(12), 1439-1458.

Schunn, C. D., & Wallach, D. (2001). Evaluating goodness-of-fit in comparison of models to data. University of Pittsburgh.

Sterman, J. D. (2000). Learning in and about complex systems. *Reflections: The SoL Journal, 1*(3), 24-51.

Sterman, J. D., & Sweeney, L. B. (2002). Cloudy skies: Assessing public understanding of global warming. *System Dynamics Review, 18*(2), 207-240.

Sweeney, L. B., & Sterman, J. D. (2000). Bathtub dynamics: initial results of a systems thinking inventory. *System Dynamics Review, 16*(4), 249-286.

# Instance-Based Decision Making Model of Repeated Binary Choice

**Christian Lebiere (cl@cmu.edu)**
Psychology Department
Carnegie Mellon University, 5000 Forbes Avenue
Pittsburgh, PA 15213 USA

**Cleotilde Gonzalez (coty@cmu.edu) and Michael Martin (mkmartin@andrew.cmu.edu)**
Dynamic Decision Making Laboratory,
Social and Decision Sciences Department
Carnegie Mellon University, 5000 Forbes Avenue
Pittsburgh, PA 15213 USA

## Abstract

We describe an instance-based model of decision-making for repeated binary choice. The model provides an accurate account of existing data of aggregate choice probabilities and individual differences, as well as newly collected data on learning and choice interdependency. In particular, the model provides a general emergent account of the risk aversion effect that does not require any metacognitive assumptions. Advantages of the model include its simplicity, its compatibility with previous models of choice and dynamic control, and the strong constraints it inherits from the underlying cognitive architecture.

**Keywords:** Learning; dynamic decision making; RELACS; memory; cognitive architectures; ACT-R.

## Introduction

Erev and Barron (2005) have discussed the tradeoffs of adaptation and maximization in repeated choice tasks. A main demonstration from their studies is that extended practice with a binary choice problem with immediate feedback does not always lead to payoff maximization.

The deviations from maximization may be due to different effects. One of them, the *payoff variability effect*, refers to a tendency to increase exploration in a noisy environment (Erev & Barron, 2005). That is, when payoff variability is associated with an alternative of higher expected value compared to the other alternative, choice behavior moves toward random choice. This payoff variability effect has been found in one-shot decisions (Busemeyer & Townsend, 1993) but it is more robust in repeated choice (Erev & Barron, 2005).

Erev and Barron (2005) proposed a model of Reinforcement Learning Among Cognitive Strategies (RELACS) to account for the payoff variability effect and other deviations from maximization. RELACS assumes that a decision maker follows one of three cognitive strategies in each choice, and that the probability of using a strategy is determined by previous experiences with the strategy.

The *fast best reply* strategy involves selecting the alternative with the highest recent payoff. The *case-based reasoning* strategy involves moving from a random selection of alternatives initially to a two-stage process in which a belief is first determined and then verified as not being associated with large losses. The *slow best reply* strategy involves choosing to explore the two alternatives initially and moving gradually toward preferring the alternative more likely to maximize earnings. According to RELACS, the three strategies are reinforced with their frequency of use and are updated according to the observed payoffs.

In their analyses and comparisons to other models, Erev and Barron determine that the slow best reply strategy is the one that best captures the payoff variability effect. They also found that the assumption of learning among the different strategies is not important because a random selection among strategies fits the data as well as RELACS does.

In our past research we have proposed a framework and computational model that characterize decision makers' preferences and utilities in terms of action-outcome links. This theory called Instance-Based Learning Theory (IBLT) (Gonzalez, Lerch & Lebiere, 2003), implemented in ACT-R (Anderson & Lebiere, 1998; Anderson et al, 2004), proposes learning (i.e., increasing maximization) occurs through a progressive accumulation of *decision instances*. Instances are discrete units of knowledge (action-outcome links) which are constructed, upgraded, and reused through experiential learning in a decision making situation. Better decision policies emerge gradually as decision makers move from using explicit rules of action to implicit recognition of familiar patterns (cf. Dienes & Fahey, 1995), similar to the gradual process proposed in Logan's (1988) instance theory of automaticity. Many decision making tasks have successfully been implemented in ACT-R using this process, including dynamic control tasks (Wallach & Lebiere, 2003), supply chain management (Gonzalez & Lebiere, 2005; Martin, Gonzalez & Lebiere, 2004), backgammon (Sanner et al, 2000) and simple 2x2 games like the Prisoner's Dilemma (Lebiere, Wallach & West, 2000).

Our main contention in this paper is that the experiential accumulation, activation, retrieval and generalization of action-outcome decision instances is a general decision making strategy applicable to multiple decision making tasks, including the simple repeated choice effects posed by Erev and Barron (2005). Accordingly we describe an instance-based decision making model that captures the learning effects and the tradeoffs of adaptation and maximization reported by Erev and Baron (2005). Our instance-based decision making model works in ways similar to the slow best reply strategy proposed by Erev and Barron (2005). The results from our ACT-R model support Erev and Barron's arguments that the slow best reply strategy is the one that best captures the payoff variability effect and that learning among different cognitive strategies is unnecessary. Thus, deviations from maximization in repeated binary choice problems can be reproduced without pre-defining a set of cognitive strategies and positing reinforcement learning as a mechanism for selecting among them.

In what follows, we discuss the example problems we have taken from Erev and Barron (2005), and discuss how we replicated their behavioral results. Next, we discuss our instance-based decision making ACT-R model and the results from our model as compared to RELACS results. Finally, we discuss some predictions of our model and possibilities for unification with models of other tasks.

## The Payoff Variability Effect

We replicated, with human participants, the payoff variability effect using the following three key problems from Erev and Barron (2005):

Problem 1. H 11 points with certainty
L 10 points with certainty

Problem 2. H 11 points with certainty
L 19 points with probability 0.5
1 otherwise

Problem 3. H 21 points with probability 0.5
1 otherwise
L 10 points with certainty

All three problems required participants to choose between a high payoff alternative H (with an expected value of 11 points) and a low payoff alternative L (with an expected value of 10 points). The problems differed only on the variance but not the mean of the two payoff distributions.

We randomly assigned 60 participants to one of the three problems. The undergraduate and graduate students at Carnegie Mellon University were paid a flat fee for performing the repeated choice task for 400 trials.

We followed almost identical instructions as in Erev and Barron's experiments: individuals did not receive any information about the payoff structure. They were told their task was to select one of the alternatives by clicking on one of two unmarked and masked buttons. They were provided with the payoff value of the button they clicked on. Individuals were not informed of the trial number. Payoffs were drawn from the distribution associated with the selected button.

There are two differences between our methods and Erev and Barron's: (1) we did not use a performance-based incentive structure and (2) we ran 400 rather than 200 trials to better explore learning effects.

Figure 1 shows the proportion of maximization (Pmax) (H) choices during the 400 trials. The average proportions of maximization are very similar to those reported in the original experiments: average Pmax for the second 100-problem block (a.k.a. Pmax2) was 0.82, 0.61 and 0.50 for Problem 1, 2, and 3 respectively (compared to .90, .71, .57 in Erev and Barron's data).

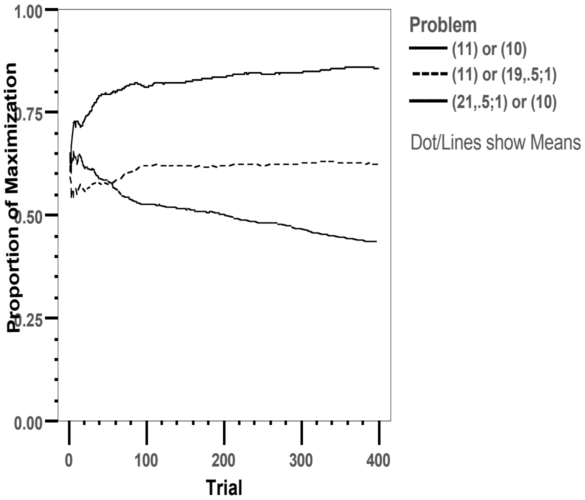


Figure 1: Proportion of maximization over practice

The learning curves shown in Figure 1 demonstrate that, as expected, an increase in payoff variability impairs maximization. In contrast to data reported by Erev and Barron (2005) the Problem 3 learning curve, where the alternative with the maximum payoff is risky, shows a *decrease* in the proportion of maximization over time.

As suggested by Erev and Barron, the difference between problems 1 and 3 demonstrates the risk aversion effect and the difference between problems 1 and 2 the risk seeking effect. We investigated the risk aversion or certainty effect (Kahneman & Tversky, 1979) in this repeated choice task by collecting data in the following problem:

Problem 4. Certain 11 points with certainty
Risky 21 points with probability 0.5
1 otherwise

Problem 4 presents participants with a tie, i.e. alternatives have the same expected value, but one is risky and the other is certain. Using the same methods as in the first 3 problems, we collected data from 20 participants. We will report detailed findings on that condition in the model comparison section.

One of the challenges for Erev and Barron's RELACS model is that it consistently underpredicts individual differences. A particular problem in RELACS seems to be the management of memory, as it does not capture the interdependency of past experiences.

Memory management and learning is a strength in ACT-R and a particular strength of our instance-based decision-making models as there are strong constraints on the effect that particular past instances would have on a future choice (Gonzalez & Quesada, 2004). As we will demonstrate, our ACT-R based models of instance-based decision making predict observed individual differences quite accurately.

## ACT-R instance-based decision making model

One advantage of instance-based learning models is that they reduce degrees of freedom in modeling. The modeler does not have to select and implement strategies, or decide upon arbitrary criteria on which a decision is made. Instead, the model represents the information immediately available to the subject in the most direct form possible, and uses that information directly to make its decisions.

Each decision-making instance in the repeated choice paradigm is composed of two elements: the choice being made and the payoff immediately received as a result. Those two elements of a decision-making instance are consciously available to the subject and thus will be represented together in declarative form.

The basic unit of declarative representation in ACT-R is the chunk. A chunk is a typed structure composed of a number of named fields, also called slots. Each slot usually contains another chunk (although it can also be empty or contain special values). Our model contains only a single chunk type, **choice**, with only two slots: **decision**, which holds the decision made by the model, and **payoff**, which holds the payoff awarded after the decision. For example, a chunk encoding the experience that pressing the left button resulted in a payoff of 10 would have the following form:

```
Decision1
     isa decision
     choice Left
     payoff 10
```

That chunk type serves both as the only type of goal for the model, and as the repository of the problem-solving experience in long-term declarative memory. The learning of that symbolic information is thus automatically accomplished by the architecture as it stores past goals into long-term memory.

The experimental paradigm covered by Erev and Barron (2005) includes three feedback conditions. In the first one called *minimal information*, payoff feedback is given only for the choice being made, and encoded as described above. In the second condition, called *complete feedback*, payoff is given for the choice made as before, but the payoff that would have resulted if the other choice had been made is also given. In that case, the model generates two chunks, one for each potential choice and its feedback. In the third condition, called *probability learning*, no numerical payoff feedback is given directly but instead the payoff is translated into a relative probability of correct choice, which is then relayed to the subject as a correct/incorrect binary feedback. In the model, that binary feedback is simply encoded as a 0/1 payoff and the same modeling approach can then apply.

How does the model use this information about choices and payoffs? The basic decision-making procedure is the same as that used in the model prisoner's dilemma and other 2x2 games (Lebiere, Wallach & West, 2000). The model evaluates each option by retrieving its expected payoff from memory, selects the one with the highest value, then registers the feedback as described above. This procedure is implemented in half-a-dozen generic production rules.

As in some past instance-based models (e.g. the Paper Rocks Scissors model of West & Lebiere, 2001), the possible combinations of symbolic information are so few (less than a handful in the payoff functions studied here) that the key knowledge of the task does not reside at the symbolic level but instead in its statistical properties. Specifically, the key information is the frequency (and recency) of each combination of decision and payoff. While subjects (and the model) could potentially keep track of those frequencies explicitly, there is no evidence that they do so. Instead, the architecture automatically learns such information in the activation values of the various chunks. Specifically, the base-level activation $A_i$ of chunk $i$ is determined by the following Bayesian learning formula:

$$A_i = \ln \sum_{j=1}^{n} t_j^{-d} \qquad \textbf{\textit{Base Level Learning}}$$

Each $t_j$ is the lag of time since the $j$th occurrence of chunk $i$. The architectural parameter $d$ is the decay rate of each occurrence, which is set to 0.5 as is (almost) always the case in ACT-R models. The power law of practice emerges from the log-summation over all references whereas the power law of forgetting results from the decay of each reference. Just as for chunks, this learning of the statistical properties of the symbolic knowledge is accomplished automatically by the architecture. Activation determines the probability of retrieving each qualifying chunk according to the following equation:

$$P_i = \frac{e^{A_i/t}}{\sum_j e^{A_j/t}} \qquad \textbf{\textit{Boltzmann Equation}}$$

This equation, also known as the softmax equation, defines retrieval as a noisy process where the probability of retrieving a given chunk is proportional to the ratio of its activation and a retrieval noise level $t$. The noise level determines the degree of stochasticity of the retrieval process and similar to the decay rate parameter it is left at its default value of 0.25.

However, the retrieval process described above has one problem. If it only retrieves one chunk associated with a given choice, it will usually not be sensitive to the magnitude of the payoff values. If one alternative has a certain payoff of 11, it will not matter whether the other has equally likely payoffs of 1 and 19 (averaging 10) or 1 and 199 (averaging 100). It will choose each about half the time in both cases, which is clearly not right. What we want is a retrieval procedure that takes into account both the frequency (and recency) of each payoff as reflected in its activation and the magnitude of the payoff itself. To that effect, Lebiere (1999) introduced a variation of the retrieval process called *blending* that has since been used in many instance-based models (e.g. Gonzalez et al., 2003; Wallach & Lebiere, 2003). The key equation controlling blended retrieval is the following:

$$V = \min \sum_i P_i \cdot \left(1 - Sim\left(V, V_i\right)\right)^2 \quad \textbf{\textit{Blending Equation}}$$

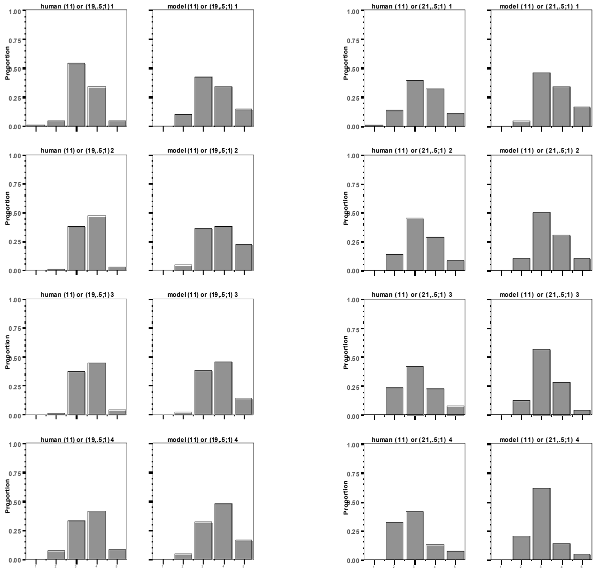
The equation states that the value *V* returned by retrieval is the one that best satisfies the constraints offered by all matching chunks *i* weighted by their probability of retrieval $P_i$ as computed in the Boltzmann equation above. Satisfying chunk constraints is defined in terms of minimizing the dissimilarity (i.e. maximizing the similarity) between the consensus answer *V* and the actual answer $V_i$ contained in chunk *i*. This process is applicable to all domains, discrete and continuous, as long as a similarity metric is defined over those values. As such it can be seen as an implementation of the generalized Bayesian framework of Tenenbaum & Griffiths (2001) or an approximation of the generalization capabilities of connectionist architectures based on distributed representations (e.g. O'Reilly & Munakata, 2000). In practice, we define linear similarity values over payoffs, which result in the retrieval process averaging their values weighted by activation.

A final point concerns the initialization of the model. If the model started with no expectations of the payoffs, it would start by deciding randomly, but then as soon as one payoff had been experienced for each choice, it would happily take the best indefinitely. To trigger exploration at the start, we initialized the model with a single chunk for each decision encoding high initial expectations (payoff of 1000). That initial value will quickly get overwhelmed by actual experience as it decays and is never reinforced, but it results in an initial period of exploration that corresponds well to human subjects without the need to arbitrarily define a specific strategy to that effect.

## Results and Comparison

Our model fits the data quite well using Erev and Barron's (2005) primary measure of performance, namely the probability of maximization in the second block of 100 problems (Pmax2).. That measure for problems 1, 2 and 3 is 0.91, 0.65 and 0.53 respectively for our model, as compared to 0.90, 0.71 and 0.57 for Erev and Barron's data and 0.82, 0.61 and 0.50 for our data. The variation between data and model is substantially smaller than the variations between data sets, suggesting the substantial role played by individual differences.

To examine individual differences, we have plotted in Figure 2 the distribution of the probability of maximization for each 100-trial block for individual subjects (and model runs) within each of five intervals: 0-20%, 20-40%, 40-60%, 60-80% and 80-100%. For reasons of space, we have selected problems 2 (Figure 2a) and 4 (Figure 2b) as the most interesting for display. Focusing for now on problem 2, one can see that the distribution of probabilities ranges across the highest 4 categories, a range well-reproduced by our model. One could argue that it is too well reproduced, with the highest category over-represented compared to the data. However, in Erev & Barron's data (which only report distribution figures for the second block), the two highest categories (60-80% and 80-100%) dominate with many fewer subjects in the 40-60% category than for our data. This would seem to explain the discrepancy between the values of Pmax2 observed by Erev and Barron and us (0.71 vs. 0.61). In this respect, our model fits comfortably between the two data sets, but it is again a reminder to be careful when comparing to aggregate data across subjects.



2a 2b

Figure 2: Individual differences for problems 2 (left) and problem 4 (right). Within each panel human data (left) and model data (right) distribution of maximization probability are shown, aggregated in 20% increments.

We now consider how deviations from maximization emerge from our model and in particular the source of the payoff variability effect. For problem 1, with deterministic payoffs, maximization is simply a matter of quickly overcoming the high initial expectations through the exploration phase. Alternative H consistently returns the highest payoff. Thus blending consistently produces a higher expected value for alternative H. For problem 2, with variable payoffs for alternative L, blending combines the distinct payoffs of 1 and 19 for alternative L to produce random fluctuations in expected value. Although blending will tend to average the two payoffs of alternative L to 10 if they are of equal activation, the noise of the activation process and the random distribution of L payoffs will tend to make activations unequal, pushing the average on either side of 10, and sometimes higher than 11, which leads to lapses in maximization. The same happens with Problem 3, except that alternative H averaging 11 is now the one with the noisy distribution, and the model does reproduce the tendency to select it less frequently than it does in Problem 2, indicating risk aversion. This brings us to problem 4, the risk aversion problem, where this symmetry argument would suggest that both options would be chosen equally often on average. However, on average both subjects and model tend to prefer the certain option, roughly 55% of the time. This risk aversion effect (and the difference in Pmax2 between Problems 2 and 3) arises from a subtle interaction in the dynamics of the task illustrated in Figure 3.

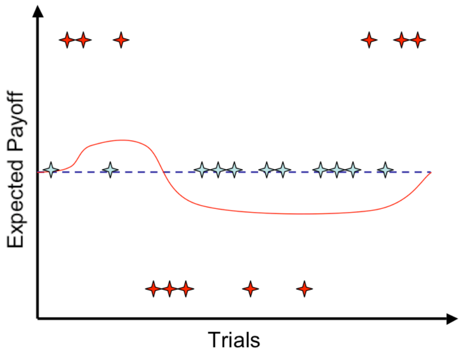


Figure 3: Emergence of risk aversion effect

The blue dashed line represents the (constant) expected value of the certain alternative while the red line represents the expected value of the risky alternative. On average, the expected values of the two alternatives are equal and they indeed start that way. Each star of a given color (red or blue) indicates an experienced payoff for the associated choice. After the start, the risky alternative provides some lucky payoffs (e.g. 21), which raises its expected value and leads to its selection more often. Luck even outs quickly however as a series of poor payoffs (e.g., 1) lowers its expected value to less than 11, which in turn leads to selection of the certain alternative most of the time. The key insight is that this bias toward certain payoffs leaves the risky alternative fewer opportunities to bring its average back to the level of the certain alternative, meaning that this interval where the certain alternative is selected most of the time is longer than the previous interval when the risky alternative was selected most often. This asymmetry is the source of the risk aversion effect in our model and its preference for certainty.

One prediction of this explanation arises from its origin in the base-level learning equation that reflects the occurrence of events into the activation of decision chunks and then into the expected outcomes of the respective choices. As experience accumulates, the impact of recent events in activation fluctuations will be gradually overcome by the increasingly long history. One would therefore expect risk aversion to disappear with practice, a prediction confirmed by Figure 4, which plots the probability of choosing the certain alternative with practice (in terms of blocks of 10 trials). In the initial exploration period, both model and subjects choose the certain alternative about 50% of the time. By around trial 50 the certain alternative is chosen over 60% of the time as the payoffs statistics are quickly learned, but the bias to select certain payoffs then gradually declines back to 50% as the increasingly long history overcomes short-term fluctuations. Figure 2b (right) illustrates this learning process across blocks of 100 trials as a quickly learned propensity to choose the certain alternative gradually reverts to the mean.

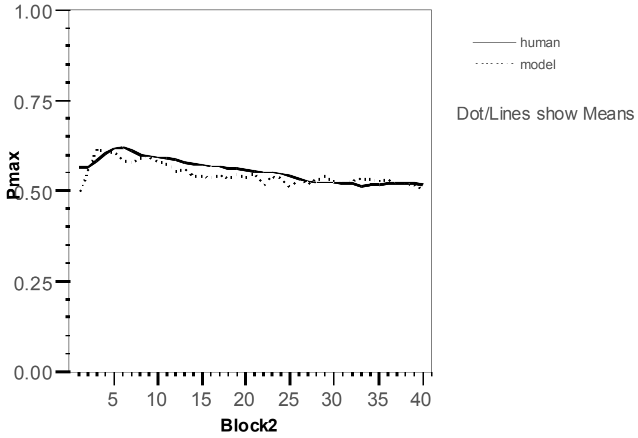


Figure 4: Time course of risk aversion effect

As we mentioned previously, one strong aspect of our model over RELACS is that it makes constrained predictions about the probability of making a given decision as a function of the recent history of choices and payoff outcomes. To study those probabilities, we used a methodology called model-tracing (Anderson et al, 1995) to force the model to make the same decisions as each human subject, thereby giving them the same context in which to make each decision. We can then directly compare each decision for model and subjects, as reported in Table 1. Columns 2 and 3 report the Pmax values for each subject and the model tracing its decisions. Columns 4 and 5 report the minimum and maximum probability of matching decisions given those base probabilities. Column 6 report the average prediction probability of agreement assuming that decisions are randomly distributed given those base probabilities while column 7 reports the actual probability of agreement. For all subjects but S8, the actual probability is higher than the predicted probability, establishing that the model is capturing some of the short-term factors used by the subjects in their decisions.

Table 1: Model tracing by subject

| ID | Subj | Model | Min | Max | Pred | Actual |
|---|---|---|---|---|---|---|
| S9 | 0.585 | 0.365 | 0.050 | 0.780 | 0.477 | 0.575 |
| S8 | 0.458 | 0.258 | 0.285 | 0.800 | 0.521 | 0.505 |
| S7 | 0.660 | 0.323 | 0.017 | 0.662 | 0.443 | 0.458 |
| S6 | 0.338 | 0.237 | 0.425 | 0.900 | 0.585 | 0.605 |
| S5 | 0.470 | 0.228 | 0.302 | 0.758 | 0.516 | 0.537 |
| S4 | 0.517 | 0.250 | 0.233 | 0.733 | 0.491 | 0.603 |
| S3 | 0.448 | 0.263 | 0.290 | 0.815 | 0.525 | 0.560 |
| S2 | 0.672 | 0.310 | 0.018 | 0.637 | 0.434 | 0.482 |
| S20 | 0.422 | 0.260 | 0.318 | 0.838 | 0.537 | 0.588 |
| S1 | 0.615 | 0.335 | 0.050 | 0.720 | 0.462 | 0.490 |
| S19 | 0.182 | 0.195 | 0.623 | 0.987 | 0.694 | 0.698 |
| S18 | 0.585 | 0.352 | 0.063 | 0.768 | 0.475 | 0.482 |
| S17 | 0.703 | 0.307 | 0.010 | 0.605 | 0.422 | 0.525 |
| S16 | 0.207 | 0.210 | 0.583 | 0.998 | 0.670 | 0.728 |
| S15 | 0.632 | 0.263 | 0.105 | 0.630 | 0.437 | 0.445 |
| S14 | 0.080 | 0.105 | 0.815 | 0.975 | 0.832 | 0.870 |
| S13 | 0.455 | 0.263 | 0.282 | 0.807 | 0.521 | 0.552 |
| S12 | 0.708 | 0.355 | 0.062 | 0.647 | 0.440 | 0.502 |
| S11 | 0.738 | 0.420 | 0.157 | 0.682 | 0.462 | 0.497 |
| S10 | 0.412 | 0.247 | 0.340 | 0.835 | 0.544 | 0.575 |

## Conclusion

Our goal in this modeling effort is to reach for breadth as well as depth from a constrained computational basis in a cognitive architecture in general and a theory of memory in particular. We primarily illustrated depth in this paper by showing how our model can account for aggregate choice and individual differences as well as new data on learning and short-term choice interdependency. In the future, we aim to emphasize breadth by unifying this model with existing models of other choice and control paradigms as well as extend its applicability to related paradigms.

## Acknowledgments

This research was partially supported by the National Science Foundation (Human and Social Dynamics: Decision, Risk, and Uncertainty, Award number: 0624228) and by the Army Research Laboratory (DAAD19-01-2-0009) awards to Cleotilde Gonzalez, and by DARPA award FA8650-05-C-7254 to Christian Lebiere.

## References

Anderson, J. R., & Lebiere, C. (1998). *The Atomic Components of Thought*. Mahwah, NJ: Erlbaum.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y . (2004). An integrated theory of the mind. *Psychological Review 111, (4).* 1036-1060.

Anderson, J. R., Corbett, A. T., Koedinger, K., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. The Journal of Learning Sciences, 4, 167-207.

Busemeyer, J. R., & Townsend, J. T. (1993). Decision field theory: A dynamic-cognitive approach to decision making in an uncertain environment. *Psychological Review, 106*(3), 432-459.

Dienes, Z., & Fahey, R. (1995). Role of specific instances in controlling a dynamic system. *Journal of Experimental Psychology: Learning, Memory and Cognition, 21*(4), 848-862.

Erev, I., & Barron, G. (2005). On adaptation, maximization, and reinforcement learning among cognitive strategies. *Psychological Review, 112*(4), 912-931.

Gonzalez, C., & Lebiere, C. (2005). Instance-based cognitive models of decision making. In D. Zizzo & A. Courakis (Eds.), *Transfer of knowledge in economic decision-making*: Macmillan (Palgrave Macmillan).

Gonzalez, C., Lerch, J. F., & Lebiere, C. (2003). Instance-based learning in dynamic decision making. *Cognitive Science, 27*(4), 591-635.

Gonzalez, C., & Quesada, J. (2003). Learning in dynamic decision making: The recognition process. *Computational and Mathematical Organization Theory, 9*(4), 287-304.

Kahneman, D., & Tversky, A. (1979). Prospect theory: An analysis of decision under risk. *Econometrica, 47*(2), 263-291.

Lebiere, C. (1999). A blending process for aggregate retrievals. In *Proceedings of the 6th ACT-R Workshop*. George Mason University, Fairfax, Va.

Lebiere, C., Wallach, D., & West, R. L. (2000). A memory-based account of the prisoner's dilemma and other 2x2 games. In *Proceedings of International Conference on Cognitive Modeling*, pp. 185-193. NL: Universal Press.

Logan, G. D. (1988). Toward an instance theory of automatization. *Psychological Review, 95*(4), 492-527.

Martin, M. K., Gonzalez, C., & Lebiere, C. (2004). Learning to make decisions in dynamic environments: ACT-R plays the beer game. In *Proceedings of the Sixth International Conference on Cognitive Modeling*, Pittsburgh, PA

O'Reilly, R. C., & Munakata, Y. (2000). *Computational Explorations in Cognitive Neuroscience*. Cambridge, MA: MIT Press.

Sanner, S., Anderson, J. R., Lebiere, C., & Lovett, M. C. (2000). Achieving efficient and cognitively plausible learning in Backgammon. *Proceedings of The Seventeenth International Conference on Machine Learning*. San Francisco: Morgan Kaufmann.

Tenenbaum, J. B., & Griffiths, T. L. (2001). Generalization, similarity and Bayesian inference. *Behavioral and Brain Sciences (24)*, 629-640.

Wallach, D. & Lebiere, C. (2003). Conscious and unconscious knowledge: Mapping to the symbolic and subsymbolic levels of a hybrid architecture. In Jimenez, L. (Ed.) *Attention and Implicit Learning*. Amsterdam, Netherlands: John Benjamins Publishing Company.

West, R. L., & Lebiere, C. (2001). Simple games as dynamic, coupled systems: Randomness and other emergent properties. *Journal of Cognitive Systems Research, 1(4)*, 221-239.

# Modeling Control Strategies in the N-Back Task

**Ion Juvina (ijuvina@cmu.edu) & Niels A. Taatgen (taatgen@cmu.edu)**
Department of Psychology, Carnegie Mellon University, 5000 Forbes Avenue
Pittsburgh, PA 15213 USA

### Abstract

Two studies aiming to investigate the use of cognitive control strategies in the N-Back task are presented. The first study identified a behavioral effect that seemed indicative of participants' proneness toward using high- versus low-control strategies. Two ACT-R models of N-Back implementing the two hypothesized strategies were developed. Model simulations were used to identify the proneness toward using high- versus low-control strategies by the individual participants in the second study. An independent measure of control – Stroop interference – was used to validate the predictions of the two models.

## Introduction and Background

Cognitive control processes are often postulated to account for behavioral effects that cannot be explained based solely on relatively better-understood cognitive processes such as perception, language or memory. Some tasks are believed to require more cognitive control than others (Garavan, Ross, Li, & Stein, 2000). This paper aims to demonstrate that there are also differences among individuals with regard to whether or not certain cognitive control processes are employed in a particular task.

The N-Back task requires judging whether a new item is identical to the *n*th-item back in a sequentially presented list of items (McElree, 2001). For example, in the sequence < $M_3$ $A_2$ $R_1$ $A_0$ > of the 2-back task, the current item ($A_0$) is identical to the 2nd-item back ($A_2$). The task requires keeping available the most recent *n* items to be compared with the incoming item. Although at each particular step only the *n*th-item can be a target, items with indexes between 0 and *n* must be remembered because they may be targets in the following steps (Awh et al., 1996). For concision, the set of the most recent *n* items will be referred to as the *rehearsal window*.

The involvement of executive control processes in N-Back is justified by the necessity to interleave different subtasks: processing incoming information, maintaining activation of recently processed and potentially relevant information (rehearsal), and discarding recently processed but irrelevant (potentially interfering) information. Evidence that these subtasks are concurrently executed comes from fMRI studies showing activation in Broca's area (BA44) indicating articulatory rehearsal, Posterior parietal area (BA40) indicating short-term storage, and Dorso-Lateral Prefrontal Cortex (BA9/46) indicating excitatory or inhibitory modulation of activation in other areas (Cohen et al., 1997; Miller & Cohen, 2001; Owen, McMillan, Laird, & Bullmore, 2005).

One of the functions of the cognitive system is to keep active the information that is relevant to the task at hand. Usually, the most frequently and recently processed information is most likely to be relevant to the current processing (Anderson, 1989). In this case, the relevant information is kept active simply because it has residual activation from recent processing. The residual activation that an item bears for a while after its use is beneficial when the item is reused, and the probability that an item will be reused gradually decreases with time. However, in the N-Back task (as well as in other tasks of this type), the switch from relevant to irrelevant is instantaneous, instead of being gradual. When an item reaches the index *n*+1 it becomes totally irrelevant. In this case, its residual activation is not only useless but it may cause interference. An active control mechanism is needed to temporarily increase or decrease activation of a particular item depending on whether or not this item is relevant for the current state of the task.

## First Study

The goal of this study was to investigate behavioral effects and individual differences in performing the N-Back task as basis for building cognitive models of this task.

### Participants

Forty-one volunteers from the Carnegie Mellon University's community participated in this study (average age 26; 23 women and 18 men). They received a fixed amount of monetary compensation for their participation.

### Design

This study used a within-subjects design with three conditions: 2-, 3-, and 4-back, administrated in this fixed order.

### Materials

The N-Back task was administrated with the aid of dedicated software. Stimuli were capital letters appearing on the computer's screen one after another at a rate of 2.5 seconds per stimulus. Each participant received approximately 10 targets and 5 foils per condition. Reactions from participants were taken with the aid of a standard keyboard, and written feedback was presented on the screen.

### Procedure

The administration of the N-Back task was based on a continuous recognition paradigm, that is, there was one stream of stimuli per condition and judgments were made after each item was presented. Based on these judgments, an item could be classified as: *target* (positive test probe) when it was identical with the *n*th-item back; *foil* or *lure* (negative test probe) when it was identical with the item presented at position *n*-1 or *n*+1 back; *distractor* (non-test probe) when it was not identical with a recently presented item. Occurrences of targets and foils in the stream of stimuli were not interleaved with one another; they were separated by a variable (random) number of distractors. Thus, the moment of occurrence for a target or a foil was unpredictable for the participants.

Participants were instructed to hit the key "M" on the keyboard when the current stimulus was identified as a target and the key "Z" when the current stimulus was identified as a foil or distractor. For the latter case, a non-reaction was also considered a valid option. Feedback was offered only for correct and erroneous reactions; feedback was not offered in case of non-reactions. A performance score increasing and decreasing in value with correct and incorrect reactions to targets, respectively, was continuously displayed on the screen.

## Results

Table 1 presents the rate of correct reactions to targets and foils by condition. In general, correctness decreases with *n*; this effect is consistent across participants and in accord with previous studies (McElree, 2001).

Table 1: The rate of correct answers by condition.

| | N2 | N3 | N4 |
|---|---|---|---|
| Targets | 0.72 | 0.55 | 0.46 |
| Foils | 0.84 | 0.57 | 0.59 |

Unexpectedly, the correlation between correctness on targets and correctness on foils was negative ($r_{39}$=-0.53, p=0.0004). Participants tended to score either high on targets and low on foils or vice-versa. This is an indication that some of the participants manifested what we called a "react-to-repetition" effect: they were tempted to react to a repeated item regardless whether they knew or not that it was a target or a foil. Since the number of targets was higher than the number of foils such a strategy would pay off overall. Other participants, who scored low on targets, scored high on foils because non-reaction to foils counted as correct answer. In both cases the correctness score was artificially increased.

An indication of possible use of different strategies was the so-called *serial position effect*. The serial position of an item is its distance from the last target or foil in the stream of stimuli. For example, the current target ($T_0$) in the following stream of targets (T) and distractors (D) <$T_1$,D,$T_1$,D,D,D,$T_0$,D,$T_0$> appears on the fourth position after the previous target ($T_1$), so its serial position is 4. Some participants decreased their performance with serial position (see Fig. 1), and this may be an indication of using a high-control strategy.

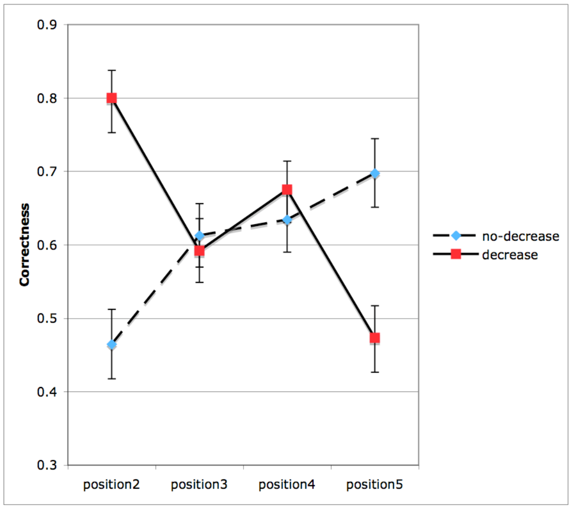


Figure 1: The serial position effect.

The two groups in figure 1 were formed by visual inspection of data of individual participants. The group showing the serial position effect ("decrease") is composed of 17 participants while the group not showing this effect ("no-decrease") contains 24 participants. The apparent increase in performance with serial position for the "no-decrease" group is most probably caused by the artificial increase in correctness due to either reaction-to-repetition or non-reaction, as described above.

## Discussion

This study allowed us to gain some initial insight into how participants approached the N-Back task. The unexpected negative correlation between correctness on targets and correctness on foils made us aware of the importance of distinguishing between *judgments of familiarity* and *judgments of recency* in the N-Back task (McElree, 2001). A judgment of familiarity refers to recognizing whether or not an item has been recently presented. A judgment of recency involves deciding whether the recently presented item appeared in a particular position (e.g., *n*-back). The latter type of judgments helps in differentiating targets from foils and is more likely to require cognitive control processes (Smith & Jonides, 1999).

When participants relied solely on correct judgments of familiarity and reacted to any repetition, their correctness on targets was artificially increased (since there were more targets than foils in the stream of stimuli) at the expense of decreased correctness on foils. When a repeated item was not recognized as familiar (i.e., recently presented), a non-reaction caused low correctness on targets and artificially high correctness on foils (because non-reaction to foils counted as correct answer). These two effects combined caused the negative correlation between targets and foils. Relying solely on judgments of familiarity could be a deliberate strategy or just a consequence of failed judgments of recency.

With regard to judgments of recency, participants seemed to employ two different strategies:

- One group of participants manifested the serial position effect – decrease in performance with serial position. This effect can be explained only by assuming that participants used some sort of rehearsal and this processes was vulnerable to distraction. Participants tried to actively maintain the rehearsal window and discard past items falling outside of it. The more distractors were to be discarded from the rehearsal window (i.e., the higher the serial position), the lower the accuracy of recency judgments.
- Another group of participants does not show any behavioral trace that could indicate the use of a rehearsal process. It is unclear on what these participants base their judgments of recency. A possible explanation is the "time tag" account of Yntema and Trask (1963). They suggested that one component of the memory trace of a past event is a tag that in some way directly indicates when the event occurred.

In conclusion, this exploratory study showed us that N-Back is a task prone to strategizing. The negative correlation between correctness on targets and correctness on foils was an artifact caused by the relative frequencies of targets and foils in the stream of stimuli and the way responses were collected (non-reactions to foils counted as correct responses). The serial position effect allowed us to hypothesize that some of the participants used a high-control strategy based on rehearsal, while other participants used a low-control strategy based on time estimation.

## ACT-R Models of N-Back

Based on the insight gained in the first study, two ACT-R models of N-Back were developed corresponding to the two aforementioned strategies participants were assumed to employ for making judgments of recency. A high-control model implements a rehearsal mechanism with the aid of the articulatory loop (Baddeley, 2000) and a low-control model implements the "time tag" account (Yntema & Trask, 1963). These two models differ from each other only with regard to the control strategy they implement; for the rest, they are identical in the sense that they have the same architectural parameters.

### High-control Model

The main assumption of this model is that participants maintain a rehearsal window of size *n*, and actively suppress items that are dropped from this window. One way to implement a rehearsal window is by making use of the phonological loop. Phonological rehearsal is supported by our own behavioral observation (sometimes participants would vocalize aloud), reported empirical effects showing decrease in performance when phonological rehearsal is suppressed (Baddeley, 2000), and brain imaging findings showing activation of Broca's area during performance on the N-Back task (Awh et al., 1996).

The model attends to incoming stimuli and judges their familiarity, that is, compares them with past items retrieved from declarative memory. Due to ACT-R's memory decay mechanism, only a few of the most recent items can be retrieved, and the chance of an item to be retrieved increases with its recency. This is the main reason for the observed decrease in performance with *n* (see Table 1 and Figure 2).

As processing progresses through the stream of stimuli, the model develops and maintains the rehearsal window. When a new stimulus is visually perceived, it is also sub-vocalized, thus its sound is made available to the auditory module. However, the auditory module cannot attend to it immediately because it is busy with attending past and rehearsed items; it just adds it to a cue of items to be attended to later as the auditory module becomes available. In the interval between two stimuli (2.5s), the model tries to sub-vocalize the most recent *n* items. They are taken from the cue of the auditory module (the phonological store).

When the current item has been found to be a repetition of a recent one (a judgment of familiarity), it is also matched against the content of the aural buffer to allow a judgment of recency: if its content is the same as the content of the item in the aural buffer it is judged as target, otherwise it is judged as foil. A judgment of recency is as accurate as the phonological loop is.

The proper functioning of the phonological loop depends on reliably maintaining its size and content. This amounts to discarding an item from the loop whenever a new one is added and preventing discarded items from reentering the loop. Discarded items can reenter the loop via retrieval. An inhibitory control process is necessary to ensure that discarded items do not reenter the rehearsal window. A temporary storage buffer (ACT-R's imaginal buffer) holds the discarded items and spreads negative activation (i.e., suppression) to their corresponding elements in declarative memory. This way the model ensures that discarded items are not retrieved and cannot reenter the rehearsal window. However, the amount of available suppression is limited and it is evenly spread among all discarded items. Thus, the more items are to be suppressed, the less effective suppression is. This is how the model shows the serial position effect. Evidence for linear increase in activation of cortical areas involved in rehearsal with working memory load has recently been reported (Zarahn, Rakitin, Abela, Flynn, & Stern, 2005).

### Low-control Model

The main assumption of this model is that participants are not rehearsing. They make judgments of recency based on learned time estimations. This assumption is inspired by the "time tag" account (Yntema & Trask, 1963) and is supported by our results from the first study showing that some participants do not manifest the serial position effect.

The low-control model makes judgments of familiarity in the same way as the high-control model. The key difference is in making judgments of recency. When an item is encoded it is attached with a time tag specifying the moment of its encoding. The temporal module of ACT-R is used for assigning time tags and making time estimations (Taatgen, Anderson, Dickison, & van Rijn, 2005 ). The default parameters of this module were used. When a recent item that is identical to the current item is retrieved, the model determines the time lag between the two presentations and tries to determine whether this time lag is equal to the target duration – the one needed for making correct judgments of recency. The model does not know in advance what the target duration is and has to learn it from its own experience. As a result of this learning process, any correct estimation of how long ago the *n*th-item back has been presented can serve as target duration. Thus, the model tries to retrieve the target duration and, if a correct estimation of it cannot be retrieved, the model reacts to a repeated item as it were a target. This reaction causes the system to produce feedback and the model uses this feedback to tag its recent estimation as correct or wrong. If this estimation happens to be correct it will be retrieved next time when the same time lag is found between a current item and a recent one. The more correct estimations are accumulated in memory the higher the chance that the model will make correct judgments of recency. Due to the intrinsic noise of the temporal module, time estimations are never perfect.

The essential characteristic of this process is that it does not depend on the serial position at which a target (or foil) is presented.

### Models Fit

Figure 2 shows how the two models fit the data from the first study. The two models were allowed to under-fit the data, as justified by the observation that the correctness score was artificially inflated, as explained in the sections describing the first study. The N4 condition was dropped because it had a low correctness score (see Table 1) and also a very high vulnerability to be affected by the artifact described above, as shown by the highest magnitude of the negative correlation between targets and foils in this condition ($r_{38}$=-0.49, p=0.002).

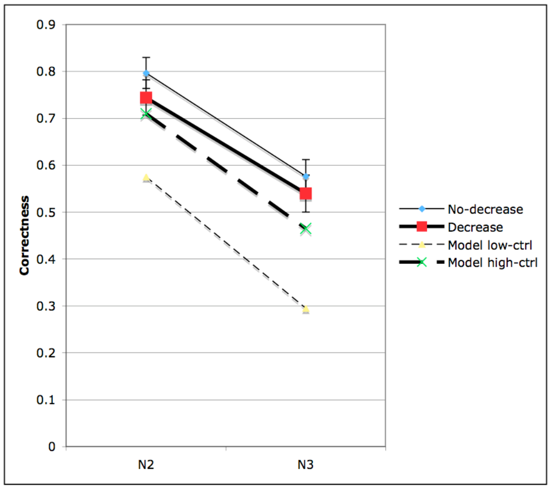


Figure 2: Data from the first study (solid lines) and model simulations (dashed lines). Thick lines indicate high control and thin lines indicate low control. Vertical bars indicate standard error of the means.

The two models show the same decrease in performance with *n* as shown in the data. The value of the retrieval threshold parameter of ACT-R was set to -0.35 (default 0.0) to fit the observed difference in the data.

The two models make qualitatively different predictions with regard to the serial position effect (Fig. 3). The high-control model predicts that maintaining a rehearsal window allows high performance at low serial positions but performance decreases at higher serial positions, as it becomes harder and harder to maintain the rehearsal window in the face of distraction.

The low-control model predicts that there is no reason for performance to vary with serial position because no rehearsal process is employed in making judgments of recency. The level of performance is given by the accuracy of time estimations, which in turn depends on the noise in the ACT-R's temporal module and the opportunities the model has to learn correct time estimations.

## Second Study

This study was intended to correct the artifact found in the first study and check the hypothesis about involvement of different control strategies in N-Back. The number of foils was made equal with the number of targets and participants were asked to explicitly reject foils, that is, non-reaction to foils did not count as correct answer. These changes were expected to bring about a more valid measure of performance in the N-Back task.

It was hypothesized that participants showing the serial position effect are prone to using a high-control strategy not only in the N-Back task but also in another control-demanding task (Stroop). It was also hypothesized that manipulating the presentation rate of stimuli (inter-stimuli interval – ISI) would trigger behavioral effects that would help us distinguishing between various control strategies.

### Participants

Fifty-two volunteers from the Carnegie Mellon University's community participated in this study (average age 24; 16 women and 36 men). They received a fixed amount of monetary compensation for their participation.

### Design

A within-subjects design has been employed with the N-Back task and the Stroop task presented one after another in this order. Only the N2 and N3 conditions from the N-Back task were retained. The N4 condition was left out based on results of the first study showing very low performance in this condition (see Table 1). The N-Back task was administrated with two presentation rates (ISI): 2.5s and 1.5s. For this manipulation, order was counterbalanced: half of the participants received the fast ISI (1.5s) first and the other half received the slow ISI (2.5s) first. The Stroop task (MacLeod, 1991) had the three standard conditions – incongruent, congruent and neutral – randomly interleaved with one another and with an equal number of trials in each condition.

### Materials

The same materials as in the first study were used for the N-Back task. Small modifications in the software were made to balance the numbers of targets and foils, collect reactions for both targets and foils, and implement the speed manipulation. A computerized version of the standard Stroop task was implemented.

## Procedure

Administration of the N-Back task followed the same procedure as in the first study, except participants received additional instructions regarding how foils must be rejected. Participants were informed that a successful rejection of a foil is rewarded with one point and a correct identification of a target is rewarded with two points. Participants were not informed about the change in speed (ISI) that would occur during the experiment.

For the Stroop task participants received a short screen-based tutorial to ensure proper understanding of the task.

## Results

As a result of the changes in the administration of N-Back, correctness rates were decreased overall and in particular for foils (see Table 2 and compare with Table 1), as compared with the first study.

Table 2: The rate of correct answers by condition.

| | N2 | N3 |
|---|---|---|
| Targets | 0.67 | 0.52 |
| Foils | 0.45 | 0.29 |

The correlation between correctness on targets and correctness on foils has become positive ($r_{51}$=0.43, p=0.001). Thus, the ability to identify targets and reject foils was better indicated by the correctness score in the second study as compared with the first study.

With regard to the serial position effect the two aforementioned models produced qualitatively different predictions (see Fig. 3). These models were used to identify each participant's proneness toward using high- vs. low-control strategies. Each participant's data were compared with the two predictions. If the data of one participant fit the prediction of the high-control model better than the prediction of the low-control model, that participant would be classified in the high-control group, and vice-versa. The root-mean-square-deviation measure was used for fitting the data of individual participants to the two model predictions. Figure 3 shows the two model predictions and the data of the two groups of participants formed based on how well individual participants fit these predictions (only serial positions 2, 3, and 4 had enough data for a reliable analysis). The high-control group was composed of 23 participants and the low-control group was composed of 29 participants. It turned out that the high-control group had also higher overall performance than the low-control group.

To verify that the two groups of participants formed based on the two different model predictions are indeed different from a cognitive control perspective, an independent measure of cognitive control was considered. Stroop interference is one of the most frequently mentioned measures of cognitive control (Miyake et al., 2000). It is computed as the difference in reaction time between incongruent and neutral trials. A one-way analysis of variance with Stroop interference as a dependent variable and the grouping variable distinguishing between high- and low-control participants as a factor was conducted and showed a significant effect in the expected direction ($F_{1,50}$=5.36, p=0.02, mean(HC)=111ms, mean(LC)=179ms). The Stroop interference manifested by high-control participants (HC) was lower in magnitude with an average of 68ms than the Stroop interference of low-control participants (LC).

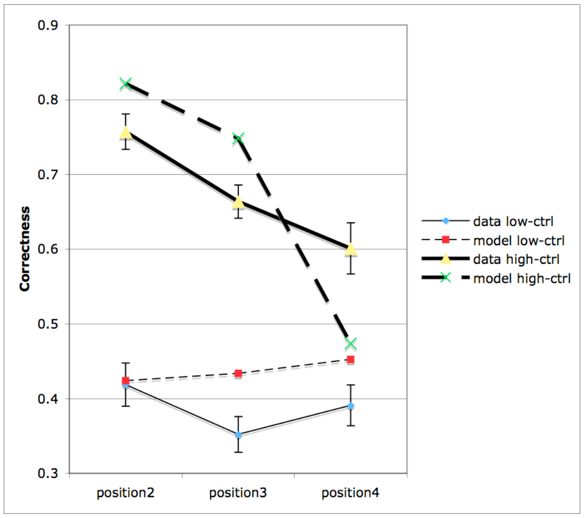


Figure 3: Model predictions (dashed lines) and their corresponding groups of participants (solid lines). Thick lines indicate high control whereas thin lines indicate low control. Vertical bars indicate standard error of the means.

It was not clear whether and to what extent performance at N-Back varied with speed. The effect of the speed change was confounded by a strong learning effect (see Table 3). Most participants who started with the fast condition (ISI=1.5s) had lower performance in this condition and most participants who started with the slow condition (ISI=2.5s) had lower performance in this condition.

Table 3: Confounding between speed and learning effects.

| | Start fast | Start slow |
|---|---|---|
| Lower performance in the slow condition | 2 | 16 |
| Lower performance in the fast condition | 24 | 9 |

## Discussion

The interpretation that scores in the first study were biased and artificially inflated by the way the task was administrated proved correct. The changes operated to the task for the second study corrected this problem. As a consequence, the correctness score for both targets and foils now accurately indicates judgments of familiarity and recency.

Participants‘ proneness toward using high- versus low-control strategies was determined by the aid of the two corresponding model predictions. Participants were assigned to two different groups based on how well their behavioral data fit the simulations of the two models. Participants that were identified as prone toward using a high-control strategy showed lower Stroop interference than participants identified as being prone toward using a low-control strategy.

The speed manipulation was confounded by a strong learning effect. Both effects deserve further investigation. Arguably, speed must have a negative influence on the high-control strategy and either none or a positive influence on the low-control strategy. It would be interesting to investigate how learning relates to using these strategies and whether and when participants switch strategies.

## General Discussion and Conclusion

The first study found behavioral traces (serial position effect) indicating the use of a high-control strategy in some of the participants and not in others. This effect was unequivocally interpreted as indicating the use of a rehearsal process vulnerable to distraction. Participants showing this effect were assumed to use the phonological loop to maintain active a rehearsal window of size *n* and inhibitory control to discard items falling outside of this window. Participants not showing this effect were assumed to use time estimations for their judgments of recency. Two ACT-R models were developed based on these assumptions.

Model predictions were used to categorize participants in the second study as prone to using high- or low-control strategies. High-control participants were shown to manifest lower Stroop interference than low-control participants. This result validates the assumption that the serial position effect is an indicator of using a high-control strategy. It can be argued that some of the specific modeling mechanisms used in these models are not unique. For example, a rehearsal process does not necessarily require the phonological loop (Logie, Venneri, Della Sala, Redpath, & Marshall, 2003) and rehearsal-independent judgments of recency can be implemented without assuming a time estimation mechanism (McElree, 2001). However, a distractor-suppression mechanism seems necessary to account for the serial position effect. To the best of our knowledge, no other modeling account has been proposed so far for this effect.

In conclusion, this paper argued that there are substantial individual differences with regard to whether or not certain cognitive control mechanisms are employed in particular tasks. It can be asserted that not only some tasks require more control than others but also some individuals are prone to using more control than others.

## Acknowledgments

This work was supported by Office of Naval Research grant N00014-06-1-005. Thanks to Daniel Dickison and Jelmer Borst for their comments on a draft of this paper.

## References

Anderson, J. R. (1989). Human Memory: An Adaptive Perspective. *Psychological Review*, 96, 703-719.

Awh, E., Jonides, J., Smith, E. E., Schumacher, E. H., Koeppe, R. A., & Katz, S. (1996). Dissociation of storage and rehearsal in verbal working memory: Evidence from positron emission tomography *Psychological Science*, 7, 25-31.

Baddeley, A. D. (2000). The episodic buffer: a new component of working memory? *Trends in Cognitive Sciences*, 4, 417-423.

Cohen, J. D., Perlstein, W. M., Braver, T. S., Nystrom, L. E., Noll, D. C., Jonides, J., et al. (1997). Temporal dynamics of brain activation during a working memory task *Nature*, 386, 604-608.

Garavan, H., Ross, T. J., Li, S.-J., & Stein, E. A. (2000). A Parametric Manipulation of Central Executive Functioning. *Cerebral Cortex*, 10, 585-592.

Logie, R. H., Venneri, A., Della Sala, S., Redpath, T. W., & Marshall, I. (2003). Brain activation and the phonological loop: The impact of rehearsal. *Brain and Cognition*, 53, 293-296.

MacLeod, C. M. (1991). Half a Century of Research on the Stroop Effect: An Integrative Review *Psychological Bulletin*, 109(2), 163-203.

McElree, B. (2001). Working memory and focal attention. *Journal of Experimental Psychology: Learning, Memory & Cognition*, 27, 817-835.

Miller, E. K., & Cohen, J. D. (2001). An integrative theory of prefrontal cortex function. *Annu. Rev. Neurosci.*, 24, 167–202.

Miyake, A., Friedman, N. P., Emerson, M. J., Witzki, A. H., Howerter, A., & Wager, T. D. (2000). The unity and diversity of executive functions and their contributions to complex "frontal lobe" tasks: A latent variable analysis. *Cognitive Psychology*, 41, 49-100.

Owen, A. M., McMillan, K. M., Laird, A. R., & Bullmore, E. (2005). N-Back Working Memory Paradigm: A Meta-Analysis of Normative Functional Neuroimaging Studies *Human Brain Mapping*, 25, 46-59.

Smith, E. E., & Jonides, J. (1999). Storage and Executive Processes in the Frontal Lobes *Science*, 283, 1657-1661.

Taatgen, N. A., Anderson, J. R., Dickison, D., & van Rijn, H. (2005 ). Time Interval Estimation: Internal Clock or Attentional Mechanism? Paper presented at the *CogSci05 - The 27th Annual Conference of the Cognitive Science Society*, Mahwah, NJ.

Yntema, D. B., & Trask, F. P. (1963). Recall as a search process. *Journal of Verbal Learning and Verbal Behavior*, 2, 65-74.

Zarahn, E., Rakitin, B., Abela, D., Flynn, J., & Stern, Y. (2005). Positive Evidence against Human Hippocampal Involvement in Working Memory Maintenance of Familiar Stimuli. *Cerebral Cortex*, 15, 303-316.

# ACT-R Models of Cognitive Control in the Abstract Decision Making Task

**Daniel Dickison (danieldickison@cmu.edu)**
**Niels A. Taatgen (taatgen@cmu.edu)**
Department of Psychology, Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15201

## Abstract

This paper discusses a method of modeling individual differences in cognitive control by developing models that differ in control structure. Such a strategy for modeling behavior is necessitated when tasks are complex and individual differences in performance vary on many measures of performance. For these tasks, merely adjusting a parameter to fit various groups of subjects may be impractical or impossible. Two such models for the Abstract Decision Making task are implemented in ACT-R to fit the performance of high and low-control subjects in the first experiment. These models are then used to accurately predict performance on a second experiment that involves novel "games" unrepresented in the prior experiment.

**Keywords:** Individual differences; cognitive control; ACT-R

## Introduction

Computational models of cognition are often developed to fit the average performance on a task by a population. While this method suffices when describing phenomena that are observed widely, it purposefully ignores individual differences. One way to model individual differences is to propose some parameter – such as working memory capacity – that varies among individuals and observe how changes in that parameter affect the model's performance (Daily, Lovett & Reder, 2001; Taatgen, 2002; Rehling, Demiral, Lebiere, Lovett, & Reder, 2003; Chuderski, Stettner & Orzechowski, 2006).

However, when tasks are more complex, manipulating isolated parameters to fit individual differences becomes more challenging. For these types of tasks, it is possible to assume that individuals adopt different control structures that manifest themselves as different problem solving strategies. The different degrees of cognitive control can be characterized as the amount of top-down control that an individual exerts when completing a task, as opposed to behavior being primarily driven by bottom-up processes (Taatgen, 2007). This paper explores the development of distinct models using the ACT-R architecture (Anderson, 2007) that differ in control structure to describe individual performance on the Abstract Decision Making (ADM) task (Joslyn & Hunt, 1998).

## Abstract Decision Making Task

ADM is a task developed by Joslyn and Hunt (1998) designed to predict individuals' performance on various real-world tasks that require decision making under time pressure. The task was used in a battery of tests designed to show individual differences in cognitive control.

ADM involves 5-minute long "games" that require subjects to classify objects into 1 of 4 bins according to their attributes. Both objects and bins have 3 attributes: size, color and shape. Only objects that match a particular bin's attributes are allowed in the bin. While objects always have 3 concrete attributes (e.g. small, red, circle), bins may specify any subset of the attributes (e.g. all circles). The number of attributes that a bin omits (i.e. the number of those that are wildcards) will be referred to as its *generality*. Thus, an "all circles" bin has a generality of 2 because it does not specify size or color. The generality of a game is defined as the greatest generality of its 4 bins.

Each game consists of 4 different bins that subjects study for as long as they want prior to starting the game. Because reviewing bins during the game is a slow and hence costly action, it is in the subject's interest to memorize the attributes of the bins. During the game, an object becomes available every 15 seconds with a pop-up notification that the subject must dismiss before continuing. None of the attributes of objects are immediately visible even when they become available. To reveal one attribute of an available object, the subject must query it by typing in the appropriate commands. To assign an object to a bin, the subject must type in a different set of commands. The subject receives points if an assignment is correct and loses points if it is incorrect. The magnitude of reward or penalty is inversely proportional to the generality of the bin – that is, more specific bins award more points.

Because there is a time limit and objects are presented quicker than most subjects can classify them, there is a time pressure urging subjects to act as quickly as possible. To analyze performance on the ADM task, we specified several measures that could be computed per subject for each game. The most obvious is the *score* measure, which is identical to the cumulative points that the subject earns for assigning objects to bins. For analysis, we normalized scores to a proportion of total possible points in a given game. Because penalties are assessed for incorrect assignments, negative scores are possible. A major problem with the score measure is that various factors play into determining the final score – including accuracy, speed, and whether subjects assigned objects to the more lucrative specific bins. To help tease those factors apart, *idealness* was defined as the proportion of correct assignments that were ideal. For any given object, there can be more than 1 bin to which it can be assigned, with 1 bin being the most specific and lucrative. *Idealness* is the proportion of correct assignments where the subject chose the best bin. Finally, *queries* was defined as the average number of attribute queries a subject made of an object before attempting to assign it for the first time. This can range from 0 meaning the subject never queried any attributes – quite a suboptimal strategy – to 3 meaning the subject always queried every attribute. Values greater than 3 are also possible if the subject queried an attribute more than once.

## Experiments

Two experiments were conducted. The first allowed the generation of models. The same models were used to predict subject performance in the second experiment, the results of which were used to validate the models.

### Experiment 1

The task consisted of 2 practice games and 4 real games. Practice games used just 3 bins and had 20 seconds between object presentations. The 4 real games consisted of 2 games each of generality 0 and generality 1. Forty-one people from the Carnegie Mellon University community participated as part of a larger experiment studying individual differences. Models were developed to fit the pattern of data observed in experiment 1.

### Experiment 2

The task consisted of just 1 practice game followed by 4 real games. The real games increased in generality from 0 in game 2 to 3 in game 5. Fifty-three subjects participated in experiment 2.

## Models

Two ACT-R models were developed to describe subject performance in the ADM task – one for a high-control strategy and one for a low-control strategy. Because it is unclear which aspects of the task give rise to particular subject performances, an attempt was made to model the task as closely as possible to the actual experiment. The model, like subjects, must type in certain commands in response to text prompts to query and assign objects. This is accomplished via ACT-R's perceptual and motor modules, which simulate the amount of time required to process visual stimuli and perform key presses. The amount of time required for the models to process textual prompts was adjusted so that the models classified roughly the same number of objects per game as did subjects. This was a simplification to reflect the amount of time people need to parse a prompt, because the models were able to deduce the current state of the textual interface as soon as the visual stimuli was encoded. One further simplification made in modeling the task was that the new object pop-up notifications were omitted for the model runs.

In both models, the objects are processed in order as they become available. Neither model moves on to a subsequent object until the current object has been successfully assigned to a bin. Each bin is stored in declarative memory. A representation of the current object is kept in the imaginal buffer, which is ACT-R's mechanism for temporarily storing the current problem state. The imaginal buffer representation includes slots for the object attributes that get filled in as they are queried. The imaginal buffer was configured to spread activation to items in declarative memory, so that when deciding which bin to assign an object to, bins that had matching attributes to those of the imaginal buffer representation were more likely to be retrieved than other bins.

The differences between the low-control and high-control models are outlined below.

### Low Control Model

This model (Figure 1) makes use of the bottom-up process of expected utility available as a module in ACT-R (Anderson, 2007) to decide at each step whether to query more object attributes or to attempt an assignment. Two exceptions are when no attributes of an object are known, then it will always query, and, conversely, when all attributes are known, it will not query any more. The "query" production rule's utility is held constant while the "retrieve bin" production rule will gain or lose utility based on past successes and failures. When an assignment is correct, a positive reward is propagated backwards through the production rules that had fired leading up to the assignment. As a result, the "retrieve bin" production rule gains utility relative to the "query" rule, thus, in the future the model is more likely to ask fewer questions before assigning. On the other hand, when an incorrect assignment is made, a negative reward (i.e. a penalty) is propagated to the "retrieve bin" rule, leading to more queries before assignment.

When a bin is retrieved, it is checked against the currently known attributes of the object in question. If there is a mismatch, the model reverts to the state where it may attempt another retrieval or query for another attribute. Otherwise, if it is a match or if it *might* be a match, it will attempt to assign.

This model makes incorrect assignments when the retrieved bin specifies attributes that have not yet been revealed from the object. For example, after only having determined that a given object is red, the model may retrieve a bin that will take red circles. Even though the red object may not be a circle, the low control model will still try to assign the object to this bin, possibly resulting in an error. This leads to a corresponding negative reward, thus slightly biasing the model towards querying more in the future.

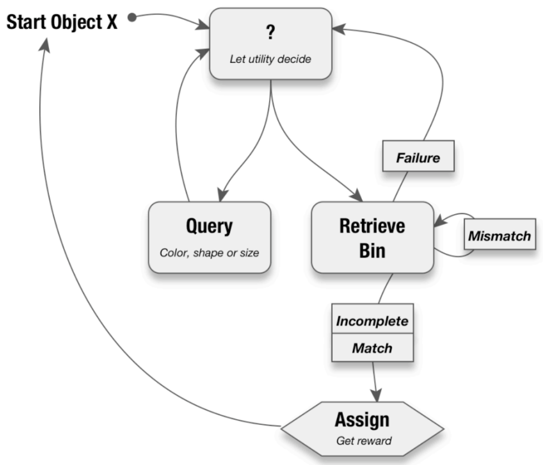


Figure 1: Low control model flow diagram.

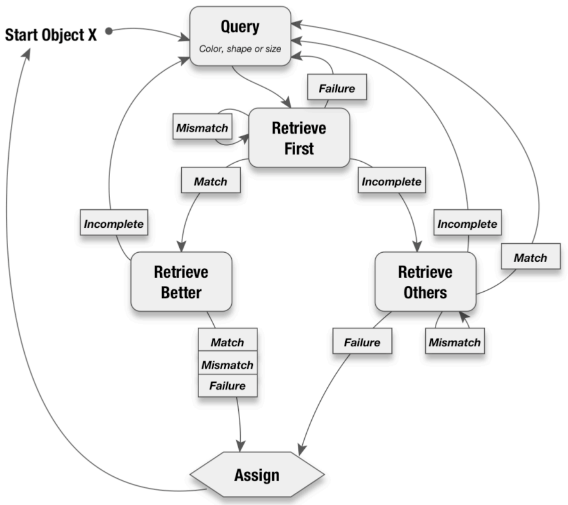


Figure 2: High control model flow diagram.

## High Control Model

The high control model (Figure 2) follows a more disciplined strategy by exerting additional top-down control. Instead of relying on the expected utility of querying versus assigning, the model attempts to maximize correctness and points gained by evaluating one bin at a time to see if it will take a given object. It also tries to minimize queries by assigning to a potentially matching bin if no other candidates can be retrieved. Specifically, it will always query first, then, after each query, it will attempt to retrieve a matching bin. If a matching bin is found, it tries to retrieve a better match – if one is found, it assigns the object to the better bin, and if one is not found, it assigns to the original match it had found earlier. If, after the original retrieval, a *maybe* matching bin is retrieved, it attempts to retrieve other bins. If no other matches are found, then it will assign to the first "maybe" bin it had retrieved, on the assumption that the object must fit the only possible bin.

An example will help illustrate the model's logic and how it minimizes the number of queries while maintaining accuracy. Suppose there are just 2 bins, one for small red circles (called A) and another for any small red shape (called B). When the model starts processing an object, it may query the object's shape. Suppose this object is a triangle. The model now tries to retrieve a bin, and retrieves the bin B. The model recognizes that the triangle may fit in this bin, but the color and size could possibly mismatch. The model now moves to the "retrieve others" state and attempts to retrieve other bins. It retrieves bin A, and sees that the triangle is a mismatch for this circles-only bin and discards it. It then fails to retrieve any other bins because there are no other bins for this game. It then concludes that the first bin – bin B – must be the correct bin, and assigns the triangle to the small red shapes bin. Thus, after just 1 query the model was able to correctly classify the object.

This model only makes errors when it fails to retrieve matching bins after first having found a possible candidate. This is, during the "retrieve others" state, it fails to retrieve a "maybe" bin even when one exists in its declarative memory. This outcome results from the fact that bin retrievals are subject to activation decay and noise.

## Results and Model Fits

In order to fit the models to subjects, subjects were divided into a high-control group and a low-control group based on certain idealness and queries criteria. Each model was then developed to fit a group of subjects in experiment 1. Apart from the control structures of the models, their speed of processing textual prompts was adjusted so that they classified roughly the same number of objects that subjects classified during each game. Finally, predictions were made for experiment 2 using the exact same models, and their outputs were compared to data observed in experiment 2.

### Control Grouping

The criteria for splitting subjects into groups was based on the idealness and queries measures for the games which had generalities of 0 and 1 – that is, games involving no wildcards in bin attributes, and games additionally involving bins with 1 wildcard, respectively. Because in games with generality 0 each object can only be assigned to 1 bin, the idealness measure is necessarily 1.0 for all subjects. With the addition of generality 1 bins, each object can be assigned to either 1 bin (the most specific bin), or to 2 bins. If a subject always assigns to the best bin, idealness will still be 1.0. If, on the other hand, the subject assigns to the more general, less lucrative bin(s), idealness will decrease. Figure 3, shows the distribution of subjects' idealness measures for experiment 1, generality 1, while Figure 4 shows the same measure for the varying game generalities. A somewhat bimodal distribution is evident at generality 1, with many subjects getting a perfect 1.0 and a cluster of subjects getting a lower value around 0.7 through 0.9. The bimodality is further exemplified in games of increasing generality. These distributions suggest that two distinct strategies were used by subjects, rather than subjects varying on one continuous dimension. We deemed subjects who had idealness greater than 0.9 in the generality 1 games to be potentially high-control, and others as low-control.

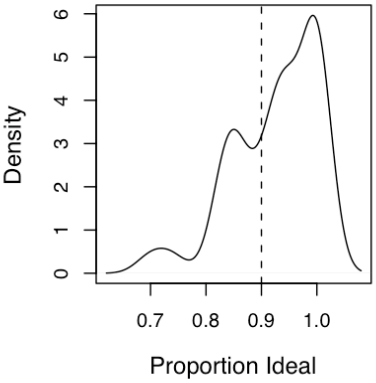


Figure 3: Distribution of experiment 1 subjects' *idealness* measure for games of generality 1 as a density plot. The dashed line shows the cutoff for high-control classification on the idealness measure.

In addition to the idealness criterion, a criterion based on the number of queries was used for control group classification. The motivation for adding another criterion is that using the idealness criterion alone, subjects would be classified as high-control if they simply attempted to assign objects to the most lucrative bins first, and, failing that, try the more general bins. This strategy maximizes idealness while minimizing the number of queries to the detriment of overall score. In order to filter out such strategy takers from high control classification, we required subjects to show an increase in queries when playing generality 1 games as compared to when playing generality 0 games. Because of the configuration of bins, more queries are required to find the ideal bin when there are wildcards involved.

In experiment 1, 27 subjects out of 41 were classified as high control on the idealness criterion, with 25 of those also matching the queries criterion for high control. For experiment 2, these numbers were 24 out of 53 for the idealness criterion and 18 of those matching the queries criterion also.

## Model Comparison

Having grouped subjects thus, the outputs produced by the models were compared to the observed data. Figure 5 shows the overall score, number of queries, and proportion ideal for each game generality in experiment 1. The two practice games are excluded from analysis, and the 2 games of each generality are collapsed.

Of note is that, regarding idealness, the low control model reflects the degree to which subjects decrease idealness in generality 1 games, while the high control model maintains a near-perfect mark. Furthermore, the high control model increases the number of queries as it moves to the more general games as do subjects. This is noteworthy because the high control model is designed to minimize the number of queries, but it is sensitive to the fact that more queries are necessary for high performance, a feat made possible by its complex control structure.

The group classifications are good predictors of overall score in experiment 1, producing a main effect of group on score, $F(1, 160) = 52.6$, $p < 0.01$, in addition to a main effect of game generality on score, $F(1, 160) = 20.4$, $p < 0.01$.

Figure 6 shows the same measures plotted for experiment 2. Note that in this experiment, there was 1 game each of the 4 levels of generality, and there was only 1 practice game as opposed to 2. That may explain the lower overall performance of subjects in this experiment compared to experiment 1.

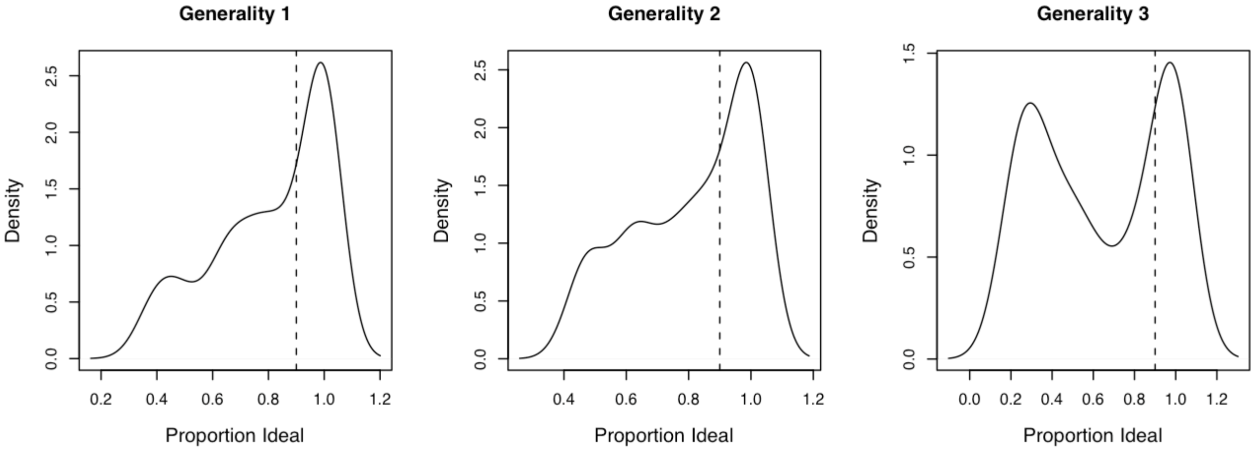


Figure 4: Distributions of experiment 2 subjects' *idealness* measure for games of varying generality.

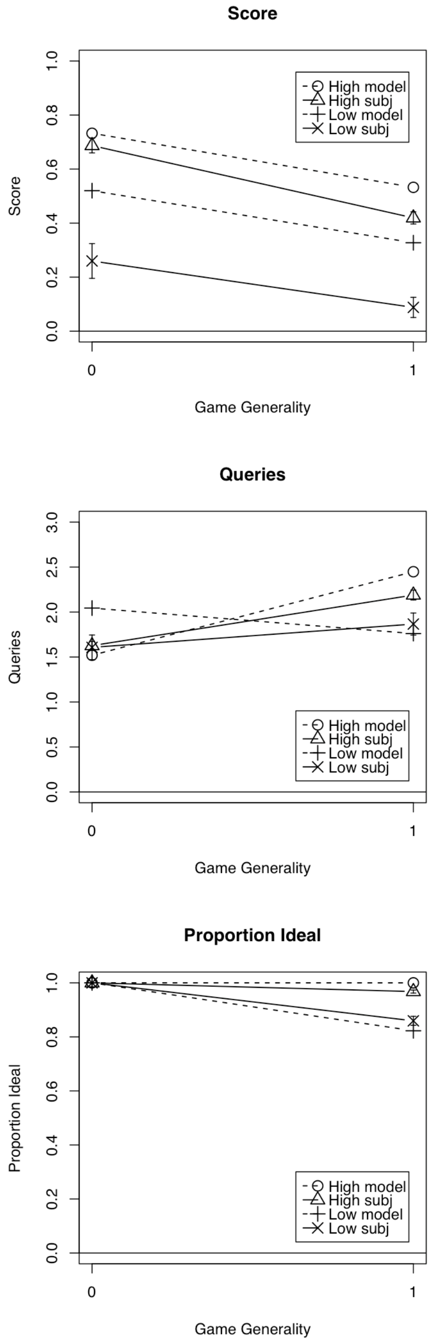


Figure 5: Experiment 1 data (solid lines) and model fits (dashed lines) for high control (open shapes) and low control (crosses). Error bars, shown only for subject data, represent standard error.

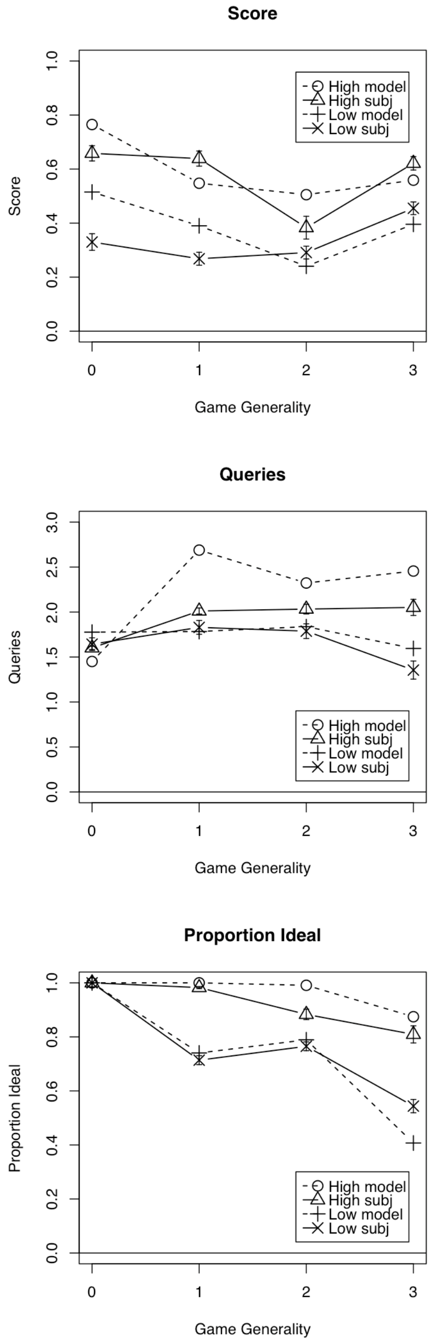


Figure 6: Experiment 2 data and model fits.

For these data, models were not explicitly designed to fit the data but instead were extrapolated from those designed for experiment 1. The fits for game generality 2 and 3, which were nonexistent in experiment 1, show that the models predict subject behavior in the right directions. The most notable deviation is the exaggeration of differences in the number of queries between the high and low control models. The high control subjects maintain the number of queries at around 2.0, while the model maintains it slightly higher. On the other hand, low control subjects show a gradual decline in the number of queries. The interaction of subject group and game generality on the number of queries was significant, $F(1, 203) = 4.27$, $p < 0.05$. The models also successfully predict the differences in the proportion of assignments that are ideal in the more general games, with the low control subjects and model showing a significant drop while the high control subjects and model show a much smaller drop. Again, the interaction of group and game generality on idealness was significant, $F(1, 202) = 6.21$, $p < 0.05$.

## Discussion

This paper explored a method of analyzing individual differences in cognitive control by developing models that differ in control structure. When tasks are more complex, simple parameter fitting may become impossible or impractical to explain the myriad differences in behavior that individuals show. For such tasks, it can be useful to assume that different people employ different control strategies and develop separate models to represent those control structures. A high control strategy means that it is less influenced by the environment and exerts more top-down cognitive control. In the ADM models, the low control model behaved according to past outcomes of various actions, while the high control model followed a strategic course of action to maximize performance. That is not to say, however, that individuals do not also vary according to parameters like working memory. In fact, fitting the working memory parameter to individual subjects in addition to selecting different control structures would probably have resulted in more accurate fits.

The possibility remains that a single model could account for the data by varying a parameter, such as a model of speed-accuracy tradeoff varying on retrieval latency. However, the fact that the data show bimodal distributions of an outcome variable suggests discreet strategic differences between subjects. It may be the case that something akin to a speed-accuracy tradeoff leads subjects to adopt different control strategies.

It is also noteworthy that the model predictions presented here for experiment 2 were indeed predictions; that is, no parameters were adjusted nor any other modifications made to generate output for experiment 2. This is an important aspect of modeling to ensure that models are not overly specific to a given experimental situation. In other words, validating model outputs on novel experimental conditions helps avoid overfitting of data. Such validation supports the view that certain aspects of a model are applicable beyond the specifics of the experiment.

## Acknowledgments

This work was supported by Office of Naval Research grant N00014-06-1-005.

## References

Anderson, J. R. (2007). *How can the human mind occur in the physical universe?* New York, NY: Oxford University Press.

Chuderski, A., Stettner, Z., & Orzechowski, J. (2006). Modeling individual differences in working memory search task. *Proceedings of the Seventh International Conference on Cognitive Modeling* (pp. 74-79). Trieste, Italy.

Daily, L. Z., Lovett, M. C., Reder, L. M. (2001) Modeling individual differences in working memory performance: a source activation account. *Cognitive Science*, 25, 315-353.

Joslyn, S. & Hunt, E. (1998). Evaluating individual differences in response to time-pressure situations. *Journal of Experimental Psychology: Applied*, 4(1), 16-43.

Rehling, J., Demiral, B., Lebiere, C., Lovett, M., & Reder, L. M. (2003). Modeling individual difference factors in a complex task environment. In F. Detje, D. Doerner, & H. Schaub (Eds.), In *Proceedings of the Fifth International Conference on Cognitive Modeling* (pp. 287-288). Bamberg, Germany: Universitats-Verlag Bamberg.

Taatgen, N. A. (2002). A model of individual differences in skill acquisition in the Kanfer-Ackerman Air Traffic Control Task. *Cognitive Systems Research, 3*(1), 103-112.

Taatgen, N. A. (2007). The minimal control principle. In Gray W. (Ed.), *Integrated Models of Cognitive Systems*. New York: Oxford University Press.

# The Importance of Action History in Decision Making and Reinforcement Learning

**Yongjia Wang (yongjiaw@umich.edu)**
University of Michigan, 2260 Hayward Street
Ann Arbor, MI 48109-2121

**John E. Laird (laird@umich.edu)**
University of Michigan, 2260 Hayward Street
Ann Arbor, MI 48109-2121

## Abstract

We investigate the hypothesis that historical information plays an important role in learning action selection via reinforcement learning. In particular, we consider the value of the history of prior actions in the classic T maze of Tolman and Honzik (Tolman & Honzik 1930). We show that including a sequence of actions in the state makes it possible to learn the task using reinforcement learning. Moreover we show that learning over sequences of length 0 ~ 4 is necessary to model rat behavior. This behavior is modeled in Soar-RL and compared to an earlier model created in ACT-R.

## Introduction

In many tasks, immediate sensory data is insufficient for decision making. Enriching the state with information about previous actions or previous situations can disambiguate between situations that would otherwise appear identical, which makes it possible not only to make correct decisions but also to learn the correct decision. Moreover, knowledge of the past can replace the need for unrealistic sensors, such as knowing the exact location in a maze.

Using historical information as part of the state representation poses some challenges. For the tasks we describe here, we use a simplified version of history – a sequence of prior actions. This leaves open the length of sequence, and how to model the relation between similar sequences to achieve proper level of generalization and specialization during learning. We demonstrate how these issues can be addressed in Soar-RL (Nason, & Laird, 2005) by proposing a simple model on an animal based experiment. We analyze the task and compare results to a recent ACT-R model (Fu & Anderson 2006).

## The T Maze Task

The task we will explore is the T maze task of Tolman and Honzik (Tolman & Hoznik 1930) in which a rat is put at the start location and it is rewarded if it gets to the end location. As shown in Figure 1, the T maze contains 14 numbered blinds (dead-ends), each corresponds to a binary choice point (the task is designed to prohibit the rats from going back at T-junctions). Whenever the rat turns into a dead-end, that is considered an error. In such a maze, there are few if any salient features. Rats are able to maintain a sense of direction, so that would provide the ability to create for different classes of T's. The only other salient features appear to be a history of the rat's behavior – that is the sequence of turns it made before coming to a T at which it must make a decision.

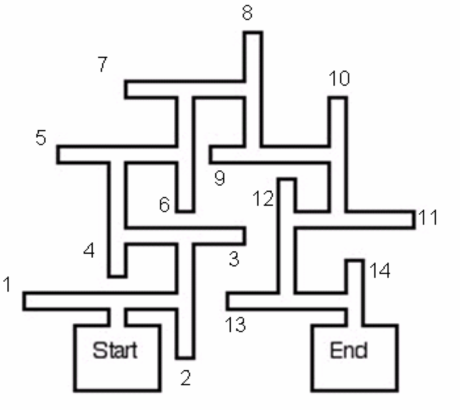


Figure 1. T maze used in Tolman and Honzik (1930)

To cast this as a problem conducive to reinforcement learning, we use the same conventions as a recent ACT-R model on this task (Fu & Anderson 2006). Moving into dead-ends and turning back results in immediate negative reward, while reaching the final goal results in positive reward. Figure 2 shows a picture of the actual environment, where the maze is embedded in a grid world and the subject moves one unit at a time. Dark boxes represent penalties, and the light box represents the final reward.

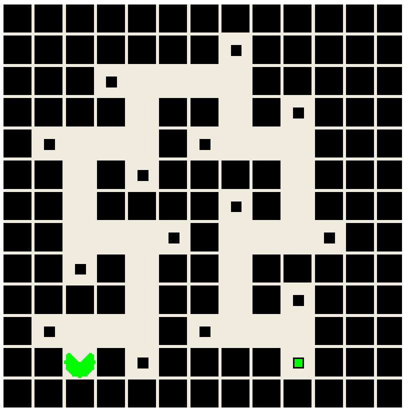


Figure 2. T-maze model

## Qualitative Analysis of Task Constraints

Given the dearth of features in the environment, the only external features available to the rat are its prior moves. Thus, we assume the representation of the state includes a sequence of previous moves. The moves could be encoded relative to the current heading: left, right, forward, backward; however, as pointed out in (Fu & Anderson 2006), the rats have strong directional bias, and thus we assume they have knowledge of absolute direction and have available the absolute directions of their movement To describe the model, we use north, east, south and west as labels for these directions. For example, at choice point 6, the state includes the sequence of [east, north, west, …] ordered left-to-right by recency, so that the first item in the sequence is the current direction.

Figure 3 shows the relationships among the choice points associated with each numbered dead-end based on the sequence representation described earlier. Choice points that are grouped together have the same previous input sequence and face with the same set of choices. Within the same group, points are further divided based on what is the correct choice. Decision points, for which moving north (2, 4, 6) or moving west (3, 11) are correct, are colored in light number with dark background; other points are colored in dark number with light background. Points in the same group but with different color are competing points in that learning to reduce the error for one type of points will simultaneously increase the error for the other type of points. Interference is most intense for the most general level (Seq 0), and disappears at the most specific level (Seq 4), where the correct decision can be learned for each choice point. The tree structure in Figure 3 therefore captures all such constraints in the task model.

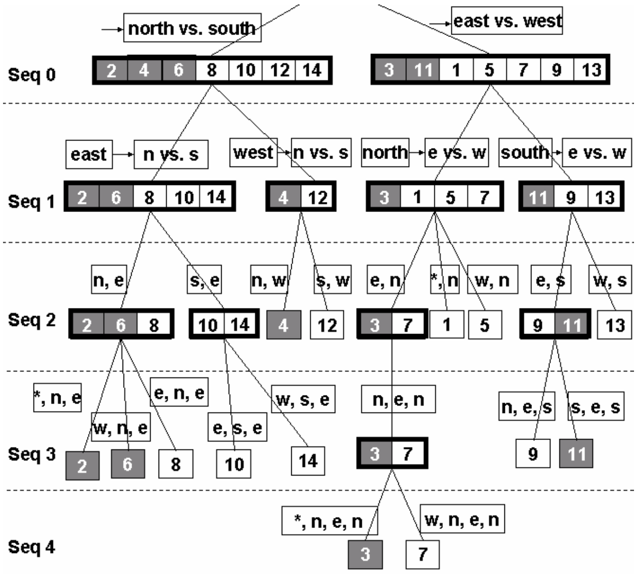


Figure 3. Relations among choice points

Our hypothesis is that choice points with similar state representations (in this case the sequence of prior moves) will appear similar to the rat and it will learn to make the same decisions in those states. Choice points with different correct directions but similar state representations will interfere with each other during learning. According to Figure 3, if the agent makes decision based on Seq 0, for example, it will tend to move south more than north and east more than west at each choice point where those options are available, since south and east correspond to the correct choice for the majority of the choice points within each group (4 south vs. 3 north and 5 east vs. 2 west). At Seq 4 (the most specific level), all choice points are completely discriminated and the correct decision can be made at each choice point.

Our assumption is that sequences of prior actions are maintained and available for decision making. Figure 3 provides the information necessary to determine what impact each sequence can have on learning. Relying solely on sequences of length 0, a rat should tend to make more errors at points 2, 4, 6, 3, and 11. Relying solely on sequences of length 1, point 4 should involve less error than point 2 and 6, since point 4 is discriminated from majority of conflicting points (especially the strongest point 14) but only interfere with point 12. Point 4 will be correctly learned at the next specificity level, while point 2 and 6 are still confused with point 8. Point 3 will involve more errors than point 11 since it is not discriminated from point 7 until sequence length of 4.

One important property of most approaches to learning these discriminations is that learning is quicker for more general levels because they are exposed to more examples. For example, there are 4 different rules (different combinations of states and legal actions) at the level of Seq 0, each of them will receive a quarter of the total training instances, while at the most specific level of Seq 4, there are 28 different rules, each of them only receives less than 4% of total training instances. This suggest there is an advantage to including selection knowledge based on all levels of the sequences so that some rough knowledge can come into play early, but more and more specific knowledge is learned over time. No deliberate mechanism is required to achieve this effect.

These conclusions are largely consistent with the experimental data from the T-maze task shown in Figure 4.

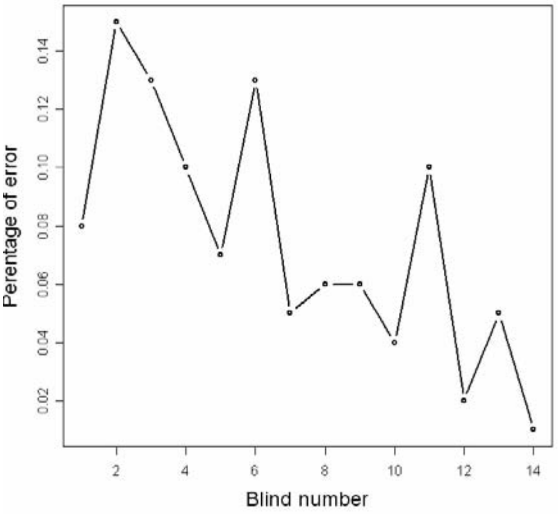


Figure 4. Percentage error in Honznik (1930)

## Soar Reinforcement Learning Model

As mentioned above, our hypothesis is that the model must consider the spectrum of specificity levels of the state representations and that these will influence learning and behavior. In Soar, this effect can be readily modeled because Soar allows knowledge for selection of an action to be encoded in multiple rules that fire together in parallel, each providing its own prediction of the expected utility of the operator. The expected utilities for the same operator are combined, producing a single, joint expected probability. Thus, when making a decision, rules match and fire for each of the levels, for each of the possible actions. Thus, we can capture all of the levels of specificity in Figure 3. Once a decision is made, all the rules that contributed to the selected action update their expected utility values.

The effect is that general rules will have the most influence for decisions at novel situations where specific rule hasn't been learned yet. In these situations, the expected values created by the specific rules will be relatively weak with values still close to the initial value of 0. As learning progresses, more and more of the specific rules will have sufficient examples so that their learning stabilizes and their values, combined with corresponding general rules, reflect the expected utility of those situations.

### Soar-RL

Soar reinforcement learning implements the general temporal-difference learning. The learned policy is represented as a Q value function as in standard Q learning. A Q value reflects the utility of taking a particular action in a particular state. In Soar-RL, a Q value is associated with each state-action pair represented as a Soar RL production rule. The update function in the case of multiple rules firing is as the following. A temporal difference is computed based on the sum of Q values for all rules that match the current condition, and is evenly distributed to update each rule. Since more general reinforcement learning rules fire more often, and a specific rule will always fire with the same general rule (there is a strict hierarchy in this task), the result is that the general rule quickly learns generalized Q value with relatively fewer trainings, while specific rules will fine tune the total Q value for specific situations and stabilize after receiving more training examples. Without general rules, the model has to learn the specific rules in novel situations without the useful initial bias that can be provided by general rules. Without specific rules, the correct behavior cannot be earned.

The probability of making a particular choice is calculated based on the Boltzmann distribution (equation 1). In the binary choice case of this task model, it can be rewritten as equation 2, therefore the probability of making the wrong choice $P_{wrong}$ is a monotonic function of the Q value difference quantity Q(s, $a_{wrong}$) – Q(s, $a_{correct}$). Here the Q value represented as a function of a state-action pair, where $a_{wrong}$ stands for the wrong action and $a_{correct}$ stands for the correct action.

$$P_i = \frac{e^{Q(s,a_i) \big/ Temperature}}{\sum_i e^{Q(s,a_i) \big/ Temperature}} \qquad \text{Equation 1}$$

$$P_1 = \frac{e^{Q(s,a_1)-Q(s,a_2)) \big/ Temperature}}{1+e^{Q(s,a_1)-Q(s,a_2)) \big/ Temperature}} \qquad \text{Equation 2}$$

Figure 5 plots the Q value difference = Q(s, $a_{wrong}$) – Q(s, $a_{correct}$) at each choice point for reinforcement learning rules with different specificity level (from Seq 0 to Seq 4). The Q values are learned separately and each is an average from 10 independent simulations for 17 trials. These Q value difference curves show the convergences trends for rules at different specificity level. The plot qualitatively illustrates how rules at each specificity level will affect the relative error rate shown in Figure 4. The initial error rate distribution should be similar to the curve Seq 0, but as more and more specific decisions are learned it eventually converges to the curve of Seq 4, the most specific level, as explained in the analysis presented in the previous section. The plot can be viewed approximately as a contour of Q value difference updating dynamics, since when all levels of rules are used in Soar, the total Q value difference will gradually converge following the path which is consistent with our empirical results (data not shown). One specific interpretation from Figure 5 is that initial error for point 4 is relatively higher than point 3, but it learns faster and results in lower total error rate. Qualitatively, the average Q value difference across all specificity levels, which is shown as a bold curve in Figure 5 approximates the relative total error rates for each dead-end. This can be confirmed by comparing with Figure 4.

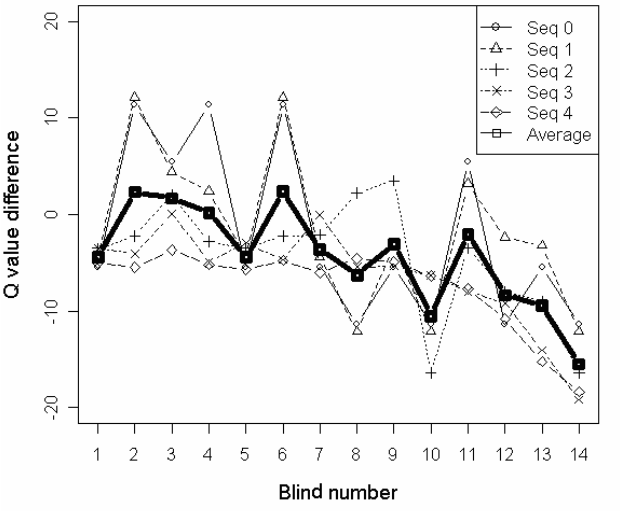


Figure 5. Effects of reinforcement learning rules with state representation at different specificity levels

Figure 5 only shows the qualitatively analysis based on separate simulations of each individual level. It is more informative to examine the combined Q value difference of all rules during learning which is shown in Figure 6 & 7.

The numbers in Figure 6 & 7 refer to trials, with 20 trial intervals. For example, the curve with 1 represents the Q value difference after trial 1, 3 represents after 21 trials. There are totally 81 trials shown in the plot to demonstrate the Q value dynamics, although the actual rat experiment only takes 17 trials. Figure 6 shows learning with only the most general rules and the most specific rules. Figure 7 shows learning with all levels of rules. One of the main differences between Figures 6 and 7 is that point 3 is learned relatively slowly when using all levels of rules. The dynamics of learning is consistent with Figure 5 and the above analysis.

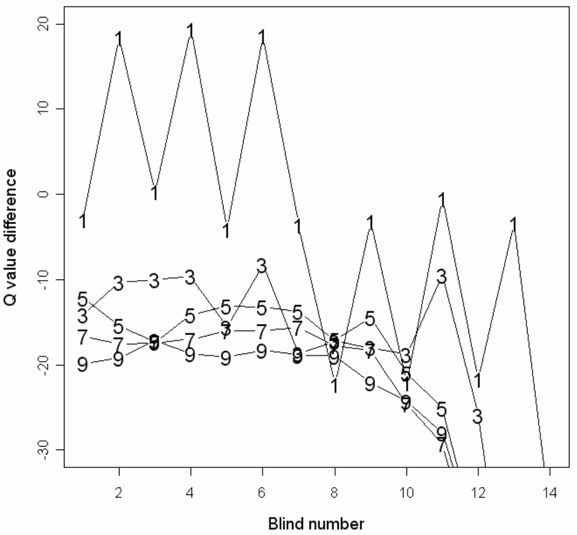


Figure 6. Change of combined Q value difference during learning, using levels 0 and 4.

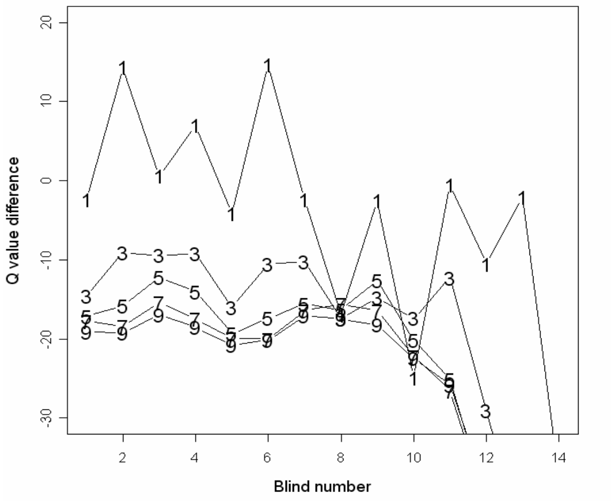


Figure 7. Change of combined Q value difference during learning, using levels 0 through 4.

## Results

Figure 8 compares observed data with prediction using all 4 levels of rules. The parameters are penalty for turning back -20, reward for reaching the goal +100, learning rate 0.1, linear discount of 10, on-policy learning with Boltzmann exploration temperature 3. We experimented with both standard exponential discount and linear discount. Figure 5-7 are generated using standard exponential discount, as that is the standard in the RL field and default in Soar. However, we discovered that a linear discount produces a better empirical match to the data, and that is what is used for the Soar results in this section. The parameter that has the most impact on matching the empirical data is the learning rate. The results are not very sensitive to the other parameters.

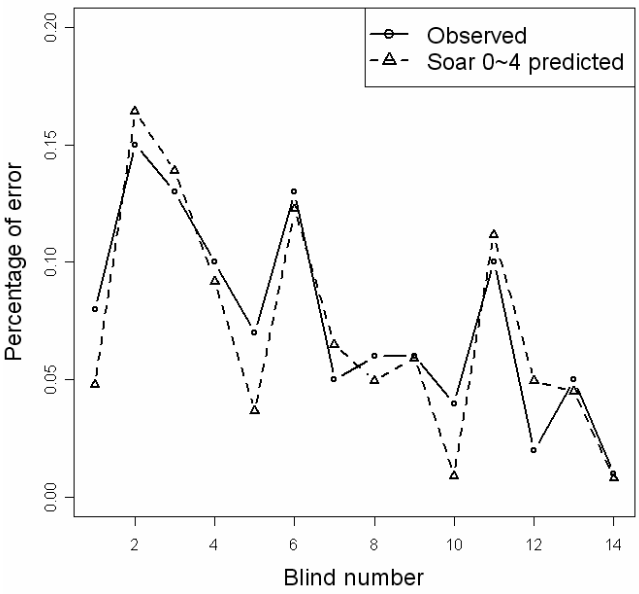
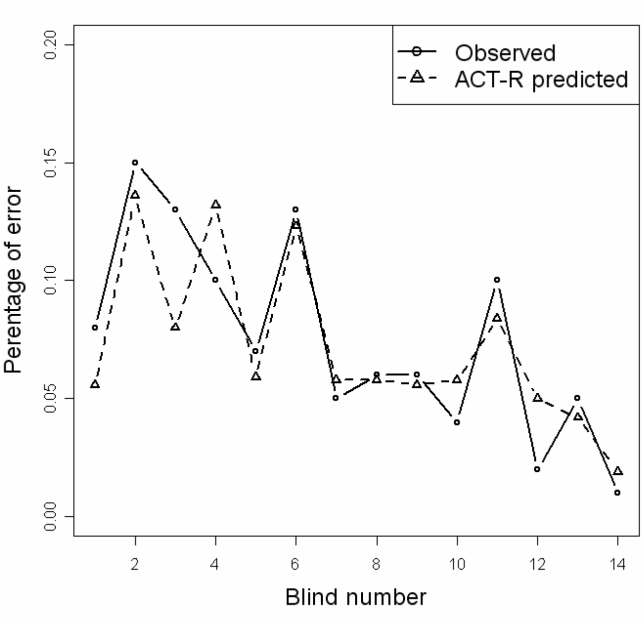
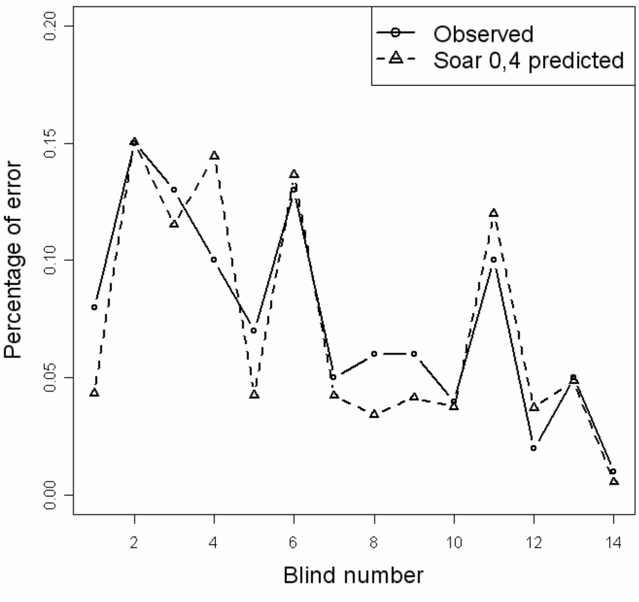


Figure 8. Soar model prediction including level 0-4



(a)



(b)

Figure 9. (a) Prediction using ACT-R model.
(b) Prediction using Soar model with equivalent rules.

## Comparison with ACT-R

An ACT-R model (Fu & Anderson 2006) was developed to model the Tolman and Honzik (1930) experiment, relying on ACT-R's native reinforcement learning component. In ACT-R, there are weights associated with rules. Learning adjusts those weights, which are used in selection. Each rule corresponds to one action and there is no explicit combination of values or joint updating of rules that are for the same action. The ACT-R model uses two sets of rules: a set of twenty-eight specific rules, two for each choice point; and a set of four general rules, one for each absolute direction. The specific rule set is equivalent to the level of seq. 4 (the model does not use sequences, assuming a rat knows its position in the maze), and the general rule set is equivalent to seq. 0 in Figure 3.

Figure 9 (a) shows the ACT-R prediction with only the most general rules and most specific rules, which is equivalent to using only Seq 0 and Seq 4 in the Soar model. Figure 9 (b) shows the prediction using Soar 0, 4 model, which is similar to the ACT-R model especially for blind 3.

Table 1 compares the correlations of the ACT-R model and Soar model. The Soar 0, 4 model predicts the ACT-R model very well (correlation 0.94), while Soar 0~4 model predicts the experimental data better (correlation 0.92) than the other models. The differences between the correlation coefficients are statistically significant. For statistical significance of difference between 0.90 and 0.92, the p value is < 0.005 based on our simulation data. This (weakly) suggests that the rats learn to make decisions using a history of prior decisions.

Table 1. Correlation Matrix comparing all models

| | Observed | Soar0~4 | Soar 0,4 | ACT-R |
|---|---|---|---|---|
| Observed | - | 0.92 | 0.90 | 0.86 |
| Soar 0~4 | - | - | - | 0.82 |
| Soar 0,4 | - | - | - | 0.94 |
| ACT-R | - | - | - | - |

Taking a closer look at the results, Soar 0~4 matches the blinds closer to the beginning much better while the Soar 0,4 model matches better for those closer to the end. Table 2 compares the correlation with partial experimental data. One hypothesis could be that for the choice points close to the end, the general rules are sufficient to produce good results, so that there is less need to use the lower level rules, which take longer to learn correctly. While for choice points at the beginning, the general rules alone produce bad results, and the more specific rules are necessary to produce good results. This hypothesis suggests that the rules at different levels are not learned and/or used uniformly for all choice points.

Table 2. Correlation with partial observed data

| | Soar0~4 | Soar 0,4 | ACT-R |
|---|---|---|---|
| Blinds 1~6 | 0.98 | 0.82 | 0.71 |
| Blinds 10~14 | 0.86 | 0.98 | 0.87 |

Table 3 compares the two levels of difference between the ACT-R model and our Soar model. The most important difference is our model's learning over multiple sequences of past actions (the model level difference). It is reasonable to assume that representing that information in the state and increasing the number of rules in ACT-R would improve the

ACT-R model's match to the observed data, especially for the early choice points.

Table 3. Comparison between the models

| | Model Level | Architectural Level |
|---|---|---|
| Soar | Use action history | Parallel rule firing |
| ACT-R | No action history | Single rule firing |

A detailed comparison between reinforcement learning in ACT-R and Soar has already been made by Nason (Nason & Laird 2004). However, this task model highlights an important difference between the two approaches. In Soar, for a single decision, multiple reinforcement learning rules are allowed to contribute to the decision making and then are updated by learning. In ACT-R, although multiple rules contribute to making a decision through competition, only one is picked and updated. Soar speeds learning with multiple reinforcement learning rules in terms of requiring fewer external actions, although the asymptotic behavior of the two approaches should be similar. This architectural level difference is secondary for the results presented here – it is the action history representation (model level difference) that makes the qualitatively different predictions in our hypothesis. However, it may be worthwhile to explore the importance of this architectural level difference in other applications.

An additional difference between the models is in the reward discount functions used in Soar and ACT-R. The default option in Soar is to multiply future expected reward with a discount factor $\gamma$ ($0 < \gamma <1$) in the step-wise update function, which results in exponential decay of rewards. As mentioned earlier, we experimented with linear discount (constant discount between steps) which generates slightly better results because exponential decay increases the differential adjustment of values near the final choice points relative to the early choice points (as evident in Figures 6 & 7). The ACT-R model uses a hyperbolic discount function, which is closer in its impact to an exponential than a linear discount. The different discount functions have more impacts on the later choice points in the result curves; they do not significantly change the relative error rates for earlier choice points (such as point 3 vs. point 4).

## Discussions

The major contributions of this paper are to examine the contribution of sequences of action histories, to decision-making and learning. The second major contribution was to evaluate the approach to representation and updating of expected values in Soar-RL and discovering that they provide an accurate model of learning dynamics by having overlapping rules at different specificity levels. Functionally, learning progresses from generalize to specific. The ACT-R model provided a useful benchmark for comparison.

We can also ask where our model falls short. Our model does not make a good prediction at blind 12. This could be due to experimental data noise, but it's more likely that there is more structure in the task that is not captured by our model. One possibility can be that instead of always using absolute directions, the rats may actually use combinations of absolute and relative directions, such as turning left and right as the state encoding strategy.

## References

Fu, Wai-Tat & Anderson, J. R (2006). From Recurrent Choice to Skill Learning: A Reinforcement-Learning Model. *Journal of Experimental Psychology: General,* 135(2) 184-206.

Nason, S. & Laird, J. E., (2005) Soar-RL: Integrating Reinforcement Learning with Soar, Cognitive Systems, 6(1) 51-59.

Nuxoll A. & Laird J.E. (2004). A Cognitive Model of Episodic Memory Integrated with a General Cognitive Architecture, *International Conference on Cognitive Modeling*.

Tolman E. C. & Honzik, C. H. (1930). Degrees of hunger, reward and non-reward, and maze learning in rats, *University of California Publications in Psychology*, 4(16), 241-256.

# Attentional Blink: An Internal Traffic Jam?

**Niels A. Taatgen (taatgen@cmu.edu)**
**Ion Juvina (ijuvina@cmu.edu)**
Department of Psychology, Carnegie Mellon University
5000 Forbes Av., Pittsburgh PA 15213

**Seth Herd (sethherd@grey.colorado.edu)**
Department of Psychology, University of Colorado
Boulder CO 80309

**David Jilk (djilk@jilk.com)**
eCortex, Inc., Boulder CO

**Sander Martens (s.martens@med.umcg.nl)**
NeuroImaging Center, University Medical Center Groningen
Antonius Deusinglaan 2, 9713 AW, Groningen, Netherlands

## Abstract

In cognitive models, cognitive control can be measured in terms of the number of control states that are used to do the task. In most cases more control leads to better performance. Attentional Blink is an example in which the opposite is true: more control leads to poorer performance. A hybrid ACT-R/Leabra model is used to model both high- and low-control participants using two and one control states, respectively.

**Keywords:** Cognitive Control, Multi-tasking, ACT-R, Leabra

## Introduction

The term *cognitive control* is used to refer to cognitive processes that help us focus on our goals and plans, and prevent external stimuli and events from interfering with them. Most of the time a higher level of control improves performance on tasks. However, too much control can make behavior brittle and less flexible. It is therefore likely that cognition strives for a level of control that is just enough for proper performance (the minimal control principle, Taatgen, 2005, 2007). In this paper we will discuss a task that demonstrates that too much control can hurt performance. The task is a Rapid Serial Visual Presentation (RSVP) task (Raymond, Shapiro, & Arnell, 1992).

In RSVP tasks, participants are presented with rapid streams of visual stimuli. Each of these streams contains 0, 1 or 2 targets that the participants have to identify. In the version that we will discuss in this paper, streams consist of 20 characters that are presented at a rate of 100ms/character. The targets are letters, while distracters are digits. The streams of interest are the ones with two targets. In these streams the time between the two targets is of importance, usually referred to by the *lag*. A lag of 1 means the two targets appear in sequence, and are 100 ms apart in time (given our presentation rate), lag 2 means that there is one distracter in between the targets, etc. Example sequences are:

Lag 1: 49204039GF3434329237
Lag 3: 0230349023Y94D324294
Lag 9: 9430R32305129K235209

The phenomenon of interest is that on lags 2-5, but mainly on lag 2 and 3, accuracy one the second target is much lower than on the first target. On other lags, including lag 1, the accuracies are the same. This phenomenon is referred to as *Attentional Blink*. The interesting aspect from the viewpoint of cognitive control is that there are indications that *less* control improves performance. Certain experimental manipulations can decrease the amount of blink people exhibit, for example if the stimuli are presented in a star field (Arend, Johnston & Shapiro, 2006), when music is played in the background (Olivers & Nieuwenhuis, 2005), or when participants receive instructions to focus less on the task (Olivers & Nieuwenhuis, 2006). In addition, there are strong individual differences in attention blink: some individuals do not exhibit attention blink at all (Martens, Munneke, Smid & Johnson, 2006).

Nieuwenhuis, Gilzenrat, Holmes and Cohen (2005) have modeled the attentional blink using a neural network. Their model consists of three layers, input, decision and detection, and a cell representing the Locus Coeruleus (LC), which is connected to the decision and detection layers. The LC provides extra activation to these layers, making them more sensitive to targets. Once the decision layer has decided that an input is a target, it is stored in the detected layer, but it also sends a signal to the LC. The result of the signal to the LC is that its contribution to activating the decision layer temporarily diminishes, decreasing the detection rate of targets that appear within 200-300 ms. This decrease is relatively slow, so it has no effect on the lag 1 trials.

Although the Nieuwenhuis et al. model is successful in predicting the outcome of several new experiments, the impact of control is outside the scope of that model. An additional finding in RSVP experiments is that in lag 1 trials, the order in which the targets are reported is often reversed, while this almost never happens in any of the other lags (Hommel & Akyürek, 2005).

Both phenomena, reduced control leads to less blink, and the reversed report of lag 1 targets, are outside the scope of the Nieuwenhuis et al. model, because it neither incorporates higher-level aspects of control nor the fine-level details of perception. In this paper we will present a model that encompasses both. To incorporate both the fine details of perception and the higher-level control aspects, we used the hybrid model that combines the Leabra neural network architecture (O'Reilly & Munakata, 2000) and the ACT-R architecture (Anderson, 2007). More specifically, visual perception is handled by a Leabra model, which passes on the information to the visual input buffer of ACT-R. ACT-R takes care of the classification of the symbol, and storing it, if it is a target.

## Experiment

### Method

Forty-one volunteers from the Carnegie Mellon student population participated in this experiment, which was part of a larger experiment on individual differences in cognitive control. The larger experiment included three other tasks that we selected to assess levels of cognitive control: the N-Back memory task (McElree, 2001), the abstract decision making task (ADM task, Joslyn & Hunt, 1998), and a dual-tasking task (DTT task, Taatgen, van Rijn & Anderson, in press). In the N-Back task, participants were shown sequences of letters, and they had to detect repetitions of letters and judge how many letters back that repetition was. In the ADM task, participants had to ask questions about properties of objects, and sort the objects into bins once they had obtained enough information. In the DTT task, participants had to do two visual tasks and a time estimation task in parallel. Four participants had to be excluded from the dataset due to a problem in the experimental software.

The stimuli consisted of sequences of 20 characters. Distracters were digits from 2 to 9, and targets were C, D, F, G, H, I, J, K, L, M, N, P, R, T, V, W and X. No consecutive characters were identical, and if there were two targets, they were different from each other. Targets were never in the first four positions in the sequence, nor in the last four.

The experiment consisted of 6 practice trials, and 4 blocks of 37 experimental trials. Each block of experimental trials consisted of 5 zero-target trials, 5 one-target trials, and 27 two-target trials. The two-target trials consisted of 3 trials of each 9 different lag lengths (1 through 9).

Each trial was preceded by a 500 ms fixation point, after which symbols in the sequence were presented one at a time for 100 ms each. At the end of the sequence, participants were asked to type all the targets (if any) they had seen, and press enter. The next trial started immediately after the participant had pressed enter.

### Results

Figure 1 shows the correctness on the second target by lag. There is a clear blink effect in lags 2-4, consistent with many earlier findings. On 16% of the Lag 1 trials, both targets were reported correctly, but in the wrong order.

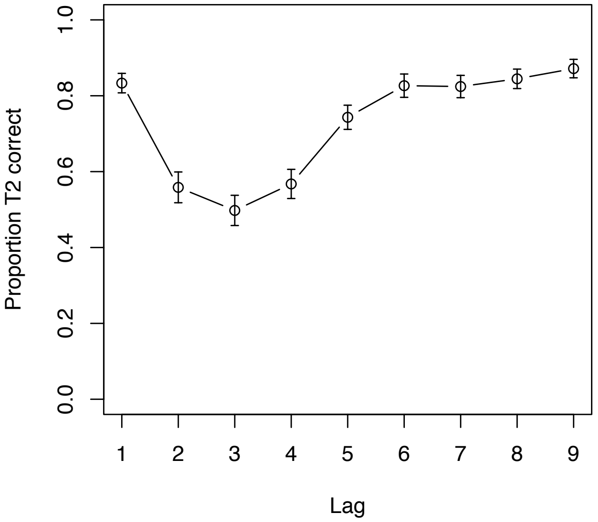


Figure 1: Blink by Lag. Bars are standard errors.

In order to assess individual differences, we calculated the AB magnitude (based on Martens et al., 2006) according to the following equation:

$$\left(\frac{\mathrm{T1}_{lag2} - \mathrm{T2}_{lag2} \mid \mathrm{T1}_{lag2}}{\mathrm{T1}_{lag2}} + \frac{\mathrm{T1}_{lag3} - \mathrm{T2}_{lag3} \mid \mathrm{T1}_{lag3}}{\mathrm{T1}_{lag3}}\right)/2$$

In this equation, T1 means proportion correct identification of the first target (given the indicated lag), and T2|T1 means correct identification of the second target given a correct identification of the first target. An AB magnitude of 0 or less means there is no blink at all, while higher values imply more blink. We correlated the AB magnitude with outcomes of the other tasks that are associated with control. For the DTT task, the outcome most associated with control is how often participants make a response on the time estimation task, the one of three tasks that is not cued by visual input. Low levels of response are an indication of low-level control. The response rate on the DTT task and the AB magnitude correlated with $r$=0.46 ($p$=0.005), indicated that more control in the DTT task implied a higher blink magnitude. Similarly, the score on the N-Back task and the AB magnitude had a correlation of $r$=0.42 ($p$=0.009). The correlation with the ADM task was only weak: $r$=0.29 ($p$=0.08). All of these correlations suggest that a higher level of control correlates with a higher AB magnitude.

For the purposes of identifying non-blinkers in the dataset, we classified participants with a AB magnitude of 0.2 or lower as non-blinker, and participants with 0.2 or higher as blinker (consistent with Martens et al., in press), amounting to 8 non-blinkers and 29 blinkers.

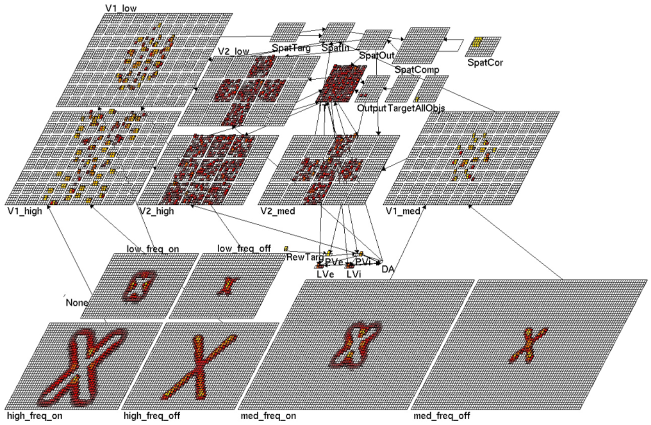


Figure 2: Visual input module

## Model

### Overview

Cognitive models allow a more precise characterization of what "more control" means. To keep outside events from completely controlling behavior, we maintain internal goals. The current goal can be in a certain state to keep track of progress of that goal. Any action or progress on the goal can change the state, or keep it as it is. Taken together, states and possible actions create a state space. The more states there are in the state space, the more it is associated with a higher level of control (Taatgen, 2007).

We designed two possible control structures for a model of attentional blink, one with two states, which models the blinkers, and one with one state, which models the non-blinkers. The first, blinker, model assumes two control states. One state is used to signify the model is searching for a target in the input stream. Once a target has been found, the model switches to a second state that is used to consolidate the target in memory. When the target is consolidated, the state switches back to the first state. When the model is in the consolidation state, it no longer fully processes the input stream. When the model switches back to the search state it has too much to do at the same time, creating an internal "traffic jam" that causes the model to sometimes miss targets in the 200-500ms range. The second, non-blinker, model uses only a single state. In other words, there is no state to protect the consolidation process, but it also misses targets less often.

### Vision

The visual input is projected on the input layer of a Leabra (O'Reilly & Munakata, 2000) neural network of the ventral visual stream (Figure 2). This model processes the input, and arrives at a classification in the output layer, in which it has a single cell for each of the possible symbols in the input. Because of the speed in the visual presentation, the network is not always able to fully settle, and there is often still residual activation of the previous symbol. Figure 3 shows an example of how activation in the output layer changes over time based on the "2829P" sequence. The consequence of the rising and falling of activations is that if the visual input is sampled at a particular moment, it is possible two output cells are active, in which case it is impossible to determine in which order the two have been presented. This explains why in lag 1 the two targets are often reported in the wrong order.

### Central Cognition and Control

The ACT-R architecture is structured as a set of interacting modules. Modules communicate through buffers, but otherwise operate asynchronously. Each module can only work on one thing at a time, but because all the modules work in parallel, the cognitive system as a whole can work on several tasks at the same time. At this level of abstraction, cognitive control is a matter of optimally engaging all modules in doing the task or tasks.

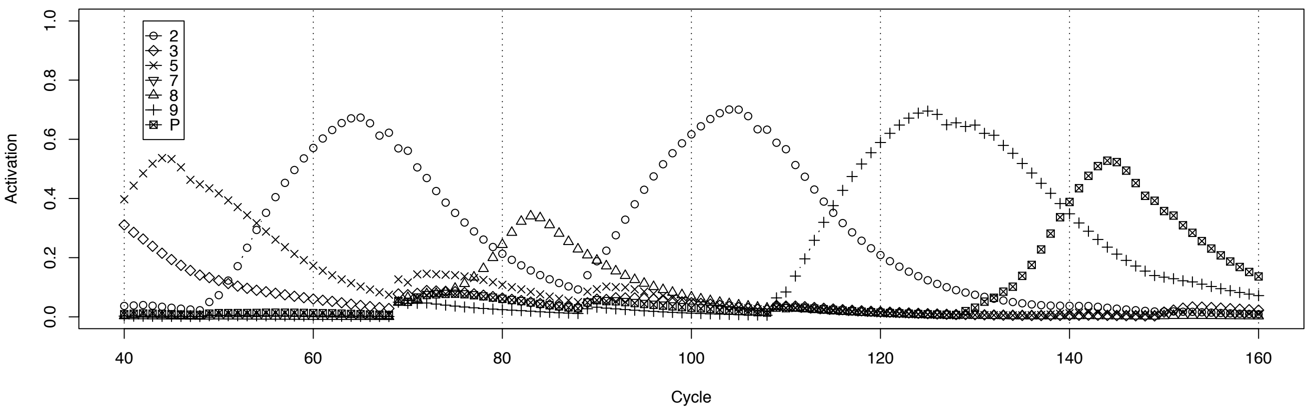


Figure 3: Example of visual module output for the "2829P" sequence. Vertical lines indicate where a new stimulus is presented, e.g. at cycle 40 the symbol "2" is presented. Each cycle corresponds to 5ms real time.

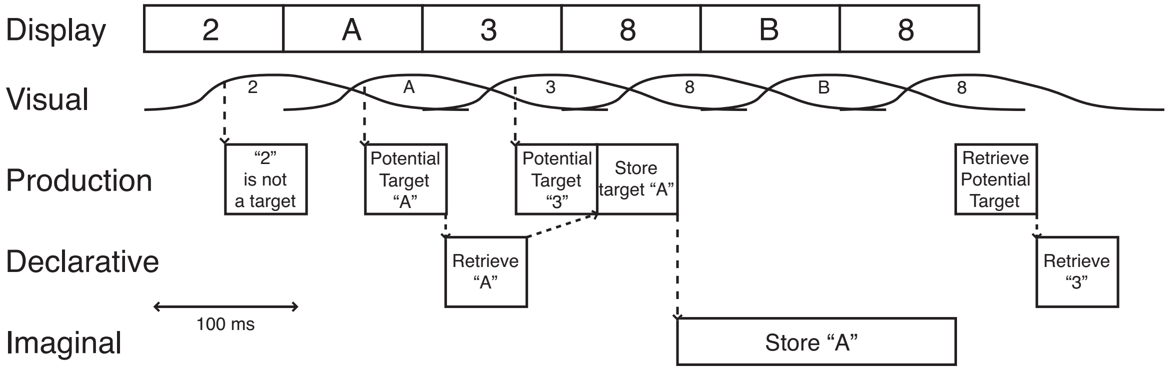


Figure 4: Time diagram of a Lag 3 presentation: blinker model.

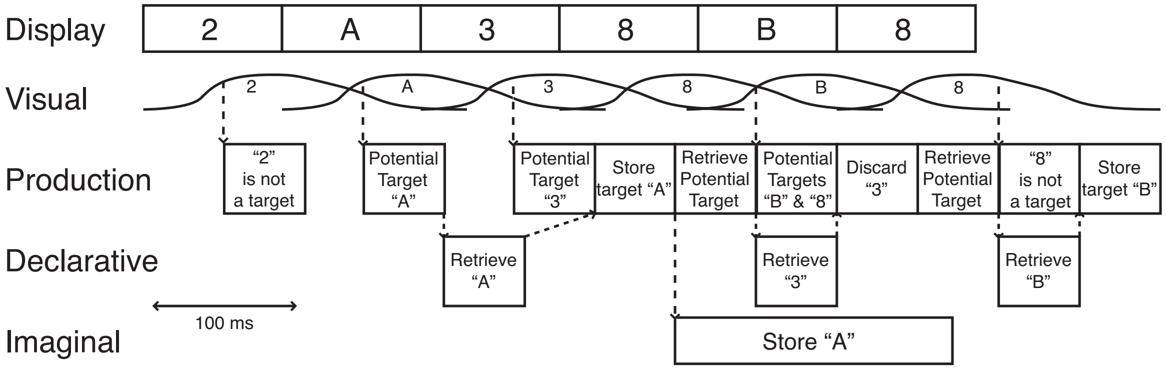


Figure 5: Time diagram of a Lag 3 presentation: non-blinker model.

In the attentional blink tasks, several modules have to operate in parallel. The visual module has to scan the incoming stream of characters. These characters have to then go through a decision process, which determines whether a character is a target, which involves the procedural and declarative modules. Once a target is found, it has to be consolidated in ACT-R's imaginal buffer, a place to temporarily store problem-related information.

Figure 4 shows a diagram of how the model operates in the case of a Lag 3 sequence. Each row in the diagram represents one of ACT-R's modules, with the exception of the top line, which shows the state of the display. Boxes in each row show activity in a module at a particular time, and the width of the box indicates the duration. The visual module follows the input, and outputs activations corresponding to the classification of the input. This input triggers production rules that try to determine whether the character is a target. The assumption of the model is that some characters can be recognized as non-targets straight away (with a probability of 60%). Other characters have to be classified by a declarative memory retrieval. In the example, the first "2" is immediately recognized as non-target. The second character ("A") is a target. Targets are always retrieved from memory to verify that they are targets. During this retrieval, the next character ("3") is identified as a potential target. However, the memory retrieval that can verify this has to be postponed until the retrieval of the "A" finishes. However, once the retrieval of "A" finishes, the model decides "A" is indeed a target, and stores it in the imaginal buffer. In this version of the model, once a target has been found the production rule that initiates storing the target changes the control state to consolidate, blocking further processing of the input. Only when the imaginal buffer is done storing the target can the target detection process resume. As a consequence, the "8" and the "B" will not be considered as targets, leading to a blink trial in this particular example. Also note that the "3" that directly followed the first target is considered a potential target by the model, although delayed, which means that if this had been a Lag 1 trial and the "3" would have been a target, both targets would have been detected.

## The non-blinker model

Although the model described above changes state to protect its memory consolidation, this is an unnecessary exertion of control. If the model does not change state when a target has been detected, detection of the second target can proceed while the first target is consolidated. Figure 5 shows an example trace of that variation of the model. Once the target "A" has been detected and transferred to the imaginal buffer, the next candidate, “3” is requested from declarative memory. The next production rule samples this visual input again, now detecting two potential targets (“B” and “3”), because both output cells of the neural network are active.

### Reversal of target on Lag 1

Figure 6 demonstrates how reversals can occur. Delayed by the possibility that “2” is a target, the model samples the visual buffer at a moment when both the “A” and “B” cells are active. Having no means to determine the order of the two, the model decides to retrieve the “B” first.

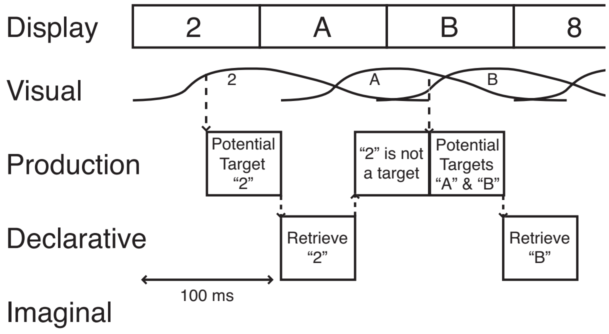


Figure 6: Example of a reversal of targets

### Parameters

The model has a number of parameters that influence its outcome. The main parameter that was used to fit the model was the probability that a character could be recognized as a foil straight away (60%). Other parameters that will probably influence the outcome when changed are the latency factor that determines how long declarative retrievals take (set so that they take about 50ms), and the time to store an item in the imaginal buffer (left at the default of 200ms). In addition, the Leabra model has several parameters, but those were left untouched for the purpose of fitting the data.

## Model Results

To assess the model we divided the dataset in blinkers and non-blinkers according to the criteria discussed earlier. Figure 7 shows the model/data comparison for the blinker group (*r*=0.97, MSE=0.03), and Figure 8 for the non-blinker group (*r*=0, MSE=0.04). The blinker model has a good correlation between model and data and a low MSE. The non-blinker model has no correlation with the data because there is no meaningful variability in the data to model. However, the MSE is comparable to that of the blinker model, confirming that there is a decent fit.

Figure 9 shows the proportion of trials in which the two targets were reported in the wrong order for both groups together (there was no difference between blinkers and non-blinkers).

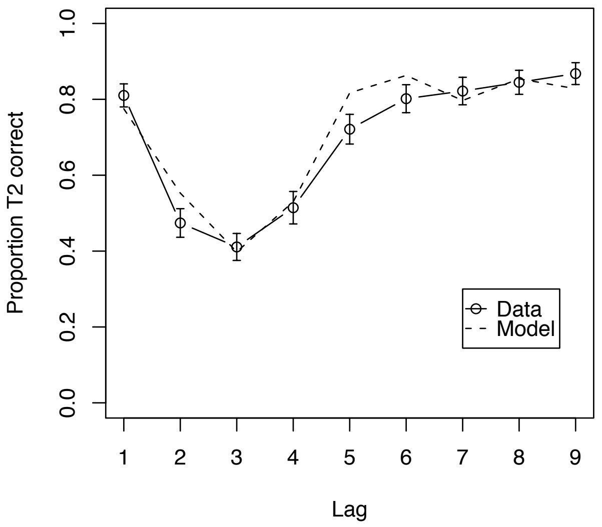


Figure 7: T2 correctness for blinkers (n=29), model/data comparison.

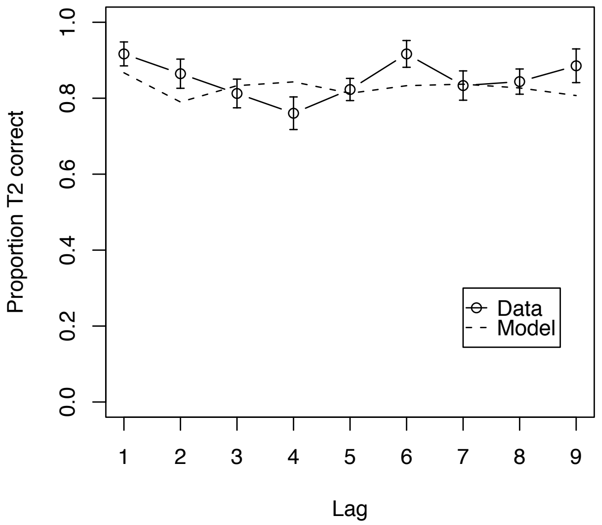


Figure 8: T2 correctness for non-blinkers (n=8), model/data comparison.

## Discussion

The central question that we tried to address in this paper is how cognitive control can be made more concrete in terms of a cognitive model. Our hypothesis is that more control is associated with more possible control states. More control states gives more top-down control of the task’s execution, but at the cost of flexibility. The experiment as a whole showed that the amount of blink in the RSVP task correlated with control aspects of other tasks, which is consistent with findings by Arend et al. (2007) and Olivers and Nieuwenhuis (2006) that attentional blink is related to control factors.

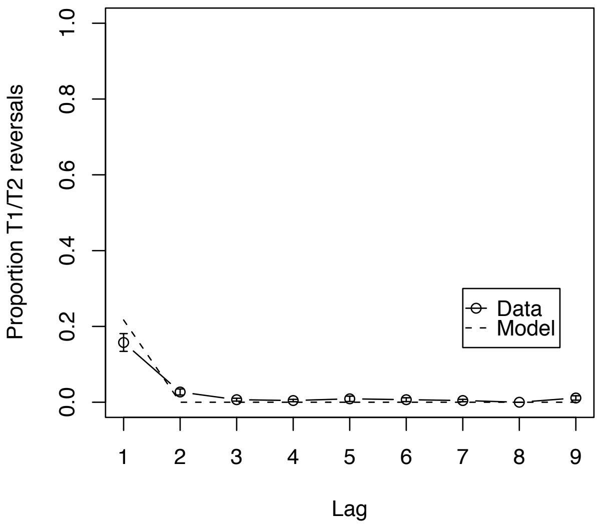


Figure 9: Proportion in which targets are reported in the wrong order, model/data comparison

The RSVP model, with two control states for high control and one control state for low control, managed to fit the data very well. Further support for the model can be found in ERP data. Martens et al. (2006) collected ERP data for both blinkers and non-blinkers. They found that the P300, which we associate with imaginal buffer activity, is present only if a target is detected. Moreover, it is later for blinkers than for non-blinkers. This is the case for both T1 and T2, but the effect on T2 is much larger. Our model currently only predicts a difference on T2. The T1 difference may be due to a different factor all together, and may be related to the proportion in which distracters can be dismissed without memory retrieval.

This model is also a demonstration of how a symbolic architecture, ACT-R, and a neural network architecture, Leabra, can work together. The perceptual part of the model was clearly outside the current capabilities of ACT-R, while the control aspects were outside of Leabra's scope.

## Acknowledgments

This work was supported by Office of Naval Research grant N00014-06-1-005 and N00014-05-1-0880, and Biologically-Inspired Cognitive Architecture, DARPA.

## References

Anderson, J. R. (2007). *How can the human mind occur in the physical universe?* New York: Oxford university press.

Arend, I., Johnston, S., & Shapiro, K. (2006). Task-irrelevant visual motion and flicker attenuate the attentional blink. *Psychonomic Bulletin & Review, 13*(3), 600-607.

Hommel, B., & Akyürek, E. G. (2005). Lag-1 sparing in the attentional blink: benefits and costs of integrating two events into a single episode. *Quarterly Journal of Experimental Psychology, 58A*, 1415-1433.

Joslyn, S., & Hunt, E. (1998). Evaluating individual differences in response to time-pressure situations. *Journal of Experimental Psychology: Applied, 4*(1), 16-43.

Martens, S., Munneke, J., Smid, H. & Johnson, A. (2006). Quick minds don't blink: electrophysiological correlates of individual differences in attentional selection. *Journal of Cognitive Neuroscience, 18*(9), 1423-1438.

McElree, B. (2001). Working Memory and Focal Attention. *Journal of Experimental Psychology: Learning, Memory and Cognition, 27*(3), 817-835.

Nieuwenhuis, S., Gilzenrat, M. S., Holmes, B. D., & Cohen, J. D. (2005). The role of the locus coeruleus in mediating the attentional blink: a neurocomputational theory. *Journal of Experimental Psychology: General, 134*(3), 291-307.

Olivers, C. N. L., & Nieuwenhuis, S. (2005). The Beneficial Effect of Concurrent Task-Irrelevant Mental Activity on Temporal Attention. *Psychological Science, 16*(4), 265-269.

Olivers, C. N. L., & Nieuwenhuis, S. (2006). The beneficial effects of additional task load, positive affect, and instruction on attentional blink. *Journal of Experimental Psychology: Human Perception and Performance, 32*(2), 364-379.

O'Reilly, R. C., & Munakata, Y. (2000). *Computational Explorations in Cognitive Neuroscience* Cambridge, MA: MIT Press.

Raymond, J. E., Shapiro, K. L., & Arnell, K. M. (1992). Temporary suppression of visual processing in a RSVP task: An attentional blink? *Journal of Experimental Psychology: Human Perception and Performance, 18*, 849-860.

Taatgen, N. A. (2005). Modeling parallelization and flexibility improvements in skill acquisition: from dual tasks to complex dynamic skills. *Cognitive Science, 29*(3), 421-455.

Taatgen, N. A. (2007). The minimal control principle. In W. Gray (Ed.), *Integrated models of cognitive systems*. Oxford: Oxford University Press.

Taatgen, N. A., Rijn, H. v., & Anderson, J. R. (in press). An Integrated Theory of Prospective Time Interval Estimation: The Role of Cognition, Attention and Learning. *Psychological Review.*

# Category Development and Reorganization Using a Bidirectional Associative Memory-inspired Architecture

**Gyslain Giguère (giguere.gyslain@courrier.uqam.ca)**
UQÀM, Département de psychologie, A/S LEINA,
C.P. 8888, Succ. CV, Montréal, Qc, H3C 3P8

**Sylvain Chartier (chartier.sylvain@gmail.com)**
Université d'Ottawa, Département de psychologie,
550 Cumberland, Ottawa, Ont, K1N 6N5

**Robert Proulx (proulx.robert@uqam.ca)**
UQÀM, Département de psychologie, A/S LEINA,
C.P. 8888, Succ. CV, Montréal, Qc, H3C 3P8

**Jean-Marc Lina (jean-marc.lina@etsmtl.ca)**
École de Technologie Supérieure, Département de génie électrique
C.P. 8888, Succ. CV, Montréal, Qc, H3C 3P8

## Abstract

This paper shows that the recently proposed Feature-Extracting Bidirectional Associative Memory (FEBAM) can create and reorganize item clusters (or categories). This model, contrary to most other cluster-creating architectures, is based on projection methods and follows associative model principles (e.g. distribution of information, pattern completion and noise tolerance). Using a bidirectional associative memory (BAM)-inspired architecture, the resulting model is tested by simulating iterative cluster development and reorganization of artificial stimuli and alphanumeric patterns. In contrast to classic clustering techniques, the model is able to reproduce predetermined categories by iteratively reevaluating cluster membership, and allows given category members to move from a category to another. Because FEBAM has been shown to possess many more interesting properties, it is argued that the model possesses more cognitive explanative power than other comparable models and algorithms.

## Introduction

### Categories

In everyday life, humans are constantly exposed to situations in which they must group perceptual patterns (such as visual objects) into categories, in order to act upon the identity and properties of the encountered stimuli. To achieve this task properly, a cognitive system must adapt to many different environments which necessitate a broad range of behaviors according to context. Cognitive scientists have historically argued over the fact that the human cognitive system either uses generic abstractions (such as prototypes) or very specific perceptual stimulations (such as complete exemplars) to achieve category learning and further classification. Seemingly, the use of one of these representations is likely to be closely linked to specific environmental demands and system goals (Murphy, 2002).

### Clustering in Neural Nets

More generally speaking, category formation, in a perceptual framework, is often seen as a "clustering" of similar patterns in common categories, a process akin to classic clustering techniques, which involve partitioning stimulus spaces in a number of finite categories ("clusters").

In artificial neural networks, clustering is a well-developed technique used mainly in competitive models (Kohonen, 1989; Grossberg, 1988). In these models, each output unit represents a specific cluster of items. When taking a decision, the association between an exemplar and its determined cluster unit in the output layer is strengthened. In "hard competitive" networks (Grossberg, 1988), exemplars may only be associated with one cluster (i.e. only one output unit at a time can be activated).

An example of hard competitive network is the Adaptive Resonance Theory (ART: Carpenter & Grossberg, 1987; Grossberg, 1988). ART networks are able to deal effectively with the exemplars-prototype scheme, while being able to answer the stability-plasticity dilemma. These unsupervised competitive models achieve the desired behavior by using a novelty detector (through "vigilance"); various degrees of generalization can be achieved with this procedure. If the value of the vigilance parameter is low, broad categories are developed; if it is high, narrow categories are developed, with the network ultimately performing exemplar learning.

In "soft computing" (Kohonen, 1989), exemplars may be associated with many clusters at differing degrees. This provides a more distributed classification; for instance, an exemplar may be geometrically positioned between two clusters, and possess various degrees of membership.

PCA networks (Diamantaras & Kung, 1993) can also be used to achieve clustering; in this case, each category is defined by a linear sum of extracted orthogonal components. Nonlinear PCA networks (Karhunen, Pajunen & Oja, 1998) are not restrained by this orthogonal requirement; hence, correlated components can be found (Hyvarinen & Oja, 2000). In all cases, once an item has been linked to a specific cluster, there is no mechanism allowing this item's category membership to be modified. Each model's internal structure is based on a specific metric that is constant over the training period.

### Model Overview

In this paper, we show that the FEBAM model can be used to form clusters using various exemplar sets. First, it is shown that prototypes can be stored in the network's memory, regardless of the number of units. In this situation, exemplars form transient memories that can be used for identification, while prototypes constitute attractors that can be used for categorization. Second, when using a unit recruiting procedure, the model can ultimately develop exemplar-based attractors. In FEBAM, it is the number of units that specifies category broadness (in comparison to novelty detection in ART models). If, for instance, additional units are recruited and trained during learning, then the model is able to develop a greater amount of narrower categories. Ultimately, the network can perform "exemplar clustering".

## Model Description

### Architecture

FEBAM's architecture is based on a BAM architecture (Kosko, 1988) proposed by Hassoun (1989) and Chartier & Boukadoum (2006a). It consists in two Hopfield-like neural networks interconnected in head-to-toe fashion.

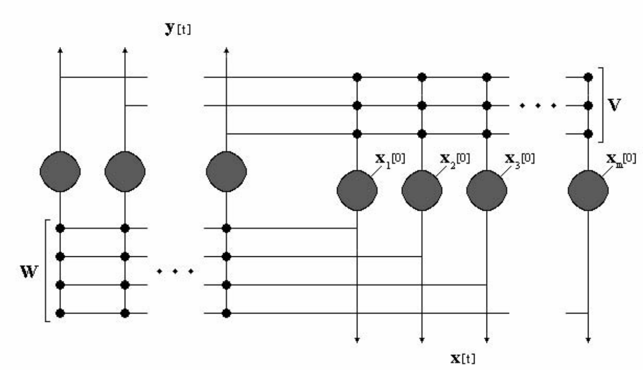


Figure 1: FEBAM network architecture.

When connected, these networks allow a recurrent flow of information to be processed bidirectionally. As shown in Figure 1, the **W** layer returns information to the **V** layer and vice versa. As in a standard BAM, both layers serve as a teacher for the other layer and the connections are explicitly depicted in the model[1]. To enable a BAM to perform clustering, one set of those explicit connections must be removed. Thus, in contrast with the standard BAM architecture, the "initial output" **y**(0) is not obtained externally, but is instead acquired by iterating once through the network, as depicted in Figure 2.

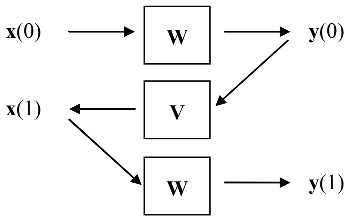


Figure 2: Output iterative process used for learning updates.

In the context of item cluster creation, the **W** layer will be used to determine the maximal number of clusters created by the network; the more units in that layer, the more possible clusters. The exact relationship between these quantities is the following: the theoretical maximal number of categories developed by the network is equal to $2^n$, where $n$ is the number of units in the network.

### Output Function

FEBAM's output function is expressed by the following equations:

$$\forall\ i, ..., N, \mathbf{y}_i(t+1) = \begin{cases} 1, & \text{If} \quad \mathbf{Wx}_i(t) > 1 \\ -1, & \text{If} \quad \mathbf{Wx}_i(t) < -1 \\ (\delta+1)\mathbf{Wx}_i(t) - \delta(\mathbf{Wx})_i^3(t), & Else \end{cases} \qquad (1)$$

and

$$\forall\ i, ..., M, \mathbf{x}_i(t+1) = \begin{cases} 1, & \text{If} \quad \mathbf{Vy}_i(t) > 1 \\ -1, & \text{If} \quad \mathbf{Vy}_i(t) < -1 \\ (\delta+1)\mathbf{Vy}_i(t) - \delta(\mathbf{Vy})_i^3(t), & Else \end{cases} \qquad (2)$$

where $N$ and $M$ are the number of units in each layer, $i$ is the index of the respective vector element, $\mathbf{y}(t+1)$ and $\mathbf{x}(t+1)$ represent the network outputs at time $t + 1$, and $\delta$ is a general output parameter. Like in any nonlinear dynamic system, to guarantee that a given output converges to a fixed point such as $\mathbf{x}^*(t)$ or $\mathbf{y}^*(t)$, the slope of the outputs function's derivative must be positive and smaller than one (Chartier & Proulx, 2005; Kaplan & Glass, 1995):

$$\frac{d\mathbf{y}(t+1)}{d\mathbf{Wx}(t)} = 0 < (\delta+1) - 3\delta(\mathbf{Wx}(t))^2 < 1 \qquad (3)$$

$$\frac{d\mathbf{x}(t+1)}{d\mathbf{Vy}(t)} = 0 < (\delta+1) - 3\delta(\mathbf{Vy}(t))^2 < 1 \qquad (4)$$

[1] In opposition, the architecture of multi-layer Perceptrons strictly illustrates a series of input-output relationships, without ever specifying the origin of the *teacher*'s information. This is less desirable in a neuropsychological perspective.

This condition is satisfied when $0 < \delta < 0.5$ for bipolar stimuli[2]. Figure 3 illustrates the shape of the output function when $\delta = 0.4$. This output function possesses the advantage of exhibiting continuous-valued (gray-level) attractor behavior. Such properties contrast with networks using a standard nonlinear output function, which can only exhibit bipolar attractor behavior (e.g. Kosko, 1988).

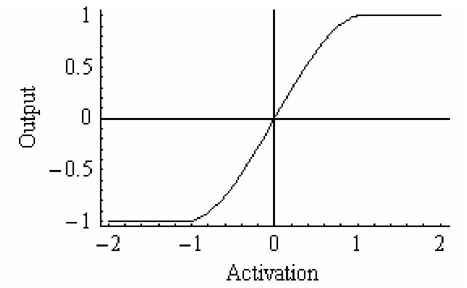


Figure 3: Output function for $\delta = 0.4$.

### Learning Function

Learning is based on time-difference Hebbian association (Chartier & Proulx, 2005; Kosko, 1990; Oja, 1989; Sutton, 1988), and is formally expressed by the following equations:

$$\mathbf{W}(k+1) = \mathbf{W}(k) + \eta(\mathbf{y}(0) - \mathbf{y}(t))(\mathbf{x}(0) + \mathbf{x}(t))^{\mathrm{T}} \tag{5}$$

and

$$\mathbf{V}(k+1) = \mathbf{V}(k) + \eta(\mathbf{x}(0) - \mathbf{x}(t))(\mathbf{y}(0) + \mathbf{y}(t))^{\mathrm{T}} \tag{6}$$

where $\eta$ represents a learning parameter; T is the usual transpose operator, $\mathbf{y}(0)$ and $\mathbf{x}(0)$, the initial patterns at $t = 0$, $\mathbf{y}(t)$ and $\mathbf{x}(t)$, the state vectors after $t$ iterations through the network, and $k$ the learning trial. The learning rule is thus very simple, and can be shown to constitute a generalization of hebbian/anti-hebbian covariation in its autoassociative memory version (Chartier & Boukadoum, 2006a). For weight convergence to occur, $\eta$ must be set according to the following condition (Chartier & Proulx, 2005; Chartier, & Boukadoum, M., 2006b):

$$\eta < \frac{1}{2(1-2\delta)Max[N,M]}, \quad \delta \neq \tfrac{1}{2} \tag{4}$$

### General Procedure

To obtain the various vectors needed for weight updates, stimuli iteration is performed as depicted in Figure 2. First, an initial stimulus ($\mathbf{x}(0)$) is introduced to the **W** layer, yielding an initial output ($\mathbf{y}(0)$). This output represents the input's classification into a distributed cluster. Second, using this initial output, the information is sent back to the input layer using the **V** layer's connections; this results in another output, $\mathbf{x}(1)$. Third, this novel output is then used to obtain the final classification output ($\mathbf{y}(1)$) by using the **W** connections once again.

As stated by Equations 5 and 6, weights can only converge when "internal feedback" is identical to the initial inputs (that is, $\mathbf{y}(1) = \mathbf{y}(0)$ and $\mathbf{x}(1) = \mathbf{x}(0)$) (in other words, when the network *resonates*). The function therefore correlates directly with network outputs, instead of activations. As a result, the learning rule is dynamically linked to the network's output (unlike most BAMs).

## Simulations

### Simulation 1

A first simulation was conducted in order to demonstrate the network's ability to cluster exemplars into categories. For this simulation, artificial pixel-based stimuli were created. Stimuli examples are shown in Figure 4.

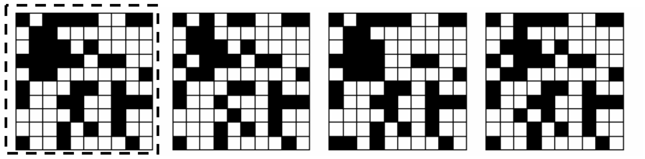


Figure 4: Examples of a prototype (left image) with three associated exemplars.

**Methodology** Eight category prototypes were produced by generating bipolar-valued vectors, for which the value of each vector position (or "feature") followed a random discrete uniform distribution. The presence of a feature (black pixel) was represented by a value of +1, and the absence of a feature (white pixel) by a value of -1. Each prototype vector comprised 100 features. Correlations between category prototypes are shown in Table 1.

Table 1: Correlations between category prototypes for Simulation 1[3].

| | P2 | P3 | P4 | P5 | P6 | P7 | P8 |
|---|---|---|---|---|---|---|---|
| **P1** | -0.04 | 0.04 | -0.02 | 0.12 | 0.14 | -0.04 | -0.16 |
| **P2** | | -0.04 | 0.06 | 0.08 | 0.30 | -0.08 | 0.12 |
| **P3** | | | 0.02 | -0.12 | 0.22 | 0.00 | -0.16 |
| **P4** | | | | 0.02 | 0.20 | 0.02 | -0.10 |
| **P5** | | | | | -0.02 | 0.04 | 0.08 |
| **P6** | | | | | | 0.02 | 0.02 |
| **P7** | | | | | | | -0.04 |

Ten exemplars were generated using each prototype, for a total of 80 items. Each exemplar was created by randomly "flipping" the value of between one to six features. Average within-category correlations are presented in Table 2.

[2] Generalization to real-valued stimuli can be found in Chartier & Boukadoum (2006a).

[3] **Px** represents the Category x prototype. In this Table, as well as all following Tables, *Pearson's r* scores are reported.

Table 2: Average within-category correlations for Simulation 1[4]

| C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 |
|---|---|---|---|---|---|---|---|
| 0.86 | 0.88 | 0.89 | 0.89 | 0.88 | 0.86 | 0.84 | 0.85 |

Learning followed the following procedure:

0. Random weight initializations;
1. Random selection of a given exemplar;
2. Weight updates according to Equations 5 & 6;
3. Repetition of 1 and 2 for 600 learning trials.

To assess the network's behavior, several simulations were performed using different numbers of units for the **W** layer.

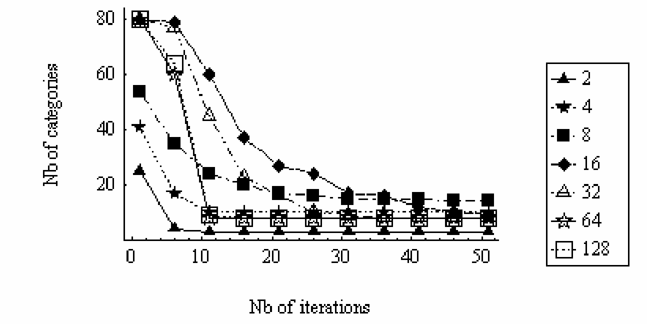


Figure 5: Number of clusters developed by the network as a function of the number of recall iterations. Each line represents a different number of units present in the **W** layer of the network.

**Results** Looking at Figure 5, one finds that for adequate clustering, the network needs at least $\log_2 n$ units. Here, because there were eight predetermined categories, the network thus needed a theoretical minimum of three units ($\log_2 8 = 3$). However, if there are much less units than exemplars, the network is likely to develop a greater number of categories than necessary. In this example, it was the case when 4, 8 or 16 units were used. If the number of units was sufficient, the number of categories developed by the network then matched the number of desired categories. In this example, when 32, 64 or 128 units were used, the network correctly developed eight categories.

In addition, when using 32 units or more, Figure 5 indicates that as the number of iterations increases, the network switches from a specific identification to a categorization process. For the first few iterations (~10 or less), there are almost as many clusters as there are exemplars. When achieving more iterations (~30 or more), the number of clusters is reduced so that it becomes similar to the number of prototypes (or categories). Consequently, if the number of units is great enough, FEBAM can be used to identify specific exemplars even though its memory has extracted the corresponding prototypes, depending on the time allowed (i.e. number of iterations performed) before an output is required.

## Simulation 2

The purpose of the second simulation was to study the number of categories developed by FEBAM as a function of the number of output units. In this simulation, not all units were initially available. A unit recruiting mechanism was introduced to allow the network to slowly converge towards a number of categories equal to the number of exemplars. A new unit was recruited by the model after a certain amount of learning trials had been achieved.

**Methodology** Four prototypes were generated using the method detailed for Simulation 1. Correlations between category prototypes are shown in Table 3.

Table 3: Correlations between category prototypes for Simulation 2[5].

| | P2 | P3 | P4 |
|---|---|---|---|
| **P1** | 0.02 | 0.08 | 0.08 |
| **P2** | | -0.02 | 0.02 |
| **P3** | | | 0.12 |

Ten exemplars were generated using each prototype, for a total of 40 items. Each exemplar was created by randomly flipping the value of between two to ten features. Average within-category correlations are presented in Table 4.

Table 4: Average within-category correlations for Simulation 2[6]

| C1 | C2 | C3 | C4 |
|---|---|---|---|
| 0.76 | 0.77 | 0.78 | 0.77 |

The simulation was conducted, starting with two initial units in the **W** layer. The number of units in the **V** layer remained constant at 100. After each phase of 600 learning trials, another unit was added to the **W** layer. This was repeated until there were 64 units. After each learning phase, a recall test was performed to estimate the number of categories developed by the network. At test, each given stimulus was iterated 200 times in the network before the final stabilized output was given. This was done to establish the nature of the network's attractors after each step.

Learning followed the following procedure:

0. Random weight initializations;

1a. Random selection of exemplars for learning, according to Equations 5 and 6;

1b. Repetition of 1a for 600 learning trials;

2. Addition of a new output unit;

3. Repetition of 1 and 2 until the number of output units is equal to 64.

[4] **Cx** represents Category x.

[5] **Px** represents the Category x prototype.

[6] **Cx** represents Category x.

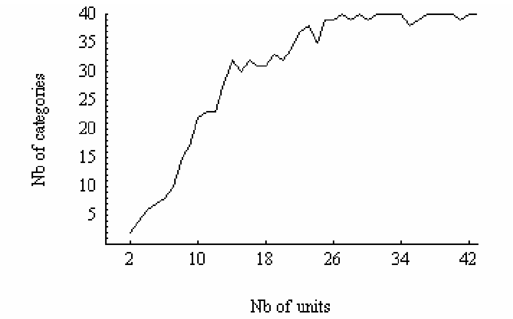


Figure 6: Simulation 2. Number of categories developed by the network as a function of the number of units iteratively added.

**Results** Figure 6 indicates that as more units are added to the network, more categories are developed. Hence, the network's architecture does "take advantage" of the recruiting mechanism properly, by separating the stimuli in more and more specific ways, iteratively going from groupings around a generic abstraction to single exemplars. Hence, this simulation shows that FEBAM can act as both an exemplar and a prototype memory, depending on the number of units that have been dynamically recruited.

This is clearly visible in Figure 7, which displays a tendency towards creating one-exemplar clusters as the number of possible clusters increases. Figure 7 also shows that while the unit recruiting mechanisms allows for some cluster reorganization (for example, some items associate with different cluster members when adding units), the exemplars always closely follow the predetermined categorical segmentation, that is they tend to cluster with items generated by the same prototype.

## Simulation 3

Simulation 2 was replicated, but this time using stimuli with no predetermined categorical (or cluster) membership. Pixel representations of letters of the alphabet were chosen because they represent a wide range of intercorrelations.

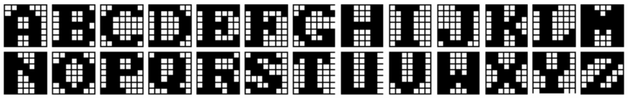


Figure 8: Set of patterns used for training.

**Methodology** The patterns used for the simulations are shown in Figure 8. Each pattern consisted of a 7 x 7 pixel matrix representing a letter of the alphabet. Once again, white and black pixels were respectively assigned corresponding values of -1 and +1. Correlations between the patterns varied from 0.02 to 0.84.

The simulation procedure was identical to that of Simulation 2, except for the number of stimuli involved (26 instead of 40).

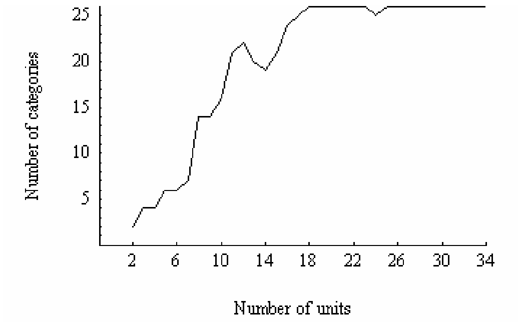
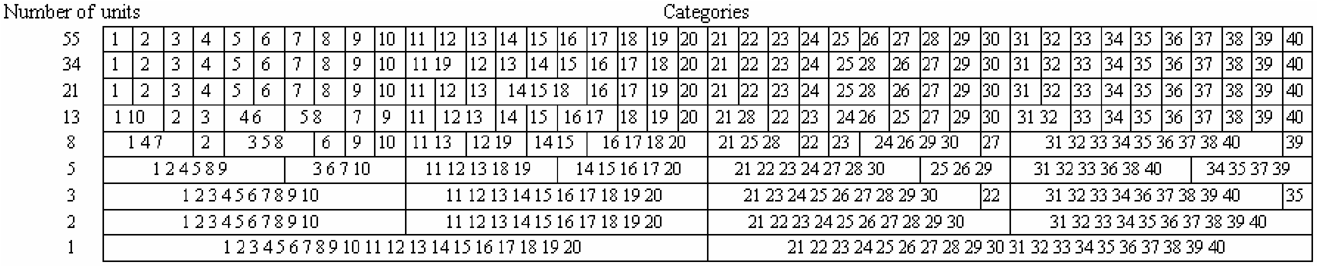


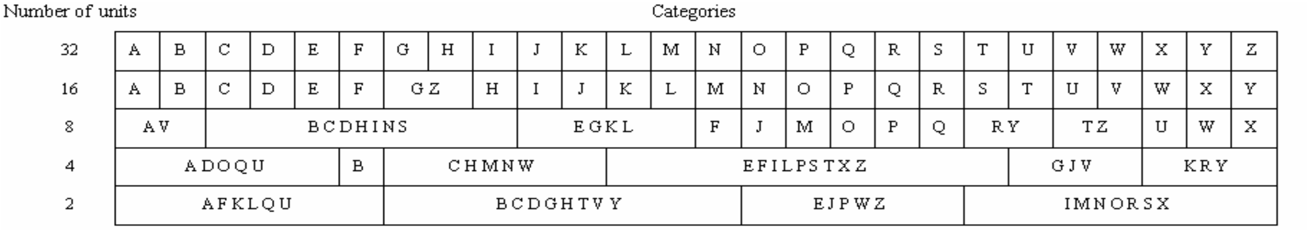
Figure 9: Simulation 3. Number of categories developed by the network as a function of the number of units iteratively added.

**Results** As in Simulation 2, Figure 9 indicates that the network takes advantage of the recruiting mechanism; as more units are added to the network, more categories are developed. However, as Figure 10 shows, a member from a given category is not tied to a specific type of categorical clustering. In fact, given exemplars can aggregate into a given cluster, and later on leave their present cluster to join another one, or form a new cluster with other exemplars. This contrasts markedly from classic clustering techniques.



| Number of units | Categories | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 55 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| 34 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 19 | | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 20 | 21 | 22 | 23 | 24 | 25 28 | | 26 | 27 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| 21 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 15 18 | | | 16 | 17 | 19 | 20 | 21 | 22 | 23 | 24 | 25 28 | | 26 | 27 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| 13 | 1 10 | | 2 | 3 | 4 6 | | 5 8 | | 7 | 9 | 11 | 12 13 | | 14 | 15 | 16 17 | | 18 | 19 | 20 | 21 28 | | 22 | 23 | 24 26 | | 25 | 27 | 29 | 30 | 31 32 | | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| 8 | 1 4 7 | | | 2 | 3 5 8 | | | 6 | 9 | 10 | 11 13 | | 12 19 | | 14 15 | | 16 17 18 20 | | | | 21 25 28 | | | 22 | 23 | 24 26 29 30 | | | | 27 | 31 32 33 34 35 36 37 38 40 | | | | | | | | | 39 |
| 5 | 1 2 4 5 8 9 | | | | | | 3 6 7 10 | | | | 11 12 13 18 19 | | | | | 14 15 16 17 20 | | | | | 21 22 23 24 27 28 30 | | | | | | | 25 26 29 | | | 31 32 33 36 38 40 | | | | | | 34 35 37 39 | | | |
| 3 | 1 2 3 4 5 6 7 8 9 10 | | | | | | | | | | 11 12 13 14 15 16 17 18 19 20 | | | | | | | | | | 21 23 24 25 26 27 28 29 30 | | | | | | | | | 22 | 31 32 33 34 36 37 38 39 40 | | | | | | | | | 35 |
| 2 | 1 2 3 4 5 6 7 8 9 10 | | | | | | | | | | 11 12 13 14 15 16 17 18 19 20 | | | | | | | | | | 21 22 23 24 25 26 27 28 29 30 | | | | | | | | | | 31 32 33 34 35 36 37 38 39 40 | | | | | | | | | |
| 1 | 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 | | | | | | | | | | | | | | | | | | | | 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 | | | | | | | | | | | | | | | | | | | |

Figure 7. Clusters developed by the network as a function of the number of units. Some exemplars cluster with different items as the number of units increases. As can be seen, as soon as the number of units (two) allows for the formation of four clusters, each formed cluster exhibits its predetermined belonging exemplars.



<table>
<tr><th>Number of units</th><th colspan="26">Categories</th></tr>
<tr><td>32</td><td>A</td><td>B</td><td>C</td><td>D</td><td>E</td><td>F</td><td>G</td><td>H</td><td>I</td><td>J</td><td>K</td><td>L</td><td>M</td><td>N</td><td>O</td><td>P</td><td>Q</td><td>R</td><td>S</td><td>T</td><td>U</td><td>V</td><td>W</td><td>X</td><td>Y</td><td>Z</td></tr>
<tr><td>16</td><td>A</td><td>B</td><td>C</td><td>D</td><td>E</td><td>F</td><td colspan="2">GZ</td><td>H</td><td>I</td><td>J</td><td>K</td><td>L</td><td>M</td><td>N</td><td>O</td><td>P</td><td>Q</td><td>R</td><td>S</td><td>T</td><td>U</td><td>V</td><td>W</td><td>X</td><td>Y</td></tr>
<tr><td>8</td><td colspan="2">AV</td><td colspan="7">BCDHINS</td><td colspan="4">EGKL</td><td>F</td><td>J</td><td>M</td><td>O</td><td>P</td><td>Q</td><td colspan="2">RY</td><td colspan="2">TZ</td><td>U</td><td>W</td><td>X</td></tr>
<tr><td>4</td><td colspan="5">ADOQU</td><td>B</td><td colspan="5">CHMNW</td><td colspan="9">EFILPSTXZ</td><td colspan="3">GJV</td><td colspan="3">KRY</td></tr>
<tr><td>2</td><td colspan="6">AFKLQU</td><td colspan="8">BCDGHTVY</td><td colspan="5">EJPWZ</td><td colspan="7">IMNORSX</td></tr>
</table>

Figure 10. Clusters developed by the network as a function of the number of units. Once again, cluster reorganization can be detected as the number of possible clusters increases.

## Discussion

In this paper, it has been shown that FEBAM, which is based on an associative learning architecture, is able to create increasingly precise clusters by recruiting additional units, and reorganize these clusters during training. Results from the first simulation have shown that the developed memory can be used to perform identification as well as categorization. This property is made possible by the dynamic memory recall process. In this case, specific exemplars are "transient" memories, while prototypes are attractors. However, simulations 2 and 3 show that by using a unit recruiting process, exemplars can also become attractors. In previous studies, FEBAM has been shown to achieve perceptual feature extraction and learning in noisy environments (Giguère, Chartier, Proulx & Lina, in press), as well as nonlinear principal component extraction and blind source extraction (Chartier, Giguère, Renaud, Lina & Proulx, in press). Moreover, FEBAM, being a special case of BAM, can also be used to simulate other applications such as categorization (Chartier & Proulx, 2005), classification (Chartier & Boukadoum, 2006a), many-to-one association and multi-step pattern recognition (Chartier & Boukadoum, 2006b). FEBAM is therefore believed to constitute a serious candidate for larger-scale cognitive modeling of human perceptual and categorical processes.

Further studies should investigate the frequency effects of both exemplars and categories. Based on results expressed by Figure 5, further studies should also investigate the categorization and classification processes as a given input (or exemplar) iterates through the network. FEBAM could ultimately link, through a dynamic memory system, both exemplar and prototype approaches. In its present form, the model is able to cluster together information if between-category variability is greater than within-category variability. Various variability clustering techniques adding an external teacher or reinforcement should therefore also be explored.

## References

Carpenter, G. A. and Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing. 37,* 54-115.

Chartier, S., Boukadoum, M. (2006a). A bidirectional heteroassociative memory for binary and grey-level patterns. *IEEE Transactions on Neural Networks, 17,* 385-396.

Chartier, S., Boukadoum, M. (2006b). A sequential dynamic heteroassociative memory for multistep pattern recognition and one-to-many association. *IEEE Transactions on Neural Networks, 17,* 59-68.

Chartier, S., Giguère, G., Renaud, P., Lina, J.M., Proulx, R. (2007, *in press*). FEBAM : a feature-extracting bidirectional associative memory. *Proceedings of the 20th International Joint Conference on Neural Networks.*

Chartier, S., Proulx, R. (2005). NDRAM: nonlinear dynamic recurrent associative memory for learning bipolar and nonbipolar correlated patterns. *IEEE Transactions on Neural Networks, 16,* 1393-1400.

Diamantaras, K.I., Kung, S.Y. (1996). *Principal Component Neural Networks.* New York: Wiley.

Giguère, G., Chartier, S., Proulx, R., Lina, J.M. (2007, *in press*). Creating perceptual features using a BAM-inspired architecture. *Proceedings of the 29th Annual Conference of the Cognitive Science Society.*

Grossberg, S. (1988). Nonlinear Neural Networks: Principles, Mechanisms, and Architectures. *Neural Networks, 1*, 17-61.

Hassoun, M.H. (1989). Dynamic heteroassociative neural memories. *Neural Networks, 2,* 275-287.

Hyvarinen, A., Oja, E. (2000). Independent component analysis: algorithms and applications, *Neural Networks, 13,* 411-430.

Kaplan, D., & Glass, L. (1995). *Understanding nonlinear dynamics* (1st ed.). New-York: Springer-Verlag.

Karhunen, J., Pajunen, P., Oja, E. (1998). The nonlinear PCA criterion in blind source separation: Relations with other approaches. *Neurocomputing, 22,* 5-20.

Kohonen, T. (1989). *Self-Organization and Associative Memory* (3rd ed.). Berlin: Springer-Verlag.

Kosko, B. (1988). Bidirectional associative memories. *IEEE Transactions on Systems, Man and Cybernetics, 18,* 49-60.

Kosko, B. (1990). Unsupervised learning in noise. *IEEE Transactions on Neural Networks*, *1*(1), 44-57.

Murphy, G.L. (2002). *The Big Book of Concepts.* Cambridge, MA: MIT Press.

Oja, E. (1989). Neural networks, principal components, and subspaces. *International Journal of Neural Systems, 1,* 61-68.

Sutton, R.S. (1988). Learning to predict by the methods of temporal difference. *Machine Learning, 3,* 9-44.

# The Emergence of Semantic Topography in a Neurally-Inspired Computational Model

**Lee I. Newman (leenewm@umich.edu)**
**Thad A. Polk (tpolk@umich.edu)**
Department of Psychology, 530 Church Street
Ann Arbor, MI 48109 USA

## Abstract

Representations in sensory cortices are organized topographically: auditory cortex is organized tonotopically, somatosensory cortex is organized somatotopically, and visual cortex is organized retinotopically. Substantial progress has been made in understanding how topography develops at a neurocomputational level, particularly in the early and middle stages of processing in the visual system. We extend this work to investigate how higher-level semantic representations could develop based on topographic input from sensory maps in the ventral visual pathway. The receptive fields of cells in these maps correspond to the loci of activity within a cortical topography rather than explicitly coded sensory features. Using this model, we show that meaningful semantic representations at increasing levels of abstraction naturally emerge as a result of exposure to a set of visual stimuli. For example, when presented with a set of simple visual features (color, texture, size, and shape) the model develops semantic representations that distinguish basic level categories (e.g., dogs, tables, cars), superordinate categories (e.g., animals, furniture, vehicles), and living versus nonliving things. This work therefore offers a computationally explicit hypothesis about how semantic representations could emerge in the brain. Our results suggest the possibility that high-order concept representations may be encoded topographically much the same way as low-order sensory representations, and that these representations may be learned based on the same principles of neural computation known to be operating in sensory cortex.

## Introduction

Topography is an important principle of neural organization and is present in all sensory cortices of the brain (Kandel, Schwartz, & Jessell, 2000). Topography represents a mapping from sensory space (e.g., location in the visual field, wavelength of light) to cortical space (e.g., location in extrastriate visual cortex). In such a mapping, nearby neurons in cortical space have similar receptive fields (they respond to nearby parts of sensory space) and are therefore selective for similar sensory features. Put simply, sensory similarity is reflected in cortical proximity.

Topographic representations have been particularly well-studied in the early processing stages in the ventral visual pathway of the primate brain. This pathway proceeds through a hierarchy of stages beginning with striate cortex (V1) and proceeding through extrastriate cortices (V2, V4) to inferotemporal cortex (IT). In the early stages of processing, receptive fields are small and neurons are highly selective for primitive sensory features (e.g., specific retinal locations, orientation of bars of light, and colors). As processing progresses receptive fields become increasingly large and neurons become selective for more abstract and complex features (e.g., specific configurations of features, and simple objects).

Physiological studies have established that topographies are present in the early and middle stages of the ventral pathway (V1, V2, V4) and the structure of these topographies has been well-described in the literature (Fujita, Tanaka, Ito, & Cheng, 1992; Hadjikhani, Liu, Dale, Cavanagh, & Tootell, 1998; Livingstone & Hubel, 1984; Shipp & Zeki, 1989, 2002a, 2002b; Tanaka, 1996; Tootell, Silverman, Hamilton, Devalois, & Switkes, 1988; Tootell, Switkes, Silverman, & Hamilton, 1988; Vanessen & Gallant, 1994). The primary organization is retinotopic, with neighboring neurons coding for sensory information located at nearby positions in the visual field. Embedded within this retinotopy are secondary topographies such as orientation columns and color blobs whose neurons are selective for particular object orientations and colors, respectively. Physiological findings also suggest that a more complex object-based topography may exist in IT cortex, but neither the structure nor learning mechanisms underlying this putative topography are fully understood (Fujita et al., 1992; Sigala & Logothetis, 2002; Tanaka, 2000).

At a computational level, significant progress has been made in understanding how sensory topographies develop in the early and middle stages of the ventral pathway. Biologically plausible models based on self-organizing learning algorithms have simulated topographic development computationally and have been able to successfully reproduce physiological data (Barrow, Bray, & Budd, 1996; Carreira-Perpinan, Lister, & Goodhill, 2005; Goodhill, 1993; Goodhill & Willshaw, 1994; Olson & Grossberg, 1998; Sirosh & Miikkulainen, 1997; Sit & Miikkulainen, 2006). All these models are based on two well-established neural mechanisms, competition and Hebbian learning, that operate throughout neocortex.

In this work we consider what role, if any, these self-organizing mechanisms might play in the learning of higher-order semantics. Specifically, we use a computational model to investigate the type of representation that develops in a later stage cortical map that (i) receives topographically organized sensory inputs, and (ii) self-organizes based on the same mechanisms of neural computation known to operate in the earlier processing stages of the ventral pathway. We find that after exposing the model to a set of visual stimuli, the topography reflects multiple levels of semantic categories more than low-level visual similarities.

An early investigation of the role of topographic structure in semantics was conducted by Ritter & Kohonen (1989). In this work, the authors showed that a SOM could produce a topography of logical word roles (e.g., subject nouns, object nouns, verbs) based on the statistical structure of word context. Applied work in content retrieval (Laaksonen, Koskela, & Oja, 2002; Laaksonen & Viitaniemi, October, 2006) have successfully used SOMs to organize images based on visual similarity. Our model differs from prior work in that it focused on the visual modality, has a hierarchical structure more closely tied to known hierarchical structure of cortex, and has the objective of understanding psychological phenomena associated with semantics.

## Self Organizing Maps

Self-organizing maps (SOMs) are a computational abstraction of cortical representation and processing (Kohonen, 1982, 1990). SOMs correspond to a locally connected population of neurons in a contiguous area of cortical tissue. As shown in Figure 1, the basic unit of representation within a SOM is the cell. Cells within a SOM are indexed based on their spatial location and each is modeled as a k-length *weight vector* specifying the preferred k-length input for the cell (i.e., the input that causes the cell to fire maximally). When presented with an input pattern, cells within a SOM compete to represent this pattern. The response of each cell is based on the similarity between its weight vector and the input pattern. The *winning cell* is the cell most similar to the input.

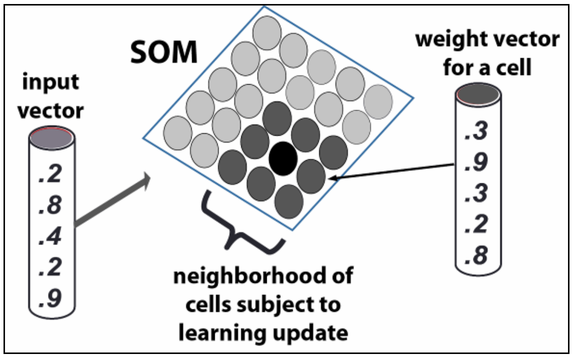


Figure 1: Self organizing map.

SOM learning is accomplished by modifying the weight vector of the winning cell so that it is more similar to the input, therefore making this cell more likely to win again with future presentations. Critically, the weight vectors of cells in close spatial proximity to the winning cells are also updated. As a result, with experience the response of a SOM tends to become spatially organized, learning a mapping from the statistical regularities discovered in its k-dimensional input space to a set of topographically organized neighborhoods within the SOM.

Mathematically, if $\boldsymbol{x}$(t) is a vector representing the input to a SOM at time t and $\boldsymbol{w}_i$(t) represents the weight vector of cell i at time t, then the winning cell c is given by

$$ArgMin_{\mathrm{c}}\ \{\| \boldsymbol{x}(\mathrm{t}) - \boldsymbol{w}_c(\mathrm{t}) \|\}, \quad (1)$$

where the $\|\cdot\|$ operator denotes vector distance. We use Euclidean distance for all distance computations in our model. The weight vector for each cell i is updated according to the following SOM learning equation

$$\boldsymbol{w}_i(\mathrm{t}+1) = \boldsymbol{w}_i(\mathrm{t}) + \alpha(\mathrm{t})\, h_{c,i}(\mathrm{t})\{ \boldsymbol{x}(\mathrm{t}) - \boldsymbol{w}_c(\mathrm{t}) \}, \quad (2)$$

where α(t) is a time-dependent learning rate, and $h_{c,i}$(t) is a kernel function that is centered on the winning cell c and that computes the magnitude of the update to cell i based on its spatial proximity to the winning cell. In our model we use the the Gaussian kernel function

$$h_{c,i}(t) = \exp\{-\frac{1}{\lambda(\mathrm{t})^2}\left\|\mathrm{i}_{(x,y)} - \mathrm{c}_{(x,y)}\right\|^2\}, \quad (3)$$

where λ(t) is a time-dependent parameter that determines the width of the kernel, and $\mathrm{i}_{(x,y)}$ and $\mathrm{c}_{(x,y)}$ denote the map coordinates of cells i and c, respectively. We compute α(t) and λ(t) as exponentially decreasing functions of time, with time denoting discrete presentations of training inputs.

While SOMs are not detailed biophysical models of cortex, the core assumptions embodied in this class of models can be mapped directly to their biophysical correlates: The weight vectors of SOM cells correspond biologically to the concept of receptive fields; winning cells are analogous to the peak location of activity within a cortical area, governed by the net result of the competitive interplay between local excitatory and inhibitory activity driven by an input; and the spatially-localized learning algorithm is a computational abstraction of Hebbian learning that occurs between neurons participating in a bump of cortical activity and the active afferent neurons providing their input. SOMs therefore provide a computationally efficient and biophysically plausible method for modeling the development of spatially structured representations in cortex.

## Methods

### Model Architecture

As shown in Figure 2, our model is a two-level hierarchy of SOMs. The first level consists of a set of four 10x10 *sensory maps* each corresponding to a particular visual feature: color, size, shape, and surface appearance. Each sensory map receives inputs in the form of real-valued sensory vectors, for example the color map receives three-vectors representing the hue, saturation and brightness of the stimuli. These maps are then exposed to a set of visual stimuli and are allowed to self-organize according to the SOM learning equations given by (1), (2) and (3).

The sensory maps capture the type of abstract featural representations found in the later stages of the ventral visual pathway when retinotopy is no longer present. With retinotopy no longer available as a basis of representation, subsequent cortical areas must somehow make sense of a more abstract sensory topography. It is important to note that the sensory maps in our model are not intended to explicitly instantiate the computations performed from early to late stage visual cortex (for one example of such a model, see Riesenhuber & Poggio, 1999; Serre, Oliva, & Poggio, 2007), but rather they serve as plausible summaries of how concrete visual information provided by a stimulus is ultimately encoded in abstract representations in the later stages of visual cortex.

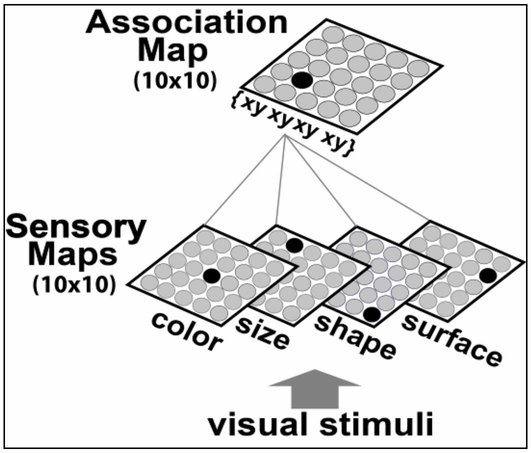


Figure 2: Architecture of the model.

The second level in the model consists of a single 10x10 *association map* that receives convergent inputs from the lower-level sensory maps. Computationally, the input to these maps is a concatenation of the spatial locations of activity in each of the sensory maps. The implication of this method of coding is that the receptive fields of cells in the association map correspond to *cortical coordinates* rather than explicitly encoded sensory-based features. We believe this is a subtle, but important aspect of the model. In the early stages of visual processing, the basis of neural organization is either driven by hardwired anatomical connections (as in the case of retinotopy) or by topography organized around concrete, low-level sensory-based features (as with orientation columns). In the later stages of visual processing, no such representational “boostrapping” is available. These later-stage cortical areas must somehow make sense of more abstract and complex representations based on the spatial location of activity in upstream maps. The association map in our model faces the same challenge: to learn meaningful representations based on spatially encoded inputs from the sensory maps.

## Simulation Procedures

**Stimuli** The model was repeatedly presented with a set of 96 visual stimuli consisting of 8 classes of objects (bicycles, bushes, cars, cats, dogs, chairs, tables, and trees) and 12 variants within each class. The variants captured characteristic within-class featural differences, for example trees of varying color and size. Each stimulus was coded as a [0,1] normalized 10-vector based on its color (hue, saturation, brightness), size (x, y, and z dimension in a typical viewing angle), shape (roundness, complexity) and surface appearance (smoothness, textural uniformity). The vector values were estimated based on images collected from Google™ Images (http://images.google.com).

**Model Learning** Weight vectors of all cells in the SOMs were initialized to random values in the range [0.1, 0.9]. Learning then proceeded in two phases. First, each of the sensory SOMs were presented with the relevant vector component of the training stimuli (e.g., the color map was presented with hue, saturation and brightness values) and the weight vectors were updated according to equations (1), (2) and (3). In each of 500 learning iterations, the presentation order of the stimuli was randomized to minimize order effects. After training, the stimuli were presented to each of the sensory maps, and the map coordinates of the winning cells for each input were computed for each of the four maps. The four pairs of coordinates for each stimulus were then concatenated and the resulting vectors were then used for the second phase of learning. In this phase, the association map were trained using an identical procedure, with the exception that the input patterns presented to the association map were the concatenated outputs from the trained sensory maps.

## Results

Figure 3 shows an example of the representation that is learned by one of the lower-level sensory maps, in this case, the map that was trained using size information for each stimulus[1]. The winning cell for each stimulus is labeled in the figure based on its object class. Inspection of this map reveals that there is no clear object-based topography, as only two of the object classes (cats and bikes) are co-located in the same region of the map. Instead, the map captures an abstract size-based topography in which larger objects are represented in the upper region of the map (smaller objects in the lower region), and taller objects are represented towards the left side of the map (wider objects towards the right). Similar results were found in each of the other sensory maps: an abstract sensory-based topography emerged, but stimuli from the same object class were not co-located in the map.

Figure 4 shows the response[2] of each cell in the size map when presented with one of the tree stimuli. As is evident, the map response to the stimulus is spatially localized to a neighborhood of activity surrounding the winning cell.

[1] Many of the variants within each object class share the same values and therefore not all 96 stimuli are visible due to overlap.

[2] Cell responses were computed based on the Euclidian distance between the stimulus and the cell’s weight vector. Smaller distances imply greater similarity and larger cell responses.

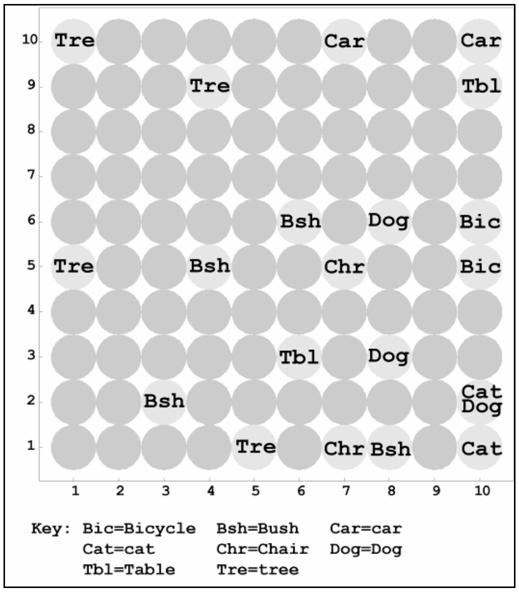


Figure 3: Sensory size map after learning.

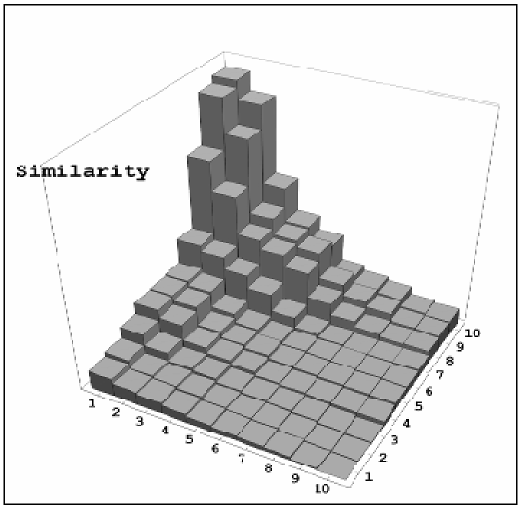


Figure 4: Response of the size map to a *tree* stimulus.

The structure of the association map after learning is shown in Figure 5. The winning cell for each stimulus is labeled in the figure based on object class. In contrast to the sensory maps which learned abstract sensory-based topographies, a clear object-based semantic topography emerged in the association map after learning. For example, a neighborhood of cells responsive to tables emerged in the lower left region of the map, and a set of cells most responsive to trees are spatially co-located in the upper right region. For some object classes, the neighborhoods overlap, as in the case of dogs and cats, and trees and bushes, indicating that the map was unable to distinguish these classes of stimuli based on available visual information. Nevertheless, for all object classes the map learned to represent the class in a spatially co-located region of cells.

Further inspection of the learned association map in Figure 5 reveals a semantic topography that captures superordinate categorical distinctions among the object classes. For example, in the lower-left region of the map there is a neighborhood of cells responsive to chairs and tables (furniture), and similarly there are distinct regions of cells whose receptive fields prefer trees and bushes (plants), bikes and cars (vehicles), and dogs and cats (animals). Furthermore, the topography encoded in the map also learned the semantic distinction between living and non-living stimuli. Cells representing dogs, cats, trees, and bushes (living things) are co-located in the upper region of the map and tables, chairs, cars, and bikes (non-living things) are co-located in neighborhood of cells in the lower region of the map.

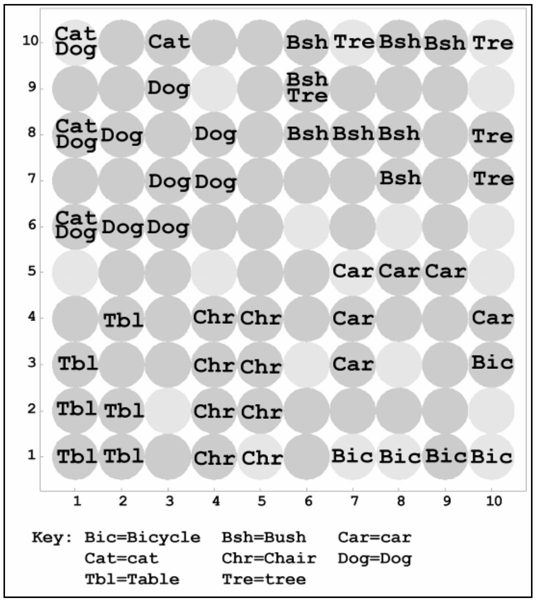

10 Cat Dog Cat Bsh Tre Bsh Bsh Tre
9 Dog Bsh Tre
8 Cat Dog Dog Dog Bsh Bsh Bsh Tre
7 Dog Dog Bsh Tre
6 Cat Dog Dog Dog
5 Car Car Car
4 Tbl Chr Chr Car Car
3 Tbl Chr Chr Car Bic
2 Tbl Tbl Chr Chr
1 Tbl Tbl Chr Chr Bic Bic Bic Bic
1 2 3 4 5 6 7 8 9 10
Key: Bic=Bicycle Bsh=Bush Car=car
Cat=cat Chr=Chair Dog=Dog
Tbl=Table Tre=tree

Figure 5: Association map after learning.

The responses of the association map to two sets of similar stimuli are shown in Figure 6. Each class of stimulus shown in the figure (table, chair, tree, and bush) produces a graded, locally organized response around the winning cell. Furthermore, the responses of the map to similar classes of stimuli (table and chair, tree and bush) share a similar subset of active cells, demonstrating the existence of topography in the form of neighborhoods of cells responsive to stimuli at multiple levels of abstraction (object class and superordinate category).

Although these results confirm the hypothesis that high-level semantic topography can be driven by the same principles of neural computation found in the lower levels of the visual pathway, we raise two concerns. First, it is possible that these results are an artifact of the learning procedure due to the random initialization of the maps and/or the random ordering in which the stimuli were presented. To address this concern, we simulated the model over a large number of random seeds and confirmed that the structure of the learned topography in the association map was consistent across all simulations. Although the specific location of the category clusters varied, the clusters themselves reliably emerged as did the superordinate organization (plants, animals, furniture, and vehicles) and the living versus non-living distinction.

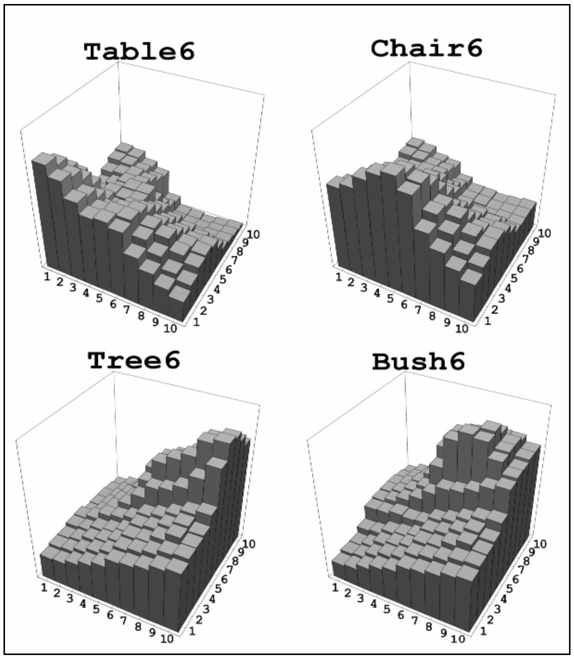


Figure 6: Association map responses.

A second concern is that the semantic organization shown in Figure 5 is an artifact of the way that the stimuli were coded. It is possible that the 96 stimulus vectors were trivially separable at multiple levels of abstraction, and therefore the resultant topography was in some sense predetermined. To address this concern, we performed a hierarchical clustering[3] of the training stimuli, producing the tree shown in Figure 7. As is evident, hierarchical clustering does not produce the same meaningful categorical clusters of stimuli produced in our model: At the level of object classes, only bicycles are distinguished from the other classes (i.e., they share a common branch within the hierarchy); at the level of superordinate category, only plants and animals are independently clustered; and at the highest level, the clustering categorically generates two heterogeneous groups: plants-bicycles and cars-tables-chairs-animals.

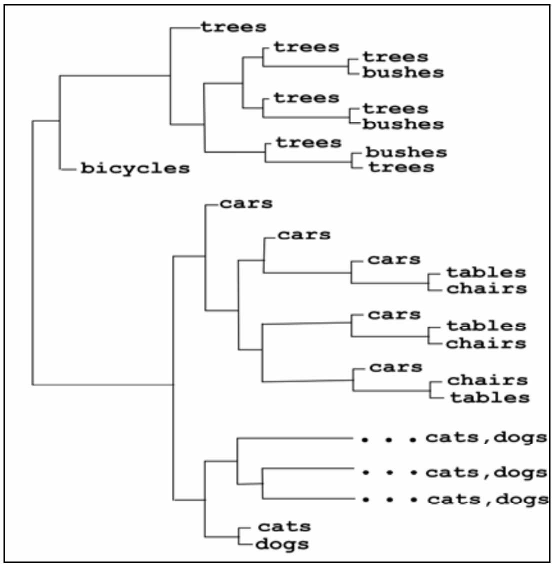


Figure 7: Clustering tree of the training stimuli.

## Discussion

The aim of this work was a plausibility proof of the hypothesis that the principles of self-organization and cortical topography at work in lower-level visual processing may also drive the learning and structure of higher-order semantics. We developed an explicit computational model in which a self-organizing map receives inputs from a hierarchy of topographic sensory maps and we found that meaningful semantic representations at increasing levels of abstraction naturally emerge as a result of exposure to a set of sensory stimuli. Specifically, when presented with a set of simple visual features (color, size, texture, shape) the higher-level map develops a topography of semantic representations that distinguishes basic level categories (bicycles, bushes, cars, cats, dogs, chairs, tables, and trees), superordinate categories (plants, animals, furniture, vehicles), and living versus nonliving things.

This work therefore offers a computationally explicit hypothesis about how semantic representations could emerge in the association cortex of the brain. Our results may also be relevant to the ongoing debate about whether conceptual knowledge is primarily organized anatomically by modality as posited by the sensory-functional hypothesis (Farah & McClelland, 1991; Warrington & McCarthy, 1983; Warrington & Shallice, 1984), or by category (Caramazza & Mahon, 2003). The results of our simulations suggest fundamental mechanisms of neural computation could lead to the emergence of topographically organized semantic representations without the need to posit that these representations are innate.

[3] The clusters were computed using the hierarchical clustering algorithm in Mathematica® using a Euclidian distance function.

## References

Barrow, H. G., Bray, A. J., & Budd, J. M. L. (1996). A self-organizing model of "color blob" formation. *Neural Computation, 8*(7), 1427-1448.

Caramazza, A., & Mahon, B. Z. (2003). The organization of conceptual knowledge: The evidence from category-specific semantic deficits. *Trends in Cognitive Sciences, 7*(8), 354-361.

Carreira-Perpinan, M. A., Lister, R. J., & Goodhill, G. J. (2005). A computational model for the development of multiple maps in primary visual cortex. *Cerebral Cortex, 15*(8), 1222-1233.

Farah, M. J., & McClelland, J. L. (1991). A computational model of semantic memory impairment - modality specificity and emergent category specificity. *Journal of Experimental Psychology-General, 120*(4), 339-357.

Fujita, I., Tanaka, K., Ito, M., & Cheng, K. (1992). Columns for visual features of objects in monkey inferotemporal cortex. *Nature, 360*(6402), 343-346.

Goodhill, G. J. (1993). Topography and ocular dominance - a model exploring positive correlations. *Biological Cybernetics, 69*(2), 109-118.

Goodhill, G. J., & Willshaw, D. J. (1994). Elastic net model of ocular dominance - overall stripe pattern and monocular deprivation. *Neural Computation, 6*(4), 615-621.

Hadjikhani, N., Liu, A. K., Dale, A. M., Cavanagh, P., & Tootell, R. B. H. (1998). Retinotopy and color sensitivity in human visual cortical area v8. *Nature neuroscience, 1*(3), 235-241.

Kandel, E. R., Schwartz, J. H., & Jessell, T. M. (2000). *Principles of neural science*. New York: McGraw-Hill, Health Professions Division.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics, 43*(1), 59-69.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE, 78*(9), 1464-1480.

Laaksonen, J., Koskela, M., & Oja, E. (2002). Picsom - self-organizing image retrieval with mpeg-7 content descriptors. *Ieee Transactions on Neural Networks, 13*(4), 841-853.

Laaksonen, J., & Viitaniemi, V. (October, 2006). *Emergence of ontological relations from visual data with self-organizing maps.* Paper presented at the 9th Scandinavian Conference on Artificial Intelligence, Espoo, Finaland.

Livingstone, M. S., & Hubel, D. H. (1984). Anatomy and physiology of a color system in the primate visual-cortex. *Journal of Neuroscience, 4*(1), 309-356.

Olson, S. J., & Grossberg, S. (1998). A neural network model for the development of simple and complex cell receptive fields within cortical maps of orientation and ocular dominance. *Neural Networks, 11*(2), 189-208.

Riesenhuber, M., & Poggio, T. (1999). Hierarchical models of object recognition in cortex. *Nature Neuroscience, 2*(11), 1019-1025.

Ritter, H., & Kohonen, T. (1989). Self-organizing semantic maps. *Biological Cybernetics, 61*(4), 241-254.

Serre, T., Oliva, A., & Poggio, T. (2007). A feedforward architecture accounts for rapid categorization. *Proceedings of the National Academy of Sciences of the United States of America, 104*(15), 6424-6429.

Shipp, S., & Zeki, S. (1989). The organization of connections between areas v5 and v1 in macaque monkey visual-cortex. *European Journal of Neuroscience, 1*(4), 309-332.

Shipp, S., & Zeki, S. (2002a). The functional organization of area v2, i: Specialization across stripes and layers. *Visual neuroscience, 19*(2), 187-210.

Shipp, S., & Zeki, S. (2002b). The functional organization of area v2, ii: The impact of stripes on visual topography. *Visual neuroscience, 19*(2), 211-231.

Sigala, N., & Logothetis, N. K. (2002). Visual categorization shapes feature selectivity in the primate temporal cortex. *Nature, 415*(6869), 318-320.

Sirosh, J., & Miikkulainen, R. (1997). Topographic receptive fields and patterned lateral interaction in a self-organizing model of the primary visual cortex. *Neural Computation, 9*(3), 577-594.

Sit, Y. F., & Miikkulainen, R. (2006). Self-organization of hierarchical visual maps with feedback connections. *Neurocomputing, 69*(10-12), 1309-1312.

Tanaka, K. (1996). Representation of visual features of objects in the inferotemporal cortex. *Neural Networks, 9*(8), 1459-1475.

Tanaka, K. (2000). Mechanisms of visual object recognition studied in monkeys. *Spatial vision, 13*(2-3), 147-163.

Tootell, R. B. H., Silverman, M. S., Hamilton, S. L., Devalois, R. L., & Switkes, E. (1988). Functional-anatomy of macaque striate cortex .3. Color. *Journal of Neuroscience, 8*(5), 1569-1593.

Tootell, R. B. H., Switkes, E., Silverman, M. S., & Hamilton, S. L. (1988). Functional-anatomy of macaque striate cortex .2. Retinotopic organization. *Journal of Neuroscience, 8*(5), 1531-1568.

Vanessen, D. C., & Gallant, J. L. (1994). Review: Neural mechanisms of form and motion processing in the primate visual-system. *Neuron, 13*(1), 1-10.

Warrington, E. K., & McCarthy, R. (1983). Category specific access dysphasia. *Brain, 106*(DEC), 859-878.

Warrington, E. K., & Shallice, T. (1984). Category specific semantic impairments. *Brain, 107*(SEP), 829-854.

# Simulating the Noun-Verb Asymmetry in the Productivity of Children's Speech

**Daniel Freudenthal (D.Freudenthal@Liv.Ac.Uk)**
**Julian M. Pine (Julian.Pine@Liv.Ac.Uk)**
School of Psychology, University of Liverpool

**Fernand Gobet (Fernand.Gobet@Brunel.Ac.Uk)**
School of Social Sciences, Brunel University

## Abstract

Several authors propose that children may acquire syntactic categories on the basis of co-occurrence statistics of words in the input. This paper assesses the relative merits of two such accounts by assessing the type and amount of productive language that results from computing co-occurrence statistics over conjoint and independent preceding and following contexts. This is achieved through the implementation of these methods in MOSAIC, a computational model of syntax acquisition that produces utterances that can be directly compared to child speech, and has a developmental component (i.e. produces increasingly long utterances). It is shown that the computation of co-occurrence statistics over conjoint contexts or frames results in a pattern of productive speech that more closely resembles that displayed by language learning children. The simulation of the developmental patterning of children's productive speech furthermore suggests two refinements to this basic mechanism: inclusion of utterance boundaries, and the weighting of frames for their lexical content.

## Introduction

Children acquiring their native language are faced with a task of considerable complexity. They need to acquire a system described by syntactic rules as well as the syntactic categories over which these rules are defined. This problem has been referred to as the 'bootstrapping problem'. Several solutions to the bootstrapping problem have been suggested. The *distributional* approach makes use of the fact that words that belong to the same word class tend to be preceded and followed by similar words. Thus, nouns tend to be preceded by determiners and adjectives, and followed by verbs. Similarly, verbs are preceded by (pro)nouns and followed by determiners and (pro)nouns. A system that tracks the overlap in the lexical items that precede and follow individual words may therefore be able to cluster these words into syntactic classes. These word classes could then potentially be used to infer phrasal categories such as noun phrase and verb phrase (see e.g. Finch & Chater, 1994).

The success of this second stage of the learning mechanism depends crucially on the quality of the syntactic classes that were derived in the first stage. For this reason, several researchers have explored different mechanisms for computing co-occurrence statistics and the effects these have on the quality of the derived classes. Finch & Chater (1994) analysed a 40,000,000 word corpus of USENET newsgroup data and used the rank order correlation between the (independent) sets of two word phrases that preceded and followed target words to inform a hierarchical cluster analysis that derives word classes. Redington, Chater & Finch (1998) perform a similar analysis on a corpus of several million words of child directed speech obtained from the CHILDES data base. They compared the performance over contexts of length one and two and found that the quality of results was very similar.

Mintz (2003) uses a slightly different approach. Mintz introduces the concept of a *frame*; two jointly occurring words with one word intervening. Computing co-occurrence statistics over conjoined pairs rather than independent sets of preceding and following words has the desirable property that it is more constraining and is therefore likely to lead to grammatical categories that are of higher quality. Mintz restricts his analysis to frames that have a high frequency in the input, and finds that the items that co-occur in these frames have a high likelihood of belonging to the same word class. Mintz does not perform a cluster analysis but does suggest that more comprehensive classes can be obtained using a relatively simple unification procedure based on overlap in the words contained in the classes.

The approach taken by Mintz, Redington et al. and Finch & Chater clearly shows there is a considerable amount of information in the distributional characteristics of the input that could potentially be used by a child acquiring language. Freudenthal et al. (2005a), however, argue that these approaches suffer from an inherent difficulty as they fail to consider how derived categories can actually be used in the production of (novel) utterances. Freudenthal et al. report work on MOSAIC, a computational model of language acquisition that produces actual utterances as output and implements a mechanism for the production of novel utterances that is very similar to that implemented by Redington et al. MOSAIC links together words that are followed and preceded by similar words and substitutes these words when producing output from the model. The fact that MOSAIC produces actual output results in potential classification errors quickly becoming apparent. Freudenthal et al. also argue that such classification errors may not always be apparent using the standard evaluation measures of accuracy and completeness as these depend on researcher's intuitions regarding an item's grammatical class. Inspection of the (representative) verb classes reported by Mintz (2003), for example, shows that these include past tense, present tense and progressive verb forms, imperative and non-imperative verb forms, and transitive and intransitive verbs. Substituting such items in production will quickly lead to errors that may not be apparent when using a metric that simply classes all items as verbs.

A further advantage of producing actual utterances is that it allows for a comparison between the output of one's language learning model and the characteristics of actual child speech. Asymmetries in children's tendency to generalize across words from different syntactic categories can then be used to inform the implementation of mechanisms that compute co-occurrence statistics. One important asymmetry that has been identified in the recent developmental literature is that children tend to be more conservative in their substitution of verbs than in their substitution of nouns. For example, several experimental studies have shown that children will readily substitute novel nouns in familiar verbal contexts, but tend to restrict their use of novel verbs to contexts in which they have heard them used in the input (see e.g. Akhtar & Tomasello, 1997; Tomasello, 2000). Moreover, Fisher (2002) points out that this pattern of generalization is precisely what one would expect given the nature of the system that the child is acquiring since restrictions on the argument structures in which different verbs can occur mean that generalizations across verbs tend to be 'riskier' than generalizations across nouns.

When taken together, these considerations place strong constraints on the development of mechanisms for extracting syntactic categories since they suggest the need for a mechanism that generalizes more readily across nouns than across verbs. They also suggest that simulating this asymmetry will not only increase the child-likeness of the model's output, but also reduce the probability of generating ungrammatical utterances.

Further constraints on the feasibility of mechanisms for the extraction of syntactic categories become apparent when considering the fact that such mechanisms are likely to be used by children acquiring a language. The most notable way in which child speech differs from adult speech is that it is considerably shorter. Children initially produce utterances that are only one or two words long. The mean length of their utterances (MLU) slowly increases as they grow older. This restriction on the length of children's speech suggests that the length of the phrases that children represent is considerably shorter than the length of the phrases they hear. This considerably reduces the number and type of contexts that could potentially feed into a system that computes co-occurrence statistics. Failure to consider this developmental component may therefore lead to researchers considering mechanisms that utilize information that is not necessarily available to a child.

The aim of this paper is to assess the relative virtues of using conjoined or independent sets of preceding and following items for the extraction of syntactic categories. This is done in the context of MOSAIC, an implemented model of language acquisition that has a developmental component. Specific attention will be given to how well the different mechanisms approximate the noun-verb asymmetry apparent in children's productive speech, as well as the constraints that result from simulating children's increasing utterance length.

The remainder of this paper is organized as follows. First, we briefly describe MOSAIC, the model that is used as a test bed for mechanisms for the extraction of syntactic categories. Next, MOSAIC's current mechanism for linking distributionally similar items is described. Finally, a number of substitution mechanisms are implemented and the output evaluated in terms of quantity and plausibility.

## MOSAIC

MOSAIC (Model of Syntax Acquisition in Children) is a computational model that has mostly been applied to the cross-linguistic simulation of the development of finiteness marking in children acquiring their native language (Freudenthal, Pine and Gobet 2006). MOSAIC learns off realistic, child directed input and learns to produce progressively longer utterances that can be directly compared to child speech. The basis of MOSAIC is an n-ary discrimination net that is used to incrementally store (fragments of) the utterances that MOSAIC has seen as input. Learning in MOSAIC is anchored at the beginning and end of the utterances to which it is exposed. That is, MOSAIC only learns a phrase when everything preceding or following that phrase in the utterance has already been encoded in the network. MOSAIC produces two types of utterances: utterance-final phrases and concatenations of utterance-initial and utterance-final phrases: utterances with missing sentence-internal elements. MOSAIC's mechanism for producing incomplete phrases has been shown to provide a good fit to the Optional Infinitive phenomenon across four languages: English, Dutch, German and French (Freudenthal et al, 2005b). Fig. 1 shows a sample MOSAIC network.

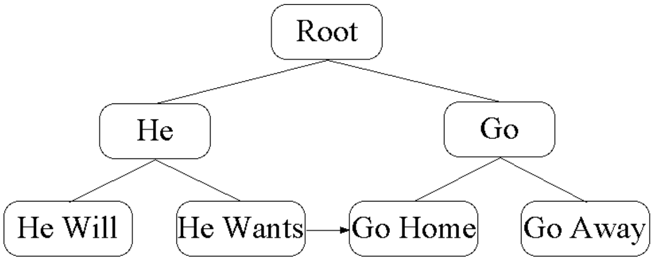


Figure 1: A partial MOSAIC network. The sentence-initial phrase *he wants*, and the sentence-final phrase *go home* have been associated, allowing the model to produce the utterance *He wants go home*. The model is also capable of producing the phrases *go home* and *go away*.

MOSAIC is capable of producing output with an increasing MLU. This is because learning is generally slow. Input is fed though MOSAIC and output is generated after every presentation of the input. The amount and length of phrases encoded in a MOSAIC network increases with every exposure to the input. Thus, developmental change can be simulated by analyzing increasingly mature models and matching the MLU of the respective models with that of children at different developmental stages.

## Productivity in MOSAIC

MOSAIC's mechanism for producing novel utterances is very similar to that described by Redington, Chater & Finch. For all nodes in the network, the preceding and following context (when encoded in the model) is stored. These contexts take the form of two independent lists of words that preceded and followed the target item. Thus, MOSAIC does not implement the notion of a frame, but assesses the preceding and following context independently. MOSAIC then considers the overlap between the contexts for pairs of words. If, for two words, the overlap in the words in both the preceding and following context exceeds a predetermined threshold, they are considered equivalent, and are connected through a *generative link*. Two words that are connected through a generative link can be substituted for each other in production. Thus, if the model has encoded the phrase *the red ball* and the words *red* and *blue* share sufficient overlap, the model is capable of producing the novel phrase *the blue ball*. MOSAIC is capable of substituting several words in an utterance. Thus, MOSAIC would be able to produce the phrase *a blue ball* if the words *a* and *the* also share a generative link.

The proportion of novel utterances in MOSAIC's output varies as a function of the mean length of the output. Typically, productivity increases from around 5% novel utterances at an MLU of 2 to around 50% productivity at an MLU of around 4. MOSAIC thus does not produce many novel utterances in the early stages of development. Productivity in MOSAIC also tends to revolve around verbs rather than nouns. Thus, MOSAIC tends to link together verbs more frequently than nouns. This runs counter to the notion that children are more conservative in substituting verbs than nouns. It also increases the risk of MOSAIC generating a high proportion of utterances that are ungrammatical, as verb-verb substitutions are more likely to be ungrammatical than noun-noun substitutions.

## Implementing Frames in MOSAIC

Mintz performed an analysis of a number of corpora of child directed speech, and analyzed the contents of the 50 most frequent frames in each corpus. Items that co-occurred in one of these frames were considered equivalent. While this approach shows the potential value of using frames, it needs to be developed into a more dynamic and probabilistic mechanism in order to be suited for a model that encodes and produces progressively longer utterances.

We implemented the mechanism in a similar way to MOSAIC's current generativity mechanism: two items are considered equivalent if there is sufficient overlap in the frames they occur in. An example may serve to illustrate how the dynamics of the frames and independent contexts may differ. Suppose the model has encoded the phrases 'A man eats', and 'The man drinks'. The preceding independent context for the word 'man' is now (A, The) while the following context is (eats, drinks). The frame context for 'man' is ((A - drinks), (The - eats)). If the model now encodes the phrases 'A woman eats' and 'The woman drinks', the independent preceding context would be (A, The), and the independent following context would be (eats, drinks) while the frame context would be ((A - eats), (The - drinks)). The overlap for the independent contexts would thus be 100%. The overlap in frames however, is 0%.

While, in the above example, the notion of a frame is clearly more constraining, it should be noted that the list of independent contexts may actually grow more quickly than the list of frames. This is because the model only needs to encode a two-word phrase to add an item to the independent preceding or following context. Thus, if the model encodes the (incomplete) phrase 'rich man', the word 'rich' is added to the independent preceding context, thus lowering the preceding overlap without affecting the context in terms of frames, as no following context has been encoded. Thus, the dynamics of the overlap between two items will be different for frames and independent contexts. As these dynamics are also affected by the frequency and variety of the contexts in which items occur, they may well affect verbs and nouns differently.

A final note concerns the status of sentence boundaries. Mintz defines a frame as two conjoint lexical items (words) with one word intervening. Thus, frames that contain sentence boundaries are excluded. While on the face of it frames including sentence boundaries are less restrictive than lexical frames it should be noted that they can actually be quite informative with respect to a word's grammatical class. Thus, the frames 'THE - END' and 'A - END' are very frequent and contain a large number of nouns. For the present simulations both lexical frames and frames with sentence boundaries were encoded in the model. Separate analyses were run utilizing all frames or lexical frames only to investigate the impact of including frames containing sentence boundaries.

# The simulations

The simulations reported here were run using the corpora of child directed Speech for two English children (Anne and Becky). The corpora, which contain approximately 33,000 and 25,000 utterances respectively were fed through MOSAIC several times and output of increasing length was generated after every exposure to the input. The version of MOSAIC used for these simulation tracks both the frames and the independent preceding and following contexts for the words it encodes. Separate analyses were run using substitution of individual words on the basis of either the frames or the independent contexts.

### Results

Simulations with MOSAIC's standard generativity mechanism were run first. Words were substituted when the overlap in terms of independent preceding and following contexts exceeded a threshold of 25%. Output was generated at three different MLU points and analysed in terms of the percentage novel items and number of noun-noun and verb-verb substitutions.

As can be seen in Table 1 the model only starts to produce substantial amounts of novel utterances in the later stages of development. It is also apparent that, except during the earliest stage, the model is more productive around verbs than around nouns.

Table 1: Descriptive statistics for MOSAIC's output using substitution on the basis of independent contexts.

| Child | MLU | Proportion novel | Noun-subs | Verb-subs |
|---|---|---|---|---|
| Anne | 2.08 | .05 | 83 | 11 |
| | 3.16 | .27 | 1873 | 2755 |
| | 4.71 | .50 | 12026 | 18897 |
| Becky | 1.93 | .02 | 20 | 9 |
| | 2.95 | .22 | 1144 | 1223 |
| | 4.25 | .49 | 4558 | 13984 |

Next, the same analysis (at 25% overlap) was performed using the frame-based generativity mechanism. In this first analysis of frame-based substitution only lexical frames were used (thus, frames containing sentence boundaries were ignored). Table 2 gives the results of this analysis.

Table 2: Descriptive statistics for MOSAIC's output using frame-based substitution.

| Child | MLU | Proportion novel | Noun-subs | Verb-subs |
|---|---|---|---|---|
| Anne | 2.08 | .05 | 15 | 0 |
| | 3.11 | .29 | 6547 | 130 |
| | 4.05 | .29 | 13111 | 740 |
| Becky | 1.93 | .01 | 24 | 0 |
| | 2.86 | .18 | 1102 | 324 |
| | 3.58 | .19 | 3273 | 307 |

As can be seen in table 2 the restriction to frames results in MOSAIC showing a clear bias towards noun-noun substitutions. It is also apparent that, for the last developmental stage, the models are considerably less generative. In order to investigate if this decreased generativity accounts for the noun bias, we also ran the frame-based simulations with an overlap percentage of 20. This increased the generativity in the final developmental stage to .47 for Anne's model and .34 for Becky's model. Noun substitutions still outnumbered verb substitutions by about 15 to 1 for Anne, and 5 to 1 for Becky's model. Thus, the frame-based generativity mechanism is genuinely more productive around nouns.

Inspection of the number and types of links that are created by the models provides some insight into why the frame based generativity mechanism is more productive around nouns than verbs. Anne's model in the second developmental stage has encoded 649 verbs and 733 nouns. The frame based mechanism has created 308 noun-noun links and 18 verb-verb links. The mechanism that links items on the basis of independent preceding and following context creates 124 noun-noun links and 80 verb-verb links. The frame-based generativity mechanism thus creates more noun-noun links, and fewer verb-verb links. Inspection of the verbs that get linked also reveals that the frame based mechanism does not simply link fewer verbs: it links different verbs, in particular verbs with a lower average frequency. The average frequency in the input corpus for verbs linked on the basis of frames is 8.89. For the independent contexts mechanism the equivalent number is 35.54. The frame-based mechanism thus appears more likely to link low-frequency items than the independent contexts mechanism. A bias towards linking low-frequency items will naturally favour the linking of nouns over verbs, as nouns are, on average, less frequent than verbs.

Some of the reasons why a frame-based mechanism is biased towards linking low-frequency items become apparent when inspecting the frames and independent contexts that the model has encoded for particular words. For the verb 'put' the model has encoded 23 preceding and 24 following contexts. These independent contexts combine to give a total of 57 frames. For the verb 'see' 10 preceding and 20 following contexts and 41 frames have been encoded. The overlap in independent contexts is 33%, yet the overlap in terms of frames is a mere 2%. For the nouns 'table' and 'door', 4 and 8 preceding and 6 and 4 following contexts have been encoded, which give rise to 4 and 5 frames respectively. The overlap in these frames is 29%, the overlap in independent contexts is 17%. It thus appears to be the case that, for frequent items, many (varied) contexts are encoded, which can potentially combine to give many different frames. Infrequent items occur in a small number of potentially more typical contexts, which do not combine to give a large number of frames. This results in the overlap in terms of frames being higher than for independent contexts for infrequent items. For frequent items, the overlap in terms of frames tends to be lower than for independent contexts.

In order to establish if this pattern holds more generally, we divided the words encoded in MOSAIC into a low, medium and high frequency group[1], and calculated the average number of preceding and following contexts as well as frames. As can be seen in Table 3, the ratio of the number of frames over the (average of) the independent contexts increases linearly with frequency: approximately 1, 1.5 and 2. While this increase is not surprising in itself (as the maximum number of frames for any word is the product of the number of preceding and following contexts), it does explain why a frame-based mechanism favours the linking of infrequent items. While, intuitively, frames are more constraining than independent contexts, they appear to be especially constraining for frequent items that appear in

[1] Frequency was measured as the number of times the node encoding a word was traversed when processing the input. Low, medium and high frequency words were defined as having a frequency count between 20 and 500, between 500 and 1000, and over 1000 respectively.

many contexts. While the high-low frequency distinction does not cut across verbs and nouns, verbs (in particular, regular present tense verbs) are, on average, more frequent, and occur in more frames. In fact, the average frequency of the main verbs[2] encoded in MOSAIC is twice that of the nouns encoded in MOSAIC. Verbs occur, on average, in 5.3 frames, compared to 3.69 for nouns. The frames in which verbs occur appear to be more varied as well. The total number of verb frames encoded in Anne's model is 1,966, which comprises 1,229 unique frames. The total number of noun frames is 2,679, which comprises 864 unique frames. Thus, on average, every unique verb frame occurs 1.6 times. Every unique noun frame occurs 3.1 times. The decreased generativity around verbs for a frame-based substitution mechanism therefore appears to be (at least partially) caused by verbs occurring in more and more varied frames compared to independent contexts.

Table 3: Average number of preceding, following contexts and frames for words of low, medium and high frequency.

| Freq. | Number of words | Preceding contexts | Following contexts | frames |
|---|---|---|---|---|
| Low | 2299 | 2.32 | 2.09 | 2.35 |
| Medium | 217 | 8.10 | 6.35 | 10.1 |
| High | 374 | 22.80 | 28.67 | 50.06 |

## Introducing utterance-boundaries

While the frame-based generativity mechanism is successful in simulating the noun-verb asymmetry, it is also apparent that the model is still not very generative (particularly in the early stages). Several reasons can be put forward for why this is the case. First, the overlap threshold of 25% may be too high. Additional simulations with an overlap parameter of 10% showed that generativity is increased, but only for the later stages. Thus, even at 10% overlap Becky's early model produces 1% novel utterances while Anne's model produces 7% novel utterances. The reason why the early models remain less generative is that they actually encode relatively few frames. This is because a (lexical) frame is actually relatively long: 3 words. Particularly in the earlier developmental phases, MOSAIC encodes relatively short utterances. It is therefore unlikely that many phrases of three words are encoded. As a result, few frames are available. One possible way to increase the number of frames used for the decision to link two items is to include the frames that contain sentence boundaries. Frames that include sentence boundaries are quite frequent and are a potentially useful source of information.

The high frequency of frames that contain sentence boundaries is illustrated by an analysis of the frames encoded in Anne's model in the earliest developmental phase. This model has encoded a total of 4,293 words. For these words a total number of 23,244 frames have been encoded. The number of frames that contain only lexical items (i.e. no sentence boundaries) is only 500. Thus, lexical frames make up approximately 2% of the frames encoded in the model's early stages. For Becky's early model lexical frames make up approximately 3% of all the frames. Given the high frequency of frames that contain sentence boundaries, it appears unlikely that children would not be sensitive to such frames. Indeed, when analyzing the frames that occur in the child directed speech for Anne and Becky, it becomes apparent that the 50 most frequent frames all contain a sentence boundary. What's more, the 2 most frequent frames ('The - END', 'A - END') are highly informative frames that each contain around 40 nouns.

Initial analyses with substitution based on all frames were aimed at establishing a suitable value for the overlap parameter. It became apparent that even at relatively high levels of overlap the model quickly became very generative and no longer showed a linear increase in the proportion of novel utterances. Instead, the model had relatively high levels of generativity at early stages of development. These values peaked at intermediate levels of development to subsequently decrease. This is not a characteristic of child speech, as children tend to become more productive with increasing MLU. It also became apparent that generativity around verbs became almost non-existent. Analysis of the developmental changes to the frames encoded in the model revealed that, over development, a larger proportion of the frames becomes lexical. Since lexical frames are more constraining than frames that contain a sentence boundary, it becomes less likely that two words have occurred in that particular frame. For instance, two nouns are more likely to share the frame 'THE - END' than the frame 'THE - KICKS'. Thus, the overlap between two items tends to decrease as the number of lexical frames that these items have occurred in increases. This effect is more pronounced for verbs, as they tend to occur in more varied (lexical) contexts. The increase in lexical frames over the three developmental stages that were simulated is quite considerable. For both models the proportion of lexical frames is around 3% at the first MLU point, 15% at the second MLU point and 32% at the last MLU point.

In order to control for this increasing 'informativeness' of the frames (and obtain a more linear development of the model's ability to generate novel utterances), it was decided to weight the lexical content of frames when calculating the overlap. In the previous simulations the overlap between two words was calculated as the number of overlapping frames divided by the union of the frames that either word has occurred in. For the weighted calculations, a lexical frame contributed 4 to the numerator, while a non-lexical frame contributed 1.

Table 4 gives the results of an analysis of MOSAIC's output when including sentence-boundaries in frames. For these simulations the overlap threshold was set to 50% as the new definition of a frame is less restrictive than before. As can be seen in Table 4, the model now shows reasonable levels of generativity even at early stages of development. The model also shows a clear asymmetry between noun and verb substitutions. This asymmetry becomes less

[2] Excluding progressives and (regular) past tense.

pronounced during the later stages of development as the relative generativity around verbs increases.

Table 4: Descriptive statistics for MOSAIC's output using frame-based substitution with utterance boundaries.

| Child | MLU | Proportion novel | Noun-subs | Verb-subs |
|---|---|---|---|---|
| Anne | 2.01 | .41 | 2232 | 14 |
| | 3.38 | .48 | 15993 | 762 |
| | 4.07 | .58 | 36000 | 3807 |
| Becky | 2.11 | .26 | 609 | 5 |
| | 3.26 | .45 | 5082 | 678 |
| | 4.12 | .55 | 12799 | 3435 |

## Conclusions

This paper set out to establish the relative merits of using frames or independent contexts as the basis for a substitution mechanism in the simulation of child speech. Particular emphasis was placed on the model's ability to simulate the verb-noun asymmetry that is apparent in child speech whilst incorporating the constraints that derive from simulating children's increasing average utterance length. The analyses reported here show that a generativity mechanism that uses independent contexts is biased towards substituting high frequency items, while a mechanism based on frames favours the substitution of lower frequency items that feature in less varied contexts. Since nouns tend to fit the latter category, and verbs fit the former category the frame based mechanism provides a better fit to the noun-verb asymmetry. It was furthermore shown that, while a frame-based generativity mechanism provides a better fit, lexical frames do not occur in meaningful numbers in a model that has only encoded short utterances. This results in low levels of generativity in early stages of development.

The inclusion of utterance boundaries in frames drastically increases the number of frames that are available to the model whilst at the same time including some frames that are potentially very informative for the formation of a noun class. The inclusion of utterance boundaries therefore results in increased generativity around nouns, particularly during the early stages of development. This increased generativity comes at a price however, as it decreases the generativity around verbs when a simple overlap threshold is used. One possible solution to this is to weight the overlap for the lexical content of frames. This results in a model which shows relatively high levels of generativity throughout development without compromising generativity around verbs.

On a more general level, the analyses reported in this paper show that the simulation of child data through the production of actual utterances and the inclusion of a developmental component highlights the fact that relatively subtle differences in the implementation of a generativity mechanism can have rather profound effects on the type of generativity that a model displays. Thus, while intuitively frames are more constraining than independent contexts, the analyses reported here show that they are particularly constraining for frequent items that appear in varied contexts. The use of frames therefore decreases generativity around verbs, while increasing generativity around nouns. Such effects may have quite profound implications for a model's ability to simulate the child data. They are, however, likely to remain hidden in approaches that simply assess the quality of derived grammatical classes without producing actual utterances.

The inclusion of a developmental component furthermore highlights the fact that, while frame-based substitution does a better job of capturing the noun-verb asymmetry, there are only a small number of lexical frames available to a system that encodes and produces short utterances. Lexical frames may therefore be of limited utility to children in early stages of development. Such effects are likely to remain hidden in approaches that track statistics across all of the input, and may result in overestimating the importance of lexical frames at the expense of far more frequent frames that include utterance boundaries.

## Acknowledgements

This research was funded by the Economic and Social Research Council under grant number RES000230211.

### References

Akhtar, N., & Tomasello, M. (1997). Young children's productivity with word order and verb morphology. Developmental Psychology, 33, 952-965.

Finch, S. & Chater, N. (1994). Distributional bootstrapping: From word class to proto-sentence. In A. Ram and K. Eiselt (Eds.). Proceedings of the 16th Annual Conference of the Cognitive Science Society (pp. 301-306). Hillsdale, NJ: Erlbaum.

Fisher, C. (2002). The role of abstract syntactic knowledge in language acquisition: a reply to Tomasello (2000). *Cognition, 82*, 259-278.

Freudenthal, D., Pine, J.M. & Gobet, F. (2006). Modelling the development of children's use of optional infinitives in English and Dutch using MOSAIC. *Cognitive Science*, *30*, *277-310*.

Freudenthal, D., Pine, J.M. & Gobet, F. (2005a). On the resolution of ambiguities in the extraction of syntactic categories through chunking. *Cognitive Systems Research*, *6*, *17-25.*

Freudenthal, D. Pine, J.M. & Gobet, F. (2005b). Simulating the cross-linguistic development of Optional Infinitive errors in MOSAIC In B.G. Bara, L. Barsalou & M. Bucciarelli (Eds.), *Proceedings of the 27th Annual Conference of the Cognitive Science Society.* Mahwah NJ: LEA.

Mintz, T. (2003). Frequent frames as a cue for grammatical categories in child directed speech. *Cognition, 90,* 91-117.

Redington, M., Chater, N. & Finch, S. (1998). Distributional information: A powerful cue for acquiring syntactic categories. *Cognitive Science, 22*, 425-469.

Tomasello, M. (2000). Do young children have adult syntactic competence? *Cognition, 74,* 209-253.

# Structural Transfer of Cognitive Skills

**Dongkyu Choi (dongkyuc@stanford.edu)**
**Tolga Könik (konik@stanford.edu)**
**Negin Nejati (negin@stanford.edu)**
**Chunki Park (chunki.park@stanford.edu)**
**Pat Langley (langley@csli.stanford.edu)**

Computational Learning Laboratory
Center for the Study of Language and Information
Stanford University, Stanford, CA 94305 USA

## Abstract

This paper investigates a computational approach to transfer: the ability to use previously learned knowledge on related but distinct tasks. We study transfer in the context of an agent architecture, Icarus, and we claim that many forms of transfer follow automatically from its use of structured concepts and skills. We show that Icarus can acquire structured representations from domain experience, and subsequently transfer that knowledge into new tasks. We present results from multiple experiments in the Urban Combat Testbed, a simulated, real-time, three-dimensional environment with realistic dynamics.

## Introduction

Many computational learning methods require far more training instances than humans to achieve reasonable performance in a domain. A key reason is that humans often reuse knowledge gained in early settings to aid learning in ones they encounter later. This phenomenon is known as *transfer* in cognitive psychology, where it has received far more attention than in AI and machine learning. Much of this research has focused on transfer of complex skills for tasks that involve action over time (e.g., Kieras & Bovair, 1986; Singley & Anderson, 1988). In this view, transfer primarily involves the reuse of cognitive structures, where the amount of shared structure has proved to be a good predictor for the degree of transfer in humans.

This paper reports on a computational approach to transfer that takes a similar perspective. We focus on the acquisition of cognitive skills from experience and on how transfer improves behavior on distinct but related tasks. We share with many psychologists the idea that transfer is mainly a structural phenomenon, rather than a consequence of statistical summaries or value functions. This suggests that transfer is linked closely to how an agent represents knowledge in memory, how its performance methods use these structures, and how its learning elements acquire this knowledge.

Theoretical commitments to representation, performance, and learning are often associated with the notion of a cognitive architecture (Newell, 1990). Thus, it seemed natural for us to study transfer in the context of Icarus (Langley & Choi, 2006), an architecture that takes positions on each of these issues. We will maintain that Icarus' commitment to relational, hierarchical, and composable knowledge structures, and to mechanisms for using and acquiring them, provide it with basic support for effective transfer. Moreover, we make the more radical claim that the architecture needs no additional mechanisms to exhibit many forms of transfer. We hold that most transfer requires no special processes beyond those already needed for other purposes.

We elaborate on these ideas in the sections that follow. First we present a virtual gaming environment that illustrates the benefits of reusing learned knowledge structures. After this, we review Icarus' assumptions about representation, performance, and learning, along with the ways in which they support transfer. Next we evaluate our claims through results of specific studies with simulated physical agents. We conclude by reviewing related efforts on structural transfer and stating our priorities for future research on this topic.

## An Example Domain

In this paper, we examine transfer between source and target problems within a single domain. We have chosen to phrase these problems in the Urban Combat Testbed[1] (UCT), a virtual 3-D environment that simulates an urban landscape, with real-time behavior and realistic dynamics. UCT contains one intelligent agent (controlled by Icarus) and, at the moment, no adversaries. Our transfer tasks focus on navigation in the presence of physical and conceptual obstacles.

Figure 1 illustrates one such transfer task. The source problem calls on the agent to find a goal and surmount obstacles encountered en route (here, to duck under and climb over obstacles it has never seen). The target problem offers the agent the opportunity to reuse its knowledge about obstacles in a different order, assuming it is acquired and represented in a modular form. In addition, the agent can reuse learned knowledge about the map. The agent exhibits (positive) transfer if it improves its behavior in the target as a result of its exposure to the source, and zero or negative transfer if it does not.

We supply the Icarus agent with minimal background to support transfer. On initialization, it has never encountered the specific objects or operators in the domain, and it has no prior knowledge of the map. However, it is initialized with useful concepts, such as a category for obstacles in general, a relation for blocked paths, plus categories for region centers and gateways (the UCT environment is divided into convex regions with passable

[1] http://gameairesearch.uta.edu/UrbanCombatTestbed.html

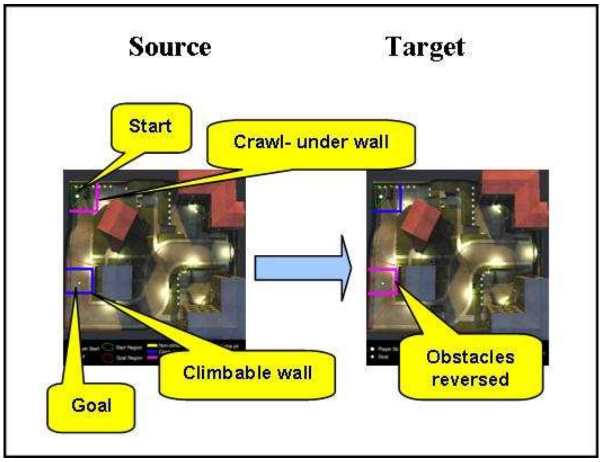


Figure 1: A transfer task in Urban Combat Testbed.

and non-passable boundaries). The agent understands the high-level goal (e.g., to find an item), and it possesses subgoals that organize search behavior. For example, it knows to overcome an obstacle in order to get a clear view of the destination, and to contain exploration within the region of the goal, once seen.

UCT is a challenging domain for both human and artificial agents. It is partially observable because the agent can only perceive nearby objects and regions, it involves uncertain action (e.g., the agent can attempt to jump over a wall but fall backwards into a ditch), and it is real time (imposing a strong constraint on agent decision making). This complexity demands a level of robustness in the mechanisms that produce transfer.

## Transfer in Icarus

Icarus achieves transfer using a hierarchical and relational representation, which encodes knowledge in a general and composable way, a goal-driven and reactive execution mechanism, which allows flexible execution of the learned knowledge structures, and a relational learning mechanism, which acquires general knowledge from observed solutions as well as background knowledge. We will discuss each of these elements and describe in turn how they contribute to transfer.

### Representation of Concepts and Skills

The Icarus architecture makes several commitments in its representation of knowledge. First, it supports two different types of knowledge; concepts and skills. Concepts describe state, while skills are methods an agent can execute in the world under certain conditions. Both have a hierarchical structure, meaning that Icarus can employ multiple layers of abstraction in describing the current state and the procedures for manipulating that state, respectively.

As shown in Table 1, concepts in Icarus resemble traditional Horn clauses in first-order logic with negations. Primitive concepts like *in-region* provide state descriptions at the lowest level of abstraction using symbolic and numeric information directly extracted from objects the agent perceives. Higher-level concepts, such as *stopped-in-region* and *climbable-gateway* have their basis in other concepts as well as primitive facts. The concept hierarchy provides relational, modular descriptions of the current state. It can also be used to represent a desired state, so concepts can express subgoals.

Table 1: Example Icarus concepts.

```
(in-region ?self ?region)
  :percepts  (self ?self region ?region)

(climbable-gateway ?gateway ?object)
  :percepts  (gateway ?gateway) (object ?object)
  :relations ((totally-blocked-gateway ?gateway
                                        ?object)
              (feature-of-object ?object
                                 CLIMBABLE))
```

Table 2: Primitive and non-primitive Icarus skills.

```
(clear ?gateway)
  :percepts ((gateway ?gateway
              dist1 ?dist1 angle1 ?angle1
              dist2 ?dist2 angle2 ?angle2))
  :start    ((close-enough-to-jump-type ?gateway))
  :actions  ((*jump-cmd (maximum ?dist1 ?dist2))
             (mid-direction ?angle1 ?angle2))

(crossable-region ?regionB)
  :percepts ((self ?self) (region ?regionB))
  :start    ((connected-region ?regionB ?gateway))
  :subgoals ((clear ?gateway))

(in-region-able ?me ?regionA ?regionB)
  :percepts ((self ?me)
             (region ?regionA)
             (region ?regionB))
  :start    ((in-region ?me ?regionA))
  :subgoals ((crossable-region ?regionB))

(in-region me region3004)
  :subgoals ((in-region-able me region3003
                                region3004)
             (in-region me region3004))
```

Taken together, Icarus skills for a given domain are a specialized form of hierarchical task networks (Nau et al., 1999). A skill's head indexes it by the goals it achieves, and since goals are naturally represented by desired concept instances, skills are tied in to the concept hierarchy. Some achieve low-level concepts, while others address broad objectives. Table 2 shows some examples of skills in Icarus. While primitive skills give simple methods using basic actions executable in the world, higher-level, non-primitive skills describe complex methods with multiple ordered subgoals. Since non-primitive skills specify subgoals, not the details of how these goals are achieved, an Icarus agent can select the most relevant method for the given subgoal depending on the current situation.

Icarus' relational and hierarchically composable representation of skills is crucial to its ability to transfer knowledge. In particular, the relational representation increases generality of the encoded skills, since they can apply in circumstances that are only qualitatively similar

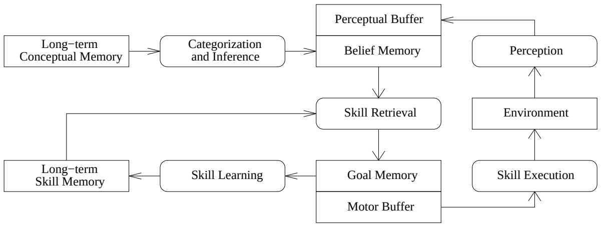


Figure 2: A schematic of memories and processes in the Icarus architecture.

to the situations where the skills are acquired.

Moreover, a hierarchically composable representation lets skills apply in new circumstances, even if they are partially incorrect, inaccurate, or inapplicable. In these cases, the undesired subskills will be either relearned from new experience or dynamically replaced during execution with other subskills. For example, Icarus can use a learned alternative that achieves the same goal, instead of an inapplicable subskill. We discuss how the architecture uses and learns hierarchically composable skills in the following sections.

## Execution of Hierarchical Skills

The Icarus architecture operates in cognitive cycles, spanning conceptual inference, skill selection, and physical execution (Figure 2). Icarus derives its beliefs via a bottom-up matching process, initiated by objects that arrive in the agent's percepts. After it infers low-level concept instances based on these objects, inference for higher-level concepts follows to build a hierarchically organized belief structure for the time step. In contrast, Icarus performs skill selection in a top-down manner, starting with the current goal. On each cycle, it finds a path from this goal through the skill hierarchy; each skill along this path is applicable given the current beliefs, with the terminal node being a primitive skill that Icarus executes in the environment. This differs from traditional production systems, which require multiple cycles and use of working memory to traverse levels of an implicit goal hierarchy.

Since the architecture repeats this procedure on each cycle, Icarus agents can react to their surroundings while pursuing goal-driven behaviors. Also, they can use new knowledge structures immediately, incorporating them for processing on the next execution cycle. Given a choice between two skills, Icarus will prefer the one more recently learned, so agents can start with some knowledge but behave more intelligently as they acquire experience. The spatial search in UCT is a good example of this feature. Initially, the agent uses basic exploration strategy to search the environment. As it explores and discovers the geography of its world, it begins to employ the new map knowledge to guide further search towards the goal.

Icarus' execution mechanism facilitates transfer by flexibly employing previously learned skills in three ways. First, it transfers learned skills to new problems by dynamically selecting and interleaving learned skills based on observed situations and achieved goals. Second, it combines skills learned from qualitatively different experiences when a novel situation has elements from these previous experiences. Finally, even if Icarus does not have sufficient knowledge to directly solve a problem, it can transfer partially applicable skills from previous solutions and patch the knowledge gap by falling back on its default search skills.

Icarus can achieve qualitatively different types of transfer by reusing high-level or low-level skills. For example, if a source and a target problem share abstract goals, it solves the target faster by transferring high-level skills. This occurs in UCT when the agent has learned how to clear a set of obstacles that block a goal. The agent uses the strategy acquired in the source (i.e., to approach the closest blocking object and overcome it) to tackle a different set of obstacles in the target, although the details depend upon the type of the obstacles and their relative configuration. On the other hand, Icarus also transfers low-level skills by composing them in novel ways to solve the target problem. For example, if Icarus learns to overcome a climbable wall in the source, it transfers that skill when it encounters a climbable wall in service of an unfamiliar route planning task, where the high-level goal might differ from that in the source problem.

One important requirement in transfer is to use previously acquired skills that are not completely correct or always applicable. This commonly occurs because conditions in the environment differ between source and target problems. For example, suppose the agent learns skills for navigating to the goal location in a UCT source problem. It partially executes those skills in the target, then abandons them in favor of exploration when it encounters a non-surmountable obstacle blocking the path. The agent reenters the skills when exploration brings it to some later step along its original path. This type of transfer results from Icarus' reactive execution module, which uses only the relevant skills in the current environment.

## Learning Hierarchical Skills

Icarus acquires structured skills via an analytical learning mechanism that it invokes whenever the agent achieves a top-level goal. This mechanism inputs the goal plus a solution trace that achieves it, described as a sequence of observed states and selected actions. Icarus generates an explanation of how the goal was achieved by interpreting this solution traces in the context of conceptual knowledge and action models. It does so by recursively examining goals, either decomposing them into subgoals using conceptual knowledge or explaining them in terms of the effects of primitive skills. The architecture converts the resulting explanation structure into hierarchical skills and add them to its skill memory. We have described this process elsewhere (Nejati et al., 2006) in more detail.

One distinctive property of this method is that it learns the structure of a skill hierarchy as well as the goals and conditions associated with each skill. This method is related to previous work on explanation-based learning (Mitchell et al., 1986), but differs in that it does not compile away the explanation structure, but rather uses it to determine the structure of the skill hierarchy.

This learning mechanism facilitates transfer by associating a hierarchy of learned skills with the goals they achieve. As a result, the component skills can be used independent of the top-level goal that motivated their construction. For example, an Icarus agent tasked to enter a building may find a solution where it jumps over a fence and then enters the building, viewed as sequence of primitive skills. The analytical learner creates a new skill to climb over a fence, as well as a higher-level skill that uses it along with primitives to reach the goal from the start location. The system associates the low-level skill (for fence climbing) with the goal for reaching a parameterized location, and it considers the component whenever a fence blocks a local goal.

The learning system also facilitates transfer by using relational background knowledge. It acquires skills that reference the agent's conceptual vocabulary, which provides flexibility in retrieving the skills. For example, an Icarus agent in UCT may acquire a skill for reaching a goal it recognizes as collectively blocked (a built-in concept that matches when multiple objects impede a path). As long as the concept is true in a situation, the agent can apply the resulting skill regardless of the actual configuration of obstacles.

## Evaluating Icarus' Account of Transfer

The basic claim in this paper is that Icarus' assumptions about representation, performance, and learning support transfer without need for any additional mechanisms. Moreover, both learning and transfer should occur at roughly the levels observed in humans, although we will not compare the architecture's behavior directly to the results of psychological studies here. We have already explained the ways in which Icarus should produce such transfer, but this is different from demonstrating such effects.

To this end, we designed and ran a controlled experiment in the Urban Combat testbed. Our dependent variable was the time taken to solve a target problem that involved achieving a physical goal in the domain, and we studied the effects of two independent factors. One concerned whether the agent had experience solving five source problems (the transfer condition) or had no such experience (the nontransfer condition). The other involved the relationship between the source and target problems, which we discuss at more length below.

### Source-Target Relationships

Analysis suggested a number of relationships between source and target problems that should support transfer of learned knowledge, six of which we focus on here. We consider two of these forms in detail and then briefly summarize the other four.

One source-target relationship, *abstracting*, involves sharing hierarchical solution structure, as the UCT scenarios in Figure 3 illustrates. Here the source and target problems involve the same start and goal locations, but the target requires solving a different subproblem to enter the chosen building. The problems can take advantage of the same route knowledge for navigating from start to goal, as well as the high-level solution structure composed of an initial route segment, an entry task, and a final approach. Other source-target pairs of this type require the agent to break into a building via different means or clear unfamiliar obstacles from its path. Problem pairs share the start and goal locations but exercise largely distinct sections of the UCT map.

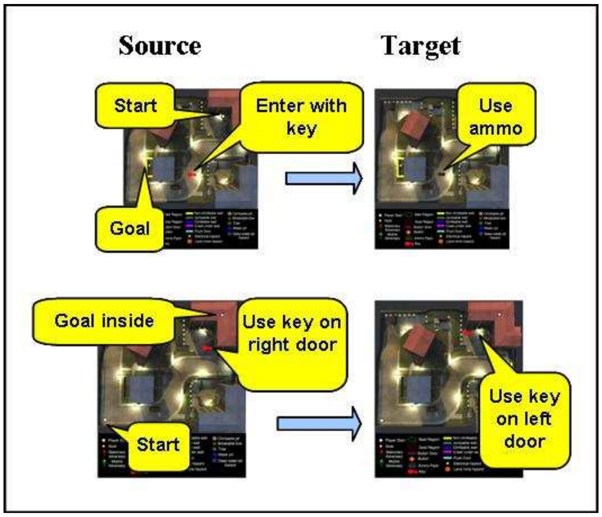


Figure 3: Two source-target pairs for abstracting.

We expect Icarus to transfer the overall decomposition of a source problem's solution into the corresponding target. This capability also lets the agent reuse route information between non-corresponding sources and targets, as well as between target problems, to the extent it explores overlapping terrain while solving tasks. However, the system cannot share solutions for new subproblems introduced into target tasks, since they do not reoccur. Those component solutions (e.g., using ammunition to enter the building vs. a key) must be discovered in each target task, so they act to increase solution time and decrease performance.

Another type of relationship between source and target problems, *restructuring*, illustrated in Figure 1, requires the agent to use solutions to subproblems in different orders. Successful transfer lets the agent solve the target problem more rapidly because it has learned how to duck under a wall and climb over a wall in the source problem, independently of when those subtasks arise in the target. Other source-target pairs of this type – surrounding the start and end states with jumpable vs. unclimbable walls, boxes vs. pits, button operated vs. push doors, and water vs. electrical hazards – follow the same pattern.

We also examined four additional relationships between source and target problems:

- reuse partial solutions from a common start state in source and target problems, despite differing goals (*extrapolating*);

- repeatedly reuse solutions to subproblems from the source problem when working on the target problem (*extending*);
- dynamically compose solutions to problems from source problems to solve a more complex target problem (*composing*); and
- reuse the skills learned on the source task to solve a target problem, but apply different operators to novel objects that occur in the target (*generalizing*).

Our experiment examined Icarus' ability to transfer learned skills from source to target problems that involved each of these six relationships. We felt that, if transfer occurred, it would provide evidence for the generality of the architecture's mechanisms.

### Experimental Design and Results

Our experimental method involves presenting the Icarus agent with a collection of source-target problem pairs. Each pair provides a known opportunity for transfer, while the set (called a scenario) supports some cross talk: a single source problem can enable transfer into multiple targets and the set of source problems supports transfer into any target.

As noted above, we ran the agent in both a transfer condition, in which it first solved a set of five source problems and then solved five target tasks, and a non-transfer condition, in which it solved only the five target problems. We ran the agent on six different sets of source-target pairs that reflected the relationships discussed above. We randomly varied the presentation order of the target problems to guard against effects of training order.

Figure 4 summarizes the results of the experiment by plotting the problem solution time for the nontransfer condition against the time for the transfer condition. Each x and y value represents the average score over 20 runs of the Icarus agent, with different icons depicting distinct forms of source-target relationship. Entries above the diagonal line indicate that positive transfer occurred, while entries below the line reflect negative transfer. The figure shows that Icarus generally exhibits positive transfer for most problems in each type of relationship. Moreover, this transfer occurs after experience with only five source problems, meaning that the rate of learning is roughly comparable to that observed in humans.

The key point is that Icarus produces this transfer without any mechanisms above those required to draw inferences, execute skills that are indexed by goals, and acquire those skills from problem solutions. It does not require any additional processes to explain transfer across a variety of different source-target relationships. Our experimental study generally supports this claim about the emergent nature of transfer effects within the Icarus architecture.

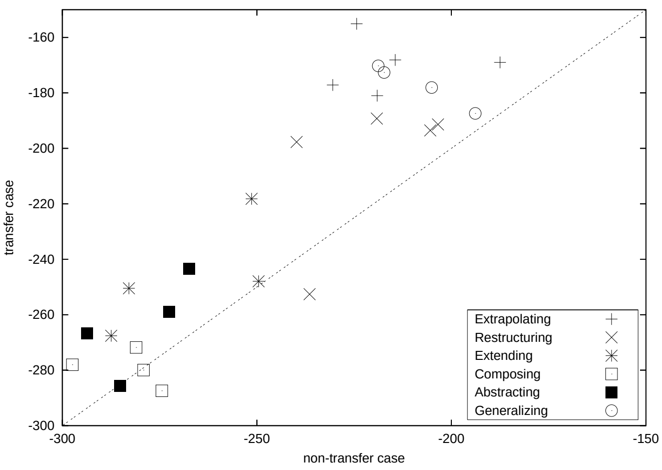


Figure 4: Solution times in seconds for transfer case plotted against those for non-transfer cases. Note that the axes are inverted so that higher scores indicate better performance.

## Related Research

Researchers in psychology have shown considerable interest in transfer, but their research has emphasized experimental studies. There have been a few computational models of this phenomenon, Kieras and Bovair (1986) and Singley and Anderson (1988) being two examples. Their models provided very accurate predictions for the degree of transfer in terms of the number of shared knowledge elements, but these were not coupled with learning mechanisms that could acquire the knowledge.

In contrast, there have been several efforts on transfer in the machine learning literature. For example, Swarup and Ray (2006) discuss transfer in the context of neural networks, and Thrun (1996) considers the case in which an agent experiences many variations of a general task. However, these systems showed much slower learning than observed in humans, and made little contact with the psychological literature.

The approach we have described in this paper aims to model reuse of knowledge structure in the context of a cognitive architecture that incorporates psychologically plausible representations and mechanisms. There have been some previous results in the same spirit. For example, Langley (1985) investigated methods for learning search-control heuristics through discrimination of production rules, and Laird et al. (1986) demonstrated learning and transfer of macro-operators through chunking in Soar. More recently, Hinrichs and Forbus (2007) have discussed the transfer of planning and strategic knowledge among subproblems in a turn-based strategy game.

## Directions for Future Work

Although our results to date have been encouraging, there clearly remains room for improving Icarus' ability to transfer its learned knowledge. One avenue involves supporting more flexible execution of skills. Currently, the system executes skills in the environment as soon as they are applicable, but since skills acquired in one setting can propose undesirable actions in another, they may lead to negative transfer. We are planning to add a module for lookahead search, constrained by the skill hierarchy, that would guard against this problem and thus improve transfer.

We are also investigating a method for making both inference and execution more flexible. This involves replacing ICARUS' current deductive inference module with one that relies on Markov logic (Richardson & Domingos, 2006), which combines first-order logical and probabilistic reasoning. This approach uses weights to soften the otherwise hard rules of a first-order knowledge base. Possible worlds that violate rules become more or less probable depending on evidence and the magnitude of weights. The result is an inference mechanism that is robust to error and uncertainty. This will let ICARUS transfer its skills more flexibly, in that it can select skills even when start conditions are likely but not deductively implied, as can happen in partially observable settings.

In this paper, we focused on forms of transfer that ICARUS can handle using its existing architectural mechanisms. But one implicit assumption of this approach is that the agent can use the same relational predicates to describe the source and target problems. This approach will not succeed in situations where the source and target problems have similar structure but have been encoded with different symbols. We aim to address this challenge by developing analytic methods that infer mappings between the representations used in the source and the target problem.

## Concluding Remarks

Although structural transfer is an important phenomenon in human learning, there are few computational models that combine learning with transfer. In this paper, we described an agent architecture that can transfer skills learned in one setting to distinct but related tasks. We showed that the framework demonstrates this capability for a number of different relations between source and target problems, and we reported experimental results on a challenging testbed.

One of our key claims was that ICARUS can achieve transfer without requiring any mechanisms beyond those needed to represent, execute, and learn skills. Our experiments with UCT supported this claim and suggested that many types of transfer arise naturally from methods that can acquire relational, hierarchical structures. We analyzed ICARUS' ability to transfer and explained how its architectural commitments support this process. In the future, we hope to model additional forms of transfer that involve more complex relations between source and target problems.

## Acknowledgments

The authors would like to thank Larry Holder and Michael Youngblood for the development and support for the Urban Combat Testbed, and John Laird and Nicholas Gorski for insightful discussions on cognitive architectures and agent development. Also, we would like to thank Dan Shapiro and Tom Fawcett for their contributions to scenario development and project evaluations, and Cynthia Matuszek and Blake Shepard for their work on domain refinement. This paper reports research sponsored by DARPA under agreement FA8750-05-2-0283. The U. S. Government may reproduce and distribute reprints for Governmental purposes notwithstanding any copyrights. The authors' views and conclusions should not be interpreted as representing official policies or endorsements, expressed or implied, of DARPA or the Government.

## References

Hinrichs, T. R., & Forbus, K. D. (2007). Analogical learning in a turn-based strategy game. *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*. Hyderabad, India.

Kieras, D. E., & Bovair, S. (1986). The acquisition of procedures from text: A production-system analysis of transfer of training. *Journal of Memory and Language, 25*, 507–524.

Laird, J., Rosenbloom, P. & Newell, A. (1986). Chunking in SOAR: The anatomy of a general learning mechanism. *Machine Learning, 1*, 11–46.

Langley P. (1985). Learning to search: From weak methods to domain-specific heuristics. *Cognitive Science, 9*, 217–260.

Langley, P., & Choi, D. (2006). Learning recursive control programs from problem solving. *Journal of Machine Learning Research, 7*, 493–518.

Mitchell, T. M., Keller, R. M., & Kedar-Cabelli, S. T. (1986). Explanation-based generalization: A unifying view. *Machine Learning, 1*, 47–80.

Nau, D., Cao, Y., Lotem, A., & Muñoz-Avila, H. (1999). SHOP: Simple hierarchical ordered planner. *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence* (pp. 968–973). Stockholm: Morgan Kaufmann.

Nejati, N., Langley, P., & Könik, T. (2006). Learning hierarchical task networks by observation. *Proceedings of the Twenty-third International Conference on Machine Learning* (pp. 665–672). New York: ACM Press.

Newell, A. (1990). *Unified theories of cognition.* Cambridge, MA: Harvard University Press.

Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine Learning, 62*, 107–136.

Singley, M. K., & Anderson, J. R. (1987). A keystroke analysis of learning and transfer in text editing. *Human-Computer Interaction, 3*, 223–274.

Swarup, S., & Ray, S. (2006). Cross-domain knowledge transfer using structured representations. *Proceedings of the Twenty-First National Conference on Artificial Intelligence*. Boston, MA: AAAI Press.

Thrun, S. (1996). *Explanation-based neural network learning: A lifelong learning approach.* Boston, MA: Kluwer Academic Publishers.

# Dialing while Driving? A Bounded Rational Analysis of Concurrent Multi-task Behavior

**Duncan P. Brumby (Brumby@cs.drexel.edu)**
**Dario D. Salvucci (Salvucci@cs.drexel.edu)**
Department of Computer Science, Drexel University
Philadelphia, PA 19104 USA

**Andrew Howes (HowesA@manchester.ac.uk)**
Manchester Business School, University of Manchester,
Manchester, M15 6PB UK

## Abstract

When people conduct multiple tasks in tandem, such as dialing a cell phone while driving a car, they often interleave the two tasks, for instance by returning attention to the primary driving task after entering bursts of three of four digits at a time. In order to explain why people tend to interleave these tasks at this particular interval, a control model of steering behavior is described that focuses on understanding how environmental and psychological constraints interact to determine driver performance. We use this model to predict the amount of time that people are prepared to stray from the driving task while engaging in a secondary in-car task and, by consequence, the degree of task interleaving. In particular, a modeling experiment was conducted to determine the consequences of systematically varying the time interval between consecutive steering updates for driving performance. The results of this analysis were then used to demonstrate why returning attention to driving after entering bursts of three of four digits at a time is a particularly efficient strategy: It does not allow driving performance to become too egregious, while at the same time keeping the additional time costs that are incurred as a result of interleaving tasks minimal.

## Introduction

While you are driving in your car it is not too difficult to sometimes direct your attention away from the road in order to engage in a secondary task, such as dialing a number on a cell phone. In this complex real-world multitasking scenario, people tend to interleave the two tasks by returning attention to driving after entering bursts of three of four digits at a time (e.g., Salvucci, 2005). One potential explanation for why people choose to interleave these tasks at this particular interval is that the representational structure of the telephone number (e.g., for a 7-digit telephone number this might follow a *xxx-xxxx* structure) provides a series of natural break points at which to return attention to driving. Although this account has intuitive appeal, it is not entirely obvious that people necessarily have to return attention to steering control after dialing three or four digits. Why not more or less digits at a time? Alternatively, if someone were engaged in some other secondary in-car task that, for instance, demanded longer interaction episodes than dialing (e.g., scrolling through a long list of media content on an Apple iPod), would they still make glances back to the road with the same regularity?

In this paper, we present a bounded rational analysis (Howes, Vera, & Lewis, 2007) of concurrent multi-task behavior, in order to better understand how long people should be prepared to look away from the road when engaging in a secondary dialing task while driving. In this analysis we focus on understanding how functional-level features of the task environment (Gray, Neth, & Schoelles, in press) and the constraints imposed by the cognitive architecture (Anderson et al., 2004) interact to make some multitasking behaviors more preferable than others. For instance, it seems rather obvious that deprived of regular attention, driving performance will rapidly fall below criterion, with potentially disastrous consequences. At the same time though the benefits of frequently interleaving play against the costs of switching between tasks (e.g., Allport, Styles, & Hsieh, 1994). In particular, switching between tasks often carries costs associated with the physical realignment of the body relative to external resources and the mental recovery of state information associated with each task. Given this trade-off between the potential costs and benefits of frequently interleaving, how do people decide when to switch back and forth between tasks?

One possible factor that might determine when task interleaving is desirable is the shape of the payoff function for the primary task (Payne, Duggan, & Neth, in press; Son & Sethi, 2006). Payne et al. conducted a series of experiments designed to investigate how people allocate time between two Scrabble tasks. Each task required participants to generate as many words as possible from a fixed set of letters in a given amount of time. Importantly, the tasks differed in the number of words that could be readily generated from their respective letter sets. This meant that the tasks had different payoff functions because the rate at which a participant could find a novel word from a particular set of letters differed between the two tasks. This difference between tasks' payoff functions were not known in advance of the study to the participants, and Payne at al. were interested in how participants would allocate their time between the two tasks. As one might expect, Payne et al. found that participants eventually learned to allocate more time to the more productive task (i.e. the task with the greater payoff function) but most still chose to switch between tasks rather frequently. Payne et al. also found that participant's "giving-up time" (i.e., the time between finding the most recent word and the decision to switch tasks) was longer in the less productive task. Taken together these two effects appear to work against each other (i.e., longer visits to the easier task, but shorter giving-up times for this task), but Payne et al. demonstrate that a stochastic model, based on

Green's (1984) assessment rule of optimal foraging theory, account for these data.

Son and Sethi (2006) present a formal analysis that shows that optimal time allocation between tasks is dependent on characteristics of the environment. Son and Sethi give the example of a learning environment where a learner's time must be allocated between multiple tasks (e.g., consider a student studying for a set of final exams). Son and Sethi demonstrate how time pressure as well as the nature of a task's learning curve can lead to different allocations of time between tasks. Moreover, the work of both Payne et al. and Son and Sethi is of interest here because it points to the potential role of a task's payoff function in determining precisely when people are likely to switch from one task to another.

In this paper, we present a bounded rational analysis (Howes, Vera, & Lewis, 2007) of possible strategic variability in how people might dial a cell phone while driving. Extending earlier work (Brumby, Howes, & Salvucci, 2007; Brumby, Salvucci, Mankowski, & Howes, 2007), a control model of steering behavior is described that focuses on understanding how environmental constraints (e.g., perturbation of the vehicle's heading over time) and psychological constraints (e.g., people's sensitivity to the lateral position of the vehicle in relation to the center of the lane) interact to determine driver performance. A modeling experiment is conducted to determine the consequences of systematically increasing the time interval between consecutive steering updates for the average lateral deviation of the vehicle from the lane center over time. We show that the particular rate that people tend to make glances back to the road while engaging in a dialing task can be understood in the context of the rate of decline in driver performance over time and the costs of switching back and forth between tasks.

## Model of Steering Control

A control model of steering behavior is developed that gives predictions of changes in a simulated vehicle's lateral deviation (i.e., distance from the lane center) over time. The model focuses on understanding how environmental and psychological constraints interact to determine driver performance. The model simulates a vehicle moving at a constant velocity down a straight road. The model performs a series of discrete steering updates that alter the heading (or lateral velocity) of the vehicle dependent on its lateral position in the lane at the time that the steering update is performed. The approach taken is similar to control theoretic accounts of lane keeping (e.g., model 1 in Hildreth et al. 2000), which assume that adjustments to the heading of a vehicle are motivated by the goal of minimizing perceptual input quantities that represent the lateral position and heading of the vehicle.

In order to parameterize the model, driver performance data from two experiments that investigated the effect of cell phone use on driving (Salvucci, 2001; Salvucci & Macuga, 2002) were analyzed to formally characterize how drivers typically adjusted the heading of the vehicle given its lateral position in the roadway. An underlying assumption of this analysis was that adjustments to the heading of the vehicle were motivated by the driver attempting to maintain a central lane position over time. In particular, the experimental software logged, at a rate of once every 30 ms, the normalized steering wheel angle of the simulated car and its divergence from the center of the lane (in meters). This steering data was then segmented into a series of *steering episodes*, which were defined as periods in which the angle of the steering wheel did not alter over time. For each of these steering episodes, a tuple was defined that represented the duration of the episode (*time*), the change in the lateral position of the vehicle (*distance*), and the average lateral velocity of the vehicle (where *lateral velocity* = *distance* / *time*). Data from all steering episodes across participants from the two studies were pooled, and the lateral velocities of all steering episodes that had a common starting lateral deviation (i.e., originated from the same lateral position in the roadway) were averaged. We report an analysis of these average data.

Figure 1 shows a scatter plot of the relationship between the lateral deviation of the vehicle at the start of a steering episode and its average lateral velocity throughout the episode. It can be seen in the figure that as the car strayed closer to the lane boundary, drivers tended to react by making sharper corrective steering movements, which in turn, increased the lateral velocity of the vehicle, returning it to a central lane position more rapidly. Furthermore, it can be seen that for many steering episodes lateral velocity was negative; indicating that the car was heading farther away from the center of the lane.

Regression analysis was conducted to estimate a best fitting curve to predict the average lateral velocity of a steering episode given the lateral deviation (*LD*) of the vehicle at the start of an episode. It was found that a quadratic function[1],

$$\textit{Velocity} = 0.2617 \text{ x } LD^2 + 0.0233 \text{ x } LD - 0.022 \qquad (1)$$

provided a high degree of correspondence with the human data ($r^2 = 0.61$), $F\ (1,80) = 62.61$, $p< .001$. This quadratic model of steering control predicts that as lateral deviation from the lane center increases, there is an increase in the lateral velocity of the vehicle, brought about at discrete steering updates, in order to return the vehicle to a central lane position more rapidly.

Furthermore, the intercept of the curve given by the model (shown in Figure 1) gives some suggestion of the driver's threshold for judging the vehicles deviation from the lane center. In particular, when the car is near the lane center (i.e., lateral deviation < 0.30 m), predicted lateral velocity is close to zero. This means that the position of the car in the roadway remains more or less constant over time. This implies that the driver was possibly satisfied with the vehicle's position in the roadway if the lateral deviation of the vehicle was less than 0.3 m from the lane center.

Although the quadratic model gave a high degree of correspondence with the data, there was also considerable variability with respect to the observed lateral velocities given a particular lateral deviation at the start of an episode. In particular, the standard deviation of the data from the mean

[1] Because of non-positive lateral velocities exponential or power functions could not be applied.

was 0.10 m/s. This suggests that people's adjustments to the heading of the vehicle were stochastic. In order to develop a stochastic model of steering control, random values were sampled from a Gaussian distribution and added to the value of the updated lateral velocity. Based on an estimate of the average standard deviation observed in the human data, the Gaussian distribution had a mean of 0.00 m/s and standard deviation of 0.10 m/s.

An important functional-level feature of the driving environment is that if left unattended, the heading of the vehicle will also be influenced by external factors, such as bumps in the road, wind, the camber of the road, etc. In order to simulate this feature of the driving environment, the heading of the vehicle was perturbed every 50 ms by a random value sampled from a Gaussian noise distribution. Following estimates from a previous model in the literature (Hildreth et al., 2000), the Gaussian noise distribution had a mean 0.00 m/s and standard deviation 0.10 m/s.

In summary, the model provides a computationally efficient formalism for predicting how drivers typically adjust the heading (or lateral velocity) of a vehicle given its lateral position in the roadway. The model focuses on how functional-level features of the task environment (e.g., perturbation of the vehicle's heading over time) and psychological constraints (e.g., people's sensitivity to the lateral position of the vehicle in relation to the center of the lane) interact to determine driver performance. Moreover, it is worth pointing out at this stage that the model does not make any theoretical commitment to the duration of a typical steering update; the model is solely dependent on parameters derived from an analysis of steering performance data and assumptions about the environment. In the next section the model is used to understand how driving performance might decline with increasing periods of driver inattention.

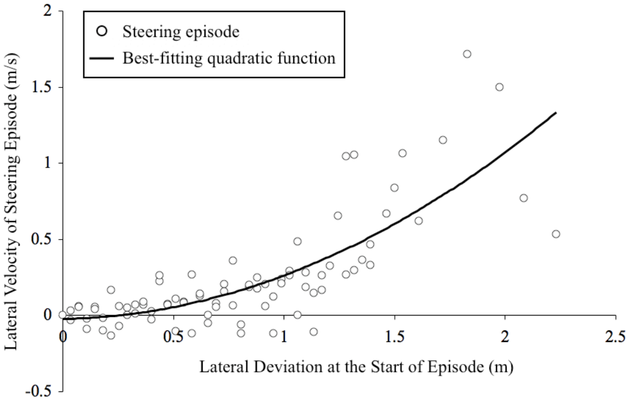


Figure 1: Relationship between lateral deviation at the start of a steering episode and lateral velocity.

## Modeling Experiment

We conducted a modeling experiment that symmetrically varied the time interval between steering updates in order to make quantitative predictions of the consequences for lateral deviation over a period of simulated driving. Specifically, we explored steering strategies that updated lateral velocity at an interval of between 50 ms and 6,000 ms, exploring performance at increasing increments of 50 ms. That is, we evaluated 120 different steering strategies that differed in terms of the duration of time between each steering update over a period of simulated driving. Each steering strategy was run for 1,000 trials, and performance averaged. The vehicle's lateral deviation at the start of each trial was 0.33 m from the lane center. This initial lateral deviation reflects the average value at the beginning of each trial in the empirical data (taken from Salvucci, 2001). Furthermore, each steering strategy was evaluated over both a single steering episode (single-event) and also over a longer measurement interval of 60-seconds simulated driving (long-term). For each steering strategy we report the lateral deviation — the root-mean-square error (RMSE) of the vehicle's lateral distance from lane center — over both a single-event and the long-term measurement interval.

### Results: The effect of interval between steering updates on lateral deviation

In order to illustrate how the movement of the vehicle is affected by the interval between steering updates, Figure 2 shows a data plot representing changes in lateral deviation over time for different illustrative strategies. Performance is for a single trial. Data points represent periods where the lateral velocity of the vehicle was altered owing to a steering update. Changes in the heading of the vehicle between steering updates are due to environmental noise.

Figure 2 offers a comparison between the performance of steering strategies that conducted relatively frequent updates to the lateral velocity of the vehicle (once every 50 ms) to strategies that updated lateral velocity less frequently (once every 600 ms). It is clear from the figure that there was very little difference in performance between these two strategies; in both cases the vehicle maintained a more or less straight heading (i.e., lateral velocity ≈ 0 m/s) and as a result kept to a consistent lateral position in the lane over time. In contrast, as the interval between consecutive steering updates increased even further (to once every 1800 ms), the vehicle tended to drift more erratically about the lane. This was partial because without frequent steering updates, the heading of the vehicle was perturbed by environmental noise. In order to compensate for this general increase in lateral deviation, the model tended to set a heading when a steering update was eventually performed that gave a large lateral velocity. As can be seen in the figure, these aggressive changes in heading lead the vehicle to move rather erratically about the lane.

We next focus on quantifying the rate at which lateral deviation increases with increasing time between updates of steering control. Figure 3 shows the performance of each strategy over a single steering update (single-event) and also over a longer measurement interval of 60-seconds simulated driving (long-term). The *x*-axis in the figure represents the interval between steering updates and the *y*-axis represents mean lateral deviation over 1,000 trials. It is clear that as the time between steering updates increases, lateral deviation generally increases, except, that is, across relatively short intervals between steering updates (< 1 sec). At these shorter intervals, the duration of time in between steering updates did not affect lateral deviation.

It is also apparent from Figure 3 that the way in which lateral deviation increased with increasing time between

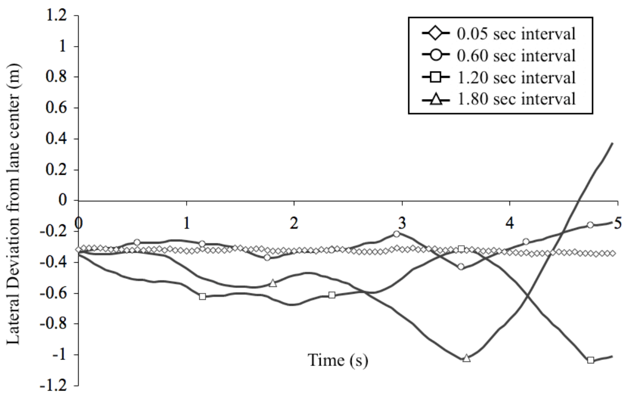


Figure 2: Data plot representing movement of the car in relation to the center of lane for illustrative steering strategies. Data points represent steering updates. Connecting lines represent movement of the car in between steering updates.

consecutive steering updates was dependent on the total period of simulated driving (i.e., single-event vs. long-term measurement interval). In particular, the rate of decline in steering performance was less when there was only a single steering update than when a strategy was maintained for a longer period of time. This is because when there is only a single steering update event, the vehicle travels at a fairly constant lateral velocity; therefore the distance traveled from the lane center will be dependent on the time until the next steering update. However, when a strategy is maintained for a longer period of time (i.e., 60 sec), the average deviation can grow quite large because, in some sense, the problems start building on each other. That is, as we described earlier, the car not only drifts farther from the lane center with increasing time away from driving, but as a consequence, it is also placed into sharper corrective headings to compensate for being farther from lane center. This interaction between increased lateral velocity and longer intervals between steering updates makes the car move erratically about the lane.

Regression analysis was conducted to estimate the best-fitting curve to account for the relationship between the interval between steering updates and lateral deviation. For performance based on a single-event, it was that an exponential function fit the data very well ($r^2$ = 0.98), $F$ (1,118) = 5828, $p$< .001, where

$$\textit{Lateral Deviation} = 0.2755^{0.2177 \times \textit{Update Interval}} \quad (2)$$

The exponent in this function increased, however, when the strategy was maintained for a longer period of simulated driving, giving

$$\textit{Lateral Deviation} = 0.2807^{0.3453 \times \textit{Update Interval}} \quad (3)$$

($r^2$ = 0.99), $F$(1,118) = 35747, $p$< .001. This meant that the rate of decline in steering performance increased more dramatically with increasing interval between steering updates. In the next section we derive predictions for driving performance under dual-task conditions by considering possible strategic variability in how people might dial a cell phone while driving.

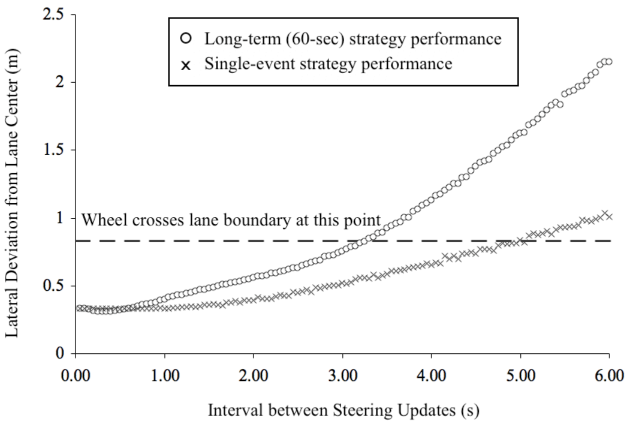


Figure 3: Data plot showing the relationship between the duration of time between each steering update and lateral deviation. Each steering strategy was run over both a single steering event and also a longer 60-second period.

## Predicting Multi-task Performance

We model data from an earlier study that investigated in-car multitasking (Salvucci, 2001). In Salvucci's experiment participants were required to dial 7-digit numbers on a cellular phone that was positioned on a hands-free device while driving. It was assumed that one "power-on" key-press preceded the 7-digit number and that one "send" key-press followed it — giving 9 key-presses in all. Salvucci reports average baseline (or single-task) dial-time for the participant's to enter the 9-keypresses of 5.21 seconds ($S.D.$ = 1.09 sec). We use this empirical estimate of dial-time to calibrate the model.

We assume that in normal conditions drivers typically adjust the heading of the vehicle once every 150 ms. This 150 ms estimate is consistent with assumptions adopted in previous computational cognitive models in the literature (e.g., Salvucci, 2005). Moreover, at this baseline interval between steering updates, lateral deviation predictions given by the model ($M$ = 0.33 m, $S.D.$ = 0.02 m, see Fig. 3) are comparable with reported baseline lateral deviation in Salvucci's (2001) experiment ($M$ = 0.35 m, $S.D.$ = 0.08 m).

We assume that engaging in a secondary task while driving disrupts the normal pattern of checking and adjusting the heading of the vehicle. In particular, we assume that steering updates cannot occur while the driver's attention is directed towards a secondary in-car task, such as when they are entering keypresses for the dialing task. This assumption is based on the idea that peripheral resources, such as the eyes, will limit the degree of parallel processing between tasks. Moreover there are numerous demonstrations in the literature of central interference affecting driver performance in dual-task conditions (e.g., Brumby, Salvucci, & Howes, submitted; Levy, Pashler, & Boer, 2006).

Furthermore, we assume that switching between tasks carries a cost overhead (or switch cost), which reflects the time required to move visual attention between the outside of the car (i.e., to focus on the road) and the inside of the car (i.e., to focus on the phone). Instead of developing a detailed model of the perceptual/motor processes involved, we use a simple timing estimate of 185 ms to move visual attention between the phone and the road, or vice versa. This timing estimate was taken from the ACT-R cognitive architecture (Anderson et al., 2004).

Given the above set of assumptions and also the estimates of single-task performance, we derive predictions for lateral deviation and task time in dual-task conditions. Brumby, Howes, and Salvucci (2007) have previously demonstrated that there are at least $2^8 = 256$ possible strategy variants for completing the dial task with more or less interleaving of steering control. Here, we attempt to abstract over this strategy space by conducting an analysis that varies the number of equal length episodes into which the dial task could be divided and explore the consequences for the interval between steering updates. It should be made clear that this level of analysis abstracts over the actual units of the dial task (i.e., entering more or less digits per episode) and instead focuses on *dividing single-task dial time in to more or less equal chunks of time*; thus, abstractly representing points in the strategy space of more or less interleaving.

Figure 4 presents a scatter plot of total time to complete the dial task and RMSE lateral deviation for strategies that systemically vary in the degree of task interleaving. In particular, at each point in the space we divide baseline dial-time (5.21 sec) by *N*, where *N* varies between 1 (no-interleave strategy) and 9 (maximum-interleave strategy). Given an estimate of the amount of time between steering updates that a particular strategy affords, Equations 2 and 3 are used to derive predictions of lateral deviation. For instance, if we consider adopting a no-interleave strategy, which completes the dial task without once returning attention to the primary task of driving, then the interval between steering updates would be 5.73 seconds (i.e., 5.21 + 0.185 x 2 + 0.15). It is clear from Figure 3 that updating steering control at this interval would likely have catastrophic consequences for the driving task, with the car being likely to cross over the lane boundary.

In contrast, if the dial task were conducted with steering updates occurring after each and every individual digit was entered, what we shall refer to as a maximum-interleave strategy, then the interval between steering updates would be only 1.09 seconds (i.e., 5.21 / 9 + 0.185 x 2 + 0.15). It can be seen in Figure 3 that updating steering control at this interval would not likely lead to an egregious lateral deviation. However, this strategy would incur 4.70 seconds of additional time costs because of frequently switching between tasks and updating steering control (i.e., 9 x (0.185 x 2 + 0.15)).

Figure 4 represents the speed/accuracy trade-off that clearly exists between dialing quickly and driving safely: The upper-left portion of the plot represents faster but less safe performance resulting from less interleaving, while the bottom-right portion represents slower but safer performance resulting from more interleaving. There are diminishing returns for interleaving, however. Such that, while interleaving tasks more often generally leads to safer performance there is a point in the space where further interleaving gives only small improvements in safety.

We compare these model-based predictions shown in Figure 4 with previous empirical data. In particular, Salvucci (2001) reports dual-task performance of 7 sec (*SD* = 1.77 sec) for the dialing task and RMSE lateral deviation of 0.49 m (*SD* = 0.10 m) for the driving task. These human data are also presented in Figure 4.

It is interesting that the human data lie close to the "turning point" where lateral deviation starts to increase dramatically within the modeled strategy space. This suggests that any less interleaving between tasks would likely result in a dramatic increase in lateral deviation, but also that more interleaving between tasks would not likely result in a significant reduction in lateral deviation given the additional time costs.

The model-based predictions demonstrate that adopting a strategy that returns attention to driving after entering three digits at a time is particularly efficient. This strategy does not allow driving performance to become too egregious because the interval between steering updates increases to only 2.26 seconds (i.e., 5.21 / 3 + 0.185 x 2 + 0.15). But at the same time the strategy keeps the additional time costs incurred as a result of interleaving tasks down to only 1.56 seconds (i.e., 3 x (0.185 x 2 + 0.15)).

Finally, notice that predictions for lateral deviation in Figure 4 were derived using the exponential loss function derived from running the model over a single-event (Eq. 2) and also over a long-term measurement interval (Eq. 3). It is interesting that for the most part the single-event model and the long-term model gave fairly consistent predictions for strategies that interleaved tasks more often. However, the model predictions differed fairly significantly for strategies that completed the dial task in only one or two bursts (i.e., the no-interleave strategy). The reason for this discrepancy is that when a particular strategy was maintained for a longer time (i.e., 60 sec), the average deviation could grow quite large at longer intervals between updates.

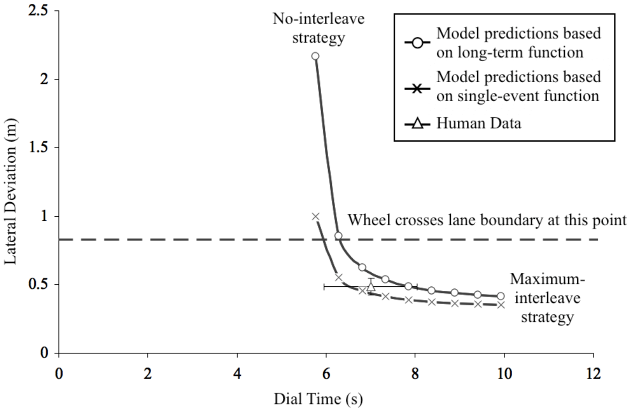


Figure 4: Data plot of dial time and average lateral deviation across strategies of varying task interleaving.

## General Discussion

The question addressed at the start of this paper was why people return attention to steering control after dialing every three or four digits of a telephone number. To address this question, a control model of steering was developed from an analysis of driver performance data. The model made minimal commitments to human cognitive architecture and minimal assumptions about the constraints imposed by the environment. The model was used to predict the average rate at which the lateral deviation of the vehicle from the lane center increases with increasing time between updates of steering control. This bounded rational analysis suggests that the *rate* of decline in driving performance with time away from steering control might determine the amount of time that people are prepared to give up to focus on a secondary task while driving and, by consequence, the degree of task interleaving. We demonstrate that returning attention to driving after entering bursts of three digits at a time is a particularly efficient strategy for completing the dial task while driving because it does not allow driving performance to become too egregious, while at the same time it keeps the time costs incurred from switching between tasks minimal. Moreover, we show that any less task interleaving would result in a dramatic increase in lateral deviation, with possibly unacceptable consequences for safety, and that any more interleaving would incur additional time costs while not affording a significant improvement for driver safety.

An open empirical question that is posed by the analysis presented here is that if the rate of decline in driving performance with time away from task were different, then people might interleave task differently. For instance, imagine if driving performance were to decline much more gradually with time away from task (i.e., when driving at a slower speed), then there would be little value in interleaving tasks: People may as well complete the dial task in a single contiguous burst in order to avoid incurring the costs of switching between tasks. Whereas, if driving performance were to decline at a much more rapid rate (i.e., when driving at a faster speed), then people might be prepared to give up less time per visit to the secondary task and consequently interleave more frequently. That is, driving speed should have an effect on both dial time and lateral deviation in dual-task conditions (see Brumby, Salvucci, & Howes, submitted, for an initial investigation into this question). Moreover, there is evidence that drivers tend to slow down on their own accord when engaging in a secondary dialing task (Salvucci, 2001; Salvucci & Macuga, 2002). This slowing behavior might reflect active attempts to reduce the consequences of directing attention away from the road for driving performance.

We might also consider applying the analysis presented here to some other secondary in-car task that demands a series of longer interaction episodes than a simple dialing task (e.g., selecting media content on an Apple iPod). The analysis presented here clearly suggests that lateral deviation should increase as the amount of time spent on the secondary task increases. An interesting question there emerges from considering a longer task, where the vehicle is more likely to drift from the lane center, is whether people give up more time to steering control (i.e., by conducting a series of steering updates in succession). The approach taken here for running the model over a long-term measurement interval was to assume that only a single corrective steering update is performed, regardless of how far from the lane center the vehicle has became. This seems like a rather implausible assumption, however. An alternative assumption is that people only resume the secondary task return when the vehicle has been to returned to a stable lateral position in the roadway (as in Salvucci's, 2005, 2001, driver models). Further work is required to explore techniques for enumerating over various durations of time given up to steering control for each of the possible multitasking strategies discussed here (see Brumby, Salvucci, Mankowski, & Howes, 2007, for some more recent progress on this issue).

## Acknowledgments

This research was supported by National Science Foundation grant #IIS-0426674. We would like to thank three anonymous reviewers for providing comments for improving this paper.

## References

Allport, A., Styles, E.A., & Hsieh, S. (1994). Shifting intentional set: Exploring the dynamic control of tasks. In C. Umilta, & M. Moscovitch (Eds.), *Attention and performance XV* (pp. 421-452). Cambridge, MA: MIT Press.

Anderson, J.R., Bothell, D., Byrne, M.D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of mind. *Psychological Review, 111*, 1036-1060.

Brumby, D.P., Salvucci, D.D., & Howes, A. (submitted). An empirical investigation into dual-task trade-offs while driving and dialing. Paper submitted to the *British HCI Group Annual Conference*. Lancaster University, UK.

Brumby, D.P., Howes, A., & Salvucci, D.D. (2007). A cognitive constraint model of dual-task trade-offs in a highly dynamic driving task. To appear in *Human Factors in Computing Systems: CHI 2007 Conference Proceedings*. New York, NY: ACM Press.

Brumby, D.P., Salvucci, D.D., Mankowski, W., & Howes, A. (2007). A cognitive constraint model of the effects of portable music-player use on driver performance. To appear in the *Proceedings of the Human Factors and Ergonomics Society 51st Annual Meeting*. Santa Monica, CA: Human Factors and Ergonomics Society.

Gray, W.D., Neth, H., & Schoelles, M.J. (in press). The functional task environment. In A. Kramer, A. Kirlik, & D. Wiegman (Eds.), *Applied Attention*. New York, NY: Oxford University Press.

Green, R. F. (1984). Stopping rules for optimal foragers. *American Naturalist, 123*, 30-43.

Hildreth, E.C., Beusmans, J.M.H., Boer, E.R., & Royden, C.S. (2000). From vision to action: experiments and models of steering control during driving. *Journal of Experimental Psychology: Human Perception and Performance, 26*, 1106–1132.

Howes, A., Vera, A., & Lewis, R.L. (2007). Bounding rational analysis: Constraints on asymptotic performance. In W.D. Gray (Ed.) *Integrated Models of Cognitive Systems* (pp. 403–413). New York, NY: Oxford University Press.

Levy, J., Pashler, H., & Boer, E. (2006). Central interference in driving: Is there any stopping the psychological refractory period? *Psychological Science, 17*, 228-235.

Payne, S.J., Duggan, G.B., & Neth, H. (in press). Discretionary task interleaving: Heuristics for time allocation in cognitive foraging. *Journal of Experimental Psychology: General.*

Salvucci, D.D. (2001). Predicting the effects of in-car interface use on driver performance: An integrated model approach. *International Journal of Human-Computer Studies, 55*, 85-107.

Salvucci, D.D. (2005). A multitasking general executive for compound continuous tasks. *Cognitive Science, 29*, 457-492.

Salvucci, D.D., & Macuga, K.L. (2002). Predicting the effects of cellular-phone dialing on driver performance. *Cognitive Systems Research, 3*, 95-102.

Son, L.K., & Sethi, R. (2006). Metacognitive Control and Optimal Learning. *Cognitive Science, 30*, 759 -774.

# From 1000ms to 650ms:
# Why Interleaving, Soft Constraints, and Milliseconds Matter

**Bella Z. Veksler (zafrib@rpi.edu)**
**Wayne D. Gray (grayw@rpi.edu)**
**Michael J. Schoelles (schoem@rpi.edu)**
Rensselaer Polytechnic Institute
110 8th St., Troy, NY 12180

## Abstract

Evaluating and modeling human performance on even simple tasks requires a great deal of attention to millisecond-level cognitive and perceptual-motor operations. Modeling human performance in a task often requires that special care be taken to understand how these millisecond level operations are interleaved and how they evolve during the execution of the task. In modeling a simple decision-making task, we found that human subjects improved their routine speed as they became more familiar with the task. Modeling was conducted using the ACT-R architecture (Anderson & Lebiere, 1998). Refinements of the model indicated that interleaving of millisecond-level perceptual-motor and cognitive operators was crucial in accounting not only for the strategy shift as per soft constraints, but also in the marked speedup in performance over the course of several trials.

## Introduction

Milliseconds matter in understanding human performance (Gray & Boehm-Davis, 2000). The soft constraints hypothesis (Gray, Sims, Fu, & Schoelles, 2006) implies that in the course of routine interactive behavior, the cognitive controller tends to select interactive routines that shave milliseconds off of task performance. Unfortunately, this local optimization may not result in optimal global performance. Hence, even in tasks that are thought of as involving higher-level cognition, such as decision-making, global performance may be suboptimal due to nearsighted, local optimization of interactive routines.

From the perspective of the soft constraints hypothesis, computational models of decision-making must encompass a full accounting of the costs of information exploration and exploitation (Fu, 2007). Hence, an initial task for the modeler is to account for the perceptual-motor costs of skilled performance. As we show in this paper, this initial task brings to the foreground the interleaving of cognitive, perceptual, and motor operations that is characteristic of skilled performance.

We describe an exploratory effort to model the interleaving of cognitive, perceptual, and motor operations required for information exploration/exploitation in a table-based, decision-making task (Lohse & Johnson, 1996). The constraints of the model/framework are examined at the level of milliseconds, contrasted against human data, and the differences are analyzed with respect to the ACT-R framework (Anderson & Lebiere, 1998).

## The Task

The experimental environment used in this research was designed to study and model how information access influences the way in which a decision is made – specifically what information is considered and how it is integrated given the environmental constraints and accessibility of information. We used a simple table task (see Figure 1) in which each of six alternatives (arranged in rows) had a value on each of six attributes (arrayed in columns). The value of the alternative was derived by summing the attribute scores so that the higher the value, the better the alternative. This environment allowed us to manipulate the way information was accessed in order to determine the cognitive and perceptual-motor tradeoffs involved.

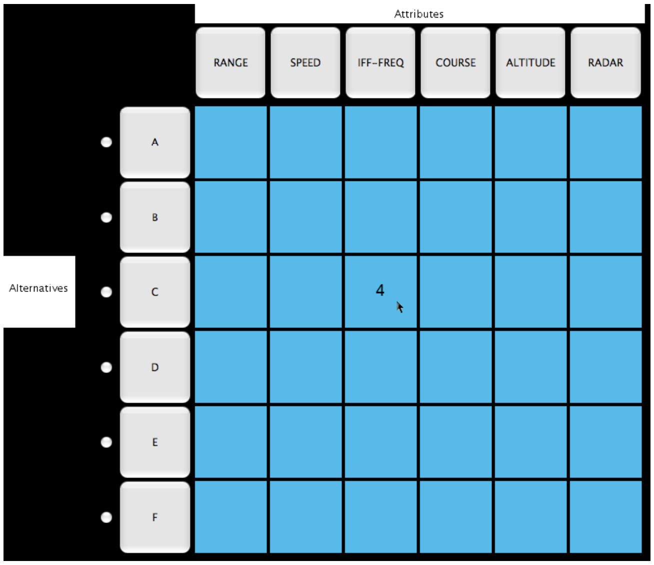


**Figure 1:** The *Table Task* environment for decision-making. The figure shows a subject clicking on the IFF-FREQ attribute (column) for alternative C (row).

We predicted that performance would vary based on exploration/exploitation costs that variations in the task environment imposed on the decision maker. In particular, we expected different costs to result in differences in the time to make a decision as well as the amount of information considered during the trial (i.e., information exploration). We also predicted that when participants were transferred to conditions with different environmental constraints, that the transfer of old strategies or the adoption of new ones would be influenced by exploration/exploitation costs of the old strategies applied to the new task environment. (Gray, 2000; Gray, Veksler, & Fu, 2004)

Although these general predictions are validated in the Results section, this analysis is beyond our current modeling effort. Exploratory modeling of this simple task revealed the necessity to focus our scope of analyses on the basic motor components prior to taking the next step into modeling the higher-level experimental effects.

# Human Data

## Method

The table task environment consisted of 6 alternatives arranged as rows and 6 attributes arranged as columns in a grid. Alternatives were military targets with attributes that contributed to their overall threat. There were values in the corresponding grid cells and it was the task of the participant to select the alternative whose corresponding attribute values summed to the highest value (see Figure 1).

There were a total of four conditions that varied how the values in the grid could be accessed. For purposes of this paper and the models presented, we only cover the "by cell" condition (CE). In this condition, participants accessed information one cell at a time by clicking on the grid cell corresponding to the value of an attribute for a particular alternative.

Each trial consisted of the participant checking the values in the grid and selecting the alternative with the highest overall value. Feedback on the number of correct answers was provided at the conclusion of the experiment. Participants completed 30 trials in this manner and for the CE condition included 18 participants.

## Results & Discussion

Our interest lays in modeling the millisecond level interactive routines of each trial in addition to the changes in information exploration/exploitation that occurred in performance within and across trials. The trial duration analysis below is intended as a benchmark for the subsequent model's performance.

### Total Trial Duration

Total trial duration averaged 23.77s, StErr = 478.32ms. However, trial durations across the 30 trials follow a power-law of learning (Figure 2). It is thus important to note that trial duration decreased from the first (M = 36.92s; StErr = 3.66s) to the last trial (M = 21.72s; StErr = 1.8s).

### Number of Cell Clicks

Participants were presented with a 6x6 grid of cells for a total of 36 cells that would need to be checked to have perfect information during a trial. Is there any indication that participants saved time by not checking every cell? Although there was some variability in the number of cells clicked across the trials, participants clicked an average of 35.81 cells. Therefore, participants roughly clicked on each cell once. The subsequent model therefore also clicks on each cell once during a trial.

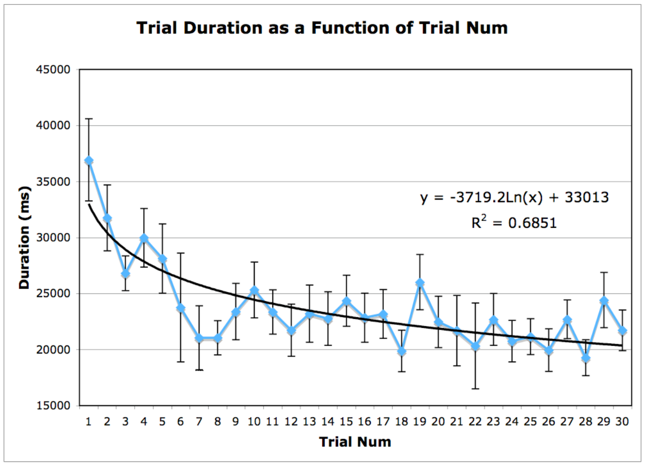


**Figure 2:** Power law of learning in trial duration

### Inter-Cell Click Interval

In addition to trial duration and number of cells clicked, we assessed how participants were spending their time during task performance. In particular, we analyzed how long they spent between cell clicks. We will call this the "inter-cell click interval". Given that not all participants clicked the same number of cells during each trial, the following analysis only shows data from the first 36 cell clicks. As will be discussed later, the inter-cell click interval provides insights into how strategies evolve over time and how we can modify our models to match human performance. It also provides insights into the cognitive and perceptual-motor shortcuts that people take and that a cognitive model needs to account for. Essentially, these are the millisecond-level operations that are crucial in many repetitive or well-practiced tasks.

Figure 3 illustrates how inter-cell click intervals changed over the course of the whole task. Initially, inter-cell click intervals averaged ~950ms whereas by the end of the 30 trials, they had decreased to ~550ms. This trend is analogous to the trend of the overall trial duration we observed in Figure 2 and is one of the ways we can determine how the strategy that the participants employed evolved.

Furthermore, within a trial (see Figure 4), we witness variability in inter-cell click intervals with respect to cell click number. The more cells are clicked within a trial, the shorter the inter-cell click interval becomes. Notice also that there is a seesaw pattern such that every 7th inter-cell click interval is longer than the surrounding ones. This is accounted for by the fact that each 7th click was a row switch. One explanation for this is that at the end of a row, participants updated their current highest value and therefore took longer transitioning to the next alternative.

Within a row, the inter-cell click interval also showed a slight increase presumably explained by the increase in cognitive load as participants added more values to their running total of the alternative's value. Figure 4 shows average data across all 30 trials.

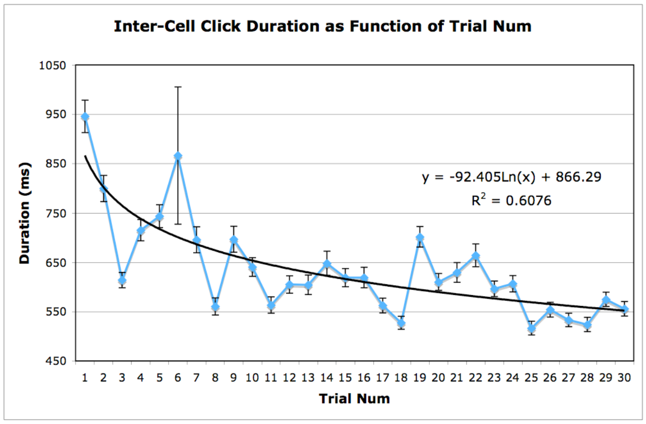


**Figure 3:** Power law of learning in inter-cell click interval

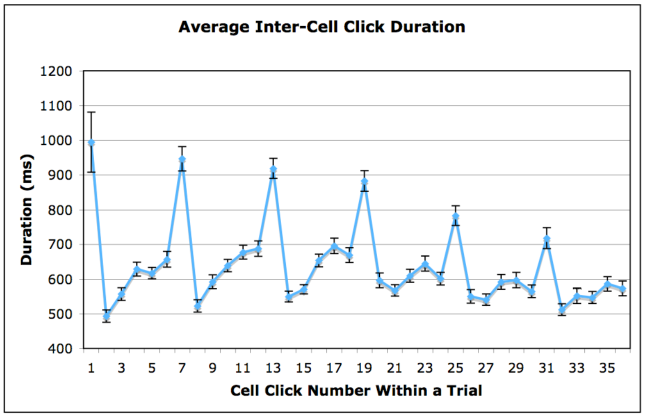


**Figure 4:** Average human inter-cell click interval within a trial. Peaks represent transitions between alternatives (rows)

### Transitioning Between Rows

Another important consideration for task performance is encompassed by the soft constraints hypothesis (Gray et al., 2006). Soft constraints guide the selection of interactive routines at the millisecond level to minimize performance costs as measured in terms of time. The utilization of soft constraints is reflected in apparent strategy shifts as performers become familiar with the task. Although initially certain biases may have caused the performer to use one set of interactive routines, the cost of exploration/exploitation ultimately shifts performance towards more efficient strategies.

This shift in strategy is most clearly seen in the transitions between alternatives (rows). Whereas initially participants were biased to "read" the values in the rows from left to right (Figure 5A), after several trials a more efficient strategy emerged. The new strategy had participants alternating the direction in which they clicked the cells based on their final position in a particular row (Figure 5B). Figure 5C shows that across trials participants increased their use of strategy B by 10%.

## The Model(s)

To model human performance on this task, we used the ACT-R cognitive architecture (Anderson et al., 2004). ACT-R is a modularized production system with a subsymbolic memory module. It has visual and motor modules to embed it in the task environment. It also has declarative memory and a procedural module. In addition, it has imaginal and goal buffers to store its working memory and goal chunks, respectively. Thus, it serves as a good framework to model human performance on this simple table task.

Several ACT-R models were developed in order to model the various components of human speed increases during this task. The essential structure of all of the models is the same: each model simply goes through each alternative, uncovers each cell value, sums the cells and updates its memory of the highest value after comparing it with the previous highest value. The differences between the models primarily lie in how they execute this list of perceptual-motor and cognitive operations.

At present, we have deliberately avoided implementing different decision-making strategies and have focused our modeling effort on getting the interleaving of cognitive, perceptual, and motor operators right. As discussed below, we do not know how to account for the obvious adaptations in interleaving that humans undergo. We view our lack in this regard as a comment on the state of the art in

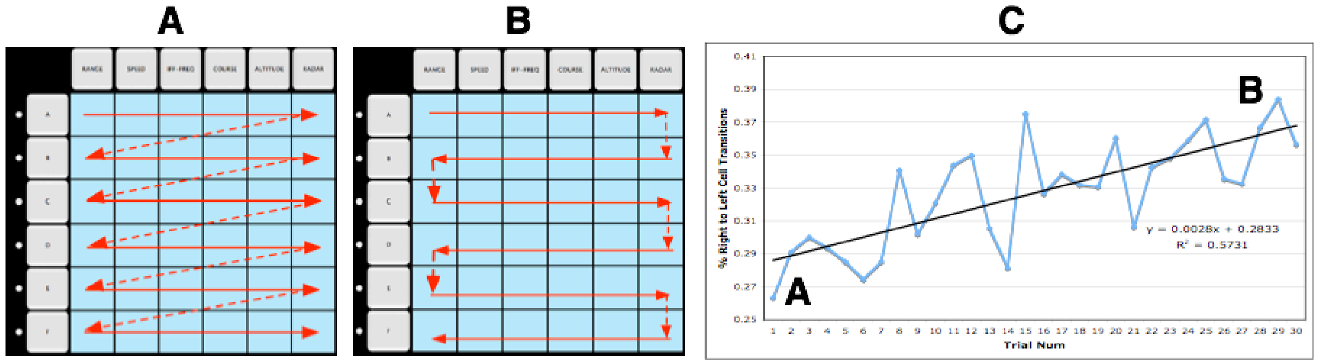


**Figure 5:** Different strategies in uncovering cell values. (A) Reading values left to right (B) Reading values alternating left-to-right and right-to-left (C) Percent of right-to-left cell click transitions as a function of trial number

interleaving which has not advanced much since the pioneering EPIC-Soar work of Chong in the late 90's (Chong, 1998a, 1998b; Chong & Laird, 1997). One method that has touched on interleaving of perceptual-motor and cognitive operators since then is Cognitive Constraint Modeling (Lewis, Howes, & Vera, 2004). Cognitive Constraint Modeling provides a description of behavior derived via constraint satisfaction. However, unlike Chong's work, this method is not at all concerned with how human interleaving strategies adapt through experience.

The absence of a mechanism that interleaves cognitive operators has led us to build models that do not change over trials but which bracket human performance (Gray & Boehm-Davis, 2000; Kieras & Meyer, 2000). Understanding the differences between these models offers some insight into how perceptual-motor and cognitive mechanisms might evolve across trials.

## Model 1: Non-Interleaved

This was an "out-of-the-box" model, composed of sequential productions that can roughly be divided into four categories. The first set of productions (Figure 6A) started each trial and switched between alternatives. The second set of productions (Figure 6B) was the workhorse of the model. This set of productions initiated the perceptual-motor operations of moving the mouse and visual attention to the various cells. It was also responsible for adding the values in the cells. It did this in a systematic left-to-right fashion for all alternatives. Thus, this model employed strategy A from Figure 5. The third set of productions (Figure 6C) compared a current alternative's value to the highest value so far and updated the model's memory of the highest alternative seen. The fourth set of productions (Figure 6D) only fired after each alternative's value had been computed and the model was ready to select its answer.

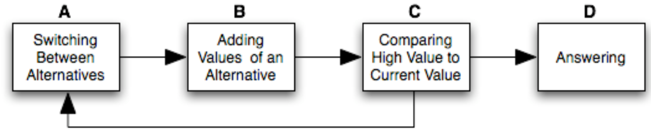

A
B
C
D
Switching Between Alternatives
Adding Values of an Alternative
Comparing High Value to Current Value
Answering

**Figure 6:** Workflow of the models

The model had declarative knowledge of addition facts and number relation facts. Thus, whenever it needed to compare whether a current alternative's value was greater than the highest value so far, it would search in its declarative memory for a relation fact involving those values.

The model also had a goal buffer that kept track of the alternative that it was currently scrutinizing, and an imaginal buffer that kept track of the highest value seen so far. Since it is beyond the scope of this paper to model human accuracy on this task, it sufficed for the model to hold the highest value and alternative in its imaginal buffer at all times, disallowing for any forgetting errors to occur.

This was the simplest model that encoded the task and for this reason, we did not expect its performance to match well with human data. It is termed non-interleaved because the perceptual-motor and cognitive operations were done largely sequentially and not interleaved with each other. We found that although this simple model failed to match duration times on the majority of trials, it did match duration times as compared to the first trial of human data (Figure 8, Model 1: Non-Interleaved).

## Model 2: Interleaving Cognitive with Perceptual-Motor Operations (I-CPM)

Examining the time plot of ACT-R's various modules over the course of a single inter-cell click interval (Figure 7), we noticed that gaps between production firings could be used to interleave perceptual-motor and cognitive operations. The interleaving was accomplished by firing productions that added the value of the last cell to the running total as the motor module was moving the mouse to the next cell. This interleaving saved ~100ms during each inter-cell click interval and decreased total trial duration by about 2.8 seconds from ~34.6s to ~31.8s, matching human duration times from Trial 2 (Figure 8, Model 2: I-CPM).

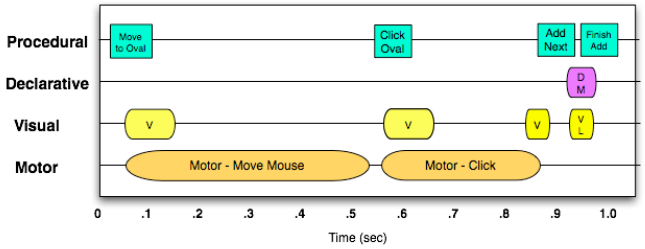


**Figure 7**: Time graph of between cell clicks in ACT-R for the non-interleaved model.

## Model 3: Interleaving Motor Preparation Time (I-CPM+MP)

Figure 7 also shows that the motor component (moving the mouse to the cell and then clicking it) comprised 80% of the time of the entire duration (786ms out of 931ms). What this means is that *just the motor component alone takes more time than the entire inter-cell click interval in the human data*.

During the course of a single trial, the repetitive sequence of moving the mouse to a cell and then clicking is done many times. In the human data, this practiced motor sequence became increasingly faster as attested by the decrease in the inter-cell click interval (see Figure 4). We decided to account for this increase in speed by taking advantage of ACT-R's motor module mechanisms.

When a motor command is issued to ACT-R's motor module, that command is executed in three phases: preparation, initiation, and execution. In cases where the model can tell ahead of time what movement will follow, it is beneficial to begin "preparing" the next movement before the current movement is finished executing. Thus, to account for the learning effects we observe in repeating the same two motor commands over and over, we allowed the model to begin preparing the next motor command prior to the finish of the current command. For example, while the move-mouse command was executing, the model already began preparing the mouse-click that would inevitably follow.

This motor preparation interleaving refinement of the model drastically decreased inter-cell click interval and, consequently, trial duration. The improvement decreased individual trial time by about 7.3 seconds, from ~31.8s to ~24.5s. This is a marked improvement over Models 1 and 2 and brought the model closer to the average human trial time of ~23.8s (Figure 8, Model 3: I-CPM+MP).

### Model 4: Alternating Transitions Between Rows (I-CPM+MP+R)

In a task in which interactive routines are on the order of hundreds of milliseconds, it is important to be able to determine where exactly it was that the model was incurring a large time cost. We therefore compared the inter-cell click interval analyses for human and model data. This comparison revealed that the major difference between human and model inter-cell click intervals was during the transitions between alternatives (where each alternative is a row in Figure 1).

As discussed earlier, participants' strategies changed over the course of the task. Initially, they clicked on cells in a left to right fashion whereas later they alternated the direction depending on their ending position in a given row. We thus incorporated this alternating behavior into the model thereby decreasing the distance the mouse had to move when a new alternative was encountered. Since move-mouse execution time in ACT-R is closely related to the distance that the mouse must move, as per Fitts' Law (Fitts, 1954; MacKenzie, 1992), this feature allowed the model to transition faster between alternatives (compare Figure 5 A and B).

This refinement in the model decreased total trial duration time by about 900ms from ~24.5s to ~23.6s (Figure 8, Model 4: I-CPM+MP+R). Although this was not a large difference, incorporating this component into the model makes it more cognitively plausible especially given that we see human participants exhibiting this shift in strategy.

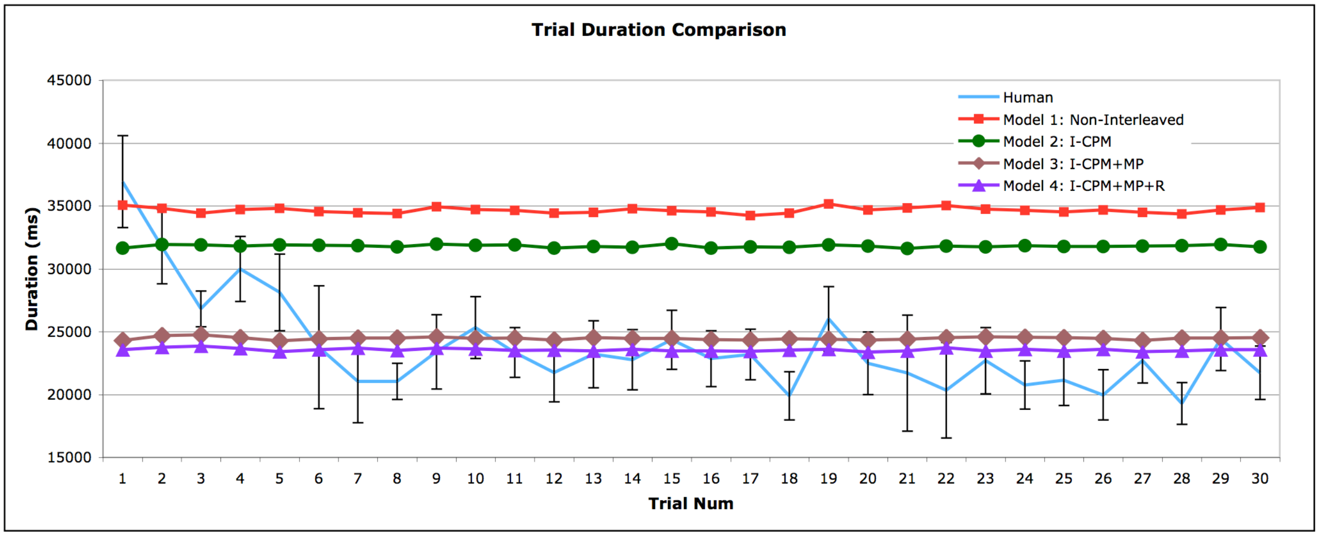


**Figure 8:** Comparison of Trial Duration Times Between the 4 Models and Human Data. Model 1: Non-Interleaved – Purely sequential model; Model 2: I-CPM – Interleave cognitive, perceptual, and motor operations; Model 3: I-CPM+MP – Interleaved motor preparation (MP) time added; Model 4: I-CPM+MP+R – Alternating transitions between rows

This final refinement of the model had the best fit to the asymptote performance in human data. Future work will include the model learning to choose between the two strategies.

## Conclusion & Future Work

Sometimes one can learn more from a modeling effort when the model does not fit the data than when it does. In fact, the lack of fit can tell us a lot about not only the limitation of the model itself and how to proceed to modify it but also about the limitation and error of the constraints with which the model was implemented. In this case, we wanted to investigate where and how the speedup in performance in humans occurs and in particular what it was about the "out-of-the-box" model that prevented it from matching human times.

### Where Does the Time Go?

According to human data, the inter-cell click interval varies as a function of trial number (Figure 3) and cell click number within a trial (Figure 4). We can see that the majority of inter-cell click intervals fall within the 600ms range. The first trials have longer durations as compared to the last trials, and the first few cell clicks in a trial take longer than subsequent cell clicks. However, the "out-of-the-box" model (Model 1) performs considerably slower in all cases, an average of around 950ms per inter-cell click interval.

One way we can speed up the model's performance is to interleave the cognitive and perceptual-motor components.

This results in at most a speedup of ~100ms per inter-cell click interval. However, if we look at human data particularly towards the end of the 30 trials (Figure 3), we see times of 520-600ms, which is considerably faster than the model's motor component alone, as per Figure 7, would allow.

Another way to speed up the model's performance is to interleave the motor preparation times with execution times. Since ACT-R does not do production compilation across perceptual and motor commands, there does not seem to be any other way of incurring this speedup in performance (Taatgen & Lee, 2003). The speedup afforded by this preparation interleaving results in a decrease of ~200ms per inter-cell click interval.

As per the soft-constraints hypothesis, a further refinement of the model altered how transitions between rows occurred. This resulted in an additional savings of ~30ms per inter-cell click interval.

Taken together, this modeling effort demonstrates the importance of millisecond-level considerations operating under even the simplest of tasks. The current model was intended to address the most perceptually motor intensive condition of the study. As such, it has led us to discover the crucial nature of interleaving and soft-constraints in attaining skilled performance.

The table task environment is a rich test bed for exploring how interactive routines in an information exploration/exploitation task evolve to produce skilled performance. Future modeling work of this task will explore how the different experimental conditions affect this evolution of interactive routines, and how these interactive routines influence performance in the decision-making task.

## Acknowledgements

Thanks to Hansjörg Neth and Christopher W. Myers for their invaluable contributions to this project. This work would not have been possible if not for the many useful discussions with Vladislav D. Veksler. The writing of this chapter was supported by a grant from the Air Force Office of Scientific Research AFOSR #FA9550-06-1-0074, Dr. Jerome Busemeyer, Program Officer.

## References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglas, S., Lebiere, C., & Quin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111*(4), 1036-1060.

Anderson, J. R., & Lebiere, C. (Eds.). (1998). *Atomic components of thought*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Chong, R. S. (1998a). Modeling dual-task performance improvements with EPIC-Soar, *The Twentieth Annual Conference of the Cognitive Science Society* (pp. 1209). Hillsdale, NJ: Lawrence Erlbaum Associates.

Chong, R. S. (1998b). Modeling single-task performance improvement using EPIC-Soar, *Ninth Midwest Artificial Intelligence and Cognitive Science Conference* (pp. 10–16). Menlo Park, CA: AAAI Press.

Chong, R. S., & Laird, J. E. (1997). Identifying dual-task executive process knowledge using EPIC-Soar. In M. G. Shafto & P. Langley (Eds.), *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society* (pp. 107–112). Palo Alto, CA: Lawrence Erlbaum Associates.

Fitts, P. M. (1954). The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement. *Journal of Experimental Psychology, 47*(6), 381–391.

Fu, W.-T. (2007). A rational–ecological approach to the exploration/exploitation trade-offs: Bounded rationality and suboptimal performance. In W. D. Gray (Ed.), *Integrated models of cognitive systems* (pp. 165-179). New York: Oxford University Press.

Gray, W. D. (2000). The nature and processing of errors in interactive behavior. *Cognitive Science, 24*(2), 205-248.

Gray, W. D., & Boehm-Davis, D. A. (2000). Milliseconds Matter: An introduction to microstrategies and to their use in describing and predicting interactive behavior. *Journal of Experimental Psychology: Applied, 6*(4), 322–335.

Gray, W. D., Sims, C. R., Fu, W.-T., & Schoelles, M. J. (2006). The soft constraints hypothesis: A rational analysis approach to resource allocation for interactive behavior. *Psychological Review, 113*(3), 461-482.

Gray, W. D., Veksler, V. D., & Fu, W. T. (2004). *Probing the Paradox of the Active User: Asymmetrical Transfer May Produce Stable, Suboptimal Performance.* Paper presented at the Twenty-Sixth Annual Meeting of the Cognitive Science Society.

Kieras, D. E., & Meyer, D. E. (2000). The role of cognitive task analysis in the application of predictive models of human performance. In J. M. Schraagen, S. F. Chipman & V. L. Shalin (Eds.), *Cognitive task analysis* (pp. 237–260). Mahwah, NJ: Lawrence Erlbaum Associates.

Lewis, R.L., Howes, A., & Vera, A. (2004). A constraint-based approach to understanding the composition of skill. *International Conference on Cognitive Modeling,* Pittsburgh, 2004.

Lohse, G. L., & Johnson, E. J. (1996). A comparison of two process tracing methods for choice tasks. *Organizational Behavior and Human Decision Processes, 68(1),* 28-43.

MacKenzie, I. S. (1992). Fitts' law as a research and design tool in human-computer interaction. *Human-Computer Interaction, 7*(1), 91–139.

Taatgen, N.A. & Lee, F.J. (2003). Production Compilation: A simple mechanism to model Complex Skill Acquisition. *Human Factors*, 45(1), 61-76.

# The Costs of Multitasking in Threaded Cognition

**Jelmer Borst (jpborst@ai.rug.nl)**[1,2,3]
**Niels Taatgen (taatgen@cmu.edu)**[1,2]
[1]Department of Psychology, Carnegie Mellon University, USA
[2]Artificial Intelligence, University of Groningen, The Netherlands
[3]School of Behavioral and Cognitive Neurosciences, University of Groningen, The Netherlands

## Abstract

Most multitasking models make use of executive processes to assign resources to tasks (Kieras et al., 2000). An alternative is to have no executive, but constrain individual processes so that they share resources in a plausible way. Salvucci and Taatgen (under revision) in their theory of threaded cognition have shown how peripheral resources and declarative memory are shared between processes without an executive. In this paper we will extend this work by showing how two tasks share a resource to store the problem representation in a dual-task paradigm where either task sometimes needs a problem representation and sometimes not. Threaded cognition predicts extra interference when both tasks need a problem representation, which is what we found in the experiment.

## Introduction

Human beings are amazingly adept at performing multiple tasks concurrently, and at combining previously unrelated tasks. This stands in sharp contrast to the current situation in cognitive modeling, where most models of multitasking make use of a so-called *Customized Executive* (Kieras et al., 2000). This is an, often complicated, control process specialized for the tasks at hand. It determines how the tasks will be interleaved, and at which point one of the tasks takes precedence. A consequence of this is that for every two tasks a different control structure is required, which, in turn, implies that we would have to learn a new control structure for every new combination of tasks. A more plausible solution would be to have a *General Executive* that could interleave any two tasks (Kieras et al., 2000; Salvucci, 2005). There have been several proposals for such a General Executive in cognitive architectures (e.g., Kieras et al., 2000 (EPIC); Salvucci, 2005 (ACT-R)). However, these proposals have not been equally successful in accounting for multitasking data as customized executive approaches.

Yet another possibility is to have no executive at all (e.g., Liu, Feyen, & Tsimhoni, 2006), which is the underlying idea of the new multitasking theory 'Threaded Cognition' of Salvucci and Taatgen (under revision). The essence of threaded cognition is that it has no central executive, but instead makes sure individual tasks in a multitasking situation interleave without top-down control. This interleaving of individual tasks sometimes leads to additional costs. Salvucci and Taatgen have already shown how declarative memory can be a contended resource, and that competition for this resource can explain differences between novices and experts on a task. In the current paper we will show evidence for a second shared central resource: the problem representation. We will use a dual-task situation with two relatively complex tasks: driving and operating a navigation device. The experimental manipulation is to have two variations of each of the two tasks, one that does require a problem representation, and one that does not.

First, we will outline threaded cognition, and show what kind of multitasking costs the theory predicts. Second, we will test in an experiment whether this prediction is correct, and finally compare the results of the experiment to a model designed with threaded cognition.

## Threaded Cognition

Threaded cognition posits that each task (in a multitasking context) is represented by a cognitive thread (Salvucci & Taatgen, under revision). Each of these threads has its own control structure: there is no central executive; threads are independent and can be run in isolation. Threaded cognition can therefore account for the flexible way humans combine previously unrelated tasks, and for the fact that many tasks can be learned in isolation first and performed together later.

All threads are processed together on a single processor, which can only execute one rule at a time, and will therefore present a bottleneck (this in contrast to the approach of Kieras et al., 2000). At any given time, production rules of all threads can be selected, when multiple rules (of different threads) match, the rule belonging to the thread that has least recently been processed will be executed. This makes sure none of the threads will starve as long as it has matching production rules.

While the central processor presents a first bottleneck, it is not the only one. The threads have to share resources like memory and vision, which creates additional interference. For instance, if two threads need to retrieve a fact from declarative memory, the one that comes first can request retrieval, and the second thread will have to wait. A second consequence of resource sharing is that threads have to be polite, in that they should not 'steal' resources from another thread, as this could result in an infinite loop.

### Costs of multitasking

As explained above, possible bottlenecks in the model are the central processor and resource sharing. In the current paper we investigate interference of sharing the problem representation resource. If a thread has to keep a problem state in mind, for instance a partial solution to a problem, and another thread has to keep track of its own problem state, both threads will have to restructure their problem state every time they take control (assuming only one problem state can be maintained at a time). Thus, threaded cognition predicts additional interference in case two threads both have to keep track of their own problem state. Additional in the sense that the problem representation has to be restored on every task switch, in contrast to the use of the visual or memory resource where threads only have to wait sometimes, but can carry on afterwards.

The current paper tests this prediction by comparing two tasks in two conditions, an easy condition in which no problem state is necessary, and a hard condition in which it is. Thus, suppose performance is 100% if both tasks are easy, and 90% when one of the two tasks is hard (because of perceptual / motor / memory resource sharing), threaded cognition predicts a task performance lower than 80% in the condition when both tasks are hard.

### Threaded Cognition & ACT-R

Because threaded cognition strives to be an "integrated" theory, it is implemented in the cognitive architecture ACT-R (Anderson et al., 2004). ACT-R is a cognitive architecture consisting of specialized modules functioning around a central production rule system. This production system works on a single goal at a time, for which it sequentially executes production rules. In order to achieve multitasking, a control structure is needed that switches between the multiple goals at the appropriate moments, essentially requiring a customized executive for each combination of tasks.

A possible solution for this problem could be, as stated above, threaded cognition. This is implemented in ACT-R in the following way. Instead of only one goal, ACT-R is now allowed to have multiple goals. Each goal represents a thread, and will have a number of dependent production rules. However, as in standard ACT-R, only one rule can fire at any given time. If production rules related to different goals match at the same time, threaded cognition will select the rule belonging to the least recently processed goal.

In ACT-R, the problem representation has to be stored in the imaginal buffer, which has to be shared by multiple tasks. In combination with threaded cognition this clearly predicts strong interference if two tasks have to keep track of a problem represenation.

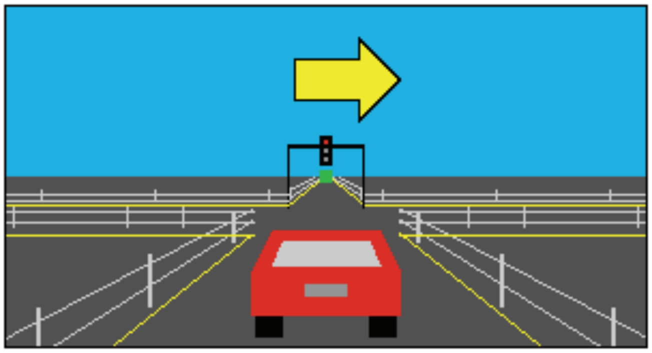


Figure 1. Example of an 'easy' intersection.

## The Experiment

To test our hypothesis we modified the discrete driving task of Salvucci, Taatgen and Kushleyeva (2006). This is a task in which participants have to steer a car down a road on the left side of the screen, while entering information into a navigation device on the right. As explained above, for our current purposes we needed two tasks, both with a hard condition in which participants have to keep track of a problem state, and an easy condition in which this is not the case. To this end we modified both parts of the discrete driving task. We will describe both tasks in detail below.

### Driving

In the driving part of the experiment the participants' main task is to keep the car in the middle of the road. Every few moments (0.5, 0.75, or 1.0 seconds, with equal probability) the car is perturbed 10 pixels to the right or to the left. It can be steered back to the middle of the road by pressing 'a' or 'd' (left or right, respectively), which also resets the perturbation timer. When the car is in the middle of the road, it will move to the left or to the right with equal probability. When it is already on one of the sides, it will move in 2/3 of the cases further to that side.

Every 15 seconds the car reaches an intersection, where it can either go left, straight, or right (keys 'q', 'w', 'e'). In the easy condition, participants are shown where to go by an arrow above the intersection, as in Figure 1. They only have to press the corresponding key on the keyboard, and do not have to keep track of past or upcoming intersections. In the hard condition, four arrows are shown at the first intersection of a set of four, and none on the other three. This means that participants have to (1) remember where to go on a series of three intersections, and (2) keep track of how many intersections they have already passed in the current set. The four arrows are shown for a maximum of 3 seconds.

### Navigation

Navigating is done using the mouse, and while it has to be performed concurrently with steering the car, participants use their left hand to steer the car with the keyboard and use their right hand for navigation with the mouse.

The navigation task starts with an initial screen with five buttons: street number, street name, city, state, and done. These buttons are used to choose the category to be entered, but as only one of them is active (and highlighted) at a time, this part of the task is trivial. When one of the buttons is clicked a keyboard appears, as in Figure 2 (in case of street name, city, or state the keyboard is completely alphabetic).

Figure 2 shows an example of the easy task. In this case the to-be-entered character is present in the display, the only thing a participant has to do is to click the corresponding key on the keyboard. As soon as the click is registered a new stimulus appears; this continues until the whole number/name is entered (the participant has no access to the full name, and can therefore not plan ahead). After all

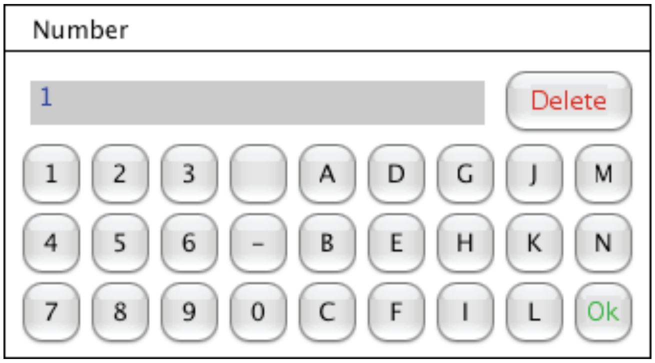


Figure 2. Navigation display in the easy variant.

characters of a name have been entered 'OK' is shown in the input field, when the participant clicks the OK-button the task returns to the initial display and the next category is highlighted. When all four parts of the address have been entered the Done-button is highlighted, when that is clicked the display disappears for 10 seconds, after which a new display appears. When the car reaches an intersection, the buttons of the navigation device become inactive, to become active again as soon as the participant steered.

In the hard condition a whole number/name is shown in the input field at once, however, it disappears as soon as the participant starts typing. Also, no feedback is offered to the participant as to what they have entered; only a 'click' can be heard every time a button is clicked. This means that the participant has to keep in mind what word they are typing and which character of the word has to be entered next.

In both conditions the numbers were three digits long, the street names six letters, the city names contained nine letters and the states were the normal two-letter abbreviations. In the hard condition, real street / city / state combinations of well-known cities were used. In the easy condition the characters of these names were scrambled to prevent participants from guessing the word.

### Eye-tracking

To investigate which of the two tasks the participants were focused on at a particular moment, we used an Eyelink II head-mounted eye-tracker (SR Research) to record eye movements.

### Participants

27 people agreed to participate in the experiment for monetary compensation. As one of them left halfway through the experiment because of a fierce headache, there are 26 complete datasets (11 female, age range 18-34, mean age 23.4). All of the participants had normal or corrected-to-normal visual acuity. Informed consent was obtained before testing. Due to technical difficulties the eye-tracking data of 6 participants could not be analyzed.

### Experimental set-up

The experiment started with five practice blocks: easy driving: 2 blocks of 4 intersections; hard driving 2x4 intersections; easy navigation: 2 complete addresses; hard navigation: 2 addresses; combination: one set of each condition combined: 4 sets of 4 intersections and a complete address. This might sound a bit overdone as the single tasks are quite easy, but as the response of many participants indicated at the combination practice ("this is impossible!"), it was necessary.

After the practice block the participants were asked to do the single tasks in isolation, to measure their base level performance (3 sets of 4 intersections in the two driving conditions, 3 addresses in the two navigation conditions). The main part of the experiment existed of two blocks of 12 4-intersection sets and addresses each, thus 24 sets in total. At the end of the experiment the single tasks were once again administered, to control for learning effects. Between the different blocks participants could take a break, which they usually only did halfway the main phase. The complete experiment lasted approximately 1.5 hours.

## The Model

To model this task we used threaded cognition and ACT-R. The experiment consists of two tasks that can be performed in isolation: driving and navigation. Thus the model will have two threads, which we describe in turn below.

### Driving thread

As long as the driving thread is the only active thread, it can constantly attend the road, and act promptly to every perturbation. However, most of the time a navigation thread is also present which needs to attend the navigation device. To know when it has to focus attention back on the road the driving thread needs a sense of time, which we implemented using the previously validated temporal module (Taatgen, Van Rijn, & Anderson, in press).

As long as the car is not on the center of the road, the driving thread will use the visual resource. It will give it up as soon as the car is on the middle of the road. As soon as it notices that the visual module is used by another thread and attends something else than the road (in this case the navigation device), the driving thread will start the timer of the temporal module. While the navigation thread is busy entering information into the navigation device, the driving thread tries to decide whether it is time to look at the road by retrieving past timing experiences, stored at the current timer value. If it retrieves an experience that says it is time to drive again, the driving thread attends the road, and steers the car back to the middle. It can also retrieve an experience saying it is still safe to continue navigation, in which case that is exactly what it does. If it fails to retrieve a past experience it will continue navigating half of the time, and go back to driving in the other half of the cases.

Where do these timing experiences come from? Every time the driving thread starts steering the car, it first stores whether this was already necessary or not (i.e., whether the car was far out of the middle of the road, or whether it was still driving safely in the middle) together with the timer value on which it looked back to the road; this forms a timing experience. It should be noted that while the driving thread is combined with a navigation thread in this particular example, this is by no means necessary. Without making any changes to the driving thread, it can be combined with any other behavior performed while driving, like using a cell phone.

The driving thread steers the car back to the middle of the road by looking whether the car is to the left or to the right of the center, and pressing the corresponding key. When the car stops at an intersection, the model tries to find an arrow. If there is only one arrow, it presses the corresponding key. If there are four arrows, the model starts memorizing them by attending them in left to right order, until the arrows disappear after 3 seconds. It also changes its problem state to represent where it is in the current set of intersections. If it now arrives on an intersection with no arrows it retrieves the arrow corresponding to the current problem state from memory, and steers into that direction. Every time the driving threads steers the car back to the middle of the road it will also retrieve the arrow for the upcoming intersection, and, if necessary the problem state.

### Navigation thread

Navigation starts with selecting a category: finding an active button and clicking it. If the task is easy, the model now perceives the stimulus and clicks the corresponding key. However, if the task is hard the model puts the to-be-entered information in its problem state and starts typing the first character. As soon as it clicks a button it starts searching for the next character of the word, and so on until the whole word has been entered.

It should be noted that both tasks are polite in the sense that they will only take over control when all resources are free, except for the problem state. There is one exception to this general rule: the driving task can request visual attention back immediately. This mimics real driving in the sense that when someone is paying attention for some time to entering information in a navigation device, at some point they will look back to check the state of the road, independent of whether they had finished entering all the information.

Whenever the model switches to the navigation task and notes that it is in a hard condition and does not have the right problem state, it will first request this from declarative memory, effectively pushing the problem state of the driving thread into declarative memory. Similarly, whenever the model switches to driving in the hard condition, it will restore the driving problem state.

## Results

A visual inspection of the data showed that all learning took place before the main phase of the experiment: there was no noticeable difference between the base level measurements before and after the experiment. Therefore the rest of this paper will only be concerned with the main two blocks of the experiment. All reported *F*- and *p*-values are from ANOVAs, all error bars depict standard errors.

### Task durations

The average duration of periods spent on one of the two subtasks can be seen in Figure 3 (driving sequence) and Figure 4 (navigation sequence). These durations are approximations, calculated in the following manner: the length of a driving sequence is defined as the time between two *navigation* actions (button clicks), with at least one driving action in between. Similarly, the length of a navigation period is the time between two *driving* actions with a navigation action in between.

**Driving** Figure 3 shows that the length of driving periods decreases when the navigation task becomes hard, *but only when driving is easy.* When navigation is hard, people know what they are going to type next ("philadel…"), which means that they do not have to find the stimulus first, but can start right away with entering navigation information.

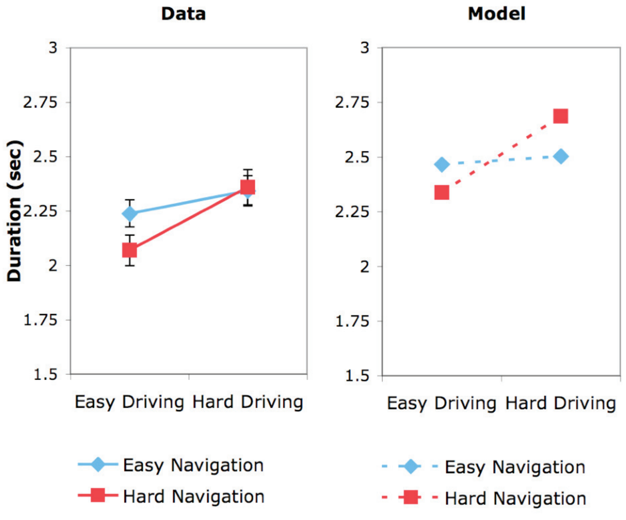


Figure 3. Duration of driving periods.

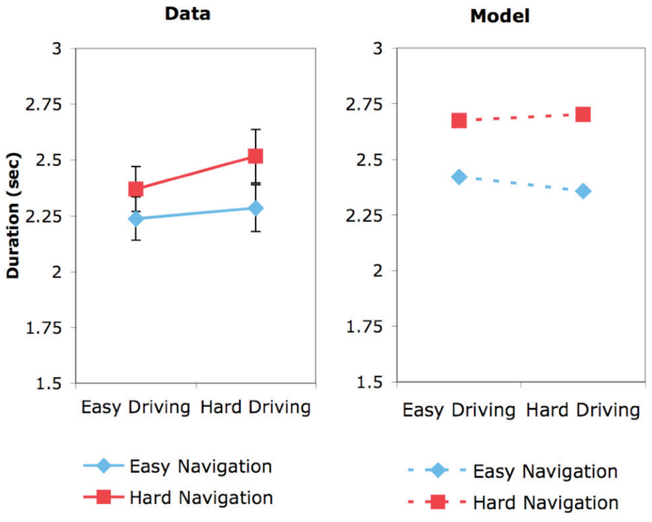


Figure 4. Duration of navigation periods.

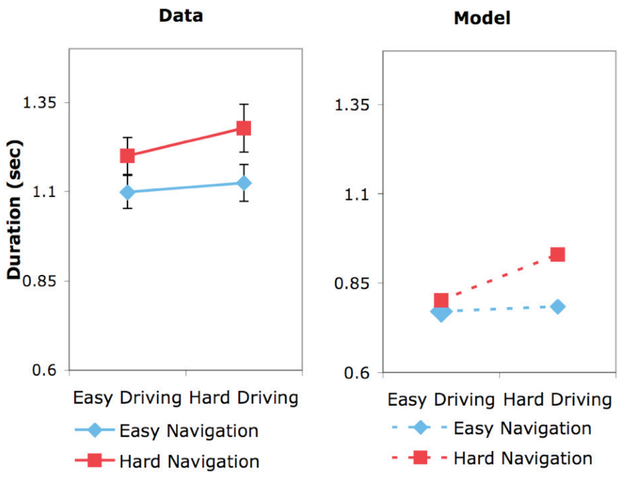


Figure 5. Duration of eye-tracking driving periods.

Because of the fact that the length of a driving sequence is measured as the time between two navigation actions with a driving action in between, the length of the driving sequence decreases when navigation becomes hard. However, this effect disappears when both navigation and driving are hard – it seems as if people have to reconstruct their problem state before they can start navigating, which increases the length of the driving periods. Overall can be seen that the length of driving periods increases with driving difficulty. An ANOVA showed indeed a main effect of driving (*F*(1,25) = 65.414, *p* < .001) and an interaction effect of driving x navigation (*F*(1,25) = 13.906, *p* < .001).

**Navigation** In Figure 4 can be seen that the duration of the navigation periods increases with task difficulty of navigation (*F*(1,25) = 16.755, *p* < .001). Driving has no significant effect on the length of the navigation periods, neither is there an interaction.

**Model** The model shows the same pattern as the experimental data: in the driving task (Figure 3, right panel) there is a significant interaction, while in the navigation task (Figure 4, right panel) there is no significant interaction. This is what threaded cognition predicted: there will be interference as soon as people have to keep track of a problem state in both tasks.

## Task durations measured with eye-tracking

Figure 5 again shows the duration of periods spent on the driving task, but now as measured by the eye-tracker. The length of a period is now determined by where a participant was looking: as long as participants were looking at the right side of the screen it was recorded as navigation, as long as they were looking at the left side as driving. These measurements are arguably more accurate than the ones before: periods without any key-presses or mouse clicks are taken into account as well. This explains why the average length of the periods is about a second shorter than what we saw earlier.

**Driving** Interestingly, instead of decreasing, the length of driving periods now increases with navigation difficulty (*F*(1,19) = 9.1367, *p* < .01). This can be explained by the fact that finding and reading the stimulus in the easy condition no longer contributes to the driving periods. The reason for the increase is probably that participants tried to finish parts of a word ("phi…"), before going back to driving, an effect that will not occur in the easy driving condition and will make for longer navigation periods. The longer participants spend on navigation, the longer they need to steer the car back to the middle of the road. There is also a significant effect of driving difficulty (*F*(1,19) = 14.455, *p* < .01). Besides these two main effects, we found an interaction effect of driving x navigation as well (*F*(1,19) = 14.931, *p* < .01). This could be explained by the fact that people have to reconstruct their problem state before entering navigation information, and this preparation is done while looking at the driving display, as people can still control the car in that case.

**Navigation** No significant effects were observed in the duration of the periods spent on navigation (no graph is shown).

**Model** The model showed the same patterns, it only predicted the duration of the driving periods to be about 250 ms shorter (Figure 5, right panel). On the other hand, the duration of the navigation periods is predicted correctly (2.25 sec), without effects of condition.

## Deviation

Due to space limitations we cannot show graphs of the average deviation of the middle of the road, but will describe it shortly. Deviation increases with task difficulty, this is both significant for driving, *F*(1,25) = 21.010, *p* < .001 and for navigation, *F*(1,25) = 18.967, *p* < .001. No interaction effect was found. The values range between 10 and 12 pixels.

However, the model shows an interaction effect. There is too much deviation in the easy driving/easy navigation condition. The duration of the navigation periods in the easy/easy condition is overestimated as well (Figure 4), and these two phenomena are connected. Because the model spends a little too much time in the easy/easy condition on navigation, it will also deviate further from the middle of the road. Furthermore, the model performed better than the participants, with deviation ranging between 6 and 8 pixels. However, about one third of our participants actually performed on that level, while some others were far worse than average. The model must be seen as a 'perfect' participant, in that it always manages to steer the car back to the middle of the road in exactly the right number of key-presses.

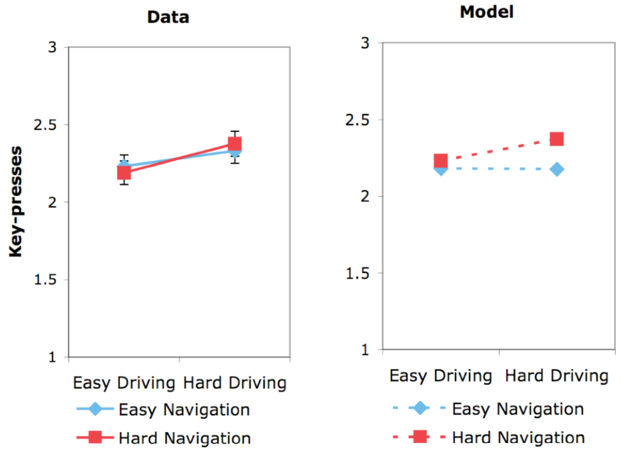


Figure 6. Mean number of key-presses per driving period.

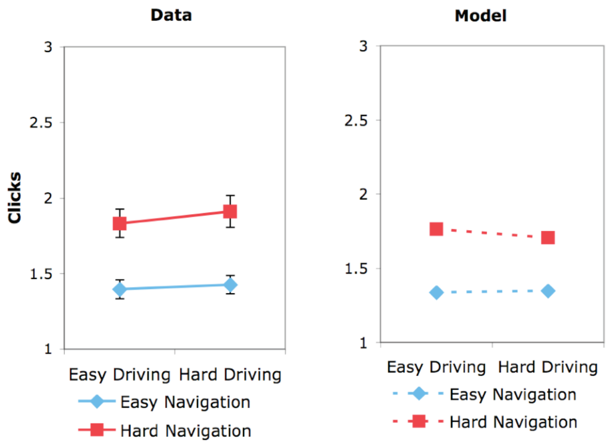


Figure 7. Mean number of clicks per navigation period.

### Number of clicks / key-presses per period

In Figure 7 and Figure 8 is respectively shown how many times participants pressed a key during a driving period, and how many times they clicked a button during a navigation period. There is only one significant effect on the number of key-presses, which is driving (F(1,25) = 13.475, p < .01). The opposite is true for the number of clicks during a navigation period, this gives a highly significant effect of navigation (F(1,25) = 229.54, p < .001), and only a marginal effect of driving (F(1,25) = 6.070, p = .02).

The model shows the same effects, as can be seen in the right panels of both figures.

## Discussion & Conclusion

As explained above, threaded cognition predicts an extra drop in performance when it is necessary to keep track of a problem state for two tasks. The results of the experiment clearly showed that this is in fact the case: we found a significant interaction effect in the length of driving periods. By modeling this task in ACT-R with threaded cognition, we showed exactly why these costs are connected to the driving task: the preparation of a problem state for both the driving and the navigation task is done while driving, and the need to reconstruct a problem state therefore increases the duration of driving periods. Eye-tracking measurements confirmed those results.

Threaded cognition is one of the first theories of multitasking without a control structure to interleave the subtasks. Salvucci & Taatgen (under revision) showed the value of this theory in a multitude of task combinations, they validated the theory on tasks ranging from simple laboratory tasks to real-world tasks, and showed the effects of sharing perceptual and memory. In the current paper we investigated whether threaded cognition can account for the costs of sharing another internal resource: the problem state. As we have made clear, threaded cognition predicted correctly in which conditions we had to expect extra costs of sharing a problem state.

## Acknowledgments

This work was supported by Office of Naval Research grant N00014-06-1-005. We would like to thank Dario Salvucci for comments on an early version of the experiment.

## References

Anderson, J.R., Bothell, D., Byrne, M.D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111*(4), 1036-1060.

Kieras, D.E., Meyer, D.E., Ballas, J.A., & Lauber, E.J. (2000). Modern computational perspectives on executive mental processes and cognitive control: Where to from here? In S. Monsell & J. Driver (Eds.), *Control of cognitive processes* (pp. 681-712). Cambridge, MA: MIT Press.

Liu, Y., Feyen, R., & Tsimhoni, O. (2006). Queueing network-model human processor (qn-mhp): A computational architecture for multitask performance in human-machine systems. *ACM Transactions on Computer-Human Interaction (TOCHI), 13*(1), 37-70.

Salvucci, D.D. (2005). A multitasking general executive for compound continuous tasks. *Cognitive Science, 29*, 457-492.

Salvucci, D.D., & Taatgen, N.A. (under revision). Threaded cognition: An integrated theory of concurrent multitasking. *Psychological Review*.

Salvucci, D.D., Taatgen, N.A., & Kushleyeva, Y. (2006). Learning when to switch tasks in a dynamic multitasking environment. In D. Fum, F. D. Missier & A. Stocco (Eds.), *Proceedings of the seventh international conference on cognitive modeling* (pp. 268-273). Trieste, Italy: Edizioni Goliardiche.

Taatgen, N.A., Van Rijn, D.H., & Anderson, J.R. (in press). An integrated theory of prospective time interval estimation: The role of cognition, attention and learning. *Psychological Review*.

# Goal and Spatial Memory Following Interruption

**Michel E. Brudzinski (mbrudzin@gmu.edu)**
**Raj M. Ratwani (rratwani@gmu.edu)**
George Mason University, Fairfax, VA 22031 USA

**J. Gregory Trafton (trafton@itd.nrl.navy.mil)**
Naval Research Laboratory
Washington, DC 20375 USA

## Abstract

The process of resuming an interrupted task has been understood by task level goals (Altmann & Trafton, 2002). Recent empirical evidence has implicated spatial memory as a component of the resumption process suggesting that spatial level representations are important as well. We collected eye track data in an interruptions paradigm to examine the perceptual processes involved in resumption. Four models were created to illustrate the importance of the role of spatial representations and further, to demonstrate how the task level and spatial representations can be integrated.

**Keywords:** Goals; Interruptions; Cognitive Modeling

## Introduction

Most computer-based tasks are described in terms of the tasks and goals that are needed to perform them. There are, in fact, many task analytic and computational methods for describing tasks and goals for computer based tasks (e.g., various GOMS methods). Another aspect of computer-related tasks that has typically received less attention is the spatial location of widgets within the task. In this paper, we explore the relationship between pure goal-based representations with spatially-motivated representations within an interruption paradigm.

Altmann and Trafton (2002) described the process of resuming a suspended goal. They proposed the activation-based Memory for Goals theory, which includes three constraints: (1) the interference level, (2) the strengthening constraint, (3) and the priming constraint. The constraints determine what goal will be most active in memory at any given time. The Memory for Goals theory proposes that the question "What was I doing" is cognitively equivalent to the retrieval of the highest activation goal memory. In general, the memory for goals theory focuses on the memory representations and processes that occur while resuming a goal and leaves unspecified any influence of spatial cognition.

Later research suggested that memory for goals was associated with at least a general memory for spatial location (Ratwani & Trafton, 2006) in simple computer-based tasks. Determining spatially where in the primary task one was prior to being interrupted was an important component when resuming an interrupting task.

Based on the memory for goals theory and additional empirical data, Ratwani and Trafton suggested several different strategies for how people resume an interrupted task: (1) restart the (sub-)task, (2) use an environmental cue, (3) and use a spatial memory for the location.

Ratwani and Trafton's results showed that participants resumed their task by perceptually retracing their steps they looked at what they had already accomplished, and then continued. The task that was used (described below) allowed Ratwani and Trafton to separate pure perceptual retracing and a spatial component. They found that spatial memory (specifically memory for spatial location) was being used as part of the resumption process after an interruption. The data showed that spatial memory was an important part of resuming a computer-based task after an interruption, but it did not specify the exact mechanisms or representations of the spatial memory or the relationship between spatial and task-related goal memory.

The different strategies for resuming an interrupted task rely on different aspects of the environment and memory. One obvious strategy is to retrace one's steps from the beginning of the task until the point where one was interrupted is reached. This is essentially a restart strategy and there is some direct evidence for restarting a suspended goal in the interruptions literature (Miller, 2002). This is an example of a more general strategy of using the structure of the task to determine where to resume the task.

A second possibility for resuming a task is that participants may use some type of environmental cue to resume the primary task (Altmann & Trafton, 2002; Trafton, Altmann, & Brock, 2005). In the task used in the experiment described here, there was an environmental cue (described below) that provided accurate information about what actions had been performed prior to the interruption, but less accurate information about the spatial location where the interruption had occurred.

Finally, memory for spatial location might be used to return to where one was in a task prior to being interrupted. Recent research in the visual search domain has shown that memory representations for the location of objects can be maintained over a delay (Lleras, Rensink & Enns, 2005). Further, the ability to remember approximate spatial information has been observed in computer based tasks (Ehret, 2002). Memory for spatial location may be represented at two levels: a fine grained level whereby one may be able to recall the precise location of an object as well as a more general category level whereby one can identify the region which contained an object (Huttenlocher, Hedges, & Duncan, 1991). While Ratwani and Trafton (2006) suggested that a memory for approximate spatial location was used to resume after an interruption, the specific process of this mechanism has not been elaborated. The spatial memory strategy requires that there is an association between the memory of what the interrupted goal was, and spatially where the interruption occurred

The ACT-R cognitive architecture has been used to model experiments in a number of psychological domains. ACT-R 6.0 is the latest software implementation of the ACT family of theories of cognition (Anderson & Lebiere, 1998; Anderson et al. 2004). The ACT-R theory distinguishes between declarative knowledge, which people are aware of and can describe to others, and procedural knowledge, which may be unconscious but can be demonstrated in behavior. Declarative knowledge is represented in the architecture as chunks with pairs of slot and values, while procedural knowledge is represented by production rules with sets of conditions and actions. ACT-R's perceptual-motor modules allow models to interact with computer interfaces.

A goal module represents current cognitive intentions and helps to organize and direct behavior towards the fulfillment of those intentions. Cognition unfolds within ACT-R as the serial firing of production rules that manipulate or make requests for chunks from modules through dedicated buffers. ACT-R allows researchers to model cognitive processes and collect quantitative measures that can be compared directly with quantitative measures of human performance.

In order to explore the role of spatial memory after an interruption and to explore the relationship between spatial and task-related memory, we improved the methodology used by Ratwani and Trafton (2006). We also built ACT-R models of the three strategies that could be used to resume an interrupted task (restart, environmental cue, and spatial memory). We expect that the experiment itself will replicate Ratwani and Trafton's earlier finding that spatial memory is implicated in the resumption task, and we expect the model that uses spatial memory to show the best fit to the data. The models will also allow us to explore which of several cognitively plausible spatial representations are most likely being used in computer-based tasks.

## Experimental Method

### Participants

Nineteen George Mason University undergraduate students participated for course credit.

### Materials

The primary task materials consisted of columns of numbers; each column contained 11 three digit numbers ranging from 100-999. Fifteen unique templates containing slots specifying which numbers were to be even or odd and the location of these numbers were used to generate the columns of numbers, each template had at least five odd numbers. Based on the templates, two sets of 15 columns of numbers were created for presentation. The specific numbers that filled the slots in the template were randomly generated for each participant. Each number subtended .6° of visual angle, each cell subtended 2.9° and each number was separated by 2.3° of visual angle.

The interrupting task was a list of 10 addition problems each containing four single digit addends ranging from 1-9. The addends were randomly generated for each interruption.

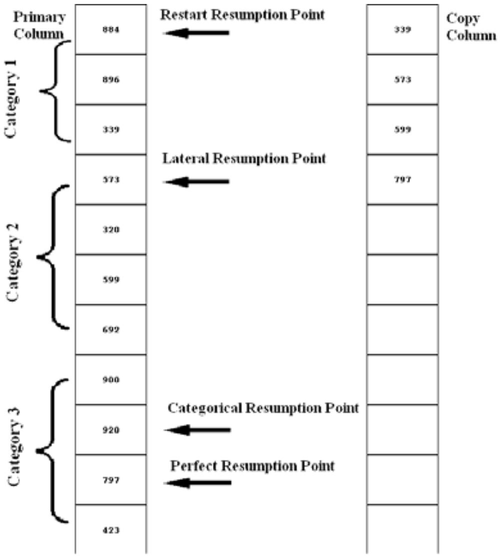


Figure 1. The experiment primary task.

### Design

A within subjects design was used; one set of 15 columns served as interruption trials and one set as control trials resulting in a total of 30 trials per participant. This allowed for matched trials between the two conditions. Presentation order of the all the trials was randomized. Each interruption trial contained a single interruption which occurred equally among three positions in the task (early, middle, and late).

### Procedure

Participants were seated 50 cm from the monitor. Stimuli were presented using E-Prime (Schneider, Eschman, & Zuccolotto, 2002). The primary task required participants to type the odd numbers from the primary column into a separate copy column (see Figure 1). Because only the odd numbers from the primary column were typed into the copy column the vertical position of the odd numbers in the copy column was generally different from their corresponding position in the primary column. The participants were instructed to start at the top of primary column and to work their way down to the bottom. Upon completion of the column the participant pressed the space bar in order to move on the next trial.

On the interruption trials the interrupting task immediately appeared and fully occluded the primary task screen. During the 15 second interruption participants were instructed to answer as many addition problems as possible. Upon resumption of the primary task, the copy column of previous responses was still displayed. The location of the last number entered in the copy column could serve as a cue as to where to resume in the primary column. However, as discussed above, because the vertical positions of the odd numbers in the primary and copy column may have been different, this position in the primary column may not be where the interruption occurred.

### Measures

Based on the reaction time data we calculated an inter-action interval (IAI) for control trials and a resumption lag for interruption trials (Altmann & Trafton 2004). The IAI was the average amount of time between actions (i.e. the average amount of time in between entering odd numbers). The resumption lag was the duration of time from the completion of the interrupting task to the first action back on the primary task (e.g. entering an odd number). Eye track were collected using a Tobii 1750 operating at 60hz. Each of the cells in the original and copy columns was defined as an area of interest. A fixation was defined as five samples.

## Model Descriptions

In order to examine the process of resuming a primary task following a secondary task interruption, we constructed a series of models using the ACT-R 6.0 cognitive architecture (Anderson & Lebiere, 1998; Anderson et al. 2004). These models systematically explored the three high-level strategies identified by Ratwani and Trafton (2006): (1) restarting the task, (2) using the spatial location of an environmental cue, (3) and using a spatial memory for the location of the interruption. Four models were selected as implementations of the high-level strategies. One model each represented the restart and lateral strategies, and two models represented spatial memory strategies, one for general location and one for specific location.

Although the models and the experimental participants did not use the exact same task environment, they shared all the critical features. The model task environment was written in LISP. The pixel coordinates of the stimuli in the models' visual environment (the visicon in ACT-R) matched the coordinates of the stimuli in the experimental task environment. The 15 patterns of even and odd three-digit numbers, each used once in a control trial, and once in an interrupted trial, were the same. The order of trial presentation, and the exact even and odd numbers used, were randomized, as in the experiment.

### Commonalities Between Models

The productions that modeled performance in the primary and secondary task were identical in all of the models. Additionally, approximately two thirds of the resumption task productions were common to all of the models. Table 1 lists the steps that occur between the end of the interruption and the end of the resumption. The task resumption process was broken into three parts for all models: (1) cue use, (2) search, and (3) primary task resumption.

Table 1: Breakdown of Overall Resumption Process

| Step | Breakdown Component |
|---|---|
| End of Interruption | |
| 1. Determine task | Cue Use |
| 2. Find cue | |
| 3. Retrieve goal | |
| 4. Determine resumption point | |
| 5. Find resumption point | Search |
| 6. Find interruption point | |
| 7. Find next primary task | Primary Resumption |
| 8. Next primary task action | |
| End of Resumption | |

The cue use portion of the resumption breakdown (Table 1, steps 1-4) incorporated the use of the environmental context to determine what task to resume and where to resume that task. In the experiment, the sudden onset of the interruption caused participants to suspend the goal of completing the primary task, entering odd numbers, and begin the secondary task, simple addition problems. Model subjects determined that the task had changed, what the new task was, and where to begin that task. In the models, the change to the visual environment (the visicon in ACT-R) caused the models to change the active goal from the odd numbers task goal to the addition task goal.

Ratwani and Trafton (2006) found that in nearly every resumption case, participants looked at the cue (i.e. the last number entered) prior to looking at the primary column of numbers. Likewise, in all of the models, the cue use portion of the resumption process involved attending to the last input odd number in the copy column and retrieving related chunks from declarative memory (see Figure 1). As suggested by Altmann and Trafton (2002), all models then retrieved the interrupted primary task goal. Different models used different strategies to determine an initial visual-location in the primary column, called the resumption point.

In the experiment, participants had to resume the primary task of entering odd numbers following the interruption. To do this, participants had to find the location in the primary column of the last odd number they entered, called the interruption point. In all of the models, the search portion of the resumption process involved finding the location of the interruption point (Table 1, steps 5-6). Exactly where the model started searching, the resumption point, was a major difference between models. The search proceeded from the resumption point to the interruption point by searching down the primary column in all of the models.

In the experiment, the end of the resumption process was demarcated by the first key-press following the interruption. The first key-press was the first digit of the next odd number in the primary column. In the models, the primary task resumption portion of the resumption process (Table 1, steps 7-8), involved finding and entering the next odd number in the primary column. This process was exactly the same in all of the models and was exactly the same as the primary task.

All of the models used the default ACT-R 6 parameter settings, except for the maximum associative strength (mas) parameter. The mas parameter controls the amount of spreading activation from the chunk representing the environmental cue to the chunk representing the primary task goal. The mas value of 15 was used in all models. This value is slightly higher than other ACT-R models because we were attempting to implement the priming constraint from Altmann and Trafton (2002).

### Differences Between Models

The models differed primarily in the process of determining the resumption point (Table 1, step 4), the starting point of the search for the interruption point. The models: (1) restart, (2) lateral, (3) perfect spatial memory, and (4) categorical spatial memory, simulated different strategies for using information, from the task environment, and from memory, to determine the resumption point. These strategies resemble the high level strategies outlined by Ratwani and Trafton (2006).

The restart model always used the top of the primary column as the resumption point. This represents the high-level strategy of using the structure of the task, in this case the spatial location of the first sub-task, without any memory for the spatial location of the interruption, to determine the initial location in the primary column.

The lateral model always moved laterally from the copy column to the primary column. The resumption point was the number in the primary column at the same vertical position as the cue in the copy column. This represents the high-level strategy of using the spatial location of the cue, without any memory for the spatial location of the interruption, to determine the initial location in the primary column.

Spatial memory strategies use memory for the spatial location of the interrupted goal to determine the resumption point. These strategies used the cue to prime the retrievals of the goal that produced the cue, and the spatial location associated with the goal. Two models represented this category of strategies. One represented perfect memory for the exact visual location of the interruption point; the other represented general memory for the categorical location of the interruption point (Huttenlocher, Hedges, & Duncan, 1991).

The categorical spatial memory model used memory of the categorical spatial location of the interrupted goal to determine the resumption point. The screen was divided into 3 approximately equal spatial categories: top, middle and bottom. The resumption point was set to be the middle of each spatial category. Depending upon the location of the interruption point within the spatial category, the resumption point was either above, below, or the same as the interruption point. The search direction switched from down the column to up the column when the bottom of the column, or the next category center was reached. [1]

## Experimental Results and Discussion

The resumption lag (m = 3755.8) was significantly longer than the inter-action interval (m = 1740.7), F(1,18) = 96.8, MSE = 381715.9, p<.001, showing that the interruptions were disruptive to primary task performance.

The eye track data were examined to explore the perceptual and spatial processes that people used as they resumed the primary task after the interruption. The focus was on the location of fixations to the primary column of numbers during the resumption lag. If participants were starting the task over again, their first fixation to the primary column should always be to the top of the primary column. In 99% of the interruption trials, participants fixated somewhere other than the top of the primary column, suggesting that participants were not starting the task over after the interruption.

Next, we examined whether participants were relying strictly on a cue to resume the primary task. Participants consistently (~99% of the time) looked to the number they last entered in the copy column immediately upon resumption of the primary task. The location of the last entered number could be used as a cue to guide them back to where they left off in the primary column. For example, participants could fixate on the cue to determine the number they last entered and then saccade directly across to the corresponding position in the primary column and continue from that point. If participants were relying on the cue to resume, the first fixation in the primary column after the interruption should be to the same cell number as the location of the cue (i.e. if the cue was in cell 6, one could resume at cell 6 in the primary column). If participants were relying strictly on the cue the average location of the cue should be the same as the average location of the first fixation on the primary column. The average cue location (m = 3.3) was significantly different from the average location of the first fixation to the primary task (m = 4.9), F (1,18) = 56.7, MSE = .418, p<.001. Thus, participants did not rely strictly on the cue to resume the primary task. Based on these results, participants did not seem to be using either a restart strategy or using the cue as a position marker. We next explore experimental evidence that participants were using a spatial strategy.

In order to examine how accurate participants were at returning to where they left off, the initial fixation to the primary column after the interruption was compared to the cell location of the number that was last entered prior to the interruption. For example, if the interruption occurred at cell 6 and the participant returned to cell 4 this distance was calculated as -2. This difference was calculated for each

[1] We created a series of models that contained different instantiations of the models – different numbers of categories, top, or middle restarts, etc. We are only presenting the best fitting models in all cases.

interruption trial for each participant to determine how close participants were able to resume. A distribution of these values showed that participants were able to return to within 2 cells of where they left off in over 60% of the cases. This strongly suggests that participants were using some kind of spatial memory to resume. Note that these results replicate the earlier Ratwani and Trafton (2006) finding that people seemed to be using some sort of spatial memory to facilitate resumption of the primary task.

What is clear from this data is that participants are using some type of spatial memory to help them resume the primary task. What is less clear is the type of and representation of spatial memory that participants are using. Our goal in modeling this task was thus twofold: (1) To build an explicit model of resumption for a simple task and (2) To explore different types of plausible spatial representations and strategies to better understand the nature of spatial cognition both within an interruption task and spatial cognition more generally. To the extent that a specific spatial model fits the pattern of experimental data well, it would provide support that people are using that type of spatial representation and spatial strategy during the resumption process.

## Model Results and Discussion

The results from the model simulations were compared to the experimental data. By comparing the overall resumption lags in Table 2, a general sense of which model more closely fits the experimental data can be observed.

Table 2. Resumption Lag experimental and model data with model fit statistics.

| | Total RL | Cue Use | Search | Primary Resumption | $R^2$ | RMSD |
|---|---|---|---|---|---|---|
| Experiment | 3.76 | 0.79 | 1.46 | 1.60 | - | - |
| Restart | 6.07 | 0.77 | 3.61 | 1.69 | .40 | 1.24 |
| Lateral | 4.64 | 0.77 | 2.19 | 1.68 | .75 | 0.42 |
| Perfect | 2.97 | 0.99 | 0.30 | 1.68 | .02 | 0.69 |
| Categorical | 3.70 | 0.83 | 1.24 | 1.63 | .88 | 0.13 |

Note: The total resumption time was not included in model fit statistics since individual components were. Model and data fits based on (Schunn & Wallach, 2001).

The true differences between the models can be seen by comparing the resumption lag components. The overall experimental resumption lag was broken down into these components as well in order to compare the models to the data. The cue use time reflects the amount of time spent looking at the last number entered in the copy column. Search time reflects the amount of time used to find where one last left off in the primary task prior to the interruption and the primary resumption is the amount of time until the next odd number is entered once one is back on track (i.e. they have found where they left off). The fit statistics comparing the models to the data are based on these three components. Notice there is little variability between models in terms of cue use and primary resumption because the models use the same productions for those processes; differences are accounted for by random noise in the model. The search time is the critical component that differentiates between the models. The point at which the model first attends to the primary column upon resumption is the driving factor behind the search time and the process for how each model determines this resumption point is different for each model. The 4 models will be discussed in turn, focusing on the non-spatial models first, then moving to the spatial models.

### Non-Spatial Models

The restart strategy results in the longest search time since the model always attends to the top of the primary column. This search time also has the largest deviation from the experimental data amongst all of the models. Empirically, the restart model is the furthest from the experimental data and results in an approximately 2 second longer resumption lag. This model shows that people are not consistently starting over their sub-task.

The lateral model produces a fit with data that is much better than the restart-strategy model (see Table 2). This model relies strictly on the cue and attends to the cell in the primary column that is directly across from the location of the cue. This model performs well for interruptions that occur early in the primary column of numbers since the correct resumption point in the primary column is near the location of the cue. However, for interruptions which occur later in the task (e.g. cell 8), returning to where the cue is may result in a rather long search time. Consequently, this model has a relatively decent fit to the data, but does not have the best fit because of the increased search time when the primary resumption point is not near the cue. Note that in our current experiment the cue moved spatially – it progressed as people entered numbers. In an unreported follow-up experiment, participants entered a cue that did not move at all. In this case, the lateral strategy is identical to the restart strategy. We are currently running our current models on the “stable” cue experiment.

### Spatial Models

The perfect spatial memory model perfectly remembers the spatial position before the interruption and returns there upon resumption. Not surprisingly, it has the shortest search time and the fastest resumption lag. This model suggests that people do not use perfect spatial memory to resume an interrupted task: their spatial location memory is approximate at best. So what type of imprecise spatial memory do people have?

We instantiated a categorical spatial memory model that divides the primary column of numbers in to three general regions of space (i.e. top, middle and bottom). The categorical model comes closest to matching the experimental data, as seen in Figure 2. This model returns to the center of the spatial category, but still has to search within the category to find the specific resumption point. This within category search produces a much smaller search time relative to the other models and search time that is much closer to the experimental data.

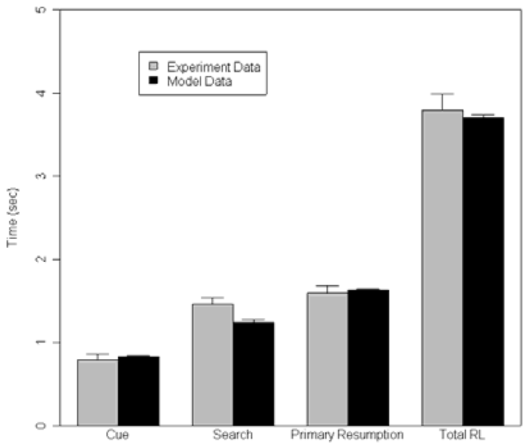


Figure 2. Experiment and categorical spatial memory model comparison.

While the spatial category model is closest to matching the experimental data in regards to the resumption lag breakdown, this gives no indication of how well the specific resumption points from the model match the actual experimental data. We generated a distribution of resumption point differences for the model just as we did for the experimental data. Differences at this level pinpoint some of the weaknesses in the spatial category model. First, the distribution based on the model is narrower than the distribution based on the experimental data. The experimental data showed that participants were able to return to within 2 cells following the interruption in over 60% of the cases; the model returns to within 2 cells in 100% of the cases. Since the model always resumes at a point within the category that it first attends to, the range of resumption points is limited by the size of the category. Thus, the inaccuracies of the model are fixed by the boundaries of these categories. Second, the experimental data is centered on -1 while the model resumes at a point centered on 0, which is perfect resumption. In the experimental data, participants tend be more conservative in where they resume, generally resuming a few cells back from where they left off. Participants may be biased towards a conservative resumption because of the relatively high cost of resuming ahead of where they once were (i.e. liberal resumption). A liberal resumption is costly in terms of search time for participants because they spend time searching ahead of where they were and prior to being interrupted and then have to search backwards. The model resumes at the center of the category with no bias towards a conservative or liberal resumption.

## General Discussion

This paper examined the relationship between task memory and spatial memory by examining the role of spatial location information in an interrupted task. Our experiment showed that spatial memory is implicated in resuming a computer-based task. Our spatial category model showed that both task memory and spatial memory are implicated in resuming a computer-based task. Additionally, our model showed that spatial location memory is approximate, not exact.

From a subgoal-resumption point of view, our model and data also suggest that the memory for goals model should be modified to include a spatial location component. Note, however, that our model does not suggest that spatial location is embedded into every goal. Rather, it suggests that spatial location can be retrieved through associative activation when needed.

Human performance may involve a combination of strategies, rather than the consistent application of a single strategy for resuming an interrupted task. The selection of a particular strategy may not be random, but may depend on aspects of the environmental context of the interruption. One way to improve this model would be to combine several of the strategies that were used in a pure sense in this report.

## Acknowledgments

This work was supported by a grant from the Office of Naval Research to J. Gregory Trafton.

## References

Altmann, E. M., & Trafton, J. G. (2002). Memory for goals: An activation-based model. *Cognitive Science*, 26, 39-83.

Altmann, E. M., & Trafton, J. G. (2004). Task interruption: Disruptive effects and the role of cues. *The proceedings of the twenty-sixth annual conference of the cognitive science society.*

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An Integrated theory of the mind. *Psychological Review* 11, (4). 1036-1060.

Anderson, J. R. & Lebiere, C. (Eds.). (1998). The atomic components of thought. Hillsdale, NJ: Erlbaum.

Ehret, B. (2002). Learning where to look: location learning in graphical user interfaces. *In Conference on Human Factors in Computing Systems Proceedings of the SIGCHI conference on Human factors in computing Systems.*

Huttenlocher, J., Hedges, L.V., & Duncan, S. (1991) Categories and Particulars: Prototype Effects in Estimating Spatial Location. *Psychological Review*, 98, pp. 352-376.

Lleras, A., Rensink, R. A., & Enns, J. T. (2005). Rapid resumption of interrupted visual search: new insights on the interaction between vision and memory. *Psychological Science*, 16, 684-688.

Miller, S. (2002) Window of opportunity: using the interruption lag to manage disruption in complex tasks. *In Proceedings of the 46th Annual Meeting of the Human Factors and Ergonomics Society.*

Ratwani, R. M., & Trafton, J. G., (2006) Now, where was I? Examining the Perceptual Processes while Resuming an Interrupted Task. *The Proceedings of the Twenty-Eighth Annual Conference of the Cognitive Science Society.*

Schneider, W., Eschman, A., & Zuccolotto, A. (2002). *E-prime user's guide*. Pittsburgh: Psychology Software Tools.

Schunn, C. and Wallach, D. (2001). Evaluating goodness of fit in comparison of models to data. Online manuscript available at http://www.lrdc.pitt.edu/schunn/gof/index.html

Trafton, J. G., Altmann, E. M., & Brock, D. P. (2005). Huh, what was I doing? How people use environmental cues after an interruption. *Proceedings of the Human Factors and Ergonomics Society 49th Annual Meeting.*

# Learning Cognitive Load Models for Developing Team Shared Mental Models

**Xiaocong Fan (XFAN@Psu.Edu)**
School of Engineering, The Behrend College
The Pennsylvania State University, Erie, PA 16563 USA

**Po-Chun Chen (PZC123@Psu.Edu)**
Department of Computer Science and Engineering
The Pennsylvania State University, University Park, PA 16802 USA

**John Yen (JUY1@Psu.Edu)**
College of Information Sciences and Technology
The Pennsylvania State University, University Park, PA 16802 USA

## Abstract

Cognitive studies indicate that members of a high performing team often develop shared mental models to predict others' needs and coordinate their behaviors. The concept of shared mental models is especially useful in the study of human-centered collaborative systems that require humans to team with autonomous agents in complex activities. We take the position that in a mixed human/agent team, agents empowered with cognitive load models of human team members can help humans develop better shared mental models. In this paper, we focus on the development of realistic cognitive load models. Cognitive experiments were conducted in team contexts to collect data about the observable secondary task performance of human participants. The data were used to train hidden Markov models (HMM) with varied number of hypothetical hidden states. The results indicate that the model spaces have a three-layer structure. Statistical analysis reveals some characteristics of top layer models, which can be used in guiding the selection of HMM-based cognitive load models.

## Introduction

Human-centered multi-agent teamwork has attracted increasing attentions in multi-agent systems field (Bradshaw et al., 2002; Norling, 2004). Human-centered teamwork, involving both humans and software agents, is about collaboratively establishing situation awareness, developing shared mental models as situation evolves, and appropriately adapting to mixed-initiative activities. Humans and autonomous agents are generally thought to be complementary: while humans are limited by their cognitive capacity in information processing, they are superior in spatial, heuristic, and analogical reasoning; autonomous agents can continuously learn expertise and tacit problem-solving knowledge from humans to improve system performance.

However, the foundation of human-agent collaboration keeps being challenged because of nonrealistic modeling of mutual awareness of the state of affairs. In particular, few researchers look beyond to assess the principles of modeling shared mental constructs between a human and his/her assisting agent. Moreover, human-agent relationships can go beyond partners to teams (Fan, Yen, & Volz, 2005). Many informational processing limitations of individuals can be alleviated by having a group perform tasks. Although groups also can create additional costs centered on communication, resolution of conflict, and social acceptance, it is suggested that such limitations can be overcome if people have *shared cognitive structures* for interpreting task and social requirements (Lord & Maher, 1990). Therefore, there is a clear demand for investigations to broaden and deepen our understanding on the principles of shared mental modeling among members of a mixed human-agent team.

There are lines of research on multi-agent teamwork, both theoretically and empirically. For instance, Joint Intention (Cohen & Levesque, 1991) and SharedPlans (Grosz & Kraus, 1996) are two theoretical frameworks for specifying agent collaborations. One of the drawbacks is that, although both have a deep philosophical and cognitive root, they do not explicitly accommodate the modeling of human team members. Cognitive studies suggested that teams which have shared mental models are expected to have common expectations of the task and team, which allow them to predict the behavior and resource needs of team members more accurately (Rouse, Cannon-Bowers, & Salas, 1992; Klimoski & Mohammed, 1994). Cannon-Bowers et al. (Rouse et al., 1992) explicitly argue that team members should hold compatible models that lead to common "expectations". We agree on this and believe that the establishment of *shared mental models* among human and agent team members is a critical step to advance human-centered teamwork research.

The long-term goal of our research is to understand how shared cognitive structures can enhance human-agent team performance. In particular, we argue that to favor human-agent collaboration, an agent system should be designed to allow the estimation and prediction of human teammates' (relative) cognitive loads, and use that to offer improvised, unintrusive help. Ideally, being able to predict the cognitive/processing capacity curves of teammates could allow team members to help *the right party at the right time* (Fan & Yen, 2007), avoiding unbalanced work/cognitive loads among the team.

The specific objective of the work reported here is to develop a computational cognitive capacity model to facilitate the establishment of shared mental models. The rest of the paper is organized as follows. In Section 2 we review studies on cognitive load and its measurements. Section 3 gives our motivation of using HMM-based approach to modeling human cognitive loads. Section 4 describes the cognitive task design and the experiment conducted to collect observable measures of secondary task performance in a team context. Section 5 reports the methodology of learning Hidden Markov Models (HMM) and the principles of selecting appropriate HMM models for an agent to estimate its human partner's dynamic cognitive load.

## Background on Cognitive Load Studies

People are information processors. Cognitive load studies (Miller, 1956; Lord & Maher, 1990; Baddeley, 1992) are, by and large, concerned about working memory capacity and how to circumvent its limitations in human problem-solving activities such as learning and decision making.

According to the cognitive load theory (Paas & Merrienboer, 1993), *cognitive load* is defined as a multidimensional construct representing the load that a particular task imposes on the performer. It has a causal dimension including causal factors that can be characteristics of the subject (e.g. expertise level), the task (e.g. task complexity, time pressure), the environment (e.g. noise), and their mutual relations. It also has an assessment dimension reflecting the measurable concepts of mental load (imposed exclusively by the task and environmental demands), mental effort (the cognitive capacity actually allocated to the task), and performance.

Lang's information-processing model (Lang, 2000) consists of three major processes: encoding, storage, and retrieval. The encoding process selectively maps messages in sensory stores that are relevant to a person's goals into working memory; the storage process consolidates the newly encoded information into chunks, forming associations and schema to facilitate subsequent recalls; the retrieval process searches the associated memory network for a specific element/schema and reactivates it into working memory. The model suggests that processing resources (cognitive capacity) are independently allocated to the three processes. In addition, working memory is used both for holding and for processing information (Baddeley, 1992). Due to limited capacity, when greater effort is required to process information, less capacity remains for the storage of information. Hence, the allocation of the limited cognitive resources has to be balanced in order to enhance human performance. This comes to the issue of measuring cognitive load, which has proven difficult for cognitive scientists.

Cognitive load can be assessed by measuring mental load, mental effort, and performance using rating scales, psychophysiological, and secondary task techniques (Paas, Tuovinen, Tabbers, & Gerven, 2003). Self-ratings may appear questionable and restricted, especially when instantaneous load needs to be measured over time. Although physiological measures are sometimes highly sensitive for tracking fluctuating levels of cognitive load, costs and work place conditions often favor task- and performance-based techniques, which involve the measure of a secondary task as well as the primary task under consideration. Secondary task techniques are based on the assumption that performance on a secondary task reflects the level of cognitive load imposed by a primary task (Sweller, 1988). From the resource allocation perspective, assuming a fixed cognitive capacity, any increase in cognitive resources required by the primary task must inevitably decrease resources available for the secondary task (Lang, 2000). Consequently, performance in a secondary task deteriorates as the difficulty or priority of the primary task increases. The level of cognitive load can thus be manifested by the secondary task performance: the subject is getting overloaded if the secondary task performance drops.

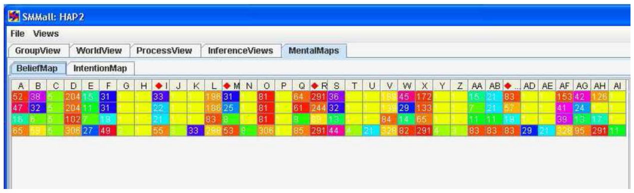
A secondary task can be as simple as detecting a visual or auditory signal but requires sustained attention. Its performance can be measured in terms of reaction time, accuracy, and error rate. However, one important drawback of secondary task performance, as noted by Paas (Paas et al., 2003), is that it can interfere considerably with the primary task (competing for limited capacity), especially when the primary task is complex. To better understand and measure cognitive load, Xie and Salvendy (2000) introduced a conceptual framework, which distinguishes instantaneous load, peak load, accumulated load, average load, and overall load. It seems that the notation of instantaneous load, which represents the dynamics of cognitive load over time, is especially useful for monitoring the fluctuation trend so that free capacity can be exploited at the most appropriate time to enhance the overall performance in human-agent collaborations.

## Modeling Cognitive Loads Using HMM

A hidden Markov model (HMM) (Rabiner, 1989) is a statistical approach to modeling systems that can be viewed as a Markov process with unknown hidden parameters. The hidden state variables are not directly visible, but influenced by certain observable variables. Each hidden state has a probability distribution over the possible observable symbols. Therefore the sequence of observable states can be used to make inference on the sequence of hidden states of a HMM. A HMM is denoted by $\lambda = \langle N, V, A, B, \pi \rangle$, where $N$ is a set of hidden states, $V$ is a set of observation symbols, $A$ is a set of state transition probability distributions, $B$ is a set of observation symbol probability distributions (one for each hidden state), and $\pi$ is the initial state distribution. Hidden Markov models have been widely applied in bioinformatics and temporal pattern recognition (such as speech, handwriting, and gesture recognition).

An intelligent agent being a cognitive aid, it is desirable that the model of its human partner implemented within the agent is *cognitively-acceptable*, if not descriptively accurate. However, building a cognitive load model that is *cognitively-acceptable* is not trivial. A HMM-based approach is used in this study for several reasons. First, cognitive load has a dynamic nature. As we mentioned above, being able to monitor the dynamics of a human partner's cognitive load over time is very useful for an agent to proactively identify collaboration opportunities in human-centered teamwork. The inference of the instantaneous cognitive load can be cast as a temporal pattern recognition problem, which is especially suitable to adopt a HMM.

Second, the HMM approach demands that the system being modeled (here, human's cognitive capacity) has both observable and hidden state variables, and the hidden variables should be correlated to the observable variables. As discussed above, there is ample evidence supporting secondary task performance as a highly sensitive and reliable technique for measuring human's cognitive load (Paas et al., 2003). We thus can use the secondary task performance as observable signals to estimate the hidden cognitive load state. For example, the secondary task performance can be measured in terms of the number of items correctly recalled. According to Miller's $7 \pm 2$ rule, the observable states will take integer values from 0 to 9 (assume it is 9 when the number of items correctly recalled is no less than 9). Hence, the strong tie, as uncovered in cognitive studies, between human's cognitive load and his/her



| A | B | C | D | E | F | G | H | ◆ I | J | K | L | ◆ M | N | O | P | Q | ◆ R | S | T | U | V | W | X | Y | Z | AA | AB | ◆ ... | AD | AE | AF | AG | AH | AI |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 52 | 38 | 5 | 204 | 15 | 31 | 1 | 1 | 33 | 1 | 1 | 196 | 31 | 1 | 81 | 1 | 64 | 291 | 36 | 1 | 1 | 189 | 45 | 172 | 1 | 1 | 15 | 21 | 83 | 1 | 1 | 153 | 42 | 126 | 1 |
| 47 | 32 | 5 | 204 | 11 | 31 | 1 | 1 | 22 | 1 | 1 | 186 | 25 | 1 | 81 | 1 | 61 | 244 | 32 | 1 | 1 | 139 | 29 | 133 | 1 | 1 | 7 | 21 | 57 | 1 | 1 | 41 | 24 | 1 | 1 |
| 16 | 6 | 5 | 102 | 7 | 18 | 1 | 1 | 21 | 1 | 1 | 83 | 8 | 1 | 81 | 1 | 9 | 99 | 13 | 1 | 1 | 84 | 14 | 65 | 1 | 1 | 11 | 11 | 18 | 1 | 1 | 39 | 13 | 17 | 1 |
| 65 | 59 | 5 | 306 | 27 | 49 | 2 | 1 | 55 | 3 | 33 | 298 | 53 | 8 | 306 | 1 | 85 | 291 | 44 | 4 | 21 | 328 | 82 | 291 | 4 | 3 | 83 | 83 | 83 | 29 | 21 | 328 | 95 | 291 | 11 |

Figure 1: Shared Belief Map.

secondary task performance also justifies the use of a HMM approach.

In the study reported below, we employed an experimental approach to collect realistic data of secondary task performance in a collaborative setting. We then used the data to learn HMM models of various number of hidden states, trying to understand the properties of HMM models that are acceptable for modeling human cognitive load.

## Cognitive Task Design and Data Collection

To study the dynamics of cognitive loads when humans are working in collaborative settings, we developed a system simulating a dynamic battlefield infosphere. A team can have several team members; each of them has limited observability (say, covering only a portion of the battlefield). The goal of a team is to selectively share information among members in a timely manner to develop global situation awareness (e.g., for making critical decisions).

Team members share information through a GUI with a shared belief map as shown in Figure 1. A shared belief map is a table with color-coded *info-cells*—cells associated with information. Each row captures the belief model of one team member, and each column corresponds to a specific information type (all columns together define the boundary of the information space being considered). Thus, info-cell $C_{ij}$ of a map encodes all the beliefs (instances) of information type $j$ held by agent $i$. Color coding applies to each info-cell to indicate the number of information instances held by the corresponding agent.

The concept of shared belief map facilitates the development of global situation awareness. It helps maintain and present a human partner with a synergy view of the shared mental models evolving within a team. Information types that are semantically related (e.g., by inference rules) can be closely organized in the map. Hence, nearby info-cells can form meaningful plateaus (or contour lines) of similar colors. Colored plateaus indicate those sections of a shared mental model that bear high overlapping degrees. In addition, the perceptible color (hue) difference manifested from a shared belief map indicates the information difference among team members, and hence visually represents the potential information needs of each team member.

We designed a primary task and a secondary task for the human subjects. The *primary task* of a human subject is to share the right information with the right party at the right time. Every time step (about 15 seconds), simulated spot reports (situational information) will be generated and randomly dispatched to team members. An info-cell on a person's belief map will be flashed (for 2 seconds) whenever that person gets new information of the type represented by that cell. The flashed cells are exactly those with newly available information that should be shared among teammates at that time step. An info-cell is frozen (the associated information is no longer sharable) when the next time step comes. Hence, a human subject has to share the newly available information with other team members under time stress. To share the information associated with an info-cell, a human subject needs to click the right mouse button on the cell to pop up a context menu, and select the receiving teammate(s) from the pop-up menu. Because information is randomly dispatched to team members, to each participant, the flashed info-cells vary from time to time, and there can be up to 12 info-cells flashed at each time step.

To choose an appropriate secondary task for the domain problem at hand is not trivial, although the general rationale is that the secondary task performance should vary as the difficulty of the primary task increases. Typically, a secondary task requires the human subjects to respond to *unpredictable stimuli* in either overt (e.g., press a button) or covert (e.g., mental counting) ways. Just for the purpose of estimating a human subject's cognitive load, any artificial task can be used as a secondary task to force the subject to go through. However, in a realistic application, we have to make sure that the selected secondary task interacts with the primary task in meaningful ways, which is not easy and often depends on the domain problem at hand. Specific to this study, the *secondary task* of a human subject is to *remember and mark* the cells being flashed (not necessarily in the exact order). Secondary task performance at step $t$ is thus measured as the number of cells marked correctly at $t$. The more number of cells marked correctly, the lower the subject's cognitive load.

While the experiment is designed in a collaborative setting with a meaningful primary task, we here especially focus on the secondary task performance. We would like to collect realistic data of secondary task performance and use the data to learn and understand the properties of HMM models of human cognitive loads. We randomly recruited 30 human subjects from undergraduate students and randomly formed 10 teams of size three. We ran the simulation system 9 times for each team and collected the secondary task performance of each team member: the number of info-cells marked correctly at each time step. Each run of the experiment has 45 time steps. We thus collected $10 \times 3 \times 9 = 270$ observation sequences of length 45.

## Learning Cognitive Load Models

### Learning Procedure

With the collected observation sequences, we took a two-phase approach. We first applied a window method to learn HMMs from sub-sequences and then evaluated each learned model with the original 270 observation sequences. Specifically, we assume human cognitive load can be modeled by HMMs with $n$ hypothetical hidden states where $3 \leq n \leq 10$. To train a $n$-state HMM, we applied a window of width $n$ on the original observation sequences to extract sub-sequences as training data. For example, to learn a 5-state HMM, a window of width 5 was used, which produced $270 \times 41 = 11,070$ training samples.

The training samples then were fed to the Baum-Welch algorithm (Rabiner, 1989) to learn HMMs. The training

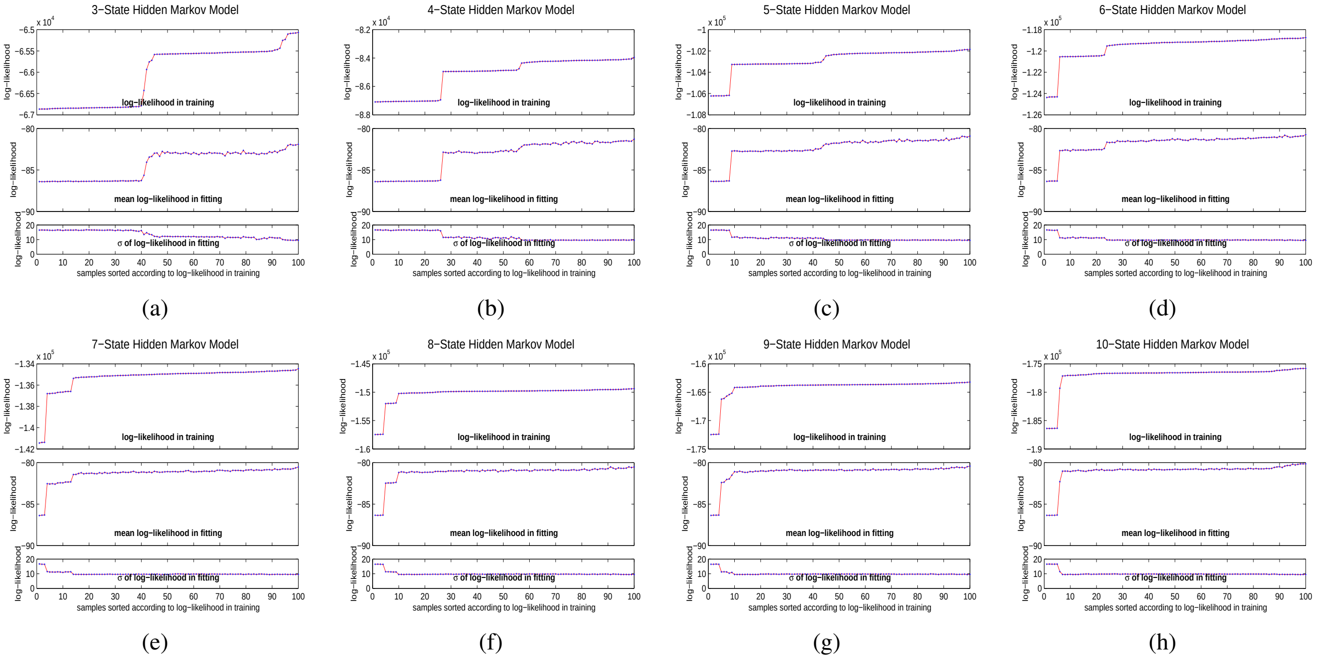


Figure 2: Each subfigure has top, middle, and bottom components, which plot the log-likelihoods of models after training, the log-likelihoods in testing, and the standard deviation of the log-likelihoods in testing. It clearly indicates (1) Maxima of each model space (from 3 to 10) form a 3-layer structure; (2) Better trained models lead to better testing log-likelihoods; and (3) Better trained models incur lower deviations. Model space also varied: as the number of hidden states increased from 3 to 10, the fraction of models at the middle and bottom levels reduced with the fraction of models at the top level increased.

process was terminated upon the convergence of its log-likelihood. Given the possibility of convergence at local maxima, we randomly generated initial guesses of parameters $(A, B, \pi)$ and repeated 100 times for each hypothetical $n$-state model. Consequently, we obtained 8 model spaces, each has 100 HMMs with $n$ hidden states ($3 \leq n \leq 10$). The top component of each subfigure in Figure 2 plots ascendingly the final log-likelihoods of the learned models of the corresponding model space.

In the second phase, for each learned HMM, we used the Forward procedure (Rabiner, 1989) to evaluate its performance by computing the occurrence probabilities (log-likelihoods) of the original 270 observation sequences, which produced 270 log-likelihoods. The middle component of each subfigure in Figure 2 plots, for each corresponding model plotted in the top component, the mean of the 270 log-likelihoods resulted from fitting (testing) the model with the original observation sequences. The bottom components plot the standard deviations of each model in fitting.

## The Model Space of Cognitive Load

Each subfigure in Figure 2(a-h) has top, middle, and bottom components, which plot the log-likelihoods of models after training, the log-likelihoods in testing, and the standard deviation of the log-likelihoods in testing. It clearly indicates that the model spaces with the number of hidden states ranging from 3 to 10 share some common properties. First, each model space (from 3 to 10) has a 3-layer structure, which means the log-likelihood maxima are clustered in three levels (models at the middle and bottom levels converged to local maxima). Second, better trained models performed better in testing: the trend of the log-likelihoods in fitting is consistent with the trend of the log-likelihoods in training (as ordered ascendingly in the top component). Third, better models produced lower deviation in fitting. Also, as the number of hidden states increased from 3 to 10, the fraction of models at the middle and bottom levels reduced with the fraction of models at the top level (converged at global maxima) increased. Extremely, most of the models in the space of 3-state HMMs are 'bad' models, while most of the models in the space of 10-state HMMs are 'good' models.

Table 1: Means of the longest hidden-state jumps (LHSJ) and mean fractions of transition pairs with stronger backward jumps (FSB) for HMMs with states from 3 to 10.

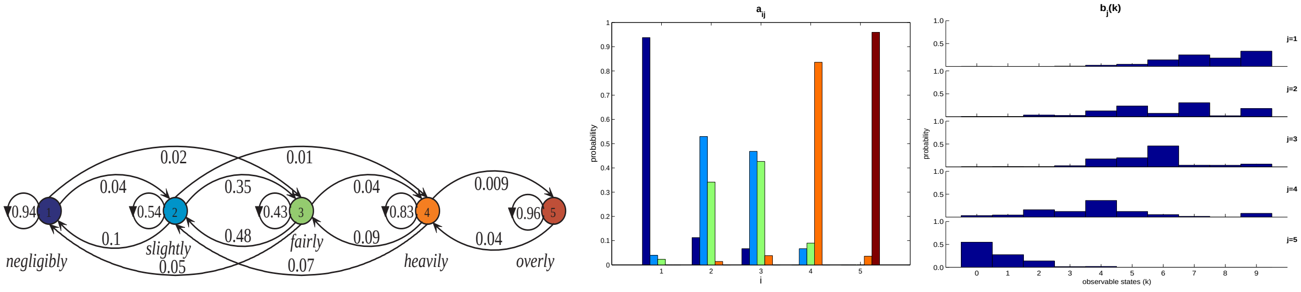
| *states* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *10* |
|---|---|---|---|---|---|---|---|---|
| LHSJ | 1.33 | 1.76 | 2.10 | 2.56 | 2.83 | 3.30 | 3.82 | 4.19 |
| FSB | 1.00 | 0.76 | 0.66 | 0.62 | 0.60 | 0.58 | 0.56 | 0.54 |

LHSJ = 0.0906 + 0.407 states

## Properties of 'Good' Cognitive Load Models

We may wonder whether there are any properties shared by the 'good' models as appeared at the top layer of each model space. We first examine $B$, the observation probability distributions. There is a strong evidence that the $B$'s of models at the top layer demonstrated more distinguishable peaks, compared with those at lower layers which typically had indistinguishable peaks or mixed distributions. Figure 3(c) gives the $B$ of one 5-state model at the top layer.

There are several statistics to check on the model parameter $A$, state transition probability distributions. With only



(a) (b) (c)

Figure 3: (a) An example 5-state HMM; (b) Transition probability distributions A; (c) Observation probability distributions B.

Table 2: For each state, the mean number of state transitions and the mean of initial state probability $\pi_i$.

| *states* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *10* |
|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 1.00/0.64 | 2.00/0.31 | *2.00*/**0.045** | | | | | | | |
| 4 | 1.89/0.44 | 2.20/0.26 | *2.75/0.27* | **1.86/0.037** | | | | | | |
| 5 | 2.32/0.41 | 2.70/0.20 | *3.04/0.18* | 2.98/0.19 | **1.70/0.028** | | | | | |
| 6 | 2.87/0.33 | 3.60/0.18 | *3.66/0.14* | 3.65/0.15 | 3.14/0.17 | **1.51/0.025** | | | | |
| 7 | 3.24/0.29 | 3.92/0.18 | 4.40/0.12 | *4.44/0.11* | 4.07/0.12 | **2.68**/0.16 | **1.23/0.019** | | | |
| 8 | 3.52/0.26 | 4.27/0.15 | 4.64/0.12 | 4.92/0.10 | *5.01/0.09* | 4.34/0.11 | **2.56**/0.15 | **1.16/0.017** | | |
| 9 | 4.09/0.24 | 4.89/0.12 | 5.41/0.11 | 5.60/0.10 | *5.70/0.09* | 5.58/0.08 | 5.12/0.09 | **2.66**/0.16 | **0.93/0.016** | |
| 10 | 4.45/0.21 | 5.21/0.12 | 5.62/0.10 | 5.85/0.09 | 6.27/0.08 | *6.37/0.07* | 6.14/0.08 | 4.77/0.09 | **2.61**/0.14 | **0.79/0.015** |

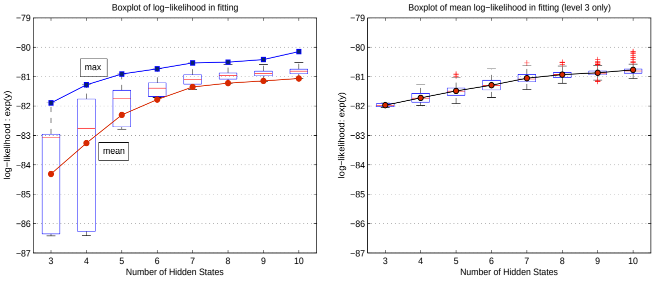
cell content: number of state jumps/$\pi$

top-layer models considered, Table 1 gives the means of the longest hidden-state jumps (LHSJ) and the mean fractions of transition pairs with stronger backward jumps (FSB). For example, for HMMs with 5 states, on average state transitions have jumps no more than 2.1 states, and of all the possible state transition pairs $(A_{ij}, A_{ji})$ where $1 \leq i < j \leq 5$ and state $j$ represents a higher cognitive load than state $i$, 66% have stronger backward transitions $(A_{ij} < A_{ji})$. Of all the models with states from 3 to 10, the means of LHSJ, ranging from 1.33 to 4.19, linearly related to the number of states with slope $0.407 \approx 2/5$. Interestingly, all models have relatively more transition pairs with stronger backward jumps. This seems to suggest that humans can more easily recover from than switch to a higher cognitive load state.

For each state and each model, Table 2 gives the mean number of transitions and the mean of initial state probability. For each category except models with 3 states, it seems that the highest one (4–6) or two (7–10) states have much few number of transitions than the other states. It may suggest that humans, once become “overly” loaded, can not easily return to cognitively favorable states. The highest state of each model category also assume extremely lower initial state probability, and the lower states have relatively higher initial state probability. This is intuitively true because humans seldomly get overloaded in the beginning. For each model category, Table 2 also indicates such a trend: as hidden state changed from low to high, the mean number of state transitions increased to its maximum (in italic), while $\pi_i$ decreased to minimum (with the highest state ignored). This may indicate that the more active a state is, the less likely a human can be initially in that state.

## The Number of Hidden States

A crucial question is, how many hidden states are appropriate for modeling cognitive load using HMMs?



(a) (b)

Figure 4: Boxplot of model log-likelihoods.

Figure 4(a) gives the Boxplot of likelihoods in fitting for all the models in each space, and Figure 4(b) gives the Boxplot of likelihoods in fitting for top-layer models only. Fig. 4(a) says that the model variance is small enough when the number of hidden states is no less than 4. Fig. 4(b) shows that there is a linear improvement on the top-layer models as state number increases. However, because the observable measure of secondary task performance ranges from 0 to 9, models with too many hidden states may overfit the given data. In addition, the trained models with more than 8 states demonstrated ‘strongly-connected’ sub-structures. Figure 5 gives a top-layer model with 10 states, where links on the upper part represent forward transitions and those on the lower part are backward ones (state transition probabilities are visualized in color densities). It is clear that this 10-state model can be reduced to a 5-state model if we view (1,2,3) and (4,5,6,7) as two compound states.

Practically, it is better to pick from top-layer models with 7 states, which have a mean likelihood $e^{-81}$–only slightly lower than 10-state models. It is also acceptable to choose from the top-layer models with 5 states; indeed, the performance could be as good as 7-state models if the best of 5-state models were picked. In addition, there is ample evi-

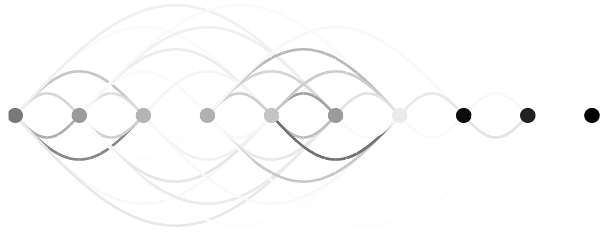


Figure 5: A 10-state model of cognitive load.

dence suggesting that human cognitive load is a continuous function over time and does not manifest sudden shifts unless there are fundamental changes in tasking demands. If we place a constraint on the state transition coefficients such that no jumps of more than 3 states are allowed, then models with 5, 6, or 7 states are best choices (ref. Table 2). Moreover, models with fewer states are more interpretable. Like the 5-state model in Fig. 3, it is easier to assign meaningful names to the hidden states.

In sum, with all the above factors considered, it seems we can follow 7 ± 2 rule to choose the number of hidden states of a HMM-based cognitive load model. To examine the efficacy of such a principle, we used 5-state HMM models and conducted a study (Fan & Yen, 2007) involving two team types: ten of the teams performed with cognitive load estimation available from agents and ten teams with no such estimation. The result indicated that teams with load estimation performed significantly better than teams with no load estimation. The reason is that being able to estimate other team members' cognitive load allows them to share the needed information with the right party at the right time.

## Conclusion

An agent empowered with a cognitive load model of its human peer can be beneficial in offering trustable autonomy and unintrusive help. We used Hidden Markov Models to capture cognitive load in a way that can be used in team contexts to make predictions about other team members' workload.

To develop realistic cognitive load models, we conducted cognitive experiments to capture human's observable secondary task performance and used that to train hidden Markov models (HMM) with varied number of hypothetical hidden states. The results indicate that each model space has a three-layer structure, and it is suggested to choose models with 7 ± 2 hidden states. With all the constraints considered, it is recommended that HMMs with 5, 6, or 7 states are best choices for modeling human cognitive load. Statistical analysis revealed that good models also share some common properties: (1) observation probability distributions have distinguishable peaks for different states; (2) highest hidden states have extremely lower initial state probabilities; and (3) the longest state jumps are linearly related to the number of states with a slope $2/5$, and there are more transition pairs with stronger backward jumps. These can be used in guiding the selection of HMM-based cognitive load models.

## References

Baddeley, A. D. (1992). Working memory. *Science*, *255*, 556-559.

Bradshaw, J., Sierhuis, M., Acquisti, A., Gawdiak, Y., Prescott, D., Jeffers, R., et al. (2002). What we can learn about human-agent teamwork from practice. In *Workshop on teamwork and coalition formation at AAMAS 02.* Bologna, Italy.

Cohen, P. R., & Levesque, H. J. (1991). Teamwork. *Nous*, *25*(4), 487-512.

Fan, X., & Yen, J. (2007). Realistic cognitive load modeling for enhancing shared mental models in human-agent collaboration. In *Proceedings of the sixth international joint conference on autonomous agents and multiagent systems* (p. 383-390). ACM Press.

Fan, X., Yen, J., & Volz, R. A. (2005). A theoretical framework on proactive information exchange in agent teamwork. *Artificial Intelligence*, *169*, 23–97.

Grosz, B., & Kraus, S. (1996). Collaborative plans for complex group actions. *Artificial Intelligence*, *86*, 269-358.

Klimoski, R., & Mohammed, S. (1994). Team mental model: Construct or metaphor? *Journal of Management*, *20*(2), 403-437.

Lang, A. (2000). The limited capacity model of mediated message processing. *Journal of Communication*, *Winter*, 46-70.

Lord, R. G., & Maher, K. J. (1990). Alternative information processing models and their implications for theory, research, and practice. *The Academy of Management Review*, *15*(1), 9-28.

Miller, G. A. (1956). The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological Review*, *63*, 81-97.

Norling, E. (2004). Folk psychology for human modelling: Extending the BDI paradigm. In *AAMAS '04: International conference on autonomous agents and multi agent systems* (pp. 202–209).

Paas, F., & Merrienboer, J. V. (1993). The efficiency of instructional conditions: an approach to combine mental-effort and performance measures. *Human Factors*, *35*, 737-743.

Paas, F., Tuovinen, J. E., Tabbers, H., & Gerven, P. W. M. V. (2003). Cognitive load measurement as a means to advance cognitive load theory. *Educational Psychologist*, *38*(1), 63-71.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*, 257-286.

Rouse, W., Cannon-Bowers, J., & Salas, E. (1992). The role of mental models in team performance in complex systems. *IEEE Trans. on Sys., man, and Cyber*, *22*(6), 1296-1308.

Sweller, J. (1988). Cognitive load during problem solving: effects on learning. *Cognitive Science*, *12*, 257-285.

Xie, B., & Salvendy, G. (2000). Prediction of mental workload in single and multiple task environments. *International Journal of Cognitive Ergonomics*, *4*, 213-242.

# Effect of Communication on Belief Dynamics in Multi-Agent Systems

**M. Afzal Upal (upal@oxy.edu)**
Cognitive Science, Occidental College
Los Angeles, CA 90041 USA

**Ravikanth Sama (ravikanth.sama@eng.utoledo.edu)**
Electrical Engineering & Computer Science Department
University of Toledo, Toledo, OH 43606 USA

## Abstract

This paper presents the architecture of a multiagent society (MAS) designed to study the dynamics of belief change in natural and artificial societies. It also presents a hierarchical model for representation of beliefs and a multiagent domain called Multiagent Wumpus World (MWW) designed to test the capabilities of the proposed MAS. It also reports on a set of experiments designed to study the formation of false social beliefs. Our results indicate that more false beliefs are likely to be generated about objects/events whose presence is harder to confirm or disconfirm. We also discovered that this behavior is slightly enhanced when the agents are allowed to communicate with other agents, in which case, the false beliefs about the objects/events which are easier to confirm or disconfirm significantly decrease while about those objects/events which are harder to confirm remain higher.

## I. Introduction

Modeling and understanding the formation, propagation, and evolution of beliefs is crucial both to the success of distributed artificial intelligence (AI) systems as well as to improve our understanding of human and animal societies. The growth of multiagent systems research in artificial intelligence [1] has been paralleled by a growing realization among cultural scientists that the traditional verbal models are too imprecise to model belief dynamics while mathematical models are too rigid and unable to be scaled up [2]. As economist Scott Moss recently lamented, “in more than half a century since the publication of von Neumann-Morgenstern (194x), no significant progress has been made in the development of models that capture the process of interaction among more than two or three agents” [3]. The alternative that Moss and others propose is to build bottom-up algorithmic models of socio-cognitive processes. The key idea behind the agent-based social simulation (ABS) approach is to encapsulate each member of a population in a software module (called an *agent*) to build bottom-up models of human or animal societies. The ABS models focus on interactions between agents and, for the most part, abstract away the internal cognitive structure of the agents. Thomas Schelling, one of the early pioneers of the ABS approach, designed 1500 agents that lived on a 500 x 500 board [4]. The agent’s cognitive structure consisted of one simple inference rule, namely, if the proportion of your different colored neighbors is above a tolerance threshold then move, otherwise stay. He showed that even populations with high tolerance end up living in highly segregated neighborhoods.

The ABS methodology illustrates that it is not necessary, or even desirable, to have a complete understanding of a social system before building computational models. Indeed, ABS systems are frequently used as theory exploration and development tools (similar to the way computer models are used as tools by AI and Cognitive Modeling researchers) because they allow theoreticians to visualize and fully explore the consequences of their models and to compare competing theories. The last few years, there has been an explosion in the development of ABS systems designed to simulate social systems from a variety of domains. Ignoring the complex internal cognitive structure not only allows ABS designers to design computationally tractable simulation systems but it also helps them show causal connections between the cognitive rules that agents use to make local decisions and social patterns that emerge at the population level–the highly desired, yet rarely achieved–identification of micro-macro links.

Few ABS systems, however, have been built to specifically model beliefs dynamics and the systems developed to date assumed overly simplistic models of individual cognition and knowledge representation. For instance, most existing ABS models of social belief change model agent-beliefs as a single bit and belief change involves flipping the bit from 0 to 1 or vice versa often to match the beliefs of the neighbors [5][6][7]. This severely limits these systems as they are unable to model most real world distributed systems applications. Complex patterns of shared beliefs such as those that characterize people’s cultural and religious beliefs are also not likely to emerge out of such systems because the ABS agents are not even able to represent them. Thus existing ABS systems cannot be used to explore or model belief dynamics in human societies.

Traditionally, artificial intelligence and cognitive modeling have studied how individuals form and modify complex belief structures [8][9][10] but have, for the most part, ignored agent interactions assuming single agents living unperturbed in closed worlds. Artificial intelligence research on *classical planning* illustrates this approach well [11]. Given the knowledge about current state of world, about goals that the agent desires to achieve, and the generalized actions that the agent can take in the world, the planning problem is to compute an ordered sequence of action instances that the agent can execute to attain its goals. The classical AI planning research assumes that the planning agent is acting alone in the world so that the world does not change while the agent is figuring out what to do next because if that happens, the agent's plan may not be executable any longer. If the world continues to change the agent may never be able to act as it will always be computing the plan for the changed situation. Abstracting away other actors allows AI researchers to eliminate additional sources of complexity to focus on complex reasoning processes that go on inside the heads of individuals and result in the rich knowledge structures such as plans. This has led to the development of successful game playing programs that work in environments with limited or no interaction with other agents. However, this approach is not useful for modeling the dynamics of cultural belief systems such as religious belief systems because they are by their very nature products of the interaction of a large number of agents.

Clearly, to simulate belief dynamics in human societies, we need to develop knowledge-rich agent-based social simulation systems (KBS) [12]. Agents in these systems must have rich knowledge representation and reasoning capabilities and they must be able to interact with other agents present in their environment. Such simulation systems must overcome computational tractability concerns without abstracting away the agent's internal cognitive structure (as done by ABS systems) or ignoring interactions with other agents (as done by much of traditional AI & CM work)? Furthermore, to be able to tell us something about belief dynamics in human societies such agents in such systems must model the cognitive tendencies that people are known to possess. We believe that people's ability to communicate, comprehend a message, and integrate the newly received information into their existing knowledge is crucial to understanding the formation, propagation, and evolution of beliefs. We have designed a knowledge-rich multiagent society, called CCI[1], to model these processes. The challenge for any KBS system is that of overcoming the computational intractability problems to design an efficient system that can be run in real time. This paper argues that one promising approach for addressing this challenge is to develop synthetic computer games like environments that are rich enough to exercise the enhanced knowledge representation and reasoning capabilities of KBS agents yet they are not so complex to make the simulation intractable and the results impossible to analyze and understand.

## II. Communicating, Comprehending, and Integrating (CCI) Agents

The CCI agents are goal directed and plan sequences of actions to achieve their goals. Some of the actions that they need to undertake to achieve their goals may be speaking actions. An agent A may decide to send a message M to an agent B if it believes that sending B the message M will result in changing B's mental state to cause it to perform an action C which can help A achieve one of its goals.

The CCI agents, similar to people [13], are comprehension driven i.e., they attempt to explain each piece of information their sensors detect. On observing an effect E, they search for a cause C that could have produced that effect.

Agents attempt to build accurate models of their environment by acquiring information about cause-effect relationships among various environmental stimuli. They store this information as cases [14]. Agents consult their case memory to form expectations about the future. If these expectations are violated, they attempt to explain the reasons for these violations and if they can find those explanations, they revise their world model. The CCI agents ignore the information received from others if they cannot find any justification for it.

We have designed the first version of a CCI society by embedding it into an artificial domain. Multiagent Wumpus World (MWW), shown in Figure 1, is an extension of Russell and Norvig's [11] single agent Wumpus World.

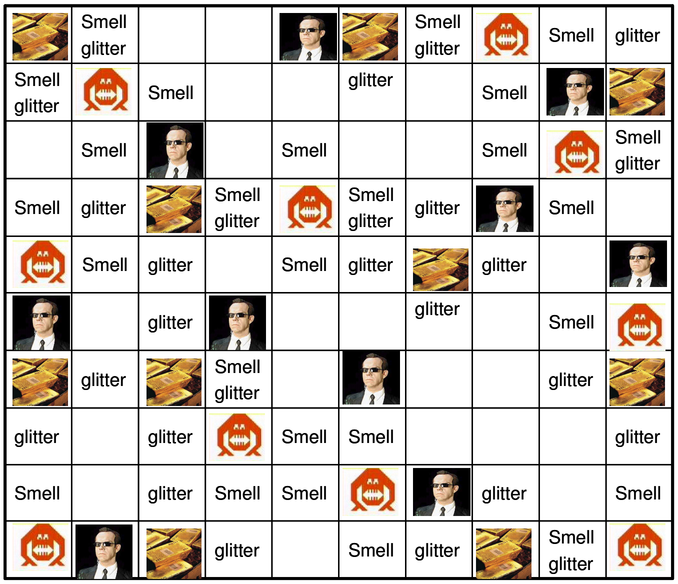


*Figure 1: A 10 x 10 version of the Multiagent Wumpus World (MWW) domain. This version has 10 agents, 10 Wumpuses, and 10 Treasures.*

### A. *Multiagent Wumpus World (MWW)*

MWW has the same basic configuration as the single agent Wumpus World (WW). MWW is an *N* x *N* board game with a number of wumpuses and treasures that are randomly placed in various cells. Wumpuses emit stench and treasures glitter. Stench and glitter can be sensed in the horizontal and vertical neighbors of the cell containing a wumpus or a treasure. Similar to the single agent WW, once the world is created, its configuration remains unchanged i.e., the wumpuses and

[1] Communicate, Comprehend, and Integrate

treasures remain where they are throughout the duration of the game. Unlike the single agent version, MWW is inhabited by a number of agents randomly placed in various cells at the start of the simulation. An agent dies if it visits a cell containing a wumpus. When that happens, a new agent is created and placed at a randomly selection location on the board.

The MWW agents have a causal model of their environment. They know that stench is caused by the presence of a wumpus in a neighboring cell while glitter is caused by the presence of treasure in a neighboring cell. Agents sense their environment and explain each stimulus they observe. While causes (such as wumpuses and treasures) explain themselves, effects (such as stench and glitter) do not. The occurrence of effects can only be explained by the occurrence of causes that could have produced the observed effects e.g., glitter can be explained by the presence of a treasure in a neighboring cell while stench can be explained by the presence of a wumpus in a neighboring cell. An observed effect, however, could have been caused by many unobserved causes e.g., the stench in cell (2, 2) in Figure observed in could be explained by the presence of a wumpus in any of the four cells:

- (1, 2)
- (3, 2)
- (2, 1)
- (2, 3)

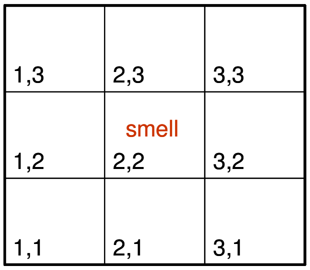


*Figure 2: A part of the MWW.*

An agent may have reasons to eliminate some of these explanations or to prefer some of them over the others. The MWW agents use their existing knowledge to select the best explanation. Agent's knowledge base contains both the game rules as well as their world model. A world model contains agent's observations and past explanations. The observations record information (stench, glitter, treasure, wumpus, or nothing) the agent observed in each cell visited in the past. The MWW agents use their past observations and game knowledge to eliminate some possible explanations e.g., if an agent sensing stench in cell (2,2) has visited the cell (1,3) in the past and did not find sense any glitter there, then it can eliminate "wumpus at (2, 3)" as a possible explanation because if there were a wumpus at (2, 3) there would be stench in cell (1, 3). Lack of stench at (1, 3) means that there cannot be a wumpus at (2, 3). Agents use their knowledge base to form expectations about the cells that they have not visited e.g., if the agent adopts the explanation that there is a wumpus in cell (2, 1) then it can form the expectation that there will be stench in cells (1, 1) and (3, 1).

In each simulation round, an agent has to decide whether to take an action or not. Possible actions include:

- the action to move to the vertically or horizontally adjacent neighboring cell
- the action to send a message to another agent present in the same cell as the agent, and
- the action to process a message that the agent has received from another agent.

### *1) Hierarchical Belief Structure*

Beliefs about the cells are incorporated into the knowledge base of the agents in a hierarchical fashion. The beliefs are classified into CERTAIN, ASSUMED, and EXPECTED. CERTAIN beliefs are those which the agent is sure about. For example, if the agent encountered a treasure in its path, it is sure that the four surrounding cells have glitter in them. ASSUMED beliefs are those which the agent is not very confident about. For example, if the agent encounters a glitter in its path, it is unsure of the source of the glitter, as the treasure could be in any one of the four neighboring cells of that cell. It then tries to explain this glitter by associating it with some treasure. If it finds none, it ASSUMES that one of the neighboring cells has the treasure and it EXPECTs to find glitter in the cells adjoining the ASSUMED treasure. While the agents have reason to hold ASSUMED and EXPECTED beliefs, they are not totally confident about them and are willing to modify these beliefs if experience suggests otherwise. The agents never accept any alterations to the CERTAIN beliefs.

The MWW agents are goal directed agents that aim to visit all treasure cells on the board. Agents create a plan to visit all treasure cells they know about. The plan must not include any cells that contain wumpuses in them. Each agent ranks all the cells by how confident it is of its knowledge about a cell. It has the highest confidence in the cells that it has already visited. Next are the cells whose neighbors the agent has visited and so on. Agents also rank cells by how urgently they need the information about that cell. The order in which the cells are to be visited determines the criticality e.g., if a cell is the next to be visited then finding information about that cell is assigned the highest priority while a cell that is not planned to be visited for another 10 rounds gets low priority. The agents then use an information seeking function that takes the two rankings (confidence and criticality) as inputs and decides whether the agent needs to communicate and if so which cell it needs the information about. If it decides to communicate and if another agent is currently present in the same cell then agent sends a request-for-information message to that agent.

The listening agent attempts to explain as to why the speaker sent it the message (e.g., was it sent to seek information or to distract me from traveling) and depending on that determination it may or may not decide to respond. If the agent decides to respond, it will offer the information about the requested cell in exchange for information about a cell about which it needs information. If the speaker agrees to the exchange then the agents communicate information about the cells.

On receiving information about a cell, an agent has to decide whether to incorporate that information into its knowledge-base or not. If it decides to incorporate the new information then it has to decide how best to revise its existing knowledge. If the new information confirms what the agent already know about the cell then no revision is done. If, on the other hand, the information received from another agent is different from what the agent expects to find in that cell then it attempts to explain the reason for the contradiction. If it can find a possible explanation that it ignored in the past that supports the received information, then it adopts the new explanation and the new information and retracts its belief in the old explanation and expectation. If the agent does not have any information about that cell, it incorporates the received information as ASSUMED belief. Otherwise, the agent rejects the newly received information.

### *2) Formation of False Beliefs*

Agents use symmetrical processes to form and revise beliefs in the presence of treasures and wumpuses. Thus agent models may contain false beliefs about the locations of the wumpuses as well the locations of treasures. However, while the agents prefer to travel towards a cell that they believe contains a treasure, they avoid cells that they believe contain wumpuses in them. This makes beliefs in the presence of wumpuses relatively harder to confirm or disconfirm than beliefs in the presence of treasures. The experiments described next were designed to investigate the impact that this asymmetry has on the patterns of false beliefs that the agents form.

### *3) Planning*

Agents are required to generate paths which they would follow to achieve their goals, known as *plans*. The planning algorithm used is kept very simple as the focus is on belief dynamics. The agents are given a goal-cell to be reached when they are born. The agents simply include all the cells in the L-shaped path that leads them to the goal. This plan is revised if and when it suspects the presence of a wumpus in its previous plan. After one goal is reached, the agent then tries to confirm all the ‘ASSUMED’ treasures and ‘EXPECTED’ stenches it has recorded in its path. Thus, the agent tries to make its world model as accurate as possible.

## III. Experiments and Results

We designed a 10 x 10 version of MWW with ten agents randomly placed at ten different locations. In the first set of experiments I designed three different versions of MWW:

- The 5x5 version has 5 wumpuses and 5 treasures
- The 10x10 version has 10 wumpuses and 10 treasures, and
- the 20x20 version has 20 wumpuses and 20 treasures.

I allowed the simulation to run for 300 rounds. At the end of that round I measured the following metrics:

- *Average agent age* is the average age of the ten agents that survive after round 300.
- *The average number of wumpus beliefs* is the average number of wumpus beliefs the surviving agents have
- *The average number of treasure beliefs* is the average number of treasure beliefs the surviving agents have
- *Average number of cells visited* is the average number of cells that 10 agent surviving at the end of round 10 have visited.

Figure 1shows the results of Experiment I. The results for average age show that 10x10 world proves to be the most taxing for agents with average agent ages only being less than 30 rounds. Agents in 5x5 and 20x20 rounds, on the other hand, have much higher average ages. This is explained by the considering the average number of cells that the agents visit. Agents in the 5x5 world visit a larger number of cells while agents live longer in the 20x20 world by visiting fewer cells and by thus deciding to stay in their cells. Agents in the 10x10 world, however, visit more cells than in 20x20 world but they pay the price of this adventurism by having smaller average ages.

In the second experiment, then we adopted the 10x10 world and measured the proportion of false wumpus and treasure beliefs that the agents had at the end of round 100. The proportion of false wumpus/treasure beliefs is the proportion between the number of false wumpus/treasure beliefs that the agent has to the total number of wumpus/treasure beliefs that the agent possesses.

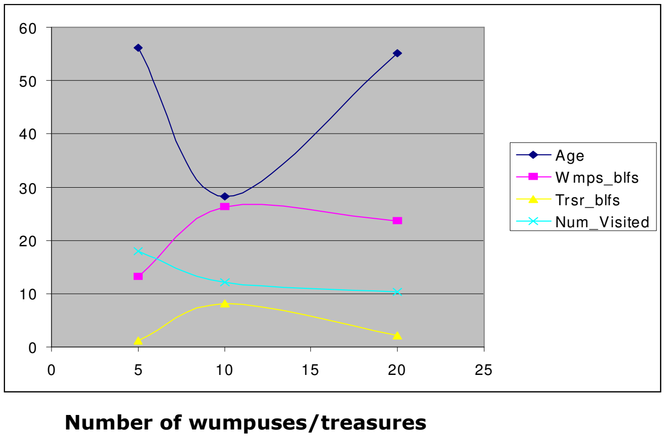


*Figure 3: Results of Experiment 1. Each point is an average of 30 runs.*

The results (Figure 4) show that after the initial drop, the proportion of false wumpus beliefs remains relatively unchanged regardless of how much the agents travel in the world. False beliefs in the presence of treasures, however, continue to decrease as agents with agent age. The agents who survive the length of simulation (100 rounds) have few, if any, false beliefs about treasures but on average 40% of their beliefs about the presence of wumpuses are false.

In the third experiment (Figure 5), we allowed the agents to communicate with other agents when they happen to meet in the same cell. The agents request information about the cells in their plan about which they are not sure of and they are about to visit. Criticality and confidence play a major role in the selection of this cell. The agents then offer information to the other agent.

The results show that communication affects false wumpus and treasure beliefs differently: the proportion of false wumpus beliefs remains almost the same as in the without-communication case while the proportion of false treasure beliefs decreases significantly.

### *A. Discussion*

Owing to our planning strategy, which focuses on improving the accuracy of the world model, each agent tries to confirm as many treasures as possible and mark as many stenches as possible. Our results indicate that explanations that are harder to confirm and disconfirm are more likely to be generated by agents that attempt to explain their observations and revise these explanations in light of the evidence. This suggests that people should have more false beliefs about things that are harder to confirm or disconfirm. There is some evidence to suggest that that is the case. Bainbridge and Stark [15] made confirmability the core of their theory of religion to argue that religious beliefs are unconfirmable algorithms to achieve rewards that are highly desired by people yet cannot be obtained. Similarly, there is some evidence to suggest that many false ethnic stereotypes people have are about things that are harder to confirm or disconfirm such as the sexual practices of the neighboring tribes. Our results also indicate that introducing the ability to communicate with other agents does not improve the condition of holding false beliefs about unconfirmable objects while it significantly reduces the false beliefs held about an object/event that is easier to confirm. This can be thought of as a traveler claiming that his city has good many skyscrapers and the listener then confirming it later when he visits that city.

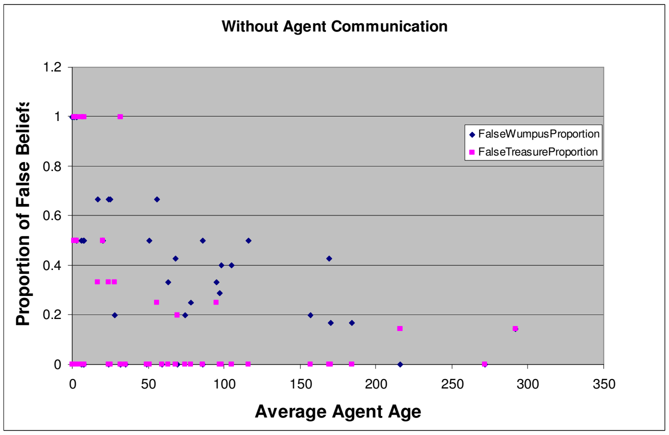


*Figure 4: Results of Experiment 2, Without agent Communication*

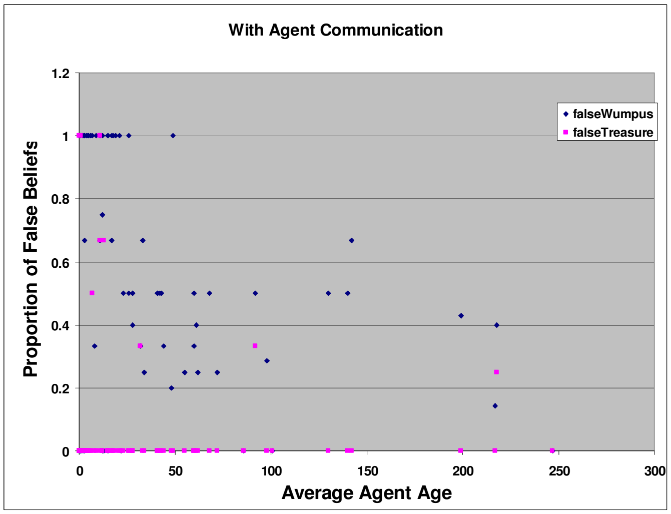


*Figure 5: Results of Experiment 3, With agent Communication*

## IV. Conclusion

This paper presents the architecture of a multiagent society designed to study the dynamics of belief change in natural and artificial societies. It also presents a multiagent domain designed to test the capabilities of the proposed systems. Preliminary results are encouraging as they indicate the potential for the use of the proposed society.

## REFERENCES

[1] G. Weiss, *Multiagent systems: A modern approach to distributed artificial intelligence*, Cambridge, MA: The MIT Press, 1999.

[2] C. Castelfranchi & B. Kokinov (editors), *Understanding the dynamics of knowledge: Integrating models of knowledge change, development and evolution in cognitive science, epistemology, philosophy, artificial intelligence, logic, and developmental, and evolutionary psychology*, Technical Report of the European Science Foundation, 2005

[3] S. Moss, Game Theory: Limitations and Alternatives, *Journal of Artificial Societies and Social Simulation*, 4(2), 2001.

[4] T. Schelling, Dynamic models of segregation, *Journal of Mathematical Sociology*, 1, 143-186, 1977.

[5] Bainbridge, W. 1995. Neural Network Models of Religious Belief , *Sociological Perspectives*, 38: 483-495.

[6] Doran, J. (1998). Simulating collective misbelief, *Journal of Artificial Societies and Social Simulation*, 1(1).

[7] Epstein, J. (2001). Learning to be thoughtless: Social norms and individual computation, *Computational Economics*, 18(1), 9-24.

[8] C. E. Alchourròn, P. Gärdenfors, and D. Makinson (1985). On the logic of theory change: Partial meet contraction and revision functions. Journal of Symbolic Logic, 50:510-530.

[9] Allen, J. (1987). Natural Language Understanding. Menlo Park, CA, Benjamin Cummings.

[10] Bringsjord, S. and Ferrucci, D. (2000). Artificial Intelligence and Literary Creativity: Inside the Mind of BRUTUS, a Storytelling Machine. Mahwah, NJ, Erlbaum.

[11] Russell, S. & Norvig P. (1995) Artificial Intelligence: A Modern Approach, Englewood Cliffs, NJ: Prentice Hall.

[12] M. A. Upal & R. Sun (editors) Cognitive Modeling and Agent-based Social Simulation: Papers from the AAAI-06 Workshop (ISBN 978-1-57735-284-6), Menlo Park, CA: AAAI Press, 2006.

[13] W. Kintsch, *Comprehension: A Paradigm for Cognition*, Cambridge, NY: Cambridge University Press, 1998.

[14] R. C. Schank & R. P. Abelson, *Scripts, plans, goals, and understanding: an inquiry into human knowledge structures*, Hillsdale, NJ: Lawrence Erlbaum, 1977.

[15] Bainbridge, W. & Stark, R. 1987. *A Theory of Religion*, New York: Lang.

[16] L. Smith, Sects and Death in the Middle East, *The Weekly standard*, 11 (39).

# Using Reflective Learning to Master Opponent Strategy in a Competitive Environment

**Mark A. Cohen (mcohen@lhup.edu)**
Department of Business Administration, Computer Science, and Information Technology, Lock Haven University,
Lock Haven, PA 17745 USA

**Frank E. Ritter (frank.ritter@psu.edu)**
**Steven R. Haynes (shaynes@ist.psu.edu)**
College of Information Sciences and Technology, The Pennsylvania State University, State College, PA 16802 USA

## Abstract

Cognitive models of people interacting in competitive environments can be useful, especially in games and simulations. To be successful in such environments, it is necessary to quickly learn the strategy used by the opponent. In addition, as the opponent adjusts its tactics, it is equally important to quickly unlearn opponent strategies that are no longer used. In this paper, we present human performance data from a competitive environment. In addition, a cognitive model that uses reflective learning is introduced and compared to the empirical findings. The model demonstrates that it is possible to simulate learning in an adversarial environment using reflection and provides insight into how such a model can be expanded.

## Introduction

Cognitive models of people interacting in competitive environments can be useful, especially in games and simulations (Jones et al., 1999; Laird, 2001a, 2001b; Ritter et al., 2002). To be successful in such environments, it is necessary to quickly learn the strategy used by the opponent. In addition, as the opponent adjusts its tactics it is equally important to quickly unlearn opponent strategies that are no longer used. The model presented here uses learning by reflection to accomplish this task. This model was created using a high-level tool that produces cognitive models quickly, and with little or no programming. We briefly take up the two important aspects of this project, leaning from reflection and the role of variability in performance.

### Leaning By Reflection

Learning by reflection (or introspection) is one technique that can be used to learn and unlearn an opponent's changing strategies while at the same time preserving the variability in which people learn (e.g. Bass, Baxter, & Ritter, 1995; Cox & Ram, 1999; Ritter & Wallach, 1998).

Learning by reflection is a form of metacognition that allows the model to learn by reflecting on its performance, and adjusting accordingly. When reflection reveals previous actions that were beneficial, the model will be more likely to repeat those same actions in similar situations. However, when reflection reveals poor performance, the actions that lead to that performance are less likely to be repeated. Thus, learning by reflection is a form of reinforcement learning (Russell & Norvig, 2003).

Reflective learning requires that both the cognitive model and its environment are fully observable (Russell & Norvig, 2003). In other words, the model must be able to observe the effects of its actions on the environment and other models.

### Variability

For a model's behavior to be believable in a game or simulation its performance must do more than match average human behavior. Cognitive models must also exhibit the same variability in behavior that a human exhibits. When playing a game or participating in a simulation, variable behavior is a crucial part of the realism that these systems must portray.

Because reflective learning strategies are based on probability, the behaviors they generate are not deterministic. This allows reflective models to exhibit variability in learning and thus performance.

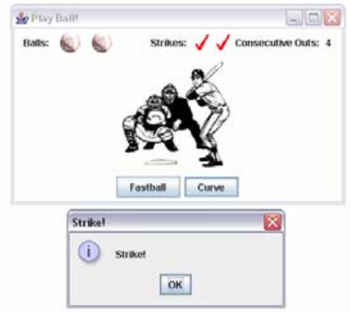
The remainder of this paper describes a study done to measure how quickly participants in a user study learn opponent strategies while performing a competitive task, and a cognitive model that was designed to exhibit similar performance with equal variability.

## Task

Lehman, Laird, and Rosenbloom (1996) in their *A Gentle Introduction to Soar* use baseball repeatedly as an example. This inspired us to implement a simple version of a baseball game to study adversarial problem solving and support people learning Soar. In a broader context, this environment provides an accessible platform for the future study of cognitive models interacting with other agents in a social simulation (Sun, 2006).

Figure 1 shows the basic interface and one of the feedback screens. In this game, participants play the role of the pitcher competing against a series of agent-operated batters. The goal of this game, as in baseball, is to get batters out.

The baseball game described here was written in Java and interacts with the Soar cognitive architecture using the Soar Markup Language (*SML Quick Start Guide*, 2005). The software and instructions on how to use it are available online (acs.ist.psu.edu/herbal).



Strike!

Figure 1: The Baseball Game Task.

## Rules of the Game

There are two ways to get a batter out in this game: The batter can get three strikes (a strike results when a batter either swings and misses or does not swing at a good pitch), or the batter can hit the ball directly at a fielder who catches the ball.

There are also two ways for a batter to get on base in this game: The batter can get four balls (a ball results when the batter does not swing at a bad pitch), or the batter can hit the ball in a way that prevents the fielders from catching it.

Acting as the pitcher, the participants in this study had a choice of throwing either a fastball or a curveball to the batter. Once they threw a pitch, the batter had a choice of either swinging at the pitch or letting it go by. Both the pitcher and batter are always aware of how many balls and how many strikes the batter has. The rules shown in Table 1 describe how to determine the outcome of each pitch.

Table 1: Determining the outcome of a pitch

| Pitcher | Batter Response | Outcome |
|---|---|---|
| Fastball | Batter swings | Contact is made that may result in either an out (50% of the time) or a hit (50% of the time). |
| Fastball | Batter does not swing | The pitch will result in a strike.[1] |
| Curveball | Batter swings | The pitch will result in a strike.[1] |
| Curveball | Batter does not swing | The pitch will result in a ball.[1] |

Based on the rules described above, the most certain way to get a batter out is to throw a curveball when the pitcher thinks the batter will be swinging and to throw a fastball when the pitcher thinks the batter is not going to swing. Naturally, if the participant can learn what strategy the batter is using then they have a better chance of getting them out.

## Batter Strategies

Each participant faced the same five different batter strategies in the same sequence during play. Strategy changes were determined by the number of consecutive outs that the participant recorded against a given strategy. When a predetermined out threshold was reached, a strategy shift by the batter would take place. The exact sequence of batter strategies and their corresponding out thresholds were defined in a configuration file that was used by the baseball environment, but is unknown to the pitcher. The batter strategies, along with their consecutive out thresholds, are shown in Table 2. The strategies shown here are the ones used in our user study in the order they are listed. However, we do not propose this as the only order, or the best order. The baseball game environment is easily configurable to use other strategies and to present them in any order. This illustrates the baseball task's usefulness for studying the effects of order on learning (Ritter, Nerb, O'Shea, & Lehtinen, 2007).

Table 2: Batter Strategies in the Baseball Environment

| Name | Strategy | Out Threshold |
|---|---|---|
| Hacker | Always swings | 4 |
| Aggressive | Swings at the first pitch and when there are fewer strikes than balls, unless there are three balls and two strikes | 7 |
| Random | Randomly chooses when to swing | 5 |
| Chicken | Never swings | 4 |
| Alternate | Swings if the last pitch was a fastball and does not swing if it is the first pitch or the last pitch was a curve | 7 |

To make it clear exactly how strategy changes took place during the game, an example is provided.

**Strategy Shift Example** The participant begins by facing batters that use the Hacker strategy. Because the consecutive out threshold for this strategy is 4, the participant continues to face batters that use the Hacker strategy until they get 4 consecutive batters out. At this point in time, the strategy shifts to the Aggressive strategy and a new out threshold of 7 is in effect. The Aggressive strategy is then used by the batters until 7 batters are retired consecutively. Game play continues in this fashion until the participant reaches the fifth and final strategy (Alternate). When 7 consecutive Alternate batters are retired by the pitcher the game ends.

[1] *If the batter gets three strikes, then they are out (called a strikeout). If the batter gets four balls, they get a free pass to first base (called a walk).*

### Performance Measure

The participant's ability to learn a particular strategy was measured quantitatively using a measure of pitching efficiency (*PE*). The following formula was used to calculate pitching efficiency:

$$PE = N_s / T_s$$

Where $N_s$ is the number of batters using strategy *s* that were faced by the participant, and $T_s$ is the consecutive out threshold for strategy *s*. A decrease in PE indicates an increase in the efficiency of the pitcher. A value of 1.0 for *PE* indicates the most efficient pitching strategy. For example, if a participant faced 14 Aggressive batters before they could retire 7 in a row, the participant's pitching efficiency would be 14 / 7, or 2.

## Method

Undergraduate Computer Science students at Lock Haven University participated. A total of 10 participants performed the baseball task. Nine of the 10 participants were male.

After signing a consent form, each participant was given instructions explaining the rules of the game. The instructions were similar to those presented here except the information in Table 2 was not provided. As a result, the participant did not know what type of strategies to expect, or when strategy changes would take place. However, the participants were aware that strategies could change during the game.

Participants were given as much time as needed to complete the task and were allowed to consult the instruction sheet during play. All the participants seemed to have no problem understanding the game and no questions were asked while performing the task.

## Models

A total of six cognitive models were written to conduct the study described here. All six models were written using the Herbal high-level language and development environment (Cohen, Ritter, & Haynes, 2005).

The Herbal high-level language is based on the Problem Space Computational Model (PSCM) (Newell, Yost, Laird, Rosenbloom, & Altmann, 1991) and produces productions that can run in both the Soar cognitive architecture (Laird & Congdon, 2005) and Jess (Friedman-Hill, 2003). In this study, the Herbal generated Soar productions were used. However, Jess productions would have also been adequate.

Because of the use of the Herbal high-level language and graphical editor, the creation of the models described here required only an understanding of the PSCM (which provided an infrastructure for model organization) and some visual modeling techniques. This serves as an example of how Herbal can provide modelers without a strong programming background access to the complicated machinery used by architectures that may traditionally be out of their reach. As these models progress towards more sophisticated learning algorithms, the simplified access to Soar and the PSCM will become even more valuable.

All of the models described here are available online as examples at the Herbal website (acs.ist.psu.edu/herbal).

### Batter Models

Five cognitive models were written to represent the strategies used by the batter (Hacker, Aggressive, Random, Chicken, and Alternate). These models are not capable of learning and served only as opponents that exhibit the behavior described in Table 2.

### Pitcher Model

A sixth model was written to play the role of the pitcher. The goal of the pitcher model was to exhibit behavior similar to that demonstrated by the participants. Unlike the batter strategy models, the pitcher model was able to learn using reflection. More specifically, this model operated within two problem spaces: one to deliberate what pitch to throw next, and one to reflect on recent performance and modify future deliberation. The formulation of an explicit reflection phase was simplified by the use of the PSCM and Herbal.

The pitcher model started with an equal probability of throwing a curveball or a fastball. Within the explicit reflection problem space, the pitcher model considers the following features of the environment: the previous number of balls and strikes on the batter, the previous pitch thrown, and the outcome of that pitch. If the outcome is positive (e.g., a strike was called or the batter struck out) the pitcher adjusts a probability so that it is more likely to throw the same pitch the next time it encounters this situation. If, on the other hand, the outcome was negative (a ball or contact by the batter, including contact resulting in an out), and the pitcher had previously experienced a positive outcome in this situation (a strike or a strikeout), the probability of throwing the same pitch in that situation was decreased.

The probability of an action occurring was adjusted by altering working memory so that more or fewer indifferent operators were proposed to throw that pitch type in a given situation. In other words, positive events lead to episodic memory that influences future action. Alternatively, negative events remove episodic memory, reducing this influence. Without prior positive outcomes in a particular situation, no episodic memory elements exist and negative outcomes in that situation are ignored.

An alternative approach to episodic memory would be to use the *numeric-indifferent* preference in Soar (Laird & Congdon, 2005). However, the Herbal high-level language did not support this at the time these models were written.

### Model Parameters

The pitcher model takes two parameters: the learning rate and the unlearning rate. The learning rate specifies how quickly the model will commit to throwing a particular pitch in a particular situation; in other words, how quickly the probability increases given a positive outcome. The unlearning rate specifies how quickly the model will reduce this learned commitment. The best values for these learning rates almost certainly depend on the nature of the particular task.

Considering the relatively simple rules in the baseball task described above, it is expected that participants will be able to learn strategies quickly. In addition, it is hypothesized that participants will at first be reluctant to unlearn until they are sure that a strategy shift has taken place. Given persistent negative feedback on a previously learned response, participants should eventually accelerate their unlearning rate.

Looking at the task environment more closely, further justification of these parameter values can be found in the fact that four of the five batter strategies are deterministic. When a particular pitch works for a batter in a specific situation, it will continue to work until a strategy shift takes place. After a particular pitch stops working for a batter, it can be assumed that a strategy shift has occurred.

As a result, in an effort to match human behavior the pitcher model described here was equipped with a fast learning rate and an initially stubborn, but later accelerating, unlearning rate. Figure 2 depicts the learning and unlearning rates used by the model.

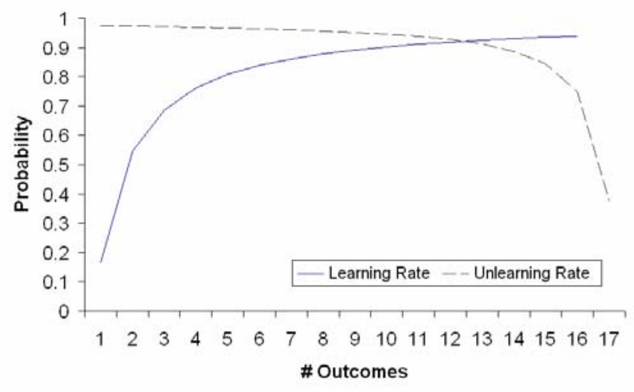


Figure 2: Learning and Unlearning Rates Used by the Model.

## Results

Because a primary goal of this work was to produce a model that not only matches the average pitching efficiency, but also matches the variability in pitching efficiency, the cognitive models created here are not deterministic. This allowed us to consider each run of the model as being equivalent to a participant run. To reduce any sampling error with this theory, the model was run 100 times.

Table 3 shows the results of the participant study and of the model executions. The average pitching efficiency and the standard deviation of the pitching efficiency are listed for all participants and all model runs. Recall that the smaller the pitching efficiency the more efficient the pitcher, and the most efficient strategy has a PE equal to 1.0.

Figure 3 visualizes the data listed in Table 3. Each bar in Figure 3 represents the average pitching efficiency as defined in the Methods section. White bars represent the participant data and shaded bars represent the model data. The error bars in Figure 3 signify one standard deviation from the average pitching efficiency.

Table 3: Pitching Efficiency against Each Batting Strategy for Participants and the Learning Pitching Model.

| | **Participants (n = 10)** | | **Model (n = 100)** | |
|---|---|---|---|---|
| **Strategy** | **Mean** | **StdDev** | **Mean** | **StdDev** |
| Hacker [4] | 1.53 | 0.80 | 1.69 | 0.70 |
| Aggressive [7] | 1.81 | 1.62 | 1.13 | 0.20 |
| Random [5] | 5.00 | 6.24 | 5.36 | 4.67 |
| Chicken [4] | 1.03 | 0.08 | 1.25 | 0.33 |
| Alternate [7] | 1.54 | 0.72 | 3.53 | 2.01 |

## Discussion

Analysis of Figure 3 reveals that the model's behavior matched both the participant's average performance, and variability in performance, for three of the five presented strategies. However, for two of the strategies the model did not satisfactorily reflect the participant's performance.

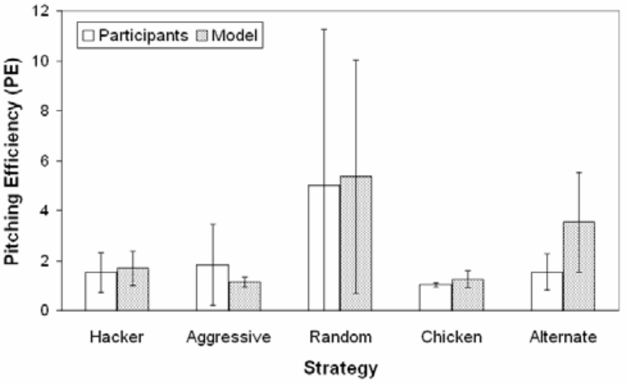


Figure 3: Comparison of Learning Pitching Model and Participants for the Batting Strategies. SDs are shown as error bars.

### Hacker and Chicken Strategies

The model's performance matched very well for both the Hacker and Chicken strategies. Given the simplicity of the learning strategy used, this is an interesting result. Both the participants and the model were able to retire the requisite number of consecutive batters quickly and without much variability. Interestingly, the Hacker strategy proved to be more difficult for both the participants and the model. This may be because the very aggressive strategy used by the Hacker makes it more likely for the batter to get a hit when the pitcher made a mistake. On the other hand, the reserved approach used by the Chicken strategy only punishes mistakes with a single ball as opposed to a hit. In this baseball task, an aggressive batter strategy is more dangerous to the pitcher than a timid one.

### Random Strategy

As expected, the variation of the pitching efficiency against the Random strategy was quite large for both the participants and the model. Both the participant and the model could not consistently figure out the random strategy, because, well, it was random. The difference between the pitching efficiency for the model, and that of the participants, might be related to the number of participants run. Due to the random nature of this strategy, additional participants might cause these averages to match more closely.

### Aggressive and Alternate Strategies

Unexpectedly, the model did not do as good of a job matching the Aggressive and Alternate strategies. The order in which these strategies are presented may play an important role here. One possible explanation for these problems is that the unlearning rate used by the model is not fast enough. While good enough to match the transitions between some strategies, the unlearning rate may need to be faster in other cases. To understand this theory, the transitions from the Hacker strategy to the Aggressive strategy, and from the Chicken strategy to the Alternate strategy, need to be examined more closely.

**Transition From Hacker to Aggressive** Because the Hacker strategy always swings, the pitcher must learn to throw a series of curveballs to get a batter out consistently. In addition, the inability to quickly unlearn the Hacker strategy is not immediately detrimental when an Aggressive batter follows the Hacker strategy. For example, if the pitcher continues to throw a series of curveballs to an Aggressive batter, the batter will not get on base until after the sixth curveball is thrown. This gives the pitcher several pitches, and therefore a lot of time to unlearn the strategy.

On the other hand, if the participant or model quickly unlearns the Hacker strategy, it will lead to throwing an early fastball which will result in a 50% chance of the batter getting a base hit. In other words, in this particular case quickly unlearning the previous strategy is not beneficial. This might explain why the model performed better against the Aggressive strategy; the model simply does not unlearn as quickly as the participants did, and this proved to be more efficient in this particular ordering of strategies.

**Transition From Chicken to Alternate** The opposite can be said about the transition from the Chicken strategy to the Alternate strategy. A series of consecutive fastballs will get a batter out using the Chicken strategy because this strategy never swings. However, if this knowledge goes unlearned, the same series of fastballs thrown to an Alternate batter will result in frequent hits because the Alternate batter swings immediately after a fastball is thrown. In this particular case, failure to quickly unlearn the Chicken strategy results in poor performance and might explain why the model did not perform as well as the participants in this case. Once again, it appears as if the model did not unlearn the learned strategy quickly enough in this particular ordering of strategies.

Unfortunately, our reflective learning strategy is fundamentally limited in how quickly it can unlearn. This limit may be a major reason for the model's inability to unlearn the Chicken strategy quickly enough. Recall that the learning algorithm used here cannot unlearn unless it has already encountered positive feedback. This causes a problem if the model's initial encounter with a strategy involves a series of negative outcomes, which is precisely the case when transitioning from Chicken to Alternate. Augmenting the algorithm to use Soar's *numeric-indifferent* preference might eliminate this limitation and possibly improve the model's fit.

**Additional Explanations** Factors other than unlearning rate may have also had an effect on the model's inability to match the participant's behavior. For example, if the pitcher follows the simple pattern of throwing a fastball, followed by curve, followed by fastball, they will always get the Alternate batter out. While speculative, it is possible that participants were quick to recognize this alternating pattern while the model did not treat alternating patterns any differently from other patterns.

## Conclusions

The paper describes a study to measure the learning of participants performing a competitive task. This task is based on the game of baseball and consists of a participant pitching to a series of batters that use a set of different strategies. Because this task is inspired by the example introduced in *A Gentle Introduction to Soar*, many Soar programmers may already be familiar with its attributes. One outcome of this work is that, over time, the continued use of baseball as a running example might help beginners learn modeling.

In addition, this paper introduced a Soar model written using the Herbal high-level language. This model used a reflective learning process to learn and unlearn various strategies. Another outcome of this work is a downloadable example of how Soar models that learn can be written without directly writing Soar productions. Easily obtainable examples like this will hopefully make Soar models available to a wider audience.

The model's behavior was compared to participants' performance and was shown to match both the participants' average performance and variability in performance against many of the presented batting strategies. This result demonstrates that cognitive models that compete in adversarial environments using introspective learning need not be complicated and can be written quickly and easily using Herbal.

Finally, for the strategies that the model did not satisfactorily master, insight into the limitations of the algorithm used, and how people possibly perform this task was gained. The relationship of the sequences of strategies and how learning is transferred was explored. These results motivate future work that will lead to improvements in the learning algorithm, and in the Herbal high-level language.

## Future Work

The results reported here provide some insights to guide future work. To start, the limitations of the learning algorithm discovered here can be addressed by exploring more sophisticated learning mechanisms (e.g. the meta-learning routines described in (Sun, 2001)).

Further work can also be done to alter the reflection strategy so that certain patterns are easier to learn than others. Patterns that people recognize quickly (such as alternating patterns) might create more intense episodic memories in the model. This change would test the theory that the participants performed well in cases where the solution consisted of a simple and quickly recognizable alternating pattern.

Additional improvements to the model could also be made by enhancing the reflective process so that positive experiences are no longer required in order to benefit from negative experiences. In the absence of positive learned events, negative reflection should still lead to a decrease in the probability of repeating the action. One solution could involve equally increasing the probability of all other possible actions when an event results in a negative outcome. This would be easier to accomplish if the Soar *numeric-indifferent* preference was used to control operator probabilities, and this capability is currently being added to the Herbal high-level language.

There is also scope to explore other parts and versions of the baseball task. For example, the environment and models can be expanded to include other batting strategies, other batter sequences, batting tournaments, and learning batters. In addition, the model created here could be transformed into an Herbal library that can be reused in future models.

Finally, because the Herbal development environment automatically creates both Soar and Jess models, the opportunity exists for comparisons of a single Herbal high-level model running in two very different architectures.

## Acknowledgments

The development of this software was supported by ONR (contract N00014-06-1-0164). Comments from Mary Beth Rosson and Richard Carlson have influenced this work. Recognition is also given to the undergraduate students that agreed to participate in this study.

## References

Bass, E. J., Baxter, G. D., & Ritter, F. E. (1995). Creating models to control simulations: A generic approach. *AI and Simulation Behaviour Quarterly, 93,* 18-25.

Cohen, M. A., Ritter, F. E., & Haynes, S. R. (2005). Herbal: A high-level language and development environment for developing cognitive models in Soar. *In Proceedings of 14th Behavior Representation in Modeling and Simulation,* 133-140. University City, CA.

Cox, M. T., & Ram, A. (1999). Introspective multistrategy learning: On the construction of learning strategies. *Artificial Intelligence, 112,* 1-55.

Friedman-Hill, E. (2003). *Jess in action: Rule-based systems in Java*. Greenwich, CT: Manning Publications Company.

Jones, R. M., Laird, J. E., Nielson, P. E., Coulter, K. J., Kenny, P., & Koss, F. V. (1999). Automated Intelligent Pilots for Combat Flight Simulation. *AI Magazine, 20,* 27-41.

Laird, J. E. (2001a). It knows what you're going to do: Adding anticipation to a Quakebot. *In Proceedings of Fifth International Conference on Autonomous Agents,* 385-392. New York, NY: ACM Press.

Laird, J. E. (2001b). Using a computer game to develop advanced AI. *IEEE Computer, 34*(7), 70-75.

Laird, J. E., & Congdon, C. B. (2005). *The Soar User's Manual Version 8.6*: The Soar Group: University of Michigan.

Lehman, J. F., Laird, J. E., & Rosenbloom, P. S. (1996). A gentle introduction to Soar: An architecture for human cognition. In D. Scarborough & S. Sternberg (Eds.), *An invitation to cognitive science* (Vol. 4). New York: MIT Press.

Newell, A., Yost, G. R., Laird, J. E., Rosenbloom, P., & Altmann, E. (1991). Formulating the problem space computational model. In R. F. Rashid (Ed.), *Carnegie Mellon Computer Science: A 25-Year commemorative* (pp. 255-293). Reading, MA: ACM-Press (Addison-Wesley).

Ritter, F. E., Nerb, J., O'Shea, T., & Lehtinen, E. (Eds.). (2007). *In order to learn: How the sequence of topics influence learning*. New York: Oxford University Press.

Ritter, F. E., Shadbolt, N. R., Elliman, D., Young, R., Gobet, F., & Baxter, G. D. (2002). *Techniques for modeling human performance in synthetic environments: A supplementary review*. Wright Patterson Air Force Base, OH: Human Systems Information Analysis Center.

Ritter, F. E., & Wallach, D. P. (1998). Models of two-person games in ACT-R and Soar. *In Proceedings of Second European Conference on Cognitive Modeling,* 202-203. Nottingham: Nottingham University Press.

Russell, S., & Norvig, P. (2003). *Artificial Intelligence: A Modern Approach* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.

*SML Quick Start Guide*. (2005): ThreePenny Software LLC.

Sun, R. (2001). Meta-learning processes in multi-agent systems. *In Proceedings of Intelligent Agent Technology,* 210-219. Maebashi, Japan: World Scientific, Singapore.

Sun, R. (Ed.). (2006). *Cognition and multi-agent interaction*. Cambridge University Press: New York.

# Toward a Large-Scale Model of Language Comprehension in ACT-R 6

**Jerry Ball[1] (Jerry.Ball@mesa.afmc.af.mil)**
**Andrea Heiberg[2] (Andrea.Heiberg@mesa.afmc.af.mil)**
**Ronnie Silber[3] (Ronnie.Silber@mesa.afmc.af.mil)**
[1]Air Force Research Laboratory/[2]L-3 Communications/[3]Lockheed-Martin
6030 S. Kent St.
Mesa, AZ 85212 USA

## Abstract

We are developing a large-scale model of language comprehension in ACT-R 6 for use in the creation of a synthetic teammate that can function as the Air Vehicle Operator (AVO) in a three-person simulation of an Unmanned Aerial Vehicle (UAV) team performing a reconnaissance mission. The use of ACT-R 6 to implement the core components of this system reflects the strong cognitive modeling orientation of this research. However, the focus is on creating cognitively plausible linguistic and associated non-linguistic representations, rather than modeling the fine-grained time course of language processing as is more typical of ACT-R models. In this regard, an empirical study aimed at discovering evidence of linguistic representations is discussed. Beside the focus on linguistic and non-linguistic representations, the large-scale nature of this effort distinguishes it from typical cognitive modeling research.

## Introduction

We are using the ACT-R 6 cognitive architecture and modeling environment (Anderson et al., 2004; Anderson & Lebiere, 1998) for development of a synthetic teammate. The long-term goal of this research is to develop language-enabled synthetic entities which can be integrated into training simulations. To achieve this goal without detriment in training, the synthetic entities must be capable of closely matching human behavior, including human language behavior. The initial application is the creation of a synthetic teammate capable of performing the functions of a UAV AVO (or pilot) in the three-person Cognitive Engineering Research on Team Tasks (CERTT) simulation (Cooke & Shope, 2005).

This paper describes the current implementation of the language comprehension component of the system in ACT-R 6. The paper begins with a short description of the theory of linguistic representation and language processing underlying model development. It continues with a description of the CERTT lab team task simulation. Following this, the current version of the model is described. A brief discussion of modeling and development tools which facilitate development follows. The paper concludes with a discussion of model validation, describing an empirical study to validate the linguistic representations which are created during processing. Overall, we follow the approach to psycholinguistic research espoused in Crocker (2005) in building a large-scale, if qualitative, model, rather than focusing on the quantitative modeling of narrowly defined and often pathological language processing phenomena.

## Linguistic Representation

The language comprehension model is founded on basic principles of Cognitive Linguistics (Langacker, 1987, 1991; Talmy, 2000; Lakoff, 1987) and Construction Grammar (Fillmore, 1988; Fillmore and Kay, 1993; Goldberg, 1995). In Cognitive Linguistics, all grammatical elements have a semantic basis, including parts of speech, grammatical markers, phrases and clauses. Understanding of language is embodied and based on experience in the world (Lakoff & Johnson, 1999). Categorization is a key element of linguistic knowledge. Categories are seldom absolute, exhibiting, instead, effects of prototypicality, base level categories (Rosch, 1978), family resemblance (Wittgenstein, 1953), fuzzy boundaries, radial structure and the like (Lakoff, 1987). Our linguistic capabilities derive from basic cognitive capabilities—there is no autonomous syntactic component separate from the rest of cognition. Knowledge of language is for the most part learned and not innate. Abstract linguistic categories (e.g., noun, verb, referring expression) are learned on the basis of experience with multiple instances of words and expressions which are members of these categories, with the categories being abstracted and generalized from experience.

Construction Grammar is a linguistic theory based on the notion of *constructions*. "Constructions are stored pairings of form and function, including morphemes, words, idioms, partially lexically filled and fully general linguistic patterns…any linguistic pattern is recognized as a construction as long as some aspect of its form and function is not strictly predictable from its component parts" and even fully predictable constructions may be stored "as long as they occur with sufficient frequency" (Goldberg, 2003).

The focus of this research is on the grammatical encoding of *Referential* and *Relational* meaning (Ball, 2005). In English, these two dimensions of meaning are typically encoded in distinct grammatical poles—a *referential pole* and a *relational pole*—with a *specifier* functioning as the locus of the referential pole and a *head* functioning as the locus of the relational pole. For example, in the expression

The pilot

the determiner "the" functions as a specifier and the noun "pilot" functions as the head. The specifier and head combine to form a *referring expression*, in this example an *object referring expression* (or nominal). Words in English divide into two basic classes: relation (verb, adjective, preposition, adverb) and object (noun, pronoun, proper noun). Relational words presume the existence of other words which express the arguments they relate. Most constructions center on some relational word (e.g., transitive verb construction, predicate adjective construction) which functions as the head of the construction, and is the locus for the encoding of relational meaning—with the construction as a whole constituting a *situation referring expression* (or clause).

Linguistic representations are perceptually grounded in non-linguistic representations of the objects and situations to which they refer. The representations of objects and situations are themselves learned from perceptual-motor (i.e., bodily) experience (cf. Barsalou, 1999). There are no purely abstract concepts that are devoid of perceptual grounding as is assumed in many cognitive theories (cf. Anderson et al., 2007). Concepts may be highly abstract, but they ultimately derive their meaning from a perceptual chain of experience (cf. Harnad, 1990)—in the limiting case perceptual experience of linguistic items themselves. A situation model (Kintsch, 1998; Zwann & Radvansky, 1998) is populated with instances of objects and situations activated by the linguistic input and non-linguistic context.

## Construction-Driven Language Processing

The processing mechanism is based on the *activation*, *selection* and *integration* of constructions corresponding to the linguistic input (Ball, 2007). Activation is a *parallel* process that biases or constrains the selection and integration of corresponding declarative memory (DM) elements into a linguistic representation. Based on the input and prior context, a collection of DM elements is activated via the parallel, spreading activation mechanism of ACT-R.

The selection mechanism is based on the *serial* retrieval mechanism of ACT-R—an alternative to the parallel *competitive inhibition* mechanism typical of connectionist models (cf. Vosse & Kempen, 2000). Retrieval occurs as a result of selection and execution of a production—only one production can be executed at a time—whose right-hand side provides a retrieval template that specifies which type of DM chunk is eligible to be retrieved. The single, most highly activated DM chunk matching the retrieval template—subject to random noise—is retrieved. The retrieval template varies in its level of specificity in accord with the production selected for execution. For example, when a production that retrieves a DM chunk of type *word* executes, the retrieval template may specify the form of the input (e.g., "airspeed") in addition to the DM type *word*. When a production that retrieves a DM chunk of type *part of speech* (POS) executes, the retrieval template may specify the word without specifying the POS—allowing the biasing mechanism to constrain POS determination. There is no assumption that humans use POS labels during language processing, but it is assumed that they categorize word into POS categories.

The retrieved DM chunk is matched on the left-hand side of another production which, if selected and executed, determines how to integrate the DM chunk into the representation of the preceding input. Production selection is driven by the matching of the left-hand side of the production against a collection of buffers (e.g., goal, retrieval, context, short-term working memory) which reflect the current goal, current input and previous context. The production with the highest utility—learned on the basis of prior experience—which matches the input and prior context, is selected for execution—subject to random noise. A default production which simply adds the retrieved DM chunk to a short-term working memory (ST-WM) stack (Ericsson & Kintsch, 1995) executes if no other production matches. The ST-WM stack—which is limited to four linguistic elements—constitutes part of the context for production selection and execution, and implements an extension to the ACT-R architecture.

A key element of the integration process is a mechanism of *context accommodation* which provides for *serial processing without backtracking*. According to Crocker (1999), there are three basic mechanisms of language processing: 1) serial processing with backtracking, 2) parallel processing, and 3) deterministic processing. Context accommodation is an alternative non-backtracking, serial processing mechanism. The basic idea behind this mechanism is that when the current input is unexpected with regard to the previously built structure, the structure is modified to accommodate the current input without backtracking. This mechanism is demonstrated using the example "no airspeed or altitude restrictions". The processing of the word "no" leads to retrieval of an object referring expression (ORE) construction containing the functional elements specifier and head (not all functional elements are shown):

$[\texttt{spec head}]_{\texttt{obj-refer-expr}}$

"No" is integrated as the specifier in this construction and expectations are established for the occurrence of the head:

$[\textit{no}_{\text{spec}}\ \texttt{head}]_{\texttt{obj-refer-expr}}$

This ORE construction is made available in the ST-WM stack to support subsequent processing. The processing of the noun "airspeed" leads to activation and selection of a head construction which contains the functional elements modifier and head, with "airspeed" functioning as the head:

$[\texttt{mod airspeed}_{\text{head}}]_{\texttt{head}}$

The head construction is integrated into the ORE construction.

$[\textit{no}_{\text{spec}}\ [\texttt{mod}\ \textit{airspeed}_{\text{head}}]_{\text{head}}]_{\texttt{obj-refer-expr}}$

The processing of the conjunction (or disjunction) "or" leads to its addition to the ST-WM stack since the category of the first conjunct of a conjunction cannot be effectively determined until the linguistic element after the conjunction is processed—due to rampant ambiguity associated with conjunctions. Note that delaying determination of the category of the first conjunct until after processing of the linguistic element following the conjunction provides a form of deterministic processing reminiscent of Marcus's deterministic parser (1980). The processing of the noun "altitude" in the context of the conjunction "or" and the ORE "no airspeed" with head noun "airspeed" results in the accommodation of "altitude" such that the head of the ORE is modified to reflect the disjunction of the nouns "airspeed" and "altitude".

$$[\mathit{no}_{\mathtt{spec}}\ [\mathtt{mod}\ (\mathit{airspeed\ or\ altitude})_{\mathtt{head}}]_{\mathtt{head}}]_{\mathtt{obj\text{-}refer\text{-}expr}}$$

The processing of "restrictions" in the context of the ORE "no airspeed or altitude" results in the accommodation of "restrictions" such that the current head "airspeed or altitude" becomes the modifier and "restrictions" becomes the head. The final representation has the form:

$$[\mathit{no}_{\mathtt{spec}}\ [(\mathit{airspeed\ or\ altitude})_{\mathtt{mod}}\ \mathit{restrictions}_{\mathtt{head}}]_{\mathtt{head}}]_{\mathtt{obj\text{-}refer\text{-}expr}}$$

This representation was arrived at using a serial processing mechanism without backtracking, despite the rampant local ambiguity of the utterance!

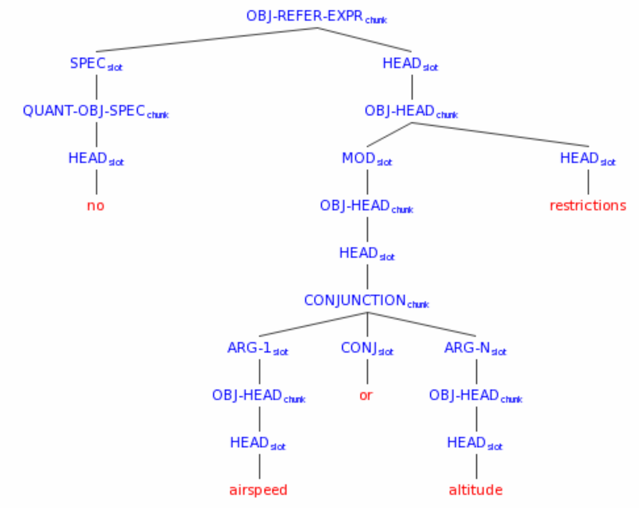


**Figure 1: No altitude or airspeed restrictions**

Context accommodation is a powerful serial processing mechanism which overcomes the limitations and cognitive implausibility of serial processing with algorithmic backtracking, full parallel processing, and full deterministic processing. Context accommodation is closely related to Lewis's notion of "limited repair parsing" (Lewis, 1998), although context accommodation is considered part and parcel of the normal processing mechanism and is not viewed as a repair mechanism. Regarding parallel processing, it is not cognitively plausible to carry forward more than a few possible representations at once, which means that a mechanism like context accommodation is needed to handle the case where the correct parse isn't in the parallel spotlight. Likewise, deterministic mechanisms require delaying integration of linguistic elements for indeterminate periods—requiring their separate representation—which is likely to exceed the limited capacity of ST-WM if used extensively.

## Synthetic Teammate

The CERTT Lab is a research facility for studying team performance and cognition in complex settings. CERTT's UAV-STE (Synthetic Task Environment) is a three-person task in which each team member is provided with distinct, though overlapping, training; has unique, yet interdependent roles; and is presented with different and overlapping information during the mission (Cooke & Shope, 2005). The overall goal is to fly the UAV to designated target areas and to take acceptable photos at these areas. The Air Vehicle Operator (AVO) controls airspeed, heading, and altitude, and monitors UAV systems. The Payload Operator (PLO) adjusts camera settings, takes photos, and monitors the camera equipment. The Data Exploitation, Mission Planning, and Communication Operator (DEMPC) oversees the mission and determines flight paths under various constraints. To successfully complete a mission, the team members need to share information with one another in a coordinated fashion.

Most communication is done via microphones and headsets, although some involves computer messaging. A set of initial speech transcripts has been collected from the UAV-STE for a number of teams. These transcripts are being used to guide development of the model. A portion of a transcript appears below.

PLO: AVO, can I please be about 3000 feet or higher, please? Cancel. Cancel.

AVO: Do I need to change my airspeed? I mean my altitude.

DEMPC: Once I get the first, uh, sequence figured out, I'll let you know. First waypoint LVN is an, uh, ROZ access point. There is no flight restrictions, but the, uh, radius is, uh, 2.5 miles. I'm pretty sure you can take a bearing towards H-area now. It looks like you're in within the 2.5 required for this entry point.

PLO: AVO, can I please, uh, keep, uh, altitude over 3000 feet for this picture, please? Can you give me a range?

DEMPC: The next target H-area has a range of 5 miles.

PLO: Copy.

AVO: Was that above 3000?

PLO: Yes, please. Can you also keep this current airspeed?

AVO: OK.

DEMPC: Next waypoint is H-area. There is no altitude restriction, but the speed restriction is between 50 & 200.

The language used by team members is not constrained; there is no special restrictive grammar. Over the three transcripts analyzed so far, the average number of utterances is 2300 per transcript; average utterance length is 7 words. There are 1300 unique words across transcripts, including 50 special vocabulary items related to the task. In each transcript, an average of 27% of the words are unique to that transcript.

The transcripts contain a number of grammatical features that are challenging from a language processing perspective:

- Multiword expressions: "Picking up the pace."
- Complex object referring expression (or nominal):
  - "First waypoint LVN is an, uh, ROZ access point."
  - "Do you have any additional altitude or speed restrictions that I need to get from you?"
  - "Can I get the 2, 3, 4, and 5 current setting ranges?"
- Anaphora:
  - "PLO, is this a photo?" (current waypoint)
  - "Does that make sense?" (previous statement)
- Complex verb argument structure: "Can I keep altitude over 3000 feet for this picture?"
- Corrections:
  - "DEMPC to PLO, effective radius is, uh, 2, uh, 5 miles, sorry about that."
  - "Do I need to change my airspeed? I mean my altitude."
- Ambiguous closed-class words, e.g., "that":
  - Complementizer: "They just told me that there's gonna be a priority target in this area that we're entering."
  - Object referring expression: "You already told me that."
  - Determiner: "Got a good photo on that one."

## The Current Model

The language comprehension model is currently capable of processing a range of grammatical constructions attested in the transcripts, including:

- Intransitive verb: "You can go."
- Transitive verb: "We already hit [$_{\text{OBJ}}$ ROW]."
- Ditransitive verb: "You can give [$_{\text{IOBJ}}$ me] [$_{\text{OBJ}}$ R-STE]."
- Verb taking clausal complement: "You told [$_{\text{IOBJ}}$ me] [$_{\text{SITCOMP}}$ the altitude restriction was below 3000 feet]."
- Auxiliary verb: "I would have had a wrong picture."
- Predicate nominal: "First waypoint is LVN."
- Predicate adjective: "Altitude is stable."
- Predicate preposition: "We are in those constraints."
- Attributive adjective modifier: "It's a good picture."
- Adverbial modifier: "Our altitude still should be fine."
- Complex nominal: "The next photographic target point is M-STR."
- Nominal conjunction: "We will maintain current airspeed and altitude."
- Sentence conjunction: "The entry is KGM and the exit is FRT."

The model creates a linguistic representation of the input, but doesn't yet map that representation to the corresponding objects and situations in the situation model.

The language comprehension model is approaching a scale and complexity atypical of most cognitive models. Verifying that the model generates theoretically motivated linguistic representations is an important on-going aspect of the project. Inputs to the model are comprised of actual utterances from the UAV-STE transcripts and a set of canonical phrases and sentences. The verification strategy includes running the model against this set of inputs, and testing that the model produces the expected output.

The model generates linguistic representations which include such information as phrase constituency, predicate/argument relations, head/modifier relations, and head/specifier relations. Linguistic representations are complex structures of DM chunks. For testing, the DM chunk structure is converted into a graphical representation (automatically generated with phpSyntaxTree, Eisenbach & Eisenbach, 2006) shown in Figure 2 (below).

At a gross level, testing is fully automated. The complex output structure (e.g., Figure 2) is traversed in left-to-right order, and the terminal symbols are reassembled into a string (e.g., "I increased the airspeed"). This output string is compared to the input string; any mismatches are flagged for further investigation. At a more detailed level of testing, the output representation is hand-checked to ensure its validity. Valid output representations are stored as the known-good baseline. A capability to dynamically visualize the evolving DM representation during the processing of each word in an input text also exists (Heiberg, Harris & Ball, 2007). Any further changes to the model may be easily regression tested by regenerating the outputs, and comparing them to the known-good baseline with an automated file comparison tool. This set of methodologies taken together helps facilitate the development of a large scale and complex model.

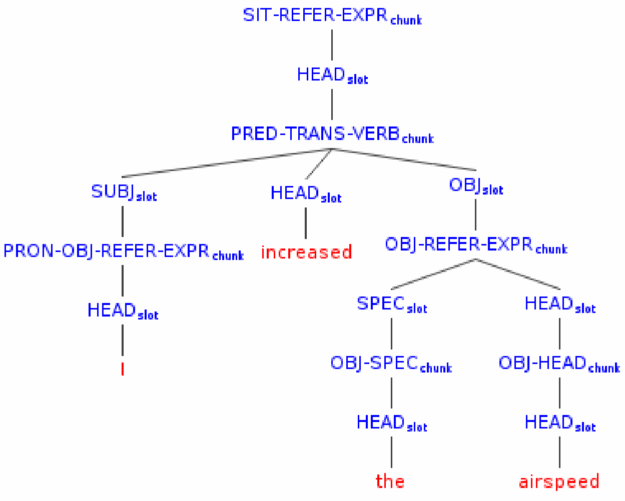


**Figure 2 Graphical Representation**

## Model Validation

We are committed to the development of a cognitively plausible model of language comprehension. However, we are not modeling the fine-grained time course of processing during language comprehension. As Just and Carpenter (1987) note, "in most cases…syntactic and semantic analyses occur concurrently with other processes that are longer and more variable in duration". It is only in the processing of unusual texts like Garden-Path and center embedded sentences that syntactic and semantic influences on processing are exposed. Instead, our focus is on validating the linguistic and non-linguistic representations that are generated during processing of more ordinary texts as reflected in our UAV team task corpus. Our approach aligns with Crocker (2005), who argues for "an alternative approach to developing and assessing theories and models of sentence comprehension" in which "a model's coverage should not be limited to a few 'interesting' construction types, but must also extend to realistically large and complex language fragments, and must account for why most processing is typically rapid and accurate".

To validate the model, we have devised a multi-part empirical study to identify the kinds of linguistic representations that humans create during language processing. Some preliminary data from a pilot study involving 20 subjects are reported.

In one part of this study subjects are asked to determine if paired expressions differ in meaning. For example, does "the man bit the dog" differ in meaning from "the dog bit the man" (all 20 subjects said "yes")? Does "this book" differ in meaning from "that book" (all 20 subjects said "yes")? Does "the old house on the hill" differ from "an old house on a hill" (14 subjects said "yes")? A difference in meaning indicates that either the different lexical items or the different structural arrangement of the lexical items in the paired expressions affects the meaning.

In another part, subjects are asked to group expressions into meaningful units of various sizes given the overall meaning of the linguistic expression. A key question for this part is whether the preferred representation of a clause aligns with the Subject-Predicator$_{\text{Head}}$-Object construction put forward in most theories of Functional Grammar (cf. Halliday & Matthiessen, 2004), the S → NP VP (i.e., Subject-Predicate) construction put forward in Generative Grammar (Chomsky, 1965) or the ReferencePoint-PredSpec-Predication construction specific to Double R Theory (Ball, 2005), but related to the Mood-Residue construction of Halliday & Matthiessen (2004). Also of interest is whether this preference varies from clause to clause and from subject to subject. For example, in "he is kicking the ball" do subjects prefer to group "is" with "kicking" (as part of the Predicator$_{\text{Head}}$ "is kicking"), with "kicking the ball" (as part of the Predicate "is kicking the ball") or with "he" (as part of the ReferencePoint-PredSpec "he is")? Preliminary results indicate the identification of the entire sentence as a meaningful group by 13 subjects. The second most common grouping, "kicking the bucket", was only identified by 4 subjects. "He is" was identified as a group by 3 subjects. The group "is kicking the ball" was not identified by any subjects. Overall, subjects tend not to include function words (e.g., "the", "is") in meaningful groups, making it difficult to assess these results, and perhaps making it necessary to revise the methodology.

In a third part subjects are asked to rank the relative contributions of various words to the overall meaning of the expression. In conjunction with the grouping task, the ranking task will allow us to identify the semantically most important word in a meaningful group, which we take to be the head of the group. For sentences containing transitive verbs, subjects identified the head of the subject as most meaningful, the main verb as second most meaningful and the head of the object as third most meaningful. The ranking of heads of subjects as more important than main verbs suggests that relational structure is not the only dimension of meaning which influences this decision.

Finally, in a fourth part, subjects are asked to identify the part of speech (POS) of a word in different expression contexts, for example, "running" in "the man is running" vs. "the running man". This part is intended to get at whether or not words are separately represented in the mental lexicon for each possible grammatical function they can fulfill (e.g., head, modifier, complement, specifier), as indicated by the POS labels they are assigned. For example, if "running" is treated as a verb in "the man is running" but as an adjective in "the running man" this might indicate separate representations in the mental lexicon. If, on the other hand, "running" is labeled a verb in both uses, a single entry in the mental lexicon is suggested. Preliminary results suggest that subjects tend to treat words in different functions as having the same part of speech. For example, 18 subjects call "boy" in "the boy" a noun; 11 subjects called "altitude" in "no altitude restrictions" a noun (6 subjects called it an adjective); 16 subjects called "home" in "he went home" a noun (only 1 subject called it an adverb); and 14 subjects called "president" in "George Bush is president" a noun (3 subjects called it an adjective).

To the extent that the linguistic representations generated by the language comprehension model are consistent with the results of this empirical study, the linguistic representations will have more validity. To some extent this validity hinges on whether or not humans have explicit knowledge of the linguistic representations they generate during language comprehension, and whether the empirical study is successful in tapping into that knowledge. It is a general assumption of the empirical study that humans have explicit knowledge of the linguistic representations they create. This assumption is motivated by the model's creation of linguistic representations composed of DM chunks, which suggests that these representations can be explicitly attended to and cognitively manipulated, and by rejection of the *autonomy of syntax* assumption of generative grammar, with its informationally encapsulated (and hence implicit) syntax module. Although the mechanisms by which linguistic representations are constructed may be largely implicit, the resulting representations are declarative and explicit. For example, humans explicitly know what “the man bit the dog” means. They explicitly know that “the man” refers to a man, “the dog” refers to a dog, “bit” establishes a relation of biting between the man and the dog, with the expression as a whole referring to a situation in which it is the man who is doing the biting, and the dog who is being bitten.

## Conclusion

We are using the ACT-R cognitive architecture in the development of a language-enabled synthetic teammate intended to closely match human behavior. To date, the research has focused on the development of the language comprehension component of the system. This component has approached a scale at which the need for development and testing tools has become important. The goal for the project is to maintain cognitive plausibility by adhering to well-established theoretical constraints from cognitive linguistics and cognitive psychology, as the system grows. We believe these constraints will actually facilitate development of a functional system (Ball, 2006).

## Acknowledgements

This research is funded by the Warfighter Readiness Research Division, Human Effectiveness Directorate, Air Force Research Laboratory. We thank Nancy Cooke and Steve Shope of CERI for providing access to the CERTT lab UAV-STE and transcripts.

## References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S, Lebiere, C, and Qin, Y. (2004). An Integrated Theory of the Mind. *Psychological Review* 111, (4). 1036-1060.

Anderson, J. R. & Lebiere, C. (1998). *The Atomic Components of Thought*. Mahwah, NJ: LEA.

Anderson, J. R., Qin, Y., Jung, K-J, & Carter, C. (2007). Information-processing modules and their relative modality specificity. *Cognitive Psychology*, 54, 185-217.

Ball, J. (2005). A Bi-Polar Theory of Nominal and Clause Structure and Function. In *Proceedings of the 27th Annual Cognitive Science Society.*

Ball, J. (2006). Can NLP Systems be a Cognitive Black Box? In *Papers from the AAAI Spring Symposium, Tech Report SS-06-02,* 1-6, Menlo Park, CA: AAAI Press.

Ball, J. (2007). Construction Driven Language Processing. In *Proceedings of the 2nd European Cognitive Science Conference.*

Barsalou, L. (1999). Perceptual Symbol Systems. In *Behavioral and Brain Sciences* 22, 577-609.

Chomksy, N. (1965). *Aspects of the Theory of Syntax*. Cambridge, MA: The MIT Press.

Cooke, N. & Shope, S. (2005). Synthetic Task Environments for Teams: CERTT’s UAV-STE. *Handbook on Human Factors and Ergonomics Methods*. 46-1-46-6. Boca Raton, FL: CLC Press, LLC.

Crocker, M. (1999). Mechanisms for Sentence Processing. In Garrod, S. & Pickering, M. (eds), *Language Processing,* London: Psychology Press.

Crocker, M. (2005)**.** Rational models of comprehension: addressing the performance paradox. In A. Culter (ed) *Twenty-First Century Psycholinguistics: Four Cornerstones*, LEA, Hillsdale.

Eisenbach, A. & Eisenbach, M. (2006). phpSyntaxTree tool, http://ironcreek.net/phpsyntaxtree.

Ericsson, K & Kintsch, W. (1995). Long-term working memory. *Psychological Review*, 102 211-245.

Fillmore, C. (1988). The Mechanisms of Construction Grammar. *BLS* 14: 35-55.

Fillmore, C. and Kay, P. (1993). *Construction Grammar Coursebook*. Berkeley, CA: Copy Central.

Goldberg, A. (1995). *A Construction Grammar Approach to Argument Structure*. Chicago: The University of Chicago Press.

Goldberg, A. (2003). Constructions: a new theoretical approach to language. *TRENDS in Cognitive* Sciences. Vol. 7, No. 5, pp. 219-224.

Halliday, MAK & Matthiessen, C (2004). *An Introduction to Functional Grammar*. NY: Oxford University Press.

Harnad, S. (1990). The Symbol Grounding Problem. *Physica D*, 42: 335-346.

Heiberg, A., Harris, J. & Ball, J. (2007). Dynamic Visualization of ACT-R Declarative Memory Structure. In *Proceedings of the 8th International Conference on Cognitive Modeling*.

Just, A. & Carpenter, P. (1987). *The Psychology of Reading and Language Comprehension*. Boston: Allyn and Bacon, Inc.

Kintsch, W. (1998). *Comprehension, a Paradigm for Cognition*. New York, NY: Cambridge Univ Press.

Langacker, R. (1987, 1991). *Foundations of Cognitive Grammar,* Volumes 1 & 2. Stanford, CA: Stanford University Press.

Lakoff, G. (1987). *Women, Fire and Dangerous Things*. Chicago: The University of Chicago Press

Lakoff, G. & Johnson, M. (1999). *Philosophy in the Flesh: The embodied mind and its challenge to Western thought*. New York: Basic Books.

Lewis, R.L. (1998). Leaping off the garden path: Reanalysis and limited repair parsing. In Fodor, J.D. and Ferreira, F. (eds), *Reanalysis in Sentence Processing*. Boston: Kluwer Academic.

Marcus, M. (1980). *A Theory of Syntactic Recognition for Natural Language*. Cambridge, MA: The MIT Press.

Rosch, E. (1978). Principles of Categorization. In *Cognition and Categorization*. Edited by E. Rosch & B. Lloyd. Hillsdale, NJ: LEA

Talmy, L. (2000). *Toward a Cognitive Semantics, Vols I and II*. Cambridge, MA: The MIT Press

Vosse, T. & Kempen, G. (2000). Syntactic structure assembly in human parsing. *Cognition*, 75, 105-143.

Wittgenstein, L. (1953). *Philosophical Investigations*. New York: MacMillan.

Zwann, R., and Radvansky, G. (1998). Situation models in language comprehension and memory. *Psychological Bulletin*, 123, 162-185.

# Attention and Association Explain the Emergence of Reasoning About False Beliefs in Young Children

**Paul Bello (Paul.Bello@rl.af.mil) & Perrin Bignoli (Perrin.Bignoli@rl.af.mil)**
Air Force Research Laboratory
Information Directorate; 525 Brooks Rd.
Rome, NY 13441 USA
**Nicholas Cassimatis (cassin@rpi.edu)**
Rensselaer Polytechnic Institute
Department of Cognitive Science; 110 8th St.
Troy, NY 12180 USA

## Abstract

Charting and explaining the development of young children's capacity to reason about mental states is a mainstay activity among developmental psychologists interested in how theory-of-mind (ToM) is acquired. These explanations are typically couched within one of the traditional frameworks for studying mental-state attribution: the *theory theory* and the *simulation theory*. This paper presents an analysis of the positions adopted on the issue of ToM development when subscribing to each of these frameworks, and argues for an alternative explanation of development based on a simple associative learning mechanism that appropriately shifts the child's cognitive focus of attention when asked to make predictions about the actions of others. We develop this notion within the confines of Wimmer and Perner's classic false belief task, and describe a cognitive model implemented within the *Polyscheme* computational cognitive architecture that realizes the development process.

## Introduction

One of the most contentious areas in the literature on cognitive development concerns the acquisition of a fully matured theory-of-mind. Theory-of-mind (ToM) is broadly defined as the capacity to predict and explain the behavior of other agents by appealing to unobservable mental states such as their beliefs, desires, and intentions. Many byproducts of higher-order cognition such as the ability to participate fruitfully in discourse with others, our ability to assign blame through ascriptions of intentionality, to our very notion of ourselves as beings who have mental states that change through time seem to be served by our possession of a mature theory-of-mind. However, it remains unclear how such a remarkable ability emerges during the developmental process. An answer to this question clearly depends on the answers to a number of other more fundamental questions concerning the nature of mental states and how they are used within the human cognitive architecture. Nonetheless, various theories have given accounts of how ToM might develop, conditioned on relatively imprecise definitions of what it means for one to have a belief, desire, or an intention. We will give an alternative theoretical account of what it means to "believe" something (or believe that someone else believes something), and show how such an account can explain ToM emergence while avoiding the pitfalls encountered by the most common theoretical frameworks for explaining development. We will employ the Polyscheme computational cognitive architecture to explain shifts in children's responses on the false belief task (Wimmer & Perner 1983), which is widely regarded as a "gold standard" for illustrating differences between children who have and have not developed a mature ToM.

## Frameworks for Explaining Development

The vast majority of researchers concerned with ToM development adopt one of three popular theoretical positions. The first and most widely adopted position is that of the "child as scientist." This position is usually called the *theory theory* (Carey 1985, Gopnik & Meltzoff 1997), and states that our knowledge of mental states is arranged in a theory-like set of interrelated concepts that subserve prediction and explanation of behavior. These "commonsense theories" comprise our domain-specific, intuitive notions about such domains as biology (Medin & Atran 1999), physics (Spelke et. al. 1992), and psychology. The second of these frameworks is the *simulation theory* and it's variants (Goldman 2006, Gordon 1986) which casts our ability to predict and explain behavior as the result of mental simulations within which we assume the perspective of the agent to be reasoned about, and use our own cognitive capacities to approximate those of the agent. We turn to the discussion of each of these positions with specific emphasis on the issue of development – the learning and/or maturational components of ToM. We do so by explaining the behavior of subjects in the classic false belief task. The task consists in showing subjects a storyboard comprised of the following pictorial representations, with appropriate narration by the experimenter:

- A little boy (Maxi) is in the kitchen with his mother, and he puts his chocolate on the counter.
- Maxi goes outside to play ball.
- While he is outside, Maxi's mother puts his chocolate in the kitchen cabinet.

The subjects are then asked where Maxi will look for his chocolate when he goes back into the kitchen. Many variations on this experiment exhibit the same general trend in the data: that between three and four years of age, children switch in the type of responses that they give from answering "the cabinet" (which is the typical three year old response) to answering with "the counter" (which is the typical four year old response) (Wellman, Cross & Watson 2001). What does modern developmental theory say about this interesting pattern of responses?

### Theory theory

One way to explain the distinction between three and four-year old behavior on the false belief task is to assume that children possess a set of interrelated concepts like belief, desire, and intention, which are used in predicting the actions that others will take. In general, to succeed on the task, the theory-theorist claims that children are able to think about the following:

> He/She believes that $p$.
>
> or:
>
> Persons who want that $p$ and believe that $q$ would be sufficient to bring about $p$ and have no conflicting wants or preferred strategies will try to bring it about that $q$.

Where $p$ and $q$ are propositions. Being in possession of such knowledge requires an explicit concept of belief (as a predicate, for example), which requires a psychological theory within which to couch it. On the theory-theory account, development in children consists of gradually acquiring knowledge, and subsequently revising fragments (or the totality) of their theory. In order to use the theory, various information-processing mechanisms need to be properly functioning. So, according to the theory-theory, children's failure on the false belief task must either be the result of lacking the necessary body of concepts/knowledge or immaturity of the information-processing mechanisms that use this information in making predictions. The developmental process is some mixture of knowledge acquisition, a maturing facility for information-processing, and the acquisition of new concepts – specifically a so-called "representational" concept of belief. This raises some interesting questions: how can one separate contributions to failure made by limitations on information processing and those resulting from conceptual deficits? How is such a theory representationally structured, and what sort of implications might this have for learning? It seems that committing oneself to representing statements of the form: "He/She believes that $p$" requires us to relearn new theory for every agent in whom we come in contact with. How does the theory-theory account for the fact that targets often arrive at different conclusions and take different actions than we as predictors would? Does this imply that we need to have theory corresponding to each agent's inferential mechanisms? Finally, how could theory-theory explain the enormous number of beliefs that we attribute to agents with whom we've had no contact at all? We are certainly able to make detailed predictions about readers of this paper without ever having met them, and as it turns out, three year-old children are capable of doing the same (Nichols & Stich 2003).

### Simulation Theory

Another way one could potentially predict and explain the behavior of others is through a process of mental simulation. The simulation theory was developed as a reaction to some of the problems that seem to be unavoidable when adopting a purely theory-driven framework for prediction. In order to succeed on the false belief task, an agent implementing simulation theory needs to be able to entertain beliefs of the form:

> $p$

Along with being able to think about such propositions, the agent must be able to imagine a counterfactual world in which it imaginatively identifies with the target it wishes to predict. Explaining development in light of this very general simulation theory is simple: children at three and four years of age do not possess different knowledge or conceptual structures, they merely become more adept at identifying with other people in the context of simulation. The imaginings necessary for simulation are ultimately linked to information processing capabilities, and changes in this capability produces observable developmental transitions in children. However, the simulation theory suffers from some immediate difficulties. Even if we are able to imaginatively identify with others and impute all of our own beliefs to them within the context of a counterfactual situation, it remains unclear how to account for differences between the simulator and the target. In the false belief task, it seems rather unsatisfying to assert that the subject "imaginatively identifies" with Maxi, and magically knows that the chocolate is on the counter. Rather, it seems as if the subject would need to employ a number of *overrides* to his current knowledge of the real world in order to make such a claim. The subject would also seemingly have to possess some form of knowledge about perceptual occlusion in order to come to an appropriate starting point for simulating Maxi's mental life while he is outside playing. While all of this extra knowledge may not be related to mental states, per se, it defeats the purpose of simulation – which is supposed to be an information-poor process as opposed to theory-laden, information rich-process.

## Prior Modeling Work

The only prior modeling work on the developmental transition between three and four years of age is a Bayesian analysis presented in (Goodman et. al. 2006). The authors adopt a theory-theoretical perspective on development, and chart the transition of the child from a "copy theorist" (CT) who maintains that all events in the real world are copied as knowledge possessed by the target, and a so-called "perspective theorist" (PT) who uses other information to mediate what gets attributed to other agents. In the case of the false-belief task, this other information takes the form of knowledge about visual access. The learning process is surprise-driven, and is as follows:

1. Children start out with two theories, CT, and PT. CT is originally preferred to make predictions, since it

includes no extra information (in this case regarding the target's visual access).

2. Predictions are made with CT. Normally these predictions are successful and unsurprising since the target is usually in the presence of the predictor-agent, and there is no discrepancy in visual access.

3. Situations in which the target possesses a false belief are incorrectly predicted by CT, resulting in a high surprise value.

4. These same situations (and typical non false-belief situations) are correctly predicted by PT. Slowly, PT becomes the preferred device for predicting and explaining the target's behavior.

This is an elegant, rational explanation for development, but it leaves a few lingering questions to be answered. The models to be selected from are presumed to be probabilistic graphical models (Pearl 1988, 2000). In these models, there are nodes representing the "belief" of the target. It seems odd that on the one hand, theory-theory claims that a conceptual change happens between three and four regarding children's understanding of belief, yet these models presume children already have a sufficient representation of belief as copy-theorists. Rather, it is understanding that visual access is coupled to belief that allows children to become perspective-theorists. Much research has gone into children's understanding of visual access as it relates to beliefs, and the verdict seems to be in favor of a very early understanding that visual access factors into what others know (Lempers et. al. 1977, Gopnik & Slaughter 1991), leaving in question that switching between these two *particular* models should play any role in explaining four year old behavior. It's also clear, given a purely Bayesian interpretation of commonsense theories, that representing beliefs about beliefs about beliefs, *et cetera* becomes prohibitively costly from a computational standpoint, since entire networks would have to be recopied in memory to keep the relevant conditional probabilities from having unwanted influence on one another.

## The Cognitive Substrate Hypothesis

One of the hallmarks of human cognition is the range of situations it is capable of adapting itself to. Presumably, our evolutionary forebears did not need to fabricate microprocessors, sail yachts, formulate grand unified theories, or do the majority of other activities that we are capable of doing in our current day and age. Instead, they most likely needed to be able to reason more generally about the nature of objects in their physical environments, and be able to make better predictions about their behavior in order to maximize their chances for survival in what we assume to be extremely inhospitable conditions. The Cognitive Substrate Hypothesis states that a (relatively) small set of properly integrated, domain-general computational mechanisms can provide a mechanistic explanation for much, if not all of higher-order cognition. While many different suggestions could potentially be made in defining a cognitive substrate as we've described above, our particular selection of domain-general mechanisms are motivated by developmental studies of physical reasoning. In order to successfully exist in a dynamic physical environment one must proficiently reason about a core set of domains including but not limited to: time, space, parts/wholes, paths, instrumental desires, events, identity/similarity, situations/worlds, and causality (including learning causal contingencies). We see computational functionality in these domains as being absolutely critical to the survival of most (if not all) primates, but especially humans - however it might be implemented.

## Polyscheme: A Substrate Implementation

We have chosen to conduct our exploration using the Polyscheme computational cognitive architecture (Cassimatis 2005), which was originally designed to integrate multiple computational mechanisms corresponding to aspects of higher and lower-level cognitive processes. Polyscheme consists of a number of specialists which maintain their own proprietary representations that communicate with one another during problem-solving through coordination via a cognitive focus of attention. The selection of this particular implementation of attention is motivated by the existence of processing interference in the Stroop effect (Stroop, 1935), which suggests that multiple mental processes operate simultaneously (word and color recognition, for example). Visual attention has also been demonstrated as an integrative mechanism for inputs from multiple sensory modalities (Triesman & Gelade, 1980). Polyscheme is based on the notion that just as the perceptual Stroop effect can be generalized to higher-level non-perceptual cognition, that integrative perceptual attention suggests the existence of a higher-level cognitive focus of attention that is the mind's principle integrative mechanism.

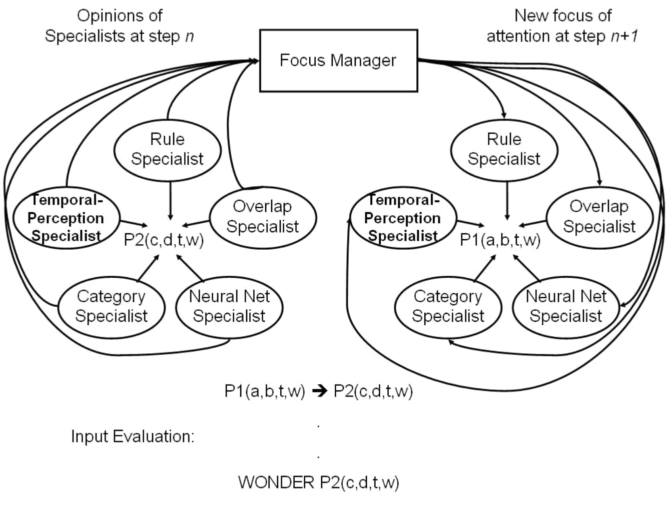


Figure 1: Focus Management in Subgoaling

Integration among specialists implementing their own computational methods is achieved through two basic principles: the multiple implementation principle (MIP), and the common function principle (CFP). The common function principle states that each specialist implements a core set of common functions, including forward inference, subgoal generation, identity matching, and simulation of alternative worlds. The multiple implementation principle states that many different computational models (i.e. rules, neural networks, etc.) implement common functions. Details concerning how the focus of attention coordinates sequences of common functions generated by multiple representations can be found in (Cassimatis 2005).

## Polyscheme: Some Formal Preliminaries

Strings of the form $P(x0,...,xn,t,w)$ are called propositions. Simply stated, P is a relation (i.e. Loves, Hates, Color, MotherOf) over the set of objects $xi$ during the temporal interval t (which could be a single time point, or a range of time points) in the world w which bears a truth value. A proposition's truth value is a tuple $< E+, E- >$ consisting of the positive (E+) and negative (E-) evidence for that proposition. Evidence takes on one of the following values: $\{C, L, l, m, ?\}$ representing certainly, very likely, likely, maybe, and unknown. So, if we are given the proposition $Likes(John, Mary, t, w)$, and come to find out its truth value is ¡m,m¿, we can say that at some time t, maybe John likes Mary, and maybe he doesn't. If at some later time say $t+1$, we find a note written by John to his friend, expressing his affection for Mary, we may update the truth value of this proposition, with $Likes(John, Mary, t+1, w)$ taking on the value $< C, ? >$. Sometimes the letter $E$ will appear in the proposition where a temporal argument would normally be. $E$ represents "eternity" and denotes the temporal interval containing all other temporal intervals. Similarly, one might observe the letter $R$ in the proposition where a world argument would normally be. $R$ represents the real world, which consists of objects and relations in the environment appropriately transduced into propositions by Polyscheme's perceptual machinery. This is the world as Polyscheme experiences it. Letters other than $R$ in the world argument of the proposition could represent hypothetical, future, counterfactual or past states of the world. We will exploit this functionality when describing how to perform ToM-driven inference.

## A Substrate Mapping for Social Cognition

Based on a variety of evidence from the empirical literature, we have minimally extended the cognitive substrate for physical reasoning to accommodate reasoning about the mental states of others. The domain-general functionality for reasoning about physical objects includes a general purpose spatial competency, a mechanism for reasoning about identity, functionality that keeps track of the truth-values of perceived objects/relations through time, a simple associative learner, a system that keeps track of the truth value of propositions in different worlds (used for planning under uncertainty), and a rule-based reasoner. In addition, we add three new pieces of functionality that we believe are well-justified in the literature on child development: a mechanism to keep track of lines-of-sight (Hood et al. 1998), a mechanism for detecting (specifically) human agency (Guajardo & Woodward 2004), and a mechanism for generating exceptions for rules about self/other identity. Specifically, we use the latter to selectively override the version of *Leibnitz' Law*, which states that two objects are the same just in case they share all of the same properties. To even claim these three mechanisms as "additions" seems to be somewhat dubious as well, since we can imagine uses for all three of these functions in non-social cognition. For example, it is plausible to assume that an agency detector could be used to constrain search while performing object tracking, and we're convinced of the fact that overrides to the identity hypothesis are used frequently, especially in the case of early pretense, where features of a source object must be replaced with imaginary features of a target object. On our account, representing the "beliefs" of other agents consists of detecting agency, which causes the simulation of a counterfactual world $w$ in which the identity $Same(self, other, E, w)$ is true. Beliefs held by "self" (i.e. propositions which are true in the real world) are *inherited* into the counterfactual world $w$ with a slightly weakened truth value (in order to prevent immediate contradictions from arising). So, if at some time $t1$ self determines the location of the chocolate is on the counter, we have a corresponding proposition $Location(chocolate, counter, t1, w)$ which is true in $w$.

## Reasoning About False Beliefs

As we have mentioned previously, the false belief task consists of the unexpected transfer of an object from one location to another that happens outside of the knowledge of a target agent, whose action toward this object is to be predicted by the subject. We claim that there is no gap in conceptual or theoretical knowledge differentiating three and four year old subjects. We claim that three year old subjects are in possession of all of the knowledge needed to pass the false belief task, but haven't yet learned to properly re-focus their attention on the target's line-of-sight. Our approach most closely resembles mental simulation, but also uses explicit knowledge in the form of rules to populate and guide the progress of simulation as it occurs. The set of rules that we assume both three and four year old children to be using consists of the following:

1. If an agent has line-of-sight on an object, then the agent knows the location of the object.

2. If an object is at a location it cannot be located anywhere else.

3. If an object is at a location at some time t, it will most probably be at that location at time t+1.

4. If an agent has line-of-sight on an object at time t, it will most likely have line-of-sight on that object at time t+1.

5. If an agent wants an object, and knows that the object is located at l, the agent will reach for the object at l.

We adopt a version of the "like me" hypothesis developed in (Meltzoff 2005), which broadly states that humans possess an innate faculty that posits equivalences between self and other. This idea has been supported through repeated observation of infants imitating the facial gestures of their care-givers even at forty minutes old. To do so, we use Polyscheme's identity predicate, $Same(x, y, E, w)$, in the context of a counterfactual world $w$, which serves as a mental simulation of $x$ taking the perspective of $y$. When Polyscheme sees a proposition of the form: $Category(x, Agent, E, R)$, it immediately creates an alternative world $w$ in which the proposition $Same(self, x, E, w)$ is true. We use an isomorphic version of the false belief task in which an agent named Sue sees a cookie in jar A, then goes outside, as in the classic task. While she is out, the cookie moves to jar B, and the subject (Polyscheme in this case) is asked where Sue thinks the cookie is.

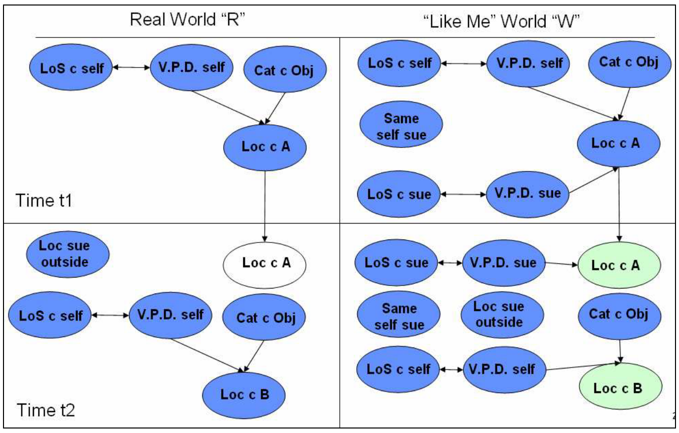


Figure 2: Inference in the False Belief Task

We will use figure 2 and figure 3 as visual references as we explain inference in the task. As we can see in 2, we represent propositions which are true in both the real world, and in the counterfactual world $w$. Upon noticing that Sue's category is $Agent$, $w$ is created and seeded with the true proposition $Same(self, Sue, E, w)$. Self has a line of sight on the cookie at time t1, and thus it's location at jar A. In the counterfactual world in which Self is identical to Sue, Sue also has a line of sight on the cookie. Information about the cookie's location in the counterfactual world is inherited from information in the real world. So in $w$, the cookie's location is also at jar A at time t1. Since the cookie's location is at jar A at time t1, we infer that the cookie is likely at jar A at time t2. Similarly, self's line of sight at time t2 is initially on the cookie. The results of these inferences are also available in $w$. Now, Sue goes outside, but self stays inside and sees the cookie move from jar A to jar B. Now, self's line of sight is on the cookie at jar B, and the location of the cookie is now known by self to be at jar B. The interesting issue is that relations such as "location" are not indexed by agent names. The location of the cookie is just the location of the cookie. If this is the case, how do we separate self's knowledge of the location of the cookie from Sue's? Our speculation is that this is what separates three and four year old subjects.

## Learning to Focus

How then, do four year old subjects successfully navigate the false belief task? We claim that four year old children selectively focus their attention on how information is acquired by the target in the simulation in order to make better predictions about how it will behave.

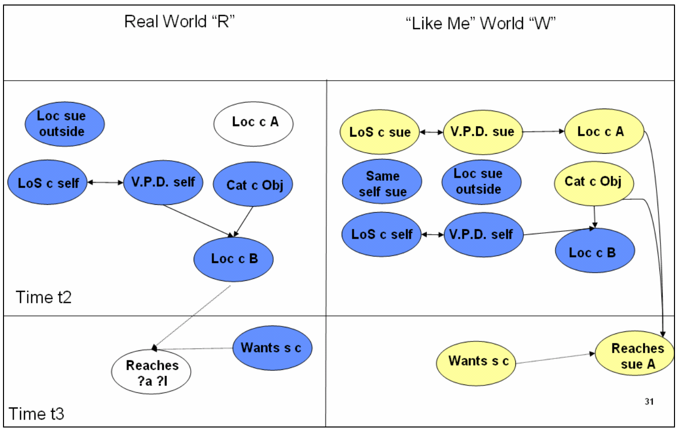


Figure 3: Re-focusing in the False Belief Task

In figure 3, we see Sue's line-of-sight highlighted. Even though self's line of sight still suggests that the cookie is in jar B, we re-focus on Sue's line of sight, and re-infer the location of the cookie to be at jar A. This re-focusing policy is a result of one of Polyscheme's specialists that monitors for conflicts in situations where self/other equivalences are drawn, and re-focuses it's cognitive focus of attention on other-specific information. In the false belief task, the self/other equivalence that causes our problem is that at time 2, and subsequently at time 3, there is a mismatch between self's line of sight and Sue's line of sight. One of Polyscheme's specialists detects this mismatch and re-focuses on Sue's line of sight, re-inferring the location of the cookie to still be at jar A. Learning what to focus on is the crucial linchpin in the developmental process. Polyscheme's associative learner keeps track of which propositions are true every time an action is either taken by self in the real world, or predicted in counterfactual worlds. The learning process is driven by bad predictions. Polyscheme learns to associate the appearance of certain propositions (such as line-of-sight) with potential contradictions. Once it has accumulated a prioritized list of these propositions, they are made available to Polyscheme's conflict-resolver. If the propositions in conflict have an agent-name other than "self" as an argument, the conflict-resolver re-focuses attention on the proposition containing the other agent's name. In the false belief task, Polyscheme makes a number a bad predictions about where Sue thinks the cookie is, and learns to associate line-of-sight with misprediction. The conflict resolver will then re-focus attention on Sue's line-of-sight, which in the context of simulating Sue's mind, will produce the correct prediction.

## Summary

We have shown that learning to keep track of situations in which there are discrepancies between line-of-sight in agents, and using these discrepancies to focus attention on the line-of-sight of the target is a plausible explanation for the emergence of facilitation on the false-belief task. This explanation avoids a number of the problematic corollaries of adopting a more classical stance on the ToM issue. Excessive duplication of propositions and rule-fragments is avoided through the simulation of counterfactual states of affairs which inherit directly from our experience of the real world. By an large, propositions are agent-independent, alleviating the need to re-tag pieces of knowledge as being associated to the various agents whom we wish to make predictions about. Difficulty in learning about mental states due to unobservability is avoided, since via the inheritance and the simulation process, we have access to these structures. We suspect that the late emergence of false belief in the third to fourth year is caused by the general lack of training examples in which we need to keep track of our own line of sight in relation to the line of sight of others. While some may claim that bouts of joint attention between infants and others constitute training examples where infants must keep track of their own line of sight in relation to the lines of sight of others, it's not clear how the specific task of action-prediction interacts with the mental accounting being performed in these cases. It might be that the added complexity delays successful performance on ToM tasks until sometime between the three and four year marks in the same way that learning past-tense information in language usage is delayed.

## References

Wimmer H., Perner, J. (1983). Beliefs about beliefs: Representation and constraining function of wrong beliefs in children's understanding of deception. *Cognition*, 13, 103-128

Carey, S. (1985). Conceptual change in childhood. MIT Press/Bradford Books, Cambridge, MA.

Gopnik, A. & Meltzoff, A.N. (1997). Words, thoughts, and theories. Cambridge, Mass. Bradford, MIT Press.

Medin, D.L. & Atran, S. (1999). Folkbiology. MIT Press.

Spelke, E.S., Breinlinger, K., Macomber, J., & Jacobson, K. (1992). Origins of knowledge. Psychological Review, 99, 605-632.

Goldman, A. (2006). Simulating Minds: The Philosophy, Psychology, and Neuroscience of Mindreading. Oxford University Press.

Gordon, R. (1986). Folk Psychology as Simulation. Mind and Language 1, 158-171; reprinted in Davies, M. and Stone T., eds., 1995, Folk Psychology: The Theory of Mind Debate. Oxford: Blackwell Publishers.

Wellman, H.M., Cross, D., & Watson, J. (2001). Meta-analysis of theory-of-mind development: the truth about false belief. Child Dev, 72(3):655684.

Nichols, S. & Stich, S. (2003). Mindreading: An Integrated Account of Pretence, Self-Awareness, and Understanding of Other Minds. Oxford University Press.

Wellman, H.M. (1990). The child's theory of mind. Cambridge, MA: MIT Press.

Goodman, N.D., Bonawitz, E.B., Baker, C.L., Mansinghka, V.K., Gopnik, A., Wellman, H., Schulz, L. and Tenenbaum, J.B. (2006). Intuitive theories of mind: A rational approach to false belief. Proceedings of the Twenty-Eighth Annual Conference of the Cognitive Science Society.

Pearl, J. (1988). Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference." Morgan Kaufmann.

Pearl, J. (2000). Causality. Cambridge.

Lempers, J.D., Flavell, E.R., & Flavell, J.H. (1977). The development in very young children of tacit knowledge concerning visual perception. Genetic Psychology Monographs 95: 353.

Gopnik, A. & Slaughter, V. (1991). Young children's understanding of changes in their mental states. Child Development, 62, 98-110.

Cassimatis, N.L. (2005). Integrating Cognitive Models Based on Different Computational Methods. In Proceedings of the Twenty-Seventh Annual Conference of the Cognitive Science Society.

Stroop, J.R. (1935). Studies of interference in serial verbal reactions. Journal of Experimental Psychology, 18, 622-643.

Treisman, A.M. & Gelade, G. (1980). A feature integration theory of attention. Cognitive Psychology, 12, 97-136.

Guajardo, J.J., & Woodward, A.L. (2004). Is agency skin-deep? Surface attributes influence infants sensitivity to goal-directed action, Infancy , 6, 361-384.

Hood, B., Willen, J., & Driver, J. (1998). Adults' eyes trigger shifts of visual attention in human infants. Psychological Science, 9 (2), 131-134.

# A 3-node Queuing Network Template of Cognitive and Neural Differences As Induced by Gray and White Matter Changes

**Marc G. Berman (bermanm@umich.edu)**
Departments of Psychology and Industrial and Operations Engineering, 530 Church Street
Ann Arbor, MI 48109-1043 USA

**Yili Liu (yililiu@umich.edu) and Changxu Wu (changxuw@umich.edu)**
Department of Industrial and Operations Engineering, 1205 Beal Avenue
Ann Arbor, MI 48109-2117 USA

## Abstract

We present a 3-node queuing network template for simulating brain activity differences for different subject populations performing simple cognitive tasks. We hypothesize that distinct areas of cortex behave similarly to queuing network servers, whose interactions are used to simulate the interactions of different brain areas. This 3-node queuing network template accurately accounts for brain activity disparities (as found with neuroimaging techniques) for different subject populations performing verbal working memory, spatial working memory, and verbal audition tasks. Further, this 3-node queuing network template provides an account explaining the interactions between different brain areas. This account suggests that reductions in service rates (due to changes in gray matter volume or white matter anisotropy) for different brain areas alters the flow or propagation of neural activity, causing different brain activity patterns for different subject populations performing the same cognitive tasks.

**Keywords:** Queuing Networks; neuroimaging; working memory

## Introduction

The brain is an enormously complex network of interconnected systems and sub-systems, which at this point cannot be easily understood. Most standard neuroimaging techniques tend to focus on singular brain regions that are hypothesized to be responsible for singular functions, either general functions (global approach) or specific functions (local approach; Nyberg and McIntosh, 2001). It seems more likely that behavior and thought result from the interactions of different brain regions rather than from singular brain region activations (Lashley, 1931; Bressler, 1995).

How do different brain areas interact with each other? A number of models and techniques have been proposed to examine this question. Such techniques include Partial Least Squares (PLS), Structural Equation Modeling (SEM), and Dynamic Bayesian Networks (DBN; Nyberg and McIntosh, 2001; Labatut et al., 2004). These techniques are all statistical techniques that can uncover the relationships between different brain areas. While these techniques are extremely useful, they do not explain why brain regions interact in such ways.

This paper offers a new research method based on queuing network theory to explore brain networks. The unique power of this queuing network approach for examining cortical interactions is illustrated in this paper through a simple 3-node queuing network architecture that explains differences in brain activity for different subject populations (young vs. old, literate vs. illiterate) performing the same cognitive tasks.

There are a number of major contributions of this paper. The first is to offer an alternative method to model connectivity in the brain and the subsequent interactions of different brain areas. The second major contribution is that the queuing network template provides plausible, novel, and causal explanations, which predict when certain brain areas will become active and offer explanations as to why they become active. The third contribution is the model's breadth where the same structural template may be the underlying architecture mediating task performance in a wide range of cognitive tasks. In addition, this architecture did not require many model parameter changes in order to model these different phenomena.

## Queuing Networks and Psychology

Queuing Networks are a mathematical discipline that are used to simulate and model a wide array of phenomena including manufacturing processes, emergency room workload, and airport traffic. The queuing network methodology has also been applied to cognitive psychology, and was used to successfully unify various psychological models of reaction time (Liu, 1996) and multitask performance (Liu, 1997). Recently, the queuing network approach has been successfully integrated with the symbolic approach (Liu et al., 2006) for both mathematical analysis and real time simulation of human performance in a multitude of settings including in-vehicle steering (Liu et al., 2006), transcription typing (Wu & Liu, 2004), and visual search tasks (Lim & Liu, 2004). The success of the queuing network methodology in these domains is evidence of its efficacy as a model of human cognition and behavior.

### Queuing Networks and Brain Activation

In this paper we attempt to model brain activation differences, as uncovered with neuroimaging, for different subject populations performing the same cognitive tasks. We will model these brain activation differences with a queuing network methodology and architecture. Rather than model changes in blood flow or volume, we attempt to model the underlying neural activity that drives the differences in blood flow as found empirically in these studies.

Positron Emission Tomography (PET) and functional Magnetic Resonance Imaging (fMRI) are two techniques used to measure blood flow changes in the brain, and both do so in different ways (Cabeza and Nyberg, 2000). fMRI measures blood flow changes via the Blood Oxygenated Level Dependent (BOLD) contrast and PET measures blood flow changes by marking blood with radioactive tracers (Cabeza and Nyberg, 2000). These hemodynamic processes are also correlated with the underlying neural activity of those brain areas exhibiting blood flow changes (Logothetis et al., 2001). This provides researchers with some confidence that while an indirect measure, these techniques can be used as to approximate underlying neural activity. In this paper we model the underlying brain activation moderating these hemodynamic changes and we do so with a queuing network framework.

## 3-Node Queuing Network (QN) Template

While there are many different types of queuing networks, this paper focuses on a simple 3-node queuing network with one server branching out into two parallel servers, as shown in Figure 1.

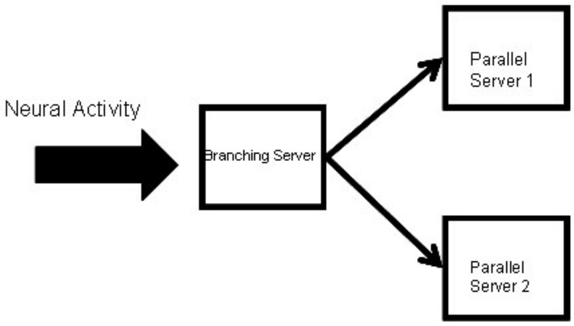


Figure 1: 3-node QN template

In all of these simulations, neural activity or neural spike trains are treated as the customers (C) in these networks that will be served by the queuing network servers in the network[1]. Each of the boxes in this network is a queuing network server (S) that provides a service to the customers that enter the network. Each of these servers will represent a unique brain area(s) that provides a unique service to the customers that enter it. Customers arrive at the branching server at some arrival rate λ customers/unit time. Once customers enter the network through the branching server, they receive some service from the branching server with a service rate of μ customers/unit time. Once customers complete service at the branching server they then travel probabilistically to either parallel server 1 or parallel server 2 for additional service. Each of these servers services customers with it's own service rate. Once customers complete service at either of those servers they have completed their full service and subsequently leave the network.

Each server also has a service capacity (the number of customers it can serve at a time) and a waiting capacity or queue capacity, which identifies how many customers can wait in front of the servers for service. The parallels between the queuing network methodology and the brain are apparent. First, it seems that different brain areas do in fact provide some unique function or service that mediate behavioral performance. Second, it is reasonable to assume that brain areas have capacity limitations in the amount of processing that they can accomplish and the speed with which they can process. Third it seems that information in the brain can be queued, as information that is not processed immediately is not immediately lost or discarded.

For these simulations we alter service rates for particular servers in the network, but we leave service capacities and queue capacities constant throughout, as it is beyond the scope of this paper to provide/hypothesize queue capacities for different brain areas. We therefore set each server's capacity to serve only one customer at a time with an infinitely large queue capacity (waiting line).

In all of these simulations we assume that each server provides a unique service and that customers need to be served by the branching server and only one of the parallel servers, but not both. Therefore, while each parallel server provides a unique service, each of the simulated tasks can be accomplished by traversing either of the parallel servers, which provide similar service.

Parameters for arrival rates, λ, were set based on the empirical parameters of the task. Service rate parameters were set based on neural evidence coming from research on aging and literacy. Zimmerman et al. (2006) have found significant gray matter volume reductions with increased age, and that these reductions have been correlated with reductions in executive functioning and working memory performance. Klingberg et al. (2000) have found that reductions in white matter anisotropy (connection fidelity), is strongly correlated with poor reading performance. We draw on these findings in setting our queuing network parameters for these simulations.

In addition one of the major assumptions of this paper is our hypothesized explanation for how neural activity flows or moves through brain neural networks. Neural activation flows in this network based on the comparative processing rates of the parallel servers, and is mediated by the following equation:

$$P_i = [\mu_i]/([\ \mu_i] + [\mu_j])\text{(Eq. 1)}$$

Where $\mu_i$ = the service rate for server i, and $P_i$ is the probability of traveling to server i.

One can see that if the service rate of one parallel server in relation to the other server is much greater, then it is more

[1] Initial neural activation at the Branching Server is time-locked with empirical stimulus onsets.

likely that neural activity or spike trains would propagate to that server and vice versa.

There is also neural evidence that supports this routing equation. It is known that stronger synaptic connection strengths of an individual neural route, increase the probability ($P_i$) that neural spike trains (the customers in our network) travel through that route (Black, 1999; Chklovskiii et al., 2004; Habib, 2003). Synaptic connection weights can also be decomposed into waiting times and processing times of customers traveling a particular route (Wu, 2007). Therefore, equation 1 is supported by the aforementioned neural evidence.

## Studies to Be Modeled

In this paper we model the neuroimaging results from 2 separate PET studies. These two studies were selected for two reasons. First each study found brain activity differences for different subject populations performing the same cognitive tasks. Second, in all the studies, subjects activated the same brain areas, in other words, different brain areas were not recruited for the different populations of subjects, only the distribution of the brain activation differed.

The first study, from Reuter-Lorenz et al. (2000), explored the difference in brain activity for old and young adults performing verbal and spatial Working Memory (WM) tasks. The second study from Petersson et al. (2000), investigated brain activity during a pseudoword generation task for illiterate and literate subjects. Though we have restricted the number of studies simulated in this paper, our results could conceivably be applied to many other studies of this kind.

All of our simulations were run for 30 minutes, roughly mimicking the total time of the empirical studies. In addition each simulation was run for 100 replications.

### Study1: Reuter-Lorenz et al., 2000

In this study young and old subjects performed verbal and spatial working memory tasks. The authors used PET to identify the brain areas activated to perform these tasks and also explored the brain activation differences between the two groups of subjects. The major finding was that for verbal working memory young subjects showed substantial left lateralized frontal activation and for spatial working memory those subjects showed substantial right lateralized frontal activation. Older adults on the other hand showed bilateral frontal activation for both spatial and verbal working memory suggesting that older adults may be recruiting other brain areas to compensate for neural declines (Reuter-Lorenz et al., 2000). In addition, few differences were found in posterior activations for these two subject populations.

**Simulation Parameters** The queuing network template used to model these data can be seen in figure 2. The anterior and posterior brain areas that compose these servers can be seen in table 1.

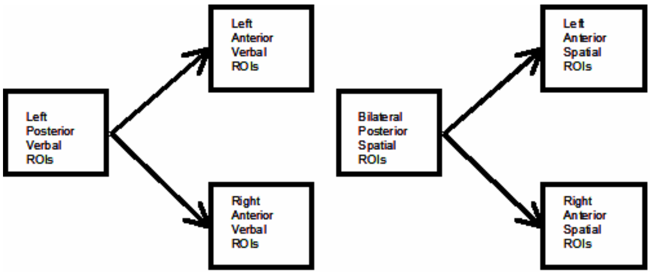

Left Posterior Verbal ROIs
Left Anterior Verbal ROIs
Right Anterior Verbal ROIs
Bilateral Posterior Spatial ROIs
Left Anterior Spatial ROIs
Right Anterior Spatial ROIs

Figure 2: 3-node QN templates used to model Reuter-Lorenz et al. (2000) data

Table 1: Brain areas that compose the Queuing Network Servers as treated singularly by Reuter-Lorenz et al. (2000).

| | Verbal Working Memory Task | Spatial Working Memory Task |
|---|---|---|
| Anterior ROIs | BA 45, 46, 10, 9 and 44 (Broca's); BA6(Supplementary Motor and premotor | BA 9, 46, 47 (DLPFC, VPFC); SMA and Premotor |
| Posterior ROIs | BA 40, 7 (parietal) and temporal sites BA 42 and 22 | BA 40, 7 (parietal); BA 18, 19 (striate and extrastriate); BA 31 (Precuneus) |

Arrival rates of the stimuli were set to be 5 seconds as this was the presentation rate of the trials to the subjects in the empirical study.

Service rate parameters were initially set to be exponentially distributed with a mean of 18 ms and have been validated by other researchers (Feyen, 2002; Wu, 2007). For young adults, service rate parameters were set in ways to show frontal lateralization. Therefore, for the verbal working memory task, the right anterior server's processing rate was treated as a free parameter and set to a value that would show lateralization (we used the same parameter value for the left anterior server for the spatial working memory task).

In addition, we feel it makes intuitive sense that left frontal areas should have disproportionately faster service rates for verbal tasks (compared to right frontal areas), and right frontal areas should have disproportionately faster service rates for spatial tasks (compared to left frontal areas), as these areas seem to be most active in the service of those respective tasks. If there were not such a difference in the processing abilities of these areas of cortex mediating performance in these tasks, we would not expect such robust lateralized activity.

For setting processing rates for older adults we used equations 2 and 3.

$$\text{Lateral frontal} = 67{,}043 - .47 * \text{Age (Eq. 2)}$$

Note: the units are in $mm^3$

$$\mu_{subject}^{-1} = \mu_{control}^{-1} + \left( \frac{graymatter_{young} - graymatter_{old}}{graymatter_{young}} \right) * \delta$$

(Eq. 3)[2].

Equation 2, was provided by Zimmerman et al. (2006) and explains how gray matter volume in lateral frontal areas decreases with increased age. Equation 3, describes how gray matter volume changes for older adults translates into slower processing rates in lateral frontal areas. Here we are assuming that gray matter reductions reduce the abilities of those cortical areas to process information, and that this reduction in processing is additive to initial processing values. Tables 2 and list the parameters used to simulate the results from Reuter-Lorenz et al. (2000). Note: the mean age for young adults was 24, and for older adults was 69.

In addition, one may note that for older adults the service rates change for frontal areas, but not for posterior areas. With aging, there is more gray matter loss in frontal areas, compared to posterior areas, which can explain more deficits in planning, organizing and performing other executive functions with age (Zimmerman et al., 2006).

Table 2. Processing rates for old and young in the verbal WM task

| Population | Older Adults | Younger Adults |
|---|---|---|
| Arrival rate (λ)[3] | 1 every 5 sec | 1 every 5 sec |
| Service rate Left Posterior Regions | Exponential mean 18 ms per neural spike train[4] | Exponential mean 18 ms per neural spike train |
| Service rate Right Anterior Regions | Exponential Mean 86 ms per neural spike train | Exponential mean 54 ms per neural spike train |
| Service rate Left Anterior Regions | Exponential Mean 50 ms per neural spike train | Exponential mean 18 ms per neural spike train |

Table 3. Processing rates for old and young adults in the spatial WM task

| Population | Older Adults | Younger Adults |
|---|---|---|
| Arrival rate (λ) | 1 every 5 sec | 1 every 5 sec |
| Service rate Left Posterior Regions | Exponential mean 18 ms per neural spike train | Exponential mean 18 ms per neural spike train |
| Service rate Right Anterior Regions | Exponential Mean 50 ms per neural spike train | Exponential mean 18 ms per neural spike train |
| Service rate Left Anterior Regions | Exponential Mean 86 ms per neural spike train | Exponential mean 54 ms per neural spike train |

From tables 2 and 3 one can see how the initial imbalance between left and right service rates for the verbal and spatial tasks would cause more neural spike train activity to propagate to left anterior areas for the verbal task, and right anterior areas for the spatial task (see equation 1). Again, for older adults, service rate parameters in anterior areas were set based on equations 2 and 3.

**Simulation Results** Figure 3 displays the simulation results and the empirical results from the Reuter-Lorenz et al. (2000) study. The fits of our simulation results have an $R^2 = 0.64$ for the verbal working memory task, and an $R^2 = 0.72$ for the spatial working memory task.

The dependent variable that Reuter-Lorenz et al. (2000) report is the % change in brain activation for experimental working memory trials compared to control trials. For the experimental trials there was a higher working memory load compared to the control trials (roughly 4 times that of controls). Therefore in our simulations we altered processing by a scalar value (4) to reflect the changes in task demands from control trials to experimental trials. We report the changes in server utilization from control trials to experimental trials.

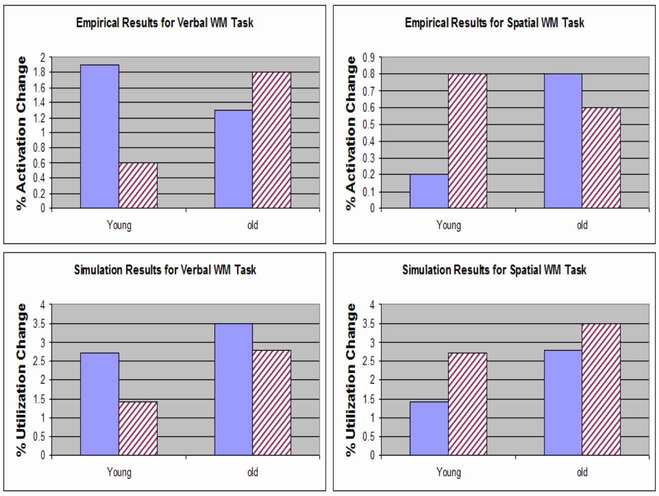


Figure 3: Empirical Results and simulation results for Verbal WM Task and spatial WM task from Reuter-Lorenz et al. (2000). Blue solid bars are left anterior areas, and magenta dashed bars represent right anterior areas. Top row shows the empirical results and the bottom row the simulation results[5]

From figure 3, one can see that our simulations do capture the empirical results well, especially the overall pattern of less lateralization with increased age. This reduced lateralization was due to processing declines mediated by gray matter loss, which reduced the ratio in processing rates of one parallel server relative to the other.

[2] δ is a scaling parameter was set to 100 for simulation 1 and set to 18 for simulation 2.

[3] The arrival rates were based on empirical stimulus presentation rates

[4] See Liu, Feyen and Tsimhoni (2006)

[5] The apparent reversal in lateralization for older adults in both the verbal and spatial working memory tasks was not significant

## Study2: Petersson et al., 2000

In this study literate and illiterate participants performed a task where they needed to repeat verbally auditorily presented words and pseudowords. It was found that literate and illiterate subjects had similar behavioral performance in repeating words, but illiterate subjects were impaired in repeating pseudowords. It was also found that the neural networks supporting pseudoword repetition were different for the two groups, suggesting that learning to read causes functional changes in brain circuitry.

Here we concentrate on path weight differences (as found with Structural Equation Modeling; SEM) between inferior Parietal Cortex (iPC; BA 7/40) with Broca's area (BA 44) and iPC with prefrontal cortex (PFC; BA 45/46). The authors found that the path weight between iPC and Broca's was higher for literate subjects (by .18), while the path weight between iPC and PFC was higher for illiterate subjects (by .26). These path weight changes may reflect more efficient phonological loop processing for literate participants, and subsequently more reliance on executive processes for illiterate subjects to perform the pseudoword repetition task. Note: we report correlations rather than path weights, but the path weights were based on the correlation matrix of the empirical study.

Table 4. Processing rates for Literate and Illiterate Subjects

| Population | Literate Subjects | Illiterate Subjects |
|---|---|---|
| Arrival rate (1/lambda) | U(6, 1) sec | U(6, 1) sec |
| Service rate iPC | Exponential mean 18ms per spike train | Exponential mean 18ms per spike train |
| Service rate Brocas | Exponential mean 18ms per spike train | Exponential mean 29 ms per spike train |
| Service rate PFC | Exponential mean 27 ms per spike train | Exponential mean 27 ms per spike train |

**Simulation Parameters** Table 4 lists the parameters that were used to simulate the data from Petersson et al., (2000). Arrival rates were set based on empirical parameters, where stimuli were presented every 6 seconds. However, we needed to include some variance so that we could calculate the correlation of neural activations in our queuing network servers.

We set parameters for literate subjects in a similar manner to that of our simulations of Reuter-Lorenz et al. (2000). We treated service rate in the PFC as a free parameter as younger adults are biased to utilize the route connecting iPC and Brocas over iPC and PFC.

For setting parameters for illiterate subjects we depended on differences in white matter anisotropy. While there may be gray matter volume differences between literate and illiterate subjects, we were guided by white matter anisotropy (connection integrity) differences between good and poor readers (Klingberg et al., 2000).

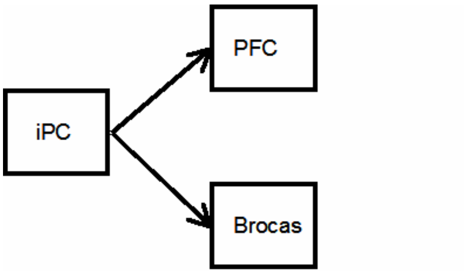


Figure 4. 3-Node queuing network used to simulate Petersson et al. (2000) data

Klingberg et al. (2000) found that white matter anisotropy in a volume connecting parietal and temporal cortices in the left hemisphere was significantly reduced in poor readers compared to normal readers, and that anisotropy in this region was significantly correlated with reading performance. We used this significant reduction in anisotropy connecting temporal and parietal cortex to change the processing rate of Broca's area for illiterate subjects as this white matter region would be connecting iPC with Broca's. We assumed that this reduction in anisotropy in this region would alter the relationship between iPC and Broca's area for illiterate subjects.

Klingberg et al. (2000) found that anisotropy in this white matter region was correlated with reading performance r = 0.84. Using behavioral data from the Petersson et al. (2000) study (via Castro-Caldas et al., 1998) and Klingberg et al.'s (2000) regression equation we calculated the anisotropy values for literate and illiterate subjects. These calculations yielded a 60% decrease in anisotropy for illiterate subjects compared to literate. Using Equation 3., we substituted white matter changes between literate and illiterate subjects (instead of gray matter) to obtain the service rate for Broca's area for illiterate subjects.

**Simulation Results** The simulation results for this task are summarized in Table 5. We obtained an $R^2 = .96$ for these simulation data (pitting our simulated correlation values against Petersson et al. (2000) path weights). One can see the changes in correlated activity match the pattern of differences in path weights exhibited from the Petersson et al. (2000) study where literate participants exhibit increased correlations between iPC and Broca's compared to illiterate subjects, and illiterate participants exhibit increased correlations between iPC and PFC compared to literate participants. We did not find the same magnitude increase in correlated activity between iPC and PFC as was found empirically, however, we did find the same overall pattern. These results indicate good coherence between our simulation and the empirical findings.

Table 5. Simulation and Empirical Results from Petersson et al. (2000)

| | Empirical | Simulation |
|---|---|---|
| Increase in iPC and Brocas Relation for Literate compared to Illiterate | +. 18 | +. 10 |
| Decrease in iPC and PFC Relation for Literate compared to illiterate | -.26 | -.10 |

## Conclusion

In sum, our 3-node queuing network templates were able to successfully model the activity of brain networks for different populations of subjects performing the same cognitive tasks. We drew on neuroscience evidence in selecting parameters and explained changes in brain networks as being caused by relative differences in service rates, which alter neural activation propagation. We hope that with the queuing network architecture we will be able to understand more complicated brain networks and make new predictions for the behavior of brain networks that underlie human cognition.

## Acknowledgments

This research was supported in part by an NSF graduate fellowship to MGB and an NSF grant to YL.

## References

Black, I.B. (1999). Trophic regulation of synaptic plasticity. *Journal of Neurobiology*, 41 (1), 108-118.

Bressler, S. L. (1995). Large-scale cortical networks and cognition. *Brain Research Reviews.* Vol. 20, 288-304.

Cabeza R. and Nyberg L.(2000). Imaging cognition II: An empirical review of 275 PET and fMRI studies. *J of Cog Neuro.* 12 (1): 1-47

Castro-Caldas A., Petersson KM, Reis A, Stone-Elander S, Ingvar M. (1998). The illiterate brain - Learning to read and write during childhood influences the functional organization of the adult brain. *BRAIN* 121(6): 1053-1063

Chklovskii, D.B., Mel, B.W., and Svoboda, K. (2004). Cortical rewiring and information storage. *Nature*, 431 (7010), 782-788.

Feyen R. (2002). Modeling Human Performance Using the Queuing Network - Model Human Processor (QN-MHP). Unpublished Dissertation, University of Michigan, Ann Arbor, Michigan.

Habib, M. (2003). Rewiring the dyslexic brain. *Trends in Cognitive Sciences*, 7 (8), 330-333.

Klingberg, T., Hedehus, M., Temple, E., Salz, T., Gabrieli, J. D. E., Moseley, M. E., et al. (2000). Microstructure of temporo-parietal white matter as a basis for reading ability: Evidence from diffusion tensor magnetic resonance imaging. *Neuron, 25*(2), 493-500.

Labatut, V., Pastor, J., Ruff, S., Demonet, J., Celsis, P. (2004). Cerebral modeling and dynamic Bayesian networks. *Artificial Intelligence in Medicine.* Vol. 30, 119-139.

Lashley, K.S. (1931). Mass Action in Cerebral Function. *Science.* Vol. 73(1888), 245-254.

Lim, J., and Liu, Y. (2004). A Queueing Network Model for Visual Search and Menu Selection. *Proceedings of the 48th Annual Conference of the HFES.*

Liu, Y. L. (1996). Queueing network modeling of elementary mental processes. *Psychological review, 103*(1), 116-136.

Liu, Y. L. (1997). Queueing network modeling of human performance of concurrent spatial and verbal tasks. *Ieee Transactions on Systems Man and Cybernetics Part A-Systems and Humans, 27*(2), 195-207.

Liu, Y., Feyen, R., & Tsimhoni, O. (2006). Queueing Network-Model Human Processor (QN-MHP): A Computational Architecture for Multi-Task Performance in Human-Machine Systems. *ACM Transactions on Computer-Human Interaction.*

Logothetis, NK; Pauls, J; Augath, M; Trinath, T; Oeltermann, A. 2001. Neurophysiological investigation of the basis of the fMRI signal. *NATURE* 412 (6843): 150-157.

Nyberg, L., & McIntosh, A. R. "Functional Neuroimaging: Network Analyses." *Handbook of Functional Neuroimaging of Cognition.* Eds. Roberto Cabeza and Alan Kingstone. A Bradford Book: MIT Press, 2001. 49-72.

Pastor, J., Ruff, S., Demonet, J., Celsis, P. (2004). Cerebral modeling and dynamic Bayesian networks. *Artificial Intelligence in Medicine.* Vol. 30, 119-139.

Petersson, K. M., Reis, A., Askelof, S., Castro-Caldas, A., & Ingvar, M. (2000). Language processing modulated by literacy: A network analysis of verbal repetition in literate and illiterate subjects. *Journal of cognitive neuroscience, 12*(3), 364-382.

Reuter-Lorenz, P. A., Jonides, J., Smith, E. E., Hartley, A., Miller, A., Marshuetz, C., et al. (2000). Age differences in the frontal lateralization of verbal and spatial working memory revealed by PET. *Journal of cognitive neuroscience, 12*(1), 174-187.

Wu C. (2007). Queueing Network Modeling of Human Performance and Mental Workload in Perceptual-Motor Tasks. Unpublished Dissertation, University of Michigan, Ann Arbor, Michigan.

Wu, C., & Liu, Y. (2004). Modeling Human Transcription Typing with QN-MHP (Queueing Network - Model Human Processor). *Proceedings of the 48th Annual Conference of the HFES.*

Zimmerman, M. E., Brickman, A. M., Pau, R. H., Grieve, S. M., Tate, D. F., et al. (2006). The relationship between frontal gray matter volume and cognition varies across the healthy adult lifespan. *American Journal of Geriatric Psychiatry, 14*(10), 823-833.

# Integrating Rational Choice and Subjective Biological and Psychological Factors in Criminal Behaviour Models

**Tibor Bosse (tbosse@few.vu.nl)** **Charlotte Gerritsen (cg@few.vu.nl)** **Jan Treur (treur@few.vu.nl)**

Vrije Universiteit Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081, 1081 HV, Amsterdam, The Netherlands

## Abstract

The Rational Choice Theory describes criminal behaviour as a form of means-end decision-making. In contrast, it is often argued that criminal behaviour involves subjective, personal biological and psychological aspects. This paper contributes an agent-based modelling approach for criminal behaviour integrating such subjective aspects with decision-making based on means-end analysis, and illustrates this for street robbery. The agent model developed is a combination of a BDI-model and a utility-based decision model in the context of such desires and beliefs. The resulting approach incorporates subjective, context-sensitive means-end analysis, where the context covers biological and psychological aspects as mentioned.

## Introduction

A longstanding debate within Criminology is whether criminal behaviour is driven by a criminal's subjective, personal biological and psychological background, or is the result of a rational, calculated choice; e.g., (Moir and Jessel, 1995; Cornish and Clarke, 1986). Often the former is considered to happen to the subject with not much freedom of choice, whereas the latter is considered to be a more or less free choice. This debate has fundamental societal implications, for example, on policies with respect to punishment and treatment of criminals. An interesting question is whether in this debate an exclusive choice has to be made between two opposed viewpoints, or a synthesis can be obtained reconciling them. If this question is to be answered in the latter sense in a convincing manner, this requires a detailed account on how exactly the two viewpoints can be integrated. This indeed is the aim of the current paper. It is shown in detail that and how such an integration is possible, by means of a formalised computational model that incorporates both sides.

As a starting point, the agent model described in (Bosse, Gerritsen and Treur, 2007b) has been taken, which focuses on a case study about the Intermittent Explosive Disorder. This model addresses action generation based on beliefs, desires and intentions (BDI), and generation of desires and beliefs in opportunities. However, for the sake of simplicity, only one action per desire was assumed, so no decision making was covered involving a choice between different options for actions to fulfil a desire. The current paper presents a model for utility-based multi-criteria decision making (e.g., Keeney and Raiffa, 1976; Raiffa, 1982) within a BDI-setting and applied to street robbery. This decision model provides a formalisation of the Rational Choice Theory within Criminology; e.g. (Cornish and Clarke, 1986). This theory as informally discussed within Criminology describes crime as an event that occurs, for example, when an offender decides to take risk breaking the law, after considering his or her own need for money, personal values or learning experiences and how well a target is protected. The criminal assesses the chances of getting caught, the expected penalty, the value to be gained by committing the act, and his or her immediate need for that value.

In the decision model introduced in this paper, this process is modelled by introducing utilities for different possible intended actions. The utility of a certain (option for an) action is then assessed according to the extent to which it fulfils the subject's desire. In this way utilities are assessed with respect to a subjective measure focusing on a specific desire, which may be affected by the subject's specific biological and psychological background. In other words, for the individual agent, rational choice means the choice to fulfil its own desires in the best possible way. Thus, the model for desire generation based on the biological and psychological factors is integrated with a rational decision model for the choice of (intended) actions.

In this paper, the next section discusses a brief summary from the literature on the role played by biological and psychological factors in criminal behaviour. After that, the dynamical modelling approach is discussed. Next, the simulation model is presented and some simulation results are shown. Finally, the approach and its possible applications are discussed.

## Biological and Psychological Factors

Since the BDI model (Georgeff and Lansky, 1987; Rao and Georgeff, 1991) does not prescribe a standard way to determine how desires are created, for a particular application usually domain-specific knowledge is used. For criminal behaviour, a number of specific biological and psychological aspects seem to play a role in the generation of desires. An extensive search has been performed into literature from areas such as Criminology and Psychology (e.g., Raine, 1993; Moir and Jessel, 1995; Delfos, 2004) for aspects to be incorporated in the model.

A *theory of mind* of a person (e.g., Baron-Cohen, 1995) describes other persons' minds by separate mental concepts, such as the person's own beliefs, desires, and intentions, and how those concepts play a role in the person's behaviour. Criminal actions are often performed by persons whose theory of mind is less developed. In recent years, more evidence is found that there often are biological reasons for this. For example, it has been found that many psychopaths have a damaged connection between the frontal lobes (concerned with conscience and remorse) and the limbic area, which generates feelings; cf. (Moir and Jessel, 1995).

Another important aspect in crimes is *aggressiveness*. Research has pointed out that there is a correlation between aggressive behaviour and the level of testosterone. In fact, 89% to 95% of all crime is performed by males (Moir and Jessel, 1995). In addition, the use of alcohol or drugs may increase the violence of behaviour.

A third aspect involved in criminal behaviour is the *desire to act*, which can be related to a high level of adrenalin. If a person's adrenalin level becomes too high, (s)he somehow has to cope with this; acting decreases the adrenalin level. Thus, if the desire to act is high, then a criminal act more easily occurs. The specific types of actions that are chosen depend on another factor, the *desire to act safely*. This factor correlates with a high level of oxytocine, a hormone mainly produced by women. Persons with a high level of oxytocine have a higher tendency to cope with their desire to act by performing 'safe' actions (e.g., taking care of the 'nest') than persons with less oxytocine; they will rather perform 'less safe' actions (e.g., fighting) (Delfos, 2004).

In addition, crimes are often committed by persons who are looking for a thrill. These persons in general have a high excitement threshold, which means that it is very difficult for them to become excited (Moir and Jessel, 1995; Raine, 1993). As a result, they are often bored, so that they generate a *desire for actions with strong stimuli*. Such actions may become criminal actions, such as stealing, joyriding, or assaulting other people. Only by performing these actions, their desire for strong stimuli is fulfilled, and they become less bored.

Furthermore, a significant amount of committed crimes can be described as *impulsive*. They are not planned, but rather triggered by occasional opportunities. An important factor causing impulsive behaviour is a low level of blood sugar, which in turn is caused by a high insulin level and a low serotonin level (Moir and Jessel, 1995).

A next factor that may affect the types of (criminal) actions that persons may perform, is the extent to which they have (positive or negative) feelings with respect to another person's wellbeing. When someone has a low amount of *positive feelings towards others*, (s)he does not really care about the other. Likewise, when someone has many *negative feelings towards others*, (s)he may wants to cause harm towards someone else. For example, in psychopaths, both attitudes are low: these persons hardly show any emotion concerning other persons, so for them, both the positive and the negative emotional attitude towards others are low (Moir and Jessel, 1995).

The last two factors chosen to incorporate in the model are the *desire for high gain* and the *desire for low loss*. These concepts were chosen on the basis of the Rational Choice Theory (Cornish and Clarke, 1986). According to this theory, to determine their actions, persons will try to minimise their expected loss or penalty (e.g., being caught, getting hurt) and maximise their gain (e.g., money, status). The theory states that criminals will make a serious decision before committing a crime, weighing pros against cons.

## Modelling Approach

Modelling the various aspects in an integrated manner poses some challenges. On the one hand, qualitative aspects have to be addressed, such as beliefs, desires, and intentions, certain brain deviations, and some aspects of the environment such as the presence of certain agents. On the other hand, quantitative aspects have to be addressed, such as testosterone and serotonin levels, and utilities.

The modelling approach based on the modelling language LEADSTO (Bosse, Jonker, Meij, and Treur, 2005) fulfils these desiderata. It integrates qualitative, logical aspects and quantitative, numerical aspects. This integration allows the modeller to exploit both logical and numerical methods for analysis and simulation. The basic building blocks of LEADSTO are so-called *executable dynamic properties*, by which direct temporal dependencies between two state properties in successive states are modelled. Their format is defined as follows. Let α and β be state properties of the form 'conjunction of ground atoms or negations of ground atoms'. In LEADSTO, the notation $\alpha \twoheadrightarrow_{e, f, g, h} \beta$, means:

*If state property α holds for a certain time interval with duration g,*
*then after some delay (between e and f) state property β will hold*
*for a certain time interval of length h.*

Here, atomic state properties can have a qualitative, logical format, such as an expression desire(d), expressing that desire d occurs, or a quantitative, numerical format such as an expression has_value(x, v) which expresses that variable x has value v. For more details of the language LEADSTO, see (Bosse, Jonker, Meij, and Treur, 2005). The overall simulation model has been built by composing two models:

1. a model to *determine desires* incorporating various biological and psychological aspects and their interactions
2. a model for reasoning about *beliefs, desires and intentions*, using a BDI-model based on *utility-based decision making*

These models have both been implemented in LEADSTO. They are described in more detail the next sections.

## Determining Desires

To determine desires, a rather complex submodel is used, incorporating dynamical system elements for the biological and psychological aspects as mentioned earlier, varying from qualitative aspects, such as anatomical aspects concerning brain deviations (e.g., the absence of certain connections) to quantitative aspects, such as biochemical aspects concerning testosterone levels. Some example LEADSTO specifications (called Local Properties, LPs) are given below (both in informal and in formal notation)[1]:

**LP9** A certain level of current testosterone will lead to a corresponding level of aggressiveness.

∀x:SCALE chemical_state(testosterone,current,x) $\twoheadrightarrow_{0, 0, 1, 1}$ desire_for_aggressiveness(x)

[1] See (Bosse, Gerritsen, and Treur, 2007b) for the complete model.

**LP20** Observation of a stimulus with an excitement level that is lower than the excitement threshold will lead to boredom.
∀s1,s2,y:INTEGER observes_stimulus(s1,s2) ∧ excitement_threshold(y) ∧ s2<y $\twoheadrightarrow_{0,0,1,1}$ boredom

**LP29a** A low blood sugar level leads to high impulsiveness.
chemical_state(blood_sugar, low) $\twoheadrightarrow_{0,0,1,1}$ desire_for_impulsiveness(high)

The variety of biological and psychological aspects that were found relevant in the literature (such as Moir and Jessel, 1995; Raine, 1993; Bartol, 2002; Delfos, 2004) and are taken into account in this model, are those described in the second section above. Different combinations of these elements lead to different types of (composed) desires; e.g., the desire to perform an exciting planned nonaggressive nonrisky action that harms somebody else (e.g., a pick pocket action in a large crowd). The following LEADSTO rule generates a composed desire out of the different ingredients mentioned earlier:

**LP30** A combination of values for theory of mind, desire for aggressiveness, desire to act, desire to act safely, desire for actions with strong stimuli, desire for impulsiveness, positive and negative emotional attitude towards others, and desire for high gain and low loss leads to a specific composed desire, represented as d(has_value(theory_of_mind, s1), ..., has_value(desire_for_low_loss, s10)).
∀s1,s2,s3,s4,s5,s6,s7,s8,s9,s10:SCALE
theory_of_mind(s1) ∧ desire_for_aggressiveness(s2) ∧ desire_to_act(s3) ∧
desire_to_act_safely(s4) ∧ desire_for_actions_with_strong_stimuli(s5) ∧
desire_for_impulsiveness(s6) ∧ emotional_attitude_towards_others(pos,s7) ∧
emotional_attitude_towards_others(neg,s8) ∧ desire_for_high_gain(s9) ∧
desire_for_low_loss(s10) $\twoheadrightarrow_{0,0,1,1}$
desire(d(has_value(theory_of_mind, s1), ...,
has_value(desire_for_low_loss, s10)))

## Utility-Based Reasoning about Intentions

As in (Bosse, Gerritsen and Treur, 2007b), part of the model for criminal behaviour is based on the BDI-model, whichbases the preparation and performing of actions on beliefs, desires and intentions (e.g., Georgeff and Lansky, 1987; Rao and Georgeff, 1991). In this model an action is performed when the subject has the intention to do this action and it has the belief that the opportunity to do the action is there. Beliefs are created on the basis of stimuli that are observed. The intention to do a specific type of action is created if there is a certain desire, and there is the belief that in the given world state, performing this action will fulfil this desire. The BDI-model was specified by:

**LP31** Desire d combined with the belief that a certain action a will lead to the fulfillment of that desire will lead to the intention to perform that action.
∀d:DESIRE ∀a:ACTION desire(d) ∧ belief(satisfies(a, d)) $\twoheadrightarrow_{0,0,1,1}$ intention(a)

**LP32** The belief that there is an opportunity to perform a certain action combined with the intention to perform that action will lead to the performance of that action.
∀a:ACTION belief(opportunity_for(a)) ∧ intention(a) $\twoheadrightarrow_{0,0,1,1}$ performed(a)

However, to assess and compare different options, and select a best option, as an extension to this basic BDI-model utilities are to be assigned and combined, addressing the degree to which an action satisfies a desire. The notion of utility to be used requires some reflection. Sometimes this may be considered a rational notion with an absolute, intersubjective (or objective) status. For two agents with a kind of standard internal functioning, considered rational, this intersubjectivity may be a reasonable assumption. However, if the internal processes are different it is less reasonable. One agent may have preferences different from those of the other agent, and hence be satisfied with a situation that is not satisfactory for the other agent. As an example, multi-attribute negotiation aims at exploiting such differences in preferences between agents in order to the benefit of both; e.g., (Keeney and Raiffa, 1976; Raiffa, 1982; Jonker and Treur, 2001; Bosse, Jonker and Treur, 2004). This shows that the meaning of utility can be subjective and personal. In particular, for a criminal subject, due to his or her specific biological and psychological characteristics, a desire can be quite deviant from what is commonly considered as the rational norm. For this subject the utility of a certain action a is assessed according to the extent to which it fulfils this personal desire. This shows how utilities are assessed with respect to a subjective measure focusing on a specific desire d, which is affected, or even largely determined by the subject's specific biological and psychological background. According to this perspective, the utility-based decision model was set up as follows:

### 1. Aspect Utility Value Representations

For any aspect $x_i$ with value $s_i$, introduce an aspect utility $v_i$ for any possible action a by

has_aspect_utility(a, has_value($x_1$, $s_1$), $v_1$)
…
has_aspect_utility(a, has_value($x_k$, $s_k$), $v_k$)

where $v_i$ is based on a closeness measure for each aspect $x_i$ of the considered option a to value $s_i$, normalised between 0 (least close, minimal utility) and 1 (most close, maximal utility). For example,

has_aspect_utility(fight,
has_value(desire_for_aggressiveness, high), 0.9)

indicates that the action of fighting contributes much to a high value for aggressiveness.

### 2. Aspect Weight Factor Representations

Introduce weight factors $w_1$, ..., $w_k$ for the different aspects $x_i$, normalised so that the sum is 1, and introduce relations weight_factor($x_i$, $w_i$) stating that aspect $x_i$ has weight factor $w_i$.

### 3. Combination of Aspect Utilities to Option Utilities

Combine the option aspect utility values $v_1$, ..., $v_k$ for a given composed desire to an overall option utility taking into account the weight factors $w_1$, ..., $w_k$, according to some combination function $f(v_1, \ldots, v_k, w_1, \ldots, w_k)$.

The combination function in **3.** can be formalised in a number of manners; two common possibilities are:

- Euclidian Distance**:** $f(v_1, \ldots, v_k, w_1, \ldots, w_k) = \sqrt{(w_1v_1^2 + \ldots + w_kv_k^2)}$
- Manhattan Distance**:** $f(v_1, \ldots, v_k, w_1, \ldots, w_k) = w_1v_1 + \ldots + w_kv_k$

The LEADSTO property for combination is:

**LP41** ∀a:ACTION ∀$x_1$,...,$x_k$:ASPECT ∀$s_1$,...,$s_k$:SCALE ∀$v_1$,...$v_k$,$w_1$,...$w_k$:REAL
belief(has_aspect_utility(a, has_value($x_1$, $s_1$), $v_1$)) ∧ … ∧
belief(has_aspect_utility(a, has_value($x_k$, $s_k$), $v_k$)) ∧
weight_factor($x_1$, $w_1$) ∧ … ∧ weight_factor($x_k$, $w_k$) $\twoheadrightarrow_{0,0,1,1}$
belief(has_utility(a, d(has_value($x_1$, $s_1$), ..., has_value($x_k$, $s_k$)),
$f(v_1, \ldots, v_k, w_1, \ldots, w_k)$ ))

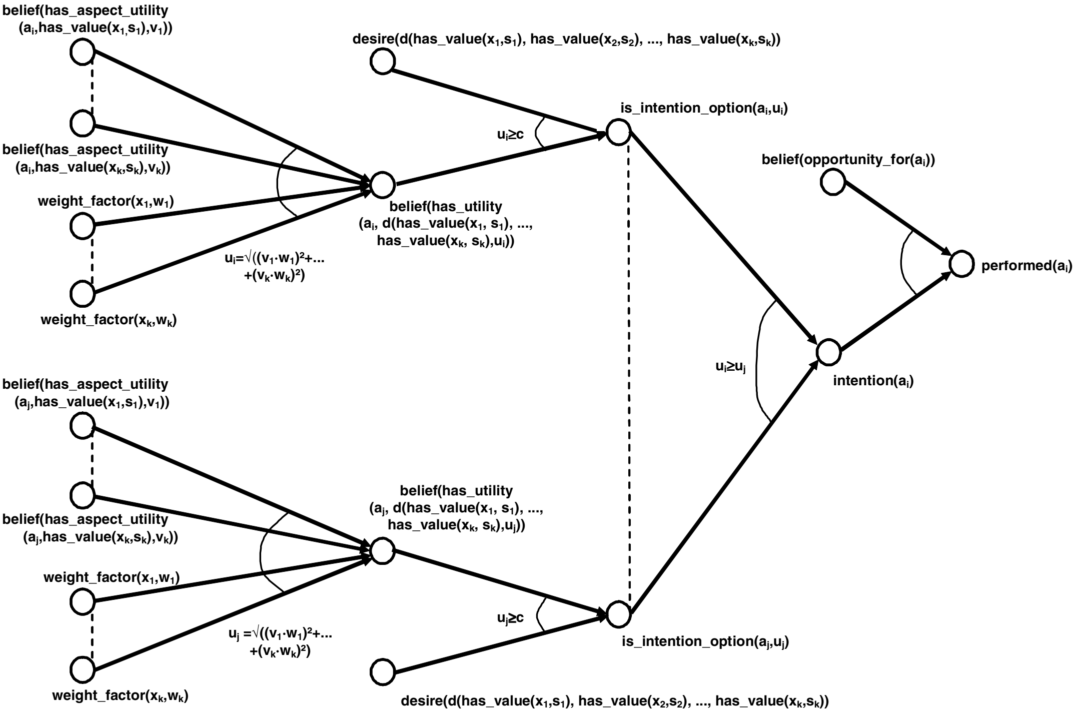

belief(has_aspect_utility ($a_i$,has_value($x_1$,$s_1$),$v_1$))
belief(has_aspect_utility ($a_i$,has_value($x_k$,$s_k$),$v_k$))
weight_factor($x_1$,$w_1$)
weight_factor($x_k$,$w_k$)
$u_i=\sqrt{((v_1 \cdot w_1)^2+\ldots+(v_k \cdot w_k)^2)}$
desire(d(has_value($x_1$,$s_1$), has_value($x_2$,$s_2$), ..., has_value($x_k$,$s_k$))
belief(has_utility ($a_i$, d(has_value($x_1$, $s_1$), ..., has_value($x_k$, $s_k$),$u_i$))
$u_i \geq c$
is_intention_option($a_i$,$u_i$)
belief(opportunity_for($a_i$))
performed($a_i$)
$u_i \geq u_j$
intention($a_i$)
belief(has_aspect_utility ($a_j$,has_value($x_1$,$s_1$),$v_1$))
belief(has_aspect_utility ($a_j$,has_value($x_k$,$s_k$),$v_k$))
weight_factor($x_1$,$w_1$)
weight_factor($x_k$,$w_k$)
$u_j =\sqrt{((v_1 \cdot w_1)^2+\ldots+(v_k \cdot w_k)^2)}$
belief(has_utility ($a_j$, d(has_value($x_1$, $s_1$), ..., has_value($x_k$, $s_k$),$u_j$))
$u_j \geq c$
is_intention_option($a_j$,$u_j$)
desire(d(has_value($x_1$,$s_1$), has_value($x_2$,$s_2$), ..., has_value($x_k$,$s_k$))

**Figure 1**: Utility-Based BDI-model

Next, the choice process is formalised. This is done in two steps. First, LP31 is replaced by LP31a, LP31b, and LP31c:

**LP31a** Desire d combined with the belief that a certain action a will lead to the fulfillment of d with utility u (≥c) will lead to the consideration of a as a possible intention option.
∀d:DESIRE ∀a:ACTION ∀u:REAL desire(d) ∧ belief(has_utility(a, d, u) ∧ u≥c)
$\twoheadrightarrow_{0.2,\, 0.2,\, 1,\, 1}$ is_intention_option(a, u)

Here c is a threshold value, for example 0.5. This is used to generate the options to be considered. To obtain only the intentions with highest utility, as a next phase, the selection process is modelled in two steps by:

**LP31b** If a1 and a2 are both intention options, but a2 has a higher utility, then a1 is ruled out as an intention option.
∀a1,a2:ACTION ∀u1,u2:REAL is_intention_option(a1,u1) ∧
is_intention_option(a2,u2) ∧ u1<u2 $\twoheadrightarrow_{0,\, 0,\, 1,\, 1}$
ruled_out_intention_option(a1, u1)

**LP31c** Eventually, an intention option that is not ruled out is selected as final intention.
∀a:ACTION ∀u:REAL is_intention_option(a, u) ∧
not ruled_out_intention_option(a, u) $\twoheadrightarrow_{0,\, 0,\, 1,\, 1}$ intention(a)

The complete utility-based decision model is depicted graphically in Figure 1. The circles denote state properties, and the arrows denote dynamic (LEADSTO) properties. Notice that the state properties of the type desire(...) are generated by the model described in the previous section.

Note that, in order to describe a specific decision making scenario with this model, the person described needs to have some expectancy about possible actions already at the start of the scenario. This expectancy may be triggered by observations (e.g., "I see a potential victim and no guardians, so I consider robbing this person"), or by other internal states (e.g., "I feel like seeking some thrill, so I consider robbing a bank this afternoon"). In the first case, the duration between the decision and the actual performance of the action is rather short, so that it is very likely that an opportunity for the considered action will indeed occur. In the second case, this duration will be longer, and it is possible that no opportunity will occur at all. The model can be used to describe both types of processes.

## An Example Simulation Trace

Based on the model shown above, a number of simulation experiments have been performed to test (for some simple scenarios) whether it shows the expected behaviour. In this section, an example simulation trace is described in detail. The example scenario involves a street robber (indicated by criminal1) who observes some possible targets, and is deliberating about whether or not to perform an assault (and if so, which assault to perform). For simplicity, we assume that there are two possible assaults to choose from (indicated by assault1 and assault2, respectively). In case of assault1, he would steal an old lady's purse, without using extreme violence. In case of assault2, he would steal a young man's brand new laptop. However, since this man seems to be rather strong, he would probably have to use violence to achieve his goal. The characteristics of both assaults, as well as criminal1's individual preferences, are shown in Table 1.

Table 1: Characteristics of a criminal and possible assaults[2]

| | weight factor (criminal1) | aspect utility (assault1) | aspect utility (assault2) |
|---|---|---|---|
| theory of mind | 0.04 | low, 0.7 | low, 0.9 |
| desire for aggressiveness | 0.04 | high, 0.3 | high, 0.8 |
| desire to act | 0.17 | high, 0.8 | high, 0.8 |
| desire to act safely | 0.02 | high, 0.1 | high, 0.1 |
| desire for actions with strong stimuli | 0.17 | high, 0.6 | high, 0.8 |
| desire for impulsiveness | 0.12 | medium, 0.5 | medium, 0.5 |
| positive emotional attitude towards others | 0.02 | low, 0.7 | low, 0.8 |
| negative emotional attitude towards others | 0.04 | low, 0.3 | low, 0.3 |
| desire for high gain | 0.19 | high, 0.5 | high, 0.8 |
| desire for low loss | 0.19 | high, 0.8 | high, 0.5 |

In the first column of the table, the different weight factors assigned to criminal1 can be seen. These weight factors, which add up to 1, show the relative importance of each aspect for the criminal. The weight factor for desire to act safely, for example, is 0.02. This means that criminal1 has a low interest in the desire to act safely. The weight factor for desire for actions with strong stimuli is 0.17, which means that he has a high desire for actions with strong stimuli. In the columns to the right of the weight factor, the utility of the different aspects is mentioned (in the column in the middle for assault1 and in the column to the right for assault2). The values describe in how far the aspect is present in this particular assault. For example, has_aspect_utility(assault1, has_value(desire_for_aggressiveness, high), 0.3) shows that assault1 does not contribute much to the high desire for aggressiveness. On the other hand, has_aspect_utility(assault2, has_value(desire_for_aggressiveness, high), 0.8) shows that assault2 contributes much to the high desire for aggressiveness.

The results of applying the simulation model to this example situation are shown in Figure 2. Here, time points are on the horizontal axis, whereas the different state properties are on the vertical axis. A box on top of a line indicates that a state property is true at that time point. As shown by this figure, the criminal immediately has a certain desire, represented as d1. Note that this stands for a complex desire represented as:

```
d(has_value(theory_of_min,low), has_value(desire_for_aggressiveness,high),
   has_value(desire_to_act,high), has_value(desire_to_act_safely,high),
   has_value(desire_for_actions_with_strong_stimuli,high),
   has_value(desire_for_impulsiveness,medium),
   has_value(positive_emotional_attitude_towards_others,low),
   has_value(negative_emotional_attitude_towards_others,low),
   has_value(desire_for_high_gain,high), has_value(desire_for_low_loss,high))
```

(which was not shown in the picture, for obvious reasons). This desire was generated by a complex process, involving a combination of biological and psychological factors. Due to space limitations, this part of the trace is not shown here either. However, more detailed simulation traces that include such processes are shown in Appendix A in (Bosse, Gerritsen and Treur, 2007b).

Based on the desire as described above, criminal1 then starts assessing the utilities of the two possible assaults (see the predicates belief(has_utility(…)) at time point 1), based on the aspect utilities and weight factors of these assaults. The action of stealing the young man's laptop (assault2) is assessed with value 0.678723, whereas the action of robbing the old lady's purse (assault1) has value 0.625532. Since both has_utility-values are higher than 0.5, both actions become possible intentions (see time point 2). Next, the criminal chooses the one with the highest utility, which leads to the intention to perform assault2 at time point 3. Later, when an opportunity for assault2 arises (time point 20), this assault is indeed performed (time point 21).

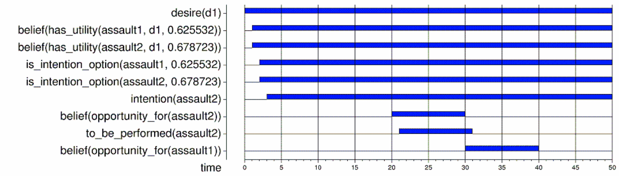


Figure 2: Example simulation trace

As illustrated by the trace in Figure 2 (and several similar traces that are not shown due to space limitations), the simulation experiments have indicated that the presented model successfully integrates personal biological and psychological aspects within the decision making process, which eventually leads to the selection of actions that correspond to the desires of the individual.

## Discussion

The few papers on simulation of criminal behaviour found in the literature usually address a limited number of aspects. For example, Brantingham and Brantingham (2004) discuss the possible use of agent modelling approaches to criminal behaviour in general, but do not report a specific model or case study. Moreover, Baal (2004) puts emphasis on the social network and the perceived sanctions. However, this model leaves the psychological and biological aspects largely unaddressed.

In this paper, an agent-based model to analyse criminal decision making is presented. The model combines a BDI-agent model (as described for the case of a criminal with Intermittent Explosive Disorder in (Bosse, Gerritsen and Treur, 2007b)) with a model for multi-attribute decision making, and applies this to the case of street robbery. It enables a choice between different options for actions fulfilling a complex desire, according to the Rational Choice Theory. The resulting agent model combines qualitative, logical aspects of a BDI-model with quantitative, numerical aspects of utility theory.

The simulation model has been made with the aim to formalise, in an abstract and computationally useful manner, the decision making behaviour of certain types of criminals as described in literature from Criminology. Such a model can be used in a number of ways. In the first place, it can be used to simulate behaviour for given scenarios of

[2] This approach assumes that an individual's preferences (i.e., the weight factors), as well as the characteristics of certain actions (i.e., aspect utilities), can be expressed by real numbers. For the presented examples, the chosen numbers are not necessarily claimed to be realistic, and should rather be seen as rough estimations by the authors.

circumstances occurring over time. This can be used to find out for such a given scenario of circumstances, whether a criminal of a certain type may show certain behaviour under these given circumstances. Second, the models can be used in the opposite direction, i.e., given a certain behaviour, to determine what kind of scenario of circumstances could have led to this behaviour. See (Bosse, Gerritsen, and Treur, 2007a) for details about how this can be done. Third, the models may be used to predict which behaviour certain types of criminals will show if circumstances are avoided or slightly changed (what-if reasoning). Using this approach, the behaviour of the subject can be modified by selecting or avoiding the appropriate circumstances, or by determining (cognitive) training programs for criminals. For all of these purposes, the model should be seen as a tool to support the user (e.g., the detective or the therapist) in its reasoning, by clarifying which scenarios are more plausible. It should however not be interpreted as a model of the absolute truth.

Validation of the model is a difficult issue. At least, the present paper has indicated that it is possible to integrate biological and psychological factors with rational factors within one model. Moreover, the model indeed shows the behaviour of different types of criminals as described in literature such as (Moir and Jessel, 1995; Raine, 1993; Delfos, 2004). In this sense the model has been validated positively. However, notice that this is a relative validation, only with respect to the literature that forms the basis of the model. In cases that the available knowledge about the behaviour and biological and psychological functioning of such criminal types is improving, the model can be improved accordingly. The modelling approach as put forward supports such an incremental development and improvement. The simulation model has been specified in a conceptual, not implementation-dependent manner, and is easy to maintain. In this sense the approach anticipates further development of the area of criminal behaviour.

In the cognitive literature, it is often claimed that cognition can be divided into two distinct systems: a low-level, emotional and unconscious system, and a high-level, evolutionary recent, conscious system, see, e.g., (Evans, 2003). At first sight, our proposed model seems to show significant similarities with this *dual process theory*. Our model to determine desires has characteristics of a low-level system, whereas the model for utility-based decision making resembles a high-level system. Future work will explore whether a more precise mapping can be made between the concepts introduced in our combined model and the concepts typically used in dual process theory. In addition, future work will explore how our model relates to models in which affective factors just 'bypass' decision making, such as in (Loewenstein, Weber, Hsee, and Welch, 2001).

## References

Baal, P.H.M. van (2004). *Computer Simulations of Criminal Deterence: from Public Policy to Local Interaction to Individual Behaviour*. Ph.D. Thesis, Erasmus University Rotterdam. Boom Juridische Uitgevers.

Baron-Cohen, S. (1995). *Mindblindness*. MIT Press.

Bartol, C.R. (2002). *Criminal Behavior: a Psychosocial Approach.* Sixth edition. Prentice Hall, New Jersey.

Bosse, T., Gerritsen, C., and Treur, J. (2007a). *Case Analysis of Criminal Behaviour.* In: Proceedings of the 20th International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems, IEA/AIE'07. Springer LNAI, to appear.

Bosse, T., Gerritsen, C., and Treur, J. (2007b). *Cognitive and Social Simulation of Criminal Behaviour: the Intermittent Explosive Disorder Case.* In: Proc. of the Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems, AAMAS'07. ACM Press, pp. 367-374.

Bosse, T., Jonker, C.M., Meij, L. van der, and Treur, J. (2005). LEADSTO: a Language and Environment for Analysis of Dynamics by SimulaTiOn. In: Eymann, T. et al. (eds.), *Proceedings of MATES'05*. LNAI, vol. 3550. Springer Verlag, 2005, pp. 165-178. Extended version in *Journal of AI Tools*, 2007, in press.

Bosse, T., Jonker, C.M., and Treur, J., (2004). Experiments in Human Multi-Issue Negotiation: Analysis and Support. In: Jennings et al. (eds.), *Proc. of the Third Int. Joint Conf. on Autonomous Agents and Multi-Agent Systems, AAMAS'04.* IEEE Computer Society Press, 2004, pp. 672-679.

Brantingham, P. L., and Brantingham, P. J. (2004). Computer Simulation as a Tool for Environmental Criminologists. *Security Journal*, 17(1), 21-30.

Cohen, L.E. and Felson, M. (1979). Social change and crime rate trends: a routine activity approach. *American Sociological Review*, vol. 44, pp. 588-608.

Cornish, D.B., and Clarke, R.V. (1986). *The Reasoning Criminal: Rational Choice Perspectives on Offending*. Springer Verlag.

Delfos, M.F. (2004). Children and Behavioural Problems: Anxiety, Aggression, Depression and ADHD; A Biopsychological Model with Guidelines for Diagnostics and Treatment. Harcourt book publishers, Amsterdam.

Evans, J.S.B.T. (2003). In two minds: dual-process accounts of reasoning. *Trends in Cognitive Sciences* 7, pp. 454-459.

Georgeff, M. P., and Lansky, A. L. (1987). Reactive Reasoning and Planning. In: *Proc. of the Sixth National Conf. on Artificial Intelligence*, AAAI'87. Menlo Park, California. American Association for Artificial Intelligence, 1987, pp. 677-682.

Jonker, C.M., and Treur, J., (2001). An Agent Architecture for Multi-Attribute Negotiation. In: B. Nebel (ed.), *Proc. of the 17th International Joint Conference on AI, IJCAI'01.* Morgan Kaufman, 2001, pp. 1195 - 1201.

Keeney, R., and H. Raiffa. (1976). *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. John Wiley & Sons, 1976.

Loewenstein, G. F., Weber, E. U., Hsee, C. K., and Welch, E, S. (2001). Risk as feelings. *Psychological Bulletin*, vol. 127, pp. 267-286.

Moir, A., and Jessel, D. (1995). *A Mind to Crime*: *the controversial link between the mind and criminal behaviour.* London: Michael Joseph Ltd; Penguin.

Raiffa, H. (1982). *The Art and Science of Negotiation*. Harvard University Press, Cambridge, MA, 1982.

Raine, A. (1993). *The Psychopathology of Crime: Criminal Behaviors as a Clinical Disorder*. New York, NY: Guilford Publications.

Rao, A.S. and Georgeff, M.P. (1991). Modelling Rational Agents within a BDI-architecture. In: Allen, J. et al. (eds.), *Proc. of the 2nd Int. Conference on Principles of Knowledge Representation and Reasoning, (KR'91).* Morgan Kaufmann, pp. 473-484.

# A Dynamical System Modelling Approach to Gross' Model of Emotion Regulation

**Tibor Bosse (tbosse@few.vu.nl) Matthijs Pontier (mpontier@few.vu.nl) Jan Treur (treur@few.vu.nl)**

Vrije Universiteit Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081, NL-1081 HV, Amsterdam, The Netherlands

## Abstract

This paper introduces a computational model for emotion regulation formalising the model informally described by Gross (1998). The model has been constructed using a high-level modelling language, and integrates both quantitative aspects (such as levels of emotional response) and qualitative aspects (such as decisions to regulate one's emotion). A number of simulation experiments have been performed, demonstrating that the computational model successfully reflects the model as described by Gross.

## Introduction

Emotions were historically seen as neural activation states without a function. However, recent research provides evidence that emotions are functional (e.g., Damasio, 2000). Emotions have a facilitating function in decision making, prepare a person for rapid motor responses, and provide information regarding the ongoing match between organism and environment. Emotions also have a social function. They provide us information about others' behavioural intentions, and script our social behaviour (Gross, 1998). In the past two decades, psychological research has started to focus more on *emotion regulation* (e.g., Gross, 1998, 2001; Ochsner and Gross, 2005; Thompson, 1994). In brief, emotion regulation is the process humans undertake in order to affect their emotional response. Recent neurological findings (such as bidirectional links between limbic centers, which generate emotion, and cortical centers, which regulate emotion) have changed the consensus that emotion regulation is a simple, top-down controlled process (Gross, 1998).

This article introduces a computational model to simulate emotion regulation, based on the process model described informally by Gross (1998, 2001). Such a model can be used for different purposes. In the first place, from a Cognitive Science perspective, it can provide insight in the process of emotion regulation. This may be useful for the purpose of developing therapies for persons that have difficulties in regulating their emotions (Burns et al., 2003; Towl and Crighton, 1996), for example, in work with forensic inpatients. In addition, a model for emotion regulation can be used in the field of Artificial Intelligence, see e.g. (Bates, 1994). For example, in the domain of virtual reality it can be used to let virtual agents show human-like behaviour regarding emotion regulation. Finally, computational models for emotion regulation may play a role within the field of Ambient Intelligence (Aarts, Harwig, and Schuurmans, 2001). For instance, when humans have to interact intensively with automated systems, it is useful if the system maintains a model of the emotional state (and the emotion regulation process) of the user. This enables the system to adapt the interaction to the user's needs.

Below, first Gross's model of emotion regulation is briefly discussed. The model describes a number of strategies humans use to adapt their emotion response levels, varying from situation selection to cognitive change and response modulation. Next, the dynamical system style modelling approach used is briefly introduced. After that, the simulation model formalising the model of Gross is described, and some simulation results are shown, both for ideal cases and for cases of over- and under-regulation. The paper concludes with a discussion.

## Gross' Model for Emotion Regulation

Gross (2001) describes a process model of emotion regulation using the following definition:

> 'Emotion regulation includes all of the conscious and nonconscious strategies we use to increase, maintain, or decrease one or more components of an emotional response'

The components he considers are (1) the *experiential* component, (the subjective feeling of the emotion), (2) the *behavioural* component (behavioural responses), and (3) the *physiological* component (responses such as heart rate and respiration). Humans use strategies to affect their level of emotional response for a given type of emotion, for example, to prevent a person from having a too high emotional or too low emotional response level. He differentiates between antecedent-focused strategies and response-focused strategies. *Antecedent-focused strategies* are applied to the process preparing for response tendencies before they are fully activated. *Response-focused strategies* are applied to the activation of the actual emotional response, when an emotion is already underway.

In his model, Gross distinguishes four different types of antecedent-focused emotion regulation strategies, which can be applied at different points in the process of emotion generation: *situation selection, situation modification, attentional deployment* and *cognitive change*. A fifth strategy, *response modulation*, is a response-focused strategy. Figure 1 shows an overview of these strategies.

The first antecedent-focused emotion regulation strategy in the model is *situation selection*: a person chooses to be in a situation that matches the emotional response level the person wants to have for a certain emotion. For example, a person can stay home instead of going to a party, because he is in conflict with someone who is going to that party. This is an example of down-regulating one's emotion.

The second antecedent-focused emotion regulation strategy in the model is *situation modification*. When this strategy is applied, a person modifies an existing situation so as to obtain a different level of emotion. For instance, when watching an irritating television program, one may zap to another channel.

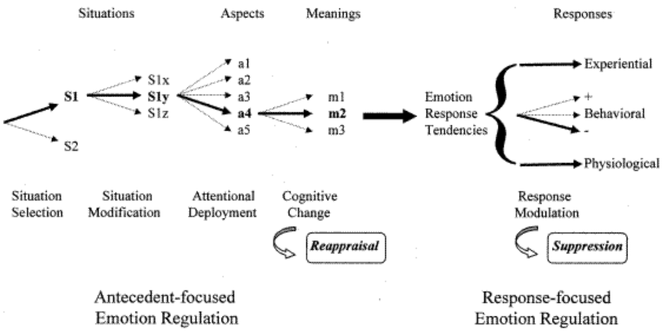


Figure 1: Emotion Regulation Model by Gross (1998).

The third antecedent-focused emotion regulation strategy is *attentional deployment*. This strategy refers to shifting your attention to a certain aspect. For example, one may close his eyes when watching an exciting penalty shoot-out. The fourth antecedent-focused emotion regulation strategy is *cognitive change*: selecting a cognitive meaning to an event. A specific type of cognitive change, which is aimed at down-regulating emotion, is *reappraisal*:

> 'Reappraisal means that the individual reappraises or cognitively re-evaluates a potentially emotion-eliciting situation in terms that decrease its emotional impact' (Gross, 2001).

An example of reappraisal is a case when a person loses a tennis match and blames the weather circumstances, instead of his own capacities. However, note that cognitive change could also be aimed at up-regulating emotion.

The fifth emotion regulation strategy, *response modulation*, a response-focused strategy, is applied after the emotion response tendencies have been generated: a person tries to affect the process of response tendencies becoming a behavioural response. A specific type of response modulation, again aimed at down-regulating, is *suppression*:

> 'Suppression means that an individual inhibits ongoing expressive behaviour.' (Gross, 2001).

An example of suppression is a person that hides being nervous when giving a presentation. As Gross considers response modulation to be not very effective, this strategy is not considered in the paper, although it would not be difficult to incorporate it in the computational model.

## Modelling Approach

Modelling the various aspects involved in Gross' model in an integrated manner poses some challenges. On the one hand, qualitative aspects have to be addressed, such as decisions to regulate one's emotion (e.g., by selecting a different situation). On the other hand, quantitative aspects have to be addressed, such as levels of emotional response.

The modelling approach based on the modelling language LEADSTO (Bosse, Jonker, Meij, and Treur, 2007) fulfils these desiderata. It integrates qualitative, logical aspects such as used in approaches based on temporal logic (e.g., Barringer et al., 1996) with quantitative, numerical aspects such as used in Dynamical Systems Theory (e.g., Ashby, 1960; Port and van Gelder, 1995). Direct temporal dependencies between two state properties in successive states are modelled by *executable dynamic properties* defined as follows. Let a and b be state properties of the form 'conjunction of ground atoms or negations of ground atoms', then the notation $a \twoheadrightarrow_{e,\, f,\, g,\, h}$ b means:

*If state property* a *holds for a certain time interval with duration g, then after some delay (between e and f) state property* b *will hold for a certain time interval of length h.*

Atomic state properties can have a qualitative, logical format (e.g., desire(d), expressing that desire d occurs), or a quantitative, numerical format (e.g., has_value(x, v) expressing that variable x has value v).

## Global Overview of the Model

Gross has described his process model for emotion regulation informally. In order to be able to formalise his model, for any given type of emotion a number of variables have been introduced. For convenience, the model concentrates on one specific type of emotion. In principle, this can be any emotion that is considered to be a basic human emotion, e.g., sadness, happiness, or anger (Ekman, Friesen, and Ellsworth, 1972).

In order to describe the regulation of such an emotion, the model takes into account a number of emotion regulation *strategies* that can be chosen. In the variant of the model as described in this paper, the four antecedent-focused emotion regulation strategies discussed by Gross are used (i.e., situation selection, situation modification, attentional deployment, and cognitive change). For the moment, response modulation is not considered. However, the model is generic in the sense that this set of strategies considered can easily be adapted. Based on the four strategies mentioned, in the formalisation four corresponding *elements* k are introduced, denoting the objects that are influenced by the particular strategies (see Table 1).

Table 1: Strategies and elements addressed in the model.

| **Strategy** | **Corresponding Element** |
|---|---|
| situation selection | situation |
| situation modification | sub_situation |
| attentional deployment | aspect |
| cognitive change | meaning |

In the model it is assumed that at each point in time, for each element k a certain choice is in effect, and this choice has a certain *emotional value* $v_k$ attached. This emotional value contributes to the *emotion response level* ERL via an element-specific weight factor $w_k$, thereby taking into account a persistency factor β indicating the degree of persistence or slowness of adjusting of the emotion response level when new emotional values are obtained. Someone whose emotions can change rapidly (e.g., who stops being angry in a few minutes after a fight) will have a low $\beta$.

Humans are always searching for a certain level of emotion depending on the person[1]. For instance, some enjoy extreme sports, while others prefer a more quiet kind of recreation. The level of emotion aimed at depends also on the type of emotion. Most humans aim at a relatively high level of emotion for happiness, while they aim at a lower level of emotion for fear. The regulation process starts by comparing the actual emotion response level ERL to the emotion response level ERL_norm aimed at. The difference d between the two is the basis for adjustment of the choices made for each of the elements k; based on these adjusted choices, each element k will have an adjusted emotional value $v_k$. The strength of such an adjustment is expressed by a modification factor $\alpha_k$, which can be seen as a flexibility or willingness (conscious or unconscious) to change one's emotional value for a certain element. For instance, the $\alpha$ for the element 'situation selection' can be seen as the flexibility to change one's situation. An overview of the variables used in the model is given in Table 2.

Table 2: Variables addressed in the model.

| Variable | Meaning |
|---|---|
| ERL | Emotion Response Level |
| ERL_norm | Emotion Response Level aimed at |
| D | Difference between ERL and ERL_norm |
| $\beta$ | Persistency factor for ERL |
| K | Elements indicating strategies incorporated |
| $w_k$ | Weight of element k in adjusting the ERL |
| $v_k$ | Emotional value for element k |
| $\alpha_k$ | Modification factor that represents the flexibility to change the emotional value of element k |

Some of these variables were chosen to be set at forehand and remain constant during the process (in particular ERL_norm, $\beta$, $w_k$, $\alpha_k$). The other variables depend on each other and on the fixed variables, as shown in a qualitative manner in the graph depicted in Figure 2.

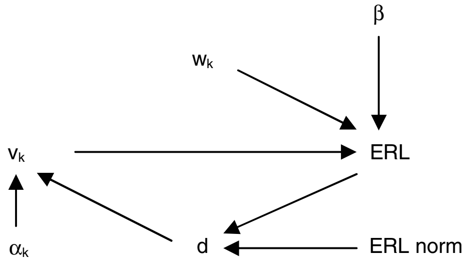


Figure 2: Dependencies between the variables.

This graph shows that the emotion response level ERL is affected by the emotional values $v_k$ for the different elements, the weights $w_k$ attached to these elements, and the persistency factor $\beta$ that indicates in how far the previous response level affects the current one. The difference d between response level and norm obviously depends on both of these factors. Finally, the emotional values $v_k$ for the different elements are affected by this difference d and the modification factor $\alpha_k$.

## The Quantitative Relations in the Model

To obtain a quantitative model, the emotion response level and the emotional values for the different elements for a given type of emotion are represented by real numbers in the interval [0, 2] (where 0 is the lowest possible emotion response level, and 2 the highest). In the model, a fixed level of emotion to aim at is assumed (the ERL norm), also expressed in a real number in the domain [0, 2]. As a simple illustration, suppose one wants to influence its state of anger by selecting an appropriate situation, and one deliberates whether to go to a party or not. This can be represented by introducing two different situations sit1 and sit2, for example with $v_{sit1}$=1.5 (since going to the party will increase the state of anger) and $v_{sit2}$=0.5 (staying home will decrease the state of anger). Moreover, the ERL norm can for instance be 0.7 (i.e., one aims at being a bit angry, but not too angry). In that case, if one's current ERL is already high, one will be likely to stay home (i.e., choose sit2), and vice versa.

The process of emotion regulation has a continuous nature. At any point in time, the characteristics of the current situation affect a person's emotional response level. Meanwhile, this emotional response level affects the person's choice for the emotional values $v_k$, which in turn influence the current situation (see also the cycle in Figure 2). An approach to model such a process is the Dynamical Systems Theory (DST) based on differential equations; e.g., (Port and van Gelder, 1995). To use differential equations for simulation, some form of *discretisation* is needed. Therefore, instead of differential equations, a set of difference equations is used, with a fixed step size s, that can be taken any size as desired.

### Updating the Emotional Response Level

Based on the above ideas, the emotion response level is recalculated each step by the following difference equation formula:

$$\text{new\_ERL} = (1-\beta) * \Sigma_k (w_k * v_k) + \beta * \text{ERL}$$

In this formula[2], new_ERL is the new emotion response level, and ERL is the old emotion response level. The persistency factor $\beta$ is the proportion of the old emotion response level that is taken into account to determine the new emotion response level. The new contribution to the emotion response level is calculated by the weighted sum of the emotional values: $\Sigma_k\ w_k * v_k$. By normalisation, the sum of all the weights $w_k$ is taken to be 1. According to the indication

[1] Although we use words like 'searching for' to describe this process, it is not claimed that this process is always a conscious, deliberate activity.

[2] Note that the formula can also be rewritten into the following difference equation format:
$\Delta\text{ERL} = (1-\beta) * (\Sigma_k (w_k * v_k) - \text{ERL})\ \Delta t$ with $\Delta\text{ERL} = \text{new\_ERL} - \text{ERL}$
This format shows more explicitly how $\beta$ determines the speed of adaptation of ERL to the new contribution $\Sigma_k\ w_k * v_k$; here $\Delta t$ is taken 1.

of Gross (2001), elements that are affected at an earlier point in the emotion regulation process have higher weights. Within the simulation model, the update of the emotional response level is expressed by the following dynamic property in LEADSTO format (where s is the step size):

**LP1 (Update Emotion Response Level)**
emotion_response_level(erl)
and has_weight(situation, w1)
and has_weight(sub_situation, w2)
and has_weight(aspect, w3)
and has_weight(meaning, w4)
and has_emotional_value(situation, v1)
and has_emotional_value(sub_situation, v2)
and has_emotional_value(aspect, v3)
and has_emotional_value(meaning, v4)
$\twoheadrightarrow_{0,\,0,\,s,\,s}$ emotion_response_level( (1-beta) *
(w1*v1 + w2*v2 + w3*v3 + w4*v4) + beta * erl)

## Updating the Emotional Values

The chosen emotional values $v_k$, which affect the emotion response level, are on their turn recalculated each step by the following set of difference equations:

$$d = ERL - ERL_{norm}$$
$$\Delta v_k = -\alpha_k * d / d_{max}\, \Delta t$$
$$new\_v_k = v_k + \Delta v_k$$

In these formulas, $new\_v_k$ is the new emotional value $v_k$, and $old\_v_k$ is the old emotional value $v_k$., while $\Delta v_k$ is the change of the emotional value $v_k$ (either positive or negative), and $\Delta t$ the time step, which is taken 1 in this paper. The change in the emotional value $v_k$ is calculated by the formula $-\alpha_k * d / d_{max}$. In this formula, $\alpha_k$ is the modification factor, and d is the difference between the actual emotion response level and the desired emotion response level (represented by ERL_norm). Here $d_{max}$ is an estimation of the maximum difference that can be reached. So $d / d_{max}$ is the proportion of the maximal reachable level of emotion above the level of emotion aimed at (or below this level, if d is negative).

When the actual emotion response level equals the desired emotion response level, then $d = 0$; this means that $\Delta v_k = 0$, so the emotion response level will not change. Moreover, a person will ‘choose’ an element with a more extreme emotional value $v_k$ when (s)he is more flexible in this emotional value $v_k$ (this is the case when $\alpha_k$ is high), or when (s)he experiences an emotion response level that is further away from the desired emotion response level (this is the case when d deviates more from 0). Within the simulation model, the update of emotional values is expressed as follows:

**LP2 (Update Emotional Values)**
emotion_response_level(erl) and erl_norm(erl_norm)
and has_emotional_value(element, v)
and has_modification_factor(element, a)
$\twoheadrightarrow_{0,\,0,\,s,\,s}$
has_emotional_value(element, v – a * (erl - erl_norm) / dmax)

# Simulation Results

A number of experiments have been performed to test what kind of behaviour can be simulated. Each subsection below addresses a specific type of scenario. Two types of cases are addressed: those with an optimal form of regulation (compared to the emotion response level aimed at), and cases of over- and under-regulation. The different scenarios are established by taking different settings for the modification factors $\alpha_k$. The values of the other variables are the same for all experiments described in this section, see Table 3.

Table 3: Values of variables used in the simulations.

| Variable | Fixed Value |
|---|---|
| $ERL_{norm}$ | 0.5 |
| $\beta$ | 0.7 |
| $w_1$ | 0.35 |
| $w_2$ | 0.30 |
| $w_3$ | 0.20 |
| $w_4$ | 0.15 |
| s | 1 |

| Variable | Initial Value |
|---|---|
| ERL | 1.85 |
| $v_1$ | 1.90 |
| $v_2$ | 1.85 |
| $v_3$ | 1.80 |
| $v_4$ | 1.75 |

As shown in the table, the person considered has an optimal level of emotion of 0.5 in the domain [0, 2]. The factor $\beta$ is set to 0.7, which means that in each step, 70% of the old emotional response level persists, and the remaining 30% is determined by the new emotional values. The weight attached to situation selection is 0.35, which means that the selected situation determines 35% of the 30% of the new emotion response level that is determined by the emotional values. Similarly, the weights for situation modification, attentional deployment, and cognitive change are set to 0.30, 0.20, and 0.15, respectively. The results of the experiments are shown and explained below.

## Optimal forms of emotion regulation

In the first experiment, all modification factors $\alpha_k$ were set to 0.15. The results are shown in Figure 3. In such figures, time is on the horizontal axis; the values of the different variables are shown on the vertical axis.

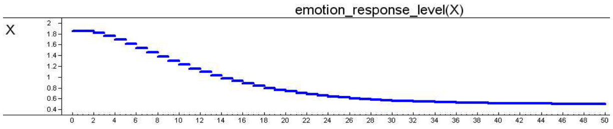


Figure 3: Results for an optimal case (equal $\alpha_k$).

The emotional response level decreases monotonically without decreasing below the level aimed at. So, the subject gradually reaches his level of emotion aimed at. The emotional values show similar behaviour (due to space limitations not shown here).

In the second experiment, the subject has for each element k a different flexibility $\alpha_k$ in emotion regulation:

$$\alpha_1 = 0.20,\ \alpha_2 = 0.15,\ \alpha_3 = 0.10,\ \alpha_4 = 0.05$$

The results of this experiment are shown in Figure 4. Here, the emotion response level reaches the emotion response level of 0.5 aimed for in a reasonable amount of time, just like in the optimal case. However, the way the emotional values change in order to achieve this differs from the first experiment. Here, it is important to note that the scale on the vertical axis is not the same for the different graphs in Figure 4. The graphs show that the emotion response levels of the elements with a higher $\alpha$ descend much quicker and further than the elements with a lower $\alpha$. For example, situation selection ($\alpha$=0.20) has reached an emotional value of 0 at the end of the simulation, whereas cognitive change ($\alpha$=0.01) changes only a little bit, and reaches an emotional value of about 1.3. This means that the subject finds a way to reach his/her level of emotion aimed for, and does this by changing his/her behaviour more for the elements for which (s)he has a higher flexibility.

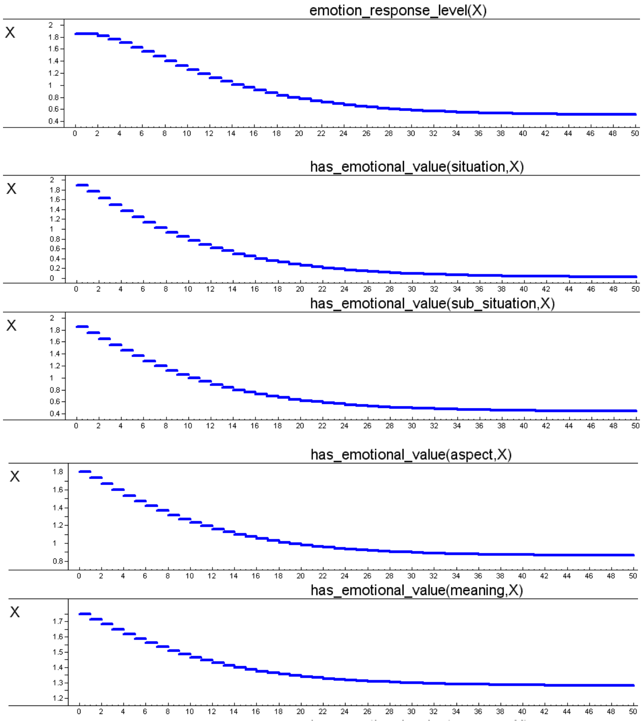


Figure 4: Results for an optimal case (different $\alpha_k$).

## Over- and under-regulation

In the third experiment, the modification factors $\alpha_k$ for all elements were set to 0.4. This means that the subject has a relatively high flexibility in emotion regulation, for all elements. The behaviour of the emotion response level in this experiment is shown in Figure 5.

In this case, the emotion response level starts to decrease rapidly, immediate after the experiment has started. However, it decreases below the level of 0.5 aimed at. It reaches its minimum after 15 steps in the simulation, at about 0.3: the subject over-regulates his/her emotion. After this, the emotion response level starts to raise until it is just above the optimal level of 0.5, and stays more or less at this value aimed at for the rest of this simulation.

The lower part of Figure 5 shows how the subject changed his/her emotional values in order to achieve this. These emotional values all show similar behaviour, since the $\alpha_k$'s, which represent the flexibility and willingness to change behaviour, were set to the same value. Also, the graphs of the emotional values are comparable to the graph of the emotion response level. The emotional values make a somewhat steeper curve, especially at the start of the graph. This makes sense, because the emotion response level is only for 30% determined by the emotional values, and for 70% by its own old value.

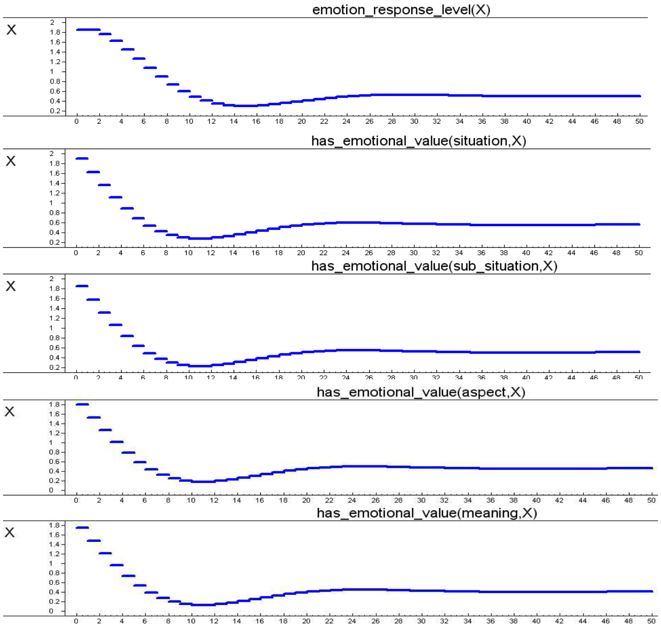


Figure 5: Results for the over-regulation case.

In the fourth experiment presented, the subject has a very low flexibility in emotion regulation, with an $\alpha_k$ value of 0.01 for all elements. The results of this experiment are shown in Figure 6. In this experiment, the emotion response level decreases extremely slowly: under-regulation. After 50 steps, it has only decreased by 0.3 until 1.55, as can be seen in the graph.

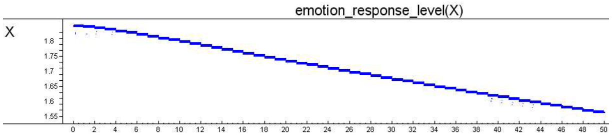


Figure 6: Results for the under-regulation case.

## Discussion

In this paper, a formal model for Gross' (informally described) model of emotion regulation has been introduced. The emotion regulation model has been constructed using the high-level simulation language LEADSTO as a modelling vehicle, and integrates both quantitative, dynamical system aspects (such as levels of emotional response) and qualitative aspects (such as decisions to regulate one's emotion). Simulation experiments have been performed for different situations, by using different settings for the modification factors $\alpha_k$: for ideal cases (all $\alpha_k$ are medium, or the $\alpha_k$ have different values), for cases of over-regulation (all $\alpha_k$ are high), and for cases of under-regulation (all $\alpha_k$ are low). The experiments show that different values for the modification factors $\alpha_k$ indeed result in different patterns.

As a preliminary validation of the model, the simulation results have been compared with the predicted behaviours for different situations as described by Gross, which are (partly) based on empirical evidence (Gross, 1998, 2001). The patterns produced by the model were found consistent with Gross' descriptions of examples of human regulation processes. Validation involving extensive comparison with detailed empirical data is left for future work.

Although the process of emotion regulation is widely investigated in the literature (e.g., Gross, 1998, 2001; Ochsner and Gross, 2005; Thompson, 1994), not so many contributions address the possibility of developing a computational model of this process. The computational models that have been developed so far either address some very specific aspects of the process at a more detailed (neurological) level, see e.g. (Thayer and Lane, 2000), or they aim at incorporating emotions into software agents, in which case they focus more on emotion elicitation (appraisal) than on emotion regulation, see e.g. (Armony et al., 1997; Bates, 1994; Velasquez, 1997). The current paper can be seen as an attempt to build a bridge between both directions. It formalises an existing theory about emotion regulation using a high-level modelling language, but still in enough detail to be able to generate useful simulation traces. As such, it has similarities with the work by Marsella and Gratch (2003), who propose an approach to incorporate both appraisal and coping behaviour into virtual humans. Their approach makes use of plan-based causal representations, augmented with decision-theoretic planning techniques, whereas our approach uses dynamical systems representations. Other differences are that they propose a "content model", in which appraisal and regulation operate on rich representations of the emotion-evoking situation, and that their work has been evaluated against clinical data.

The presented model is still in an early stage of development. For example, the modification factors $\alpha_k$ are currently fixed. In order to make the model adaptive, these factors can be made adjustable. A way to accomplish this is to adapt the values of the $\alpha_k$ to one's satisfaction about the past emotion regulation process. This way, the model could simulate cases in which humans learn to select the ideal situations, as in certain types of therapy. Another possible extension to the model would be to make the desired emotion response level ERL_norm dynamic, so that it can depend on specific circumstances. A final extension would be to represent the different elements k using more complex knowledge structures, and to enable the model to dynamically derive the different emotional values from these structures, as is done, for example, in (Marinier and Laid, 2004). Future work will explore such possibilities.

## References

Aarts, E., Harwig, R., and Schuurmans, M. (2001). Ambient Intelligence. In *The Invisible Future: The Seamless Integration of Technology into Everyday Life*, McGraw-Hill, 2001.

Armony, J.L., Servan-Schreiber, D., Cohen, J.D., and Ledoux, J.E. 1997). Computational modeling of emotion: Explorations through the anatomy and physiology of fear conditioning. *Trends in Cognitive Sciences*, vol. 1, pp. 28-34.

Ashby, R. (1960). *Design for a Brain*. Second Edition. Chapman & Hall, London. First edition 1952.

Barringer, H., Fisher, M., Gabbay, D., Owens, R., and Reynolds, M. (1996). *The Imperative Future: Principles of Executable Temporal Logic*, John Wiley & Sons, 1996.

Bates, J. (1994). The role of emotion in believable agents. *Communications of the ACM*, Vol. 37, No. 7, pp. 122-125.

Bosse, T., Jonker, C.M., Meij, L. van der, and Treur, J. (2007). A Language and Environment for Analysis of Dynamics by SimulaTiOn. *Int. Journal of Artificial Intelligence Tools*, to appear, 2007. Short version in: Eymann, T. et al. (eds.), Proc. MATES'05. Springer LNAI, vol. 3550, 2005, pp. 165-178.

Burns, M., Bird, D., Leach, C., and Higgins, K. (2003). Anger management training: the effects of a structured programme on the self-reported anger experience of forensic inpatients with learning disability. *Journal of Psychiatric and Mental Health Nursing*, Vol. 10, pp. 569-577.

Damasio, A. (2000). *The Feeling of What Happens: Body, Emotion and the Making of Consciousness*. MIT Press.

Ekman, P., Friesen, W.V., and Ellsworth, P. (1972). *Emotion in the human face: guidelines for research and integration of Findings*. New York: Pergamon.

Gross, J.J. (1998). The Emerging Field of Emotion Regulation: An Integrative Review. *Review of General Psychology*, vol. 2, No. 3, pp. 271-299.

Gross, J.J. (2001). Emotion Regulation in Adulthood: Timing is Everything. *Current directions in psychological science*, Vol. 10, No. 6, pp. 214-219.

Marinier, R.P., and Laird, J.E. (2004). Toward a Comprehensive Computational Model of Emotions and Feelings. In: *Proc. of the Sixth International Conference on Cognitive Modeling, ICCM'04*. Lawrence Erlbaum, Mahwah, NJ, pp. 172-177.

Marsella, S., and Gratch, J. (2003). Modeling coping behavior in virtual humans: Don't worry, be happy. In *Proceedings of Second International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS'03*. ACM Press, pp. 313-320.

Ochsner, K.N., and Gross, J.J. (2005). The cognitive control of emotion. *Trends in Cognitive Sciences*, vol. 9, pp. 242-249.

Port, R.F., and Gelder, T. van (eds.). (1995). *Mind as Motion: Explorations in the Dynamics of Cognition*. MIT Press, Cambridge, Mass.

Thayer, J.F., and Lane, R.D. (2000). A model of neurovisceral integration in emotion regulation and dysregulation. *Journal of Affective Disorders*, Vol. 61, pp. 201-216.

Thompson, R.A. (1994). Emotion regulation: A theme in search of definition. In N.A. Fox (Ed.), *The development of emotion regulation: Biological and behavioral aspects. Monographs of the Society for Research in Child Development*, Vol. 59 (Serial No. 240), pp. 25-52.

Towl, G.J., and Crighton, D.A. (1996). *The Handbook of Psychology for Forensic Practitioners*. Routledge, New York.

Velasquez, J. (1997). Modeling Emotions and Other Motivations in Synthetic Agents. In: *Proceedings of the Fourteenth National Conference on Artificial Intelligence, AAAI'97*, pp. 10-15.

# Modelling Animal Behaviour Based on Interpretation of Another Animal's Behaviour

**Tibor Bosse (tbosse@few.vu.nl) Zulfiqar A. Memon (zamemon@few.vu.nl) Jan Treur (treur@few.vu.nl)**

Vrije Universiteit Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081, NL-1081 HV, Amsterdam, The Netherlands

## Abstract

For certain animals, the capability to interpret and anticipate on another animal's behaviour may be crucial for survival. To this end, as is often claimed informally, an animal may apply a Theory of Mind to estimate what the other animal has on its mind. This paper uses a formal BDI-based agent model for Theory of Mind to formalise and simulate such a situation. The model uses BDI-concepts to describe a form of metacognition: a cognitive process of an agent about the cognitive process of another agent, which is also based on BDI-concepts. This paper explores whether this formal model is applicable to certain animal species. A specific case study is addressed, which involves the scenario of a prey that manipulates the behaviour of a predator. For this scenario, simulation experiments have been performed, and their results are discussed.

## Introduction

For certain animals, to function effectively in interaction with other animals, it is useful if they are able to interpret, estimate and anticipate on potential behaviour of animals around it. It is often assumed that this requires metacognition in some form of Theory of Mind (Baron-Cohen, 1995; Bogdan, 1997; Malle, Moses, and Baldwin, 2001). Such a Theory of Mind can be exploited by an animal in two different manners. The first manner is just to predict the behaviour in advance, in order to be prepared that it will occur. A second manner is to affect the occurrence of behaviour by manipulating the occurrence of circumstances that are likely to lead to it.

One of the ways to model an agent B exploiting a Theory of Mind about an agent A is to use a BDI-model (based on beliefs, desires and intentions) to describe agent A's cognitive processes and actions. To model the agent B's own behaviour a BDI-model can be used as well; in this way within agent B's cognitive processes, at two levels BDI-models play a role. This type of model will be exploited in this paper to model the behaviour of higher animals such as primates and dogs. For example, for agent B the desire is generated that agent A will not perform the action to kill B, and that agent A will in particular not generate the desire or intention to do so. Based on this desire of B, for example, the refined desire of B can be generated that agent A will not believe that agent B is reachable. Based on the latter desire, an intention and action can be generated to present circumstances to agent A that will make A believe that B is not reachable.

The vehicle used to model the two-level BDI-model is the modelling language LEADSTO (Bosse, Jonker, Meij, and Treur, 2007). In this language, direct temporal dependencies between two state properties in successive states are modelled by *executable dynamic properties*. The LEADSTO format is defined as follows. Let $\alpha$ and $\beta$ be state properties of the form 'conjunction of ground atoms or negations of ground atoms'. In the LEADSTO language the notation $\alpha \twoheadrightarrow_{e, f, g, h} \beta$, means:

*If state property $\alpha$ holds for a certain time interval with duration g,*
*then after some delay (between e and f) state property $\beta$ will hold*
*for a certain time interval of length h.*

Here, atomic state properties can have a qualitative, logical format, such as an expression desire(d), expressing that desire d occurs, or a numerical format such as an expression has_value(x, v) which expresses that variable x has value v.

In this paper, first the general BDI-model is explained. This BDI-model is illustrated by a case study about a predator that desires to kill a prey. The next section describes how the simple model can be extended to a two-level BDI-model of an agent that also involves another agent's BDI-model. This two-level BDI-model is illustrated by a case study that elaborates upon the previous example: it addresses the scenario of a prey that has metacognition addressing analysis of the behaviour of a predator, and prevents being attacked. Based on this model, some simulation experiments and their results are discussed.

## The BDI-Model

The BDI-model bases the preparation and performing of actions on beliefs, desires and intentions (e.g., Georgeff and Lansky, 1987; Jonker, Treur, and Wijngaards, 2003; Rao and Georgeff, 1991; 1995). This model shows a long tradition in the literature, going back to Aristotle's analysis of how humans (and animals) can come to actions; cf. (Aristotle, 350 BCa; 350BCb). He discusses how the occurrence of certain internal (mental) state properties within the living being entail or cause the occurrence of an action in the external world. Based on this, Aristotle introduced the following pattern to explain action (called practical syllogism):

| | |
|---|---|
| If | A has a desire D |
| and | A has the belief that X is a (or: the best) means to achieve D |
| then | A will do X |

The BDI-model incorporates such a pattern to explain behaviour in a refined form. Instead of a process from desire to action in one step, as an intermediate stage first an intention is generated, and from the intention the action is generated. Thus the process is refined into a two-step process. See Figure 1 for the generic structure of the BDI-model in causal-graph-like style, as often used to visualise LEADSTO specifications. Here the box indicates the borders of the agent, the circles denote state properties, and the arrows indicate dynamic properties expressing that one state property leads to (or causes) another state property. In this model, an action is performed when the subject has the intention to do this action and it has the belief that certain circumstances in the world are fulfilled such that the opportunity to do the action is there. Beliefs are created on the basis of observations. The intention to do a specific type of action is created if there is some desire D, and there is the belief that certain circumstances in the world state are there, that make it possible that performing this action will fulfil this desire (this is the kind of rationality criterion discussed above; e.g., what is called means-end analysis is covered by this). Whether or not a given action is adequate to fulfil a given desire depends on the current world state; therefore this belief may depend on other beliefs about the world state. Instantiated relations within the general BDI-model as depicted by arrows in graphical format in Figure 1 can be specified in formal LEADSTO format as follows:

$$\mathrm{desire}(D) \wedge \mathrm{belief}(B1) \twoheadrightarrow \mathrm{intention}(P)$$
$$\mathrm{intention}(P) \wedge \mathrm{belief}(B2) \twoheadrightarrow \mathrm{performs}(P)$$

with appropriate desire D, action P and beliefs B1, B2. Note that the beliefs used here both depend on observations, as shown in Figure 1. Furthermore, ∧ stands for the conjunction operator (and) between the atomic state properties (in the graphical format denoted by an arc connecting two (or more) arrows). Often, dynamic properties in LEADSTO are presented in *semi-formal* format, as follows:

At any point in time
if desire D is present
and the belief B1 is present
then the intention for action P will occur

At any point in time
if the intention for action P is present
and the belief B2 is present
then the action P will be performed

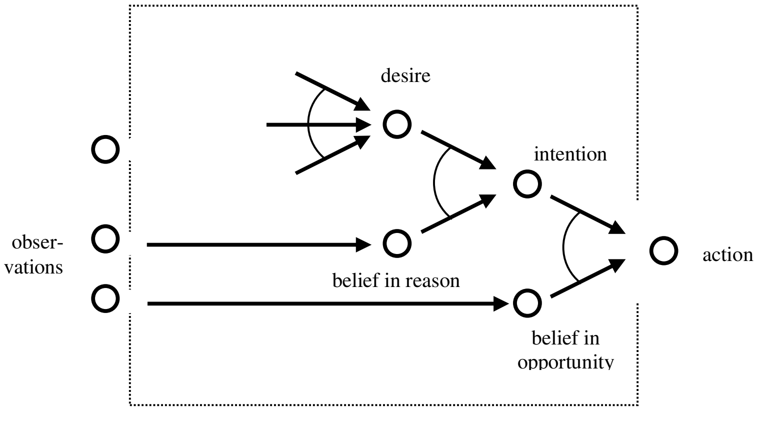


Figure 1: Structure of the general BDI-model.

As a generic template, including a reference to the agent X concerned, this can be expressed by:

For any desire D, world state property Z, and action Y such that has_reason_for(X, D, Z, Y) holds:

$$\mathrm{desire}(X, D) \wedge \mathrm{belief}(X, Z) \twoheadrightarrow \mathrm{intention}(X, Y)$$

For any world state property Z and action Y such that is_opportunity_for(X, Z, Y) holds:

$$\mathrm{intention}(X, Y) \wedge \mathrm{belief}(X, Z) \twoheadrightarrow \mathrm{performs}(X, Y)$$

Here has_reason_for(X, D, Z, Y) is a relation that can be used to specify which state property Z is considered a reason to choose a certain intention Y for desire D. Similarly is_opportunity_for(X, Z, Y) is a relation that can be used to specify which state property Z is considered an opportunity to actually perform an intended action Y.

Assuming that beliefs are available, what remains to be generated in this model are the desires. For desires, there is no generic way (known) in which they are to be generated in the standard model. Often, in applications, generation of desires depends on domain-specific knowledge.

## A BDI-Model for Animal Behaviour

To illustrate the BDI-model described above by a specific example, a specific scenario is addressed, in the domain of a predator that wants to attack a prey. This scenario was inspired by (Bogdan, 1997), who introduces the notion of a *goal setting for interpretation* (i.e., a situation in which an organisms needs to interpret the behaviour of another organism in order to satisfy its private goals), which he illustrates as follows:

> 'To illustrate, suppose that organism *A* (interpreter) has a private goal (say resting). It interferes with the goal of another organism *S* (subject), which is to eat *A*. Those *A*-type organisms will be selected who manage to form the social or *S*-regarding goal of avoiding the nasty type *S* by countering their inimical behavior, say by threat or deception. The latter goal in turn selects for interpretation, specifically, for interpretation goals such as desire identification and behavior prediction. Those *A*-type organisms are selected who form and reach such interpretation goals. The environment that selected for such accomplishments is a goal setting *of a certain kind*, say of behavior manipulation by behavior prediction and desire identification. There could be as many kinds of goal settings for interpretation as there are interpretation goals and tasks to achieve them, and hence as many skills.' (Bogdan, 1997), p. 111

Based on this description, a scenario is considered that involves a predator (agent A) and a prey (agent B). Assume that, under certain circumstances, the predator tries to kill the prey, and the prey tries to avoid this by manipulation. First, only the behaviour of the predator is addressed (in which no Theory of Mind is involved). However, in a later section, the cognitive process of the prey involving Theory of Mind is addressed as well. Using the BDI-model as introduced above, the example is made more precise as follows. The *desire* to eat the prey is created after time t by the predator if the following holds at time t:

- the predator has the belief that the prey is alone
(i.e., not surrounded by other animals)

The *intention* to kill the prey is generated after time t if the following holds at time t:

- the predator has the desire to eat the prey
- the predator has the belief that the prey is weak
(i.e., that it does not show strong, aggressive behaviour)

The *action* to kill the prey is generated after time t if the following holds at time t:

- the predator has the intention to kill the prey
- the predator has the belief that the prey is slow
(i.e., that it does not run very fast, so that it can be caught)

Using the generic template discussed, via the relations

has_reason_for(predator, eat_prey,
not(prey_shows_aggressive_behaviour), kill_prey)
is_opportunity_for(predator, not(prey_runs_fast), kill_prey)

the following model for agent predator is obtained:

belief(predator, not(prey_surrounded_by_other_animals)) ↠
desire(predator, eat_prey)
desire(predator, eat_prey) ∧
belief(predator, not(prey_shows_aggressive_behaviour)) ↠
intention(predator, kill_prey)
intention(predator, kill_prey) ∧ belief(predator, not(prey_runs_fast)) ↠
performs(predator, kill_prey)

## The Two-Level BDI-Model

According to the intentional stance (Dennett, 1987, 1991), an agent is assumed to decide to act and communicate based on intentional notions such as beliefs about its environment and its desires and intentions. These decisions, and the intentional notions by which they can be explained and predicted, generally depend on circumstances in the environment, and, in particular, on the information on these circumstances just acquired by interaction (i.e., by observation and communication), but also on information acquired by interaction in the past. To be able to analyse the occurrence of intentional notions in the behaviour of an observed agent, the observable behavioural patterns over time form a basis; cf. (Dennett, 1991).

In the model presented in this paper, the instrumentalist perspective is taken as a point of departure for a Theory of Mind. More specifically, the model describes the cognitive process of an agent B that applies the intentional stance to another agent A by attributing beliefs, desires and intentions. Thus, for agent B a Theory of Mind is obtained using concepts for agent A's beliefs, desires and intentions. For example, in case a prey (agent B) fears to be attacked by a predator (agent A), it may analyse in more detail under which circumstances the predator may generate the desire and intention to attack.

As a next step, the model is extended with BDI-concepts for agent B's own beliefs, desires and intentions as well. By doing this, agent B is able to not only *have* a theory about the mind of agent A, but also to *use* it within its own BDI-based cognitive processes to generate its actions. To this end, a number of meta-representations expressed by meta-predicates are introduced, e.g.:

belief(B, desire(A, D))

This expresses that agent B believes that agent A has desire D.

desire(B, not(intention(A, X)))

This expresses that agent B desires that agent A does not intend action X.

belief(B, depends_on(performs(A, X), intention(A, X)))

This expresses that agent B believes that, whether A will perform action X depends on whether A intends to do X. Note that the third meta-statement has a more complex structure than the other two, since it represents a statement about a *dynamic property*, rather than a statement about a *state property*. These dependencies can be read from a graph such as depicted in Figures 1 and 2 (right hand side). For example, it is assumed that agent B knows part of this graph in his Theory of Mind, expressed by beliefs such as:

belief(B, depends_on(performs(A, X), intention(A, X)))
belief(B, depends_on(performs(A, P), belief(A, B2)))
belief(B, depends_on(intention(A, P), desire(A, D)))
belief(B, depends_on(intention(A, P), belief(A, B1)))
belief(B, depends_on(desire(A, D), belief(A, B3)))
belief(B, depends_on(belief(A, X), hears(A, X)))

Desire refinement in the BDI-model for an agent B attributing motivations to an agent A is formulated (in LEADSTO format) by:

desire(B, X) ∧ belief(B, depends_on(X, Y)) ↠ desire(B, Y)
desire(B, X) ∧ belief(B, depends_on(X, not(Y))) ↠ desire(B, not(Y))
desire(B, not(X)) ∧ belief(B, depends_on(X, Y)) ↠ desire(B, not(Y))
desire(B, not(X)) ∧ belief(B, depends_on(X, not(Y))) ↠ desire(B, Y)

Moreover the following schemes for intention and action generation are included in the model. For any desire D, world state property Z, and action Y such that has_reason_for(B, D, Z, Y) holds:

desire(B, D) ∧ belief(B, Z) ↠ intention(B, Y)

For any world state property Z and action Y such that is_opportunity_for(B, Z, Y) holds:

intention(B, Y) ∧ belief(B, Z) ↠ performs(B, Y)

Moreover, some dynamic properties of the world are needed:

performs(B, Y) ∧ has_effect(Y, C) ↠ holds_in_world(C)
holds_in_world(C) ↠ observes(A, C)

For an overview of the complete two-level BDI-model, see Figure 2.

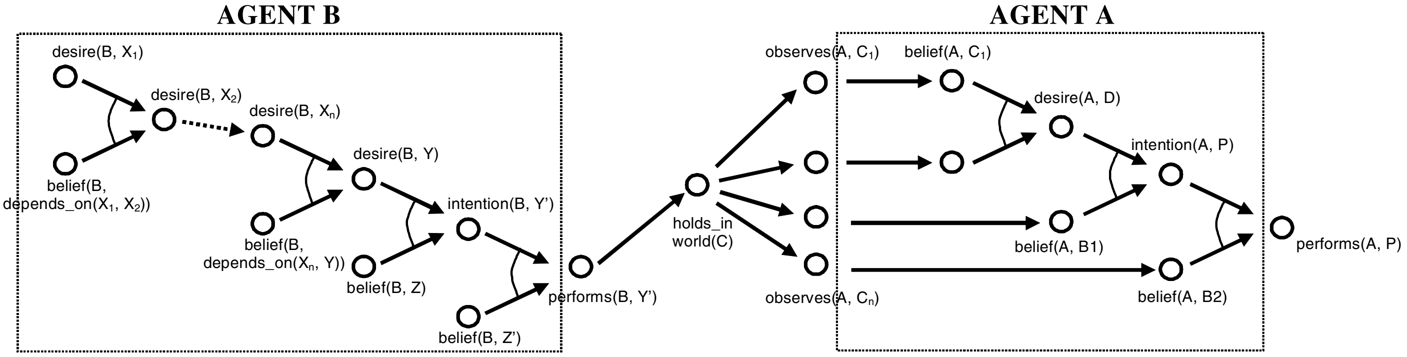


Figure 2: Structure of the two-level BDI-model.

## A Two-Level BDI-Model for Animal Behaviour

The above model was used to describe how the prey agent (from the case described earlier) acts in an anticipatory manner to avoid the predator's desire, intention and/or action to occur. The initial desire of the prey is that the predator does not perform the action to kill it:

desire(prey, not(performs(predator, kill(prey))))

Fulfilment of this desire can be obtained in the following three manners:

*Avoiding the predator's desire to occur*

This can be obtained when the predator observes that the prey is surrounded by other animals. This will make the condition in the predator's desire generation as described earlier fail.

*Avoiding the predator's intention to occur (given that the desire occurs)*

This can be obtained by refutation of the belief that plays the role of the reason to generate the intention in the predator's intention generation as described earlier, i.e., the belief that the prey is weak (and does not show aggressive behaviour).

*Avoiding the predator's action to occur (given that the intention occurs)*

This can be obtained by refutation of the belief that plays the role of opportunity in the predator's desire action as described, i.e., the belief that the prey is slow (and does not run fast).

For convenience, the model does not make a selection but addresses all three options to prevent the killing action. This means that the prey generates *desires* for:

- The predator observes that the prey is surrounded by other animals
  observes(predator, prey_surrounded_by_other_animals)
- The predator observes that the prey shows aggressive behaviour
  observes(predator, prey_shows_aggressive_behaviour)
- The predator observes that the prey runs fast
  observes(predator, prey_runs_fast)

To fulfil these desires, *intentions* are to be generated by the prey to actions such as:

- call for help of other animals: call_for_help
- show aggressive behaviour: show_aggressive_behaviour
- run fast: run_fast

*Reasons* for the prey to choose for these intentions are *beliefs* in, respectively:

- The predator is paying attention to the prey's gaze (so that it will notice it when the prey calls for help of other animals)
  predator_is_noticing_preys_gaze
- The predator is paying attention to the prey's gesture (so that it will notice it when the prey shows aggressive behaviour)
  predator_is_noticing_preys_gesture
- The predator is at a reasonable distance away (so that it is able to run away without being caught)
  predator_is_reasonable_distance_away

Moreover, the intentions of the prey can lead to the corresponding actions when the following *beliefs* of the prey in *opportunities* are there:

- Other animals are around (so that it is possible to call for their help)
  other_animals_around
- The predator is about to attack (so that it is possible to show aggressive behaviour)
  predator_about_to_attack
- No obstacle is blocking the escape route of the prey (so that it is possible to run away)
  no_obstacle

In addition to the generic BDI-model shown before, the following specific relations were used to model the case study:

```
belief(prey(depends_on(performs(predator, kill(prey)), intention(predator,
        kill(prey)))))
belief(prey(depends_on(performs(predator, kill(prey)), not(belief(predator,
        prey_runs_fast)))))
belief(prey(depends_on(intention(predator, kill(prey)), desire(predator,
        eat(prey)))))
belief(prey(depends_on(intention(predator, kill(prey)), not(belief(predator,
        prey_shows_aggressive_behaviour)))))
belief(prey(depends_on(desire(predator, eat(prey)), not(belief(predator,
        prey_surrounded_by_other_animals)))))
belief(prey(depends_on(belief(predator, prey_surrounded_by_other_animals),
        observes(predator, prey_surrounded_by_other_animals))))
belief(prey(depends_on(belief(predator, prey_shows_aggressive_behaviour),
        observes(predator, prey_shows_aggressive_behaviour))))
belief(prey(depends_on(belief(predator, prey_runs_fast), observes(predator,
        prey_runs_fast))))

has_reason_for(prey, observes(predator, prey_surrounded_by_other_ani-
        mals), predator_is_noticing_preys_gaze, call_for_help)
has_reason_for(prey, observes(predator, prey_shows_aggressive_behaviour),
        predator_is_noticing_preys_gesture, show_aggressive_behaviour)
has_reason_for(prey, observes(predator, prey_runs_fast),
        predator_is_reasonable_distance_away, run_fast)

is_opportunity_for(prey, other_animals_around, call_for_help)
is_opportunity_for(prey, predator_about_to_attack,
        show_aggressive_behaviour)
is_opportunity_for(prey, no_obstacle, run_fast)

has_effect(call_for_help, prey_surrounded_by_other_animals)
has_effect(show_aggressive_behaviour, prey_shows_aggressive_behaviour)
has_effect(run_fast, prey_runs_fast)
```

By combining these relations with the generic LEADSTO rules provided in the previous section, a complete executable LEADSTO specification for the two-level BDI-model has been created. This simulation model is shown in the appendix at http://www.cs.vu.nl/~tbosse/tom/ICCM.pdf.

## Simulation Experiments

In simulation experiments, the two-level BDI-model has been applied to the case study as described above. To this end, the LEADSTO software environment (Bosse, Jonker, Meij, and Treur, 2007) has been used. In Figure 3 and 4, examples of resulting simulation traces are shown. In these figures, time is on the horizontal axis; the state properties are on the vertical axis. The dark boxes indicate that a state property is true. Note that, due to space limitations, only a selection of the relevant atoms is shown.

Figure 3 is the resulting simulation trace of the situation in which *no* Theory of Mind is involved, i.e., only the behaviour of the predator is addressed, without manipulation by the prey. The trace depicts that the predator initially receives some inputs (e.g., indicated by the state property

observes(predator, not(prey_surrounded_by_other_animals))

at time point 1).

As a result, the predator has made some beliefs (e.g., the state property

belief(predator, not(prey_surrounded_by_other_animals))

at time point 2), which persists for a longer time. Due to this belief, it generates the desire to eat the prey at time point 3

desire(predator, eat(prey))

Based on this desire and the belief

belief(predator, not(prey_shows_aggressive_behaviour))

the predator generates the intention to kill the prey at time point 4:

intention(predator, kill(prey))

Based on this intention and the belief

belief(predator, not(prey_runs_fast))

the predator eventually performs the action of killing the prey at time point 5.

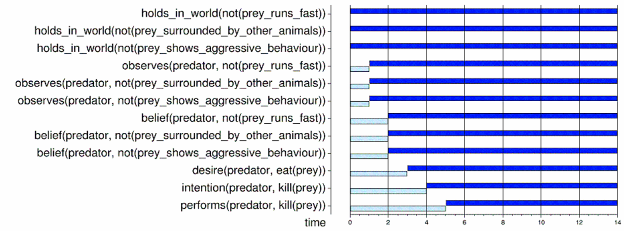


Figure 3: Simulation trace of the predator's behaviour

Figure 4 is the resulting simulation trace of the extended case study, in which the prey agent can act in an anticipatory manner to avoid the predator's desire to eat the prey, and intention and/or action to kill it. Figure 4 shows, among others, that the prey initially desires that the predator does not perform the action to kill it:

desire(prey, not(performs(predator, kill(prey))))

Based on this, the prey eventually generates a number of more detailed desires about what the predator should observe (see, for example, the state property

desire(prey, observes(predator, prey_shows_aggressive_behaviour))

at time point 3). Next, the prey uses these desires to generate some intentions to fulfill these desires (e.g., the state property

intention(prey, show_aggressive_behaviour)

at time point 4). Eventually, when the opportunities are there, these intentions are performed, and the predator observes some new inputs (e.g., the state property

observes(predator, prey_shows_aggressive_behaviour)

at time point 8). As a result, the predator eventually does not generate the action to kill the prey.

Note that in the scenario sketched in Figure 4, the prey takes all possible actions (within the given conceptualization) to fulfill its desires. This is a rather extreme case, since according to the prey's BDI-model, modifying only one of the predator's inputs will be sufficient to make sure that it does not kill the prey. Other traces can be generated in which the prey takes fewer actions to fulfill its desires.

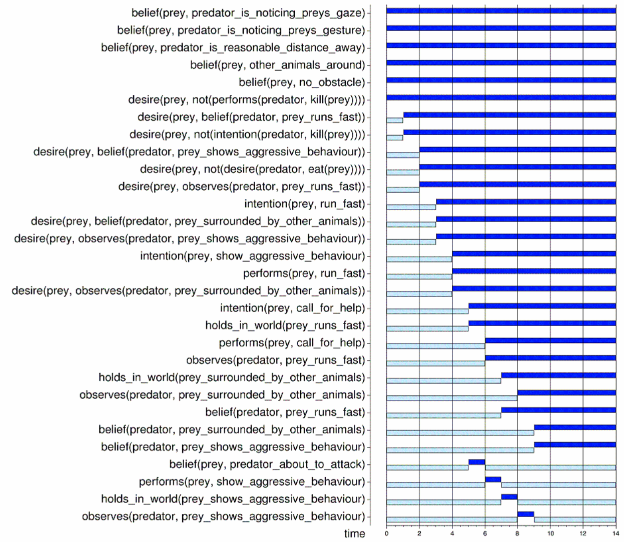


Figure 4: Simulation trace of the prey's manipulation of the predator's behaviour.

## Discussion

In order to function well in interaction with other agents, it is very helpful for an agent to have capabilities to predict in which circumstances the agents in its environment will show certain behaviours. To this end, such an agent will have to perform interpretation based on a Theory of Mind (Baron-Cohen, 1995). This type of metacognition is studied in the context of human social interaction, but also in the area of animal behaviour it is addressed; e.g., (Barrett and Henzi, 2005; Bogdan, 1997; Heyes, 1998). In this paper the latter area is addressed. A model for Theory of Mind is applied, which makes use of BDI-concepts at two different levels. First, the model uses BDI-concepts *within* the Theory of Mind (i.e., it makes use of beliefs, desires and intentions to describe the cognitive process of another agent). Second, it uses BDI-concepts for *interpretation of* the Theory of Mind (i.e., it makes use of beliefs, desires and intentions to describe an agent's meta-cognition about the cognitive process of another agent). At this second level, meta-statements are involved, such as 'B believes that A desires d' or 'B desires that A does not intend a'. These meta-statements are about the states occurring within the other agent. In addition, meta-statements are involved about the dynamics occurring within the other agents. An example of such a (more complex) meta-statement is 'B believes that, if A performs a, then earlier he or she intended a'.

The two-level BDI-based model as presented can be exploited both in order to be prepared for the behaviour of another agent, and in order to affect the behaviour of another agent at forehand. The model has been formalised using the modelling language LEADSTO, which describes dynamics in terms of direct temporal dependencies between state properties in successive states. The model not only addresses analysis of the other agent's beliefs, desires and intentions, but also integrates this with the agent's own beliefs, desires and intentions, and actions.

Obviously, empirical validation of the model is a difficult issue. At least, the present paper has indicated that it is possible to apply computational models for Theory of Mind to animal behaviour. Moreover, the model indeed shows the anticipatory behaviour of higher animals as described in literature such as (Bogdan, 1997). In this sense the model has been validated positively. However, notice that this is a relative validation, only with respect to the literature that forms the basis of the model. In cases that the available knowledge about the functioning of such animals is improving, the model can be improved accordingly. In this sense the approach anticipates further development.

Concerning related work, there is a large body of literature on Theory of Mind in non-human primates (e.g., Barrett and Henzi, 2005; Heyes, 1998), in particular in chimpanzees (Matsuzawa, Tomonaga, and Tanaka, 2006) and macaques (Sinha, 2003). This literature illustrates that non-human primates use Theories of Mind about other primates while interacting socially with them in specific types of behaviour like imitation, social relationships, deception, and role-taking. Moreover, recent literature suggests that dogs use a certain kind of Theory of Mind as well (e.g., Horowitz, 2002; Virányi, Topál, Miklósi, and Csányi, 2006). However, none of these papers contains a computational model of Theory of Mind in non-human primates. In contrast, the current paper presents such a model, and illustrates how it can be applied to simulate the behaviour of a prey animal that tries to manipulate the attacking behaviour of a predator. Moreover, a number of other papers propose computational models of Theory of Mind (e.g., Gmytrasiewicz and Durfee, 1995; Marsella, Pynadath, and Read, 2004), but these are not applied explicitly to animal behaviour. For an extensive comparison of our approach to these models, the reader is referred to (Bosse, Memon, and Treur, 2007).

For future research, it is planned to exploit the features of the LEADSTO language for modelling more quantitative, numerical concepts. For example, the possibility to add probabilities to the simulation rules will be explored. In addition, more precise values can be chosen for the timing parameters e, f, g, h mentioned in the introduction. Doing this also makes it possible to make a better comparison between the traces shown in Figure 3 and 4. Currently, the trace in Figure 4 does not contain the first three world states shown in Figure 3. If these were present, the predator would kill the prey before the prey had the chance to manipulate it. By allowing different timing parameters, this problem could be solved. In addition, being able to experiment with the timing parameters would allow the modeller to make the model more realistic.

## References

Aristotle (350 BCa). Nicomachean Ethics (translated by W.D. Ross).

Aristotle (350 BCb). De Motu Animalium On the Motion of Animals (translated by A. S. L. Farquharson) .

Baron-Cohen, S. (1995). Mindblindness. MIT Press.

Barrett, L. and Henzi, P. (2005). The social nature of primate cognition. *Proceedings of The Royal Society of Biological Sciences*, vol. 272, pp. 1865-1875.

Bogdan, R.J. (1997). Interpreting Minds. MIT Press.

Bosse, T., Jonker, C.M., Meij, L. van der, and Treur, J. (2007). A Language and Environment for Analysis of Dynamics by Simulation. *Int. Journal of Artificial Intelligence Tools.* To appear, 2007. Earlier version in: Eymann, T. et al. (eds.), Proc. of the 3rd German Conf. on Multi-Agent System Technologies, MATES'05. Springer LNAI, vol. 3550, pp. 165-178.

Bosse, T., Memon, Z.A., and Treur, J. (2007). A Two-Level BDI-Agent Model for Theory of Mind and its Use in Social Manipulation. In: *Proceedings of the AISB 2007 Workshop on Mindful Environments*, pp 335-342.

Dennett, D.C. (1987). The Intentional Stance. MIT Press. Cambridge Mass.

Dennett, D.C. (1991). Real Patterns. The Journal of Philosophy, vol. 88, pp. 27-51.

Georgeff, M. P., and Lansky, A. L. (1987). Reactive Reasoning and Planning. In: Forbus, K. and Shrobe, H. (eds.), Proceedings of the Sixth National Conference on Artificial Intelligence, AAAI'87. Menlo Park, California. American Association for Artificial Intelligence, 1987, pp. 677-682.

Gmytrasiewicz P. J., and Durfee. E. H. (1995). A rigorous, operational formalization of recursive modeling. In: Lesser, V. (ed.), *Proceedings of the First International Conference on Multiagent Systems*, pp. 125-132.

Heyes, C.M. (1998). Theory of mind in nonhuman primates. *Behavioural and Brain Sciences*, vol. 21, pp. 101-134.

Horowitz, A. (2002). The behaviors of theories of mind, and a case study of dogs at play. PhD. Thesis, University of California, 2002.

Jonker, C.M., Treur, J., and Wijngaards, W.C.A., (2003). A Temporal Modelling Environment for Internally Grounded Beliefs, Desires and Intentions. Cognitive Systems Research Journal, vol. 4(3), 2003, pp. 191-210.

Malle, B.F., Moses, L.J., Baldwin, D.A. (2001). Intentions and Intentionality: Foundations of Social Cognition. MIT Press.

Marsella, S.C., Pynadath, D.V., and Read, S.J. (2004). PsychSim: Agent-based modeling of social interaction and influence. In: Lovett, M., Schunn, C.D., Lebiere, C., and Munro, P. (eds.), *Proc. of the Sixth Int. Conference on Cognitive Modeling, ICCM 2004*, pp. 243-248 Pittsburg, Pensylvania, USA.

Matsuzawa, T., Tomonaga, M., and Tanaka, M. (2006). Cognitive Development in Chimpanzees. Springer Verlag, Tokyo, 2006.

Rao, A.S. and Georgeff, M.P. (1995) BDI-agents: from theory to practice. In: Lesser, V. (ed.), Proceedings of the International Conference on Multiagent Systems, pp. 312 – 319.

Rao, A.S. and Georgeff, M.P. (1991). Modelling Rational Agents within a BDI-architecture. In: Allen, J., Fikes, R. and Sandewall, E. (eds.), Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning, (KR'91). Morgan Kaufmann, pp. 473-484.

Sinha, A. (2003). A beautiful mind: Attribution and intentionality in wild bonnet macaques. *Current Science*, vol. 85. no. 7, 2003.

Virányi, Zs., Topál, J., Miklósi, Á, and Csányi, V. (2006). A nonverbal test of knowledge attribution: a comparative study of dogs and children. *Animal Cognition*, vol. 9, no. 1, pp. 13-26.

# From a Formal Cognitive Task Model to an Implemented ACT-R Model

**Fiemke Both[1] (fboth@few.vu.nl)**
**Annerieke Heuvelink[1,2] (heuvel@few.vu.nl)**
[1]Vrije Universiteit Amsterdam, Department of Artificial Intelligence,
De Boelelaan 1081a, 1081 HV Amsterdam, the Netherlands
[2]TNO Defence, Security and Safety, Department of Training and Instruction
Kampweg 5, 3769 DE, Soesterberg, the Netherlands

### Abstract

In order to generate behavior that can be validated, a cognitive task model needs to be implemented in software. This paper first introduces a cognitive task model based on a BDI framework and then focuses on the translation of that model into the ACT-R theory and software. Problems encountered during this translation process are described and further implications are discussed. It is found that the model's control structure matches well with ACT-R, but that the translation of its reasoning rules is complex. ACT-R's theory of cognition does not match with properties of our task model on three issues: 1) the number of memory items that can be stored in working memory, 2) the way memory items are retrieved from long-term memory, 3) the ability to execute complex computations.

## Introduction

The development and implementation of cognitive task models in order to create agents that can replace humans for performing certain tasks in certain environments is a promising research activity (Gluck & Pew, 2005). Two important activities in the development of a cognitive model are the modeling of relevant task knowledge and the modeling of a cognitive valid theory of task execution. These modeling activities typically yield a formal design on paper of the cognitive task model. However, for studying the model's behavior and, to use the model as a replacement for a human in a certain environment it needs to be implemented in software.

The work presented in this paper is part of a greater research project that has as its goal the development of a cognitive agent that can perform the task of compiling a tactical picture on board of a ship in a training simulation. The current paper describes and reflects on the translation of the cognitive task model into the cognitive architecture ACT-R (Anderson et al, 1998). It will not go into details of the task, or into the validity of the behavior that the resulting cognitive agent shows.

First, we will briefly introduce the developed cognitive task model with its main properties concerning task knowledge and task control. Next, we discuss the translation of this model into ACT-R. To test the resulting cognitive agent we coupled it to an external simulation environment. This coupling and the general task performance of the ACT-R model are discussed. Furthermore, we reflect upon problems encountered during the translation as well as their implications. Finally, we lay down further research plans.

## Research Domain

The task we modeled is the Tactical Picture Compilation Task (TPCT) within the domain of naval warfare. This task revolves around the classification and identification of entities in the surroundings. The warfare officer responsible for the TPCT monitors the radar screen for radar contacts and reasons with the available information in order to determine the type and intent of the contacts on the screen. The cognitive model of the TPCT is based on a Belief, Desire and Intention (BDI) framework (Georgeff and Lansky 1987). This choice facilitates the translation of domain knowledge into the model since domain experts tend to talk about their knowledge in terms similar to beliefs, desires and intentions. The domain knowledge needed for performing the TPCT has been elicited from naval experts by van Dam and Arciszewski (2002).

The goal of our research is the development of cognitive agents that can be used for training purposes. We therefore want to model cognitive behavior, which can vary in level of rationality. To make this possible, we developed a specific belief framework (Heuvelink, 2006). Three arguments are added to beliefs: a timestamp, the source of the information and a certainty level. Usually in BDI models, beliefs are thrown away as soon as a new belief is created that causes an inconsistency. However, because we want to reason over time, every belief is labeled with the time it is created and is never thrown away. The source and certainty labels make it possible to model an agent that can reason about information from multiple sources and with uncertainty, and that might do this in a biased way. A belief according to this new framework consists of a predicate P with an attribute A and value V, a timestamp T, source S and certainty C. An example belief is shown below.

Belief (Identification (contact1, friendly), 12, determine-identification, 0.7)

## Cognitive Task Model

A cognitive task model typically consists of declarative knowledge, denoting facts, as well as procedural knowledge, denoting reasoning rules. Besides modeling how to reason, it is also necessary to model when to reason about what. In this paper, we mainly address the modeling of the declarative and procedural knowledge necessary for performing the TPCT. The modeling of cognitive control is not our current focus, and therefore we limit the complexity of our control structure. First, we describe the format of the declarative and procedural knowledge embedded in the cognitive model. Then we elaborate on the control structure of the model and at the end, we present the conceptual design of our agent.

## Reasoning over Beliefs

The goal of the TPCT is to correctly classify and identify all contacts in the surroundings. To draw these kind of conclusions about contacts, the agent needs knowledge about their behavior. There are two ways to gather such information. The first is from the external world, e.g., the agent can watch the screen that displays information from sensors such as the radar system. Additionally, the agent can decide to perform actions that lead to more knowledge about the situation, such as activating radar or sending a helicopter to investigate a contact. The second method to gain information is through the internal process of reasoning about beliefs to deduce new beliefs. In the reasoning process, often multiple beliefs form the evidence for the formation of a new belief. Any uncertainty in the source beliefs will be transferred to the new belief.

The following rule is an example of how a new belief is derived using other beliefs and domain knowledge. The position of the contact that the agent is currently reasoning about is compared to the position of every other contact that is detected by the radar system. A new belief is created for every pair that indicates how certain the agent is that they are within formation distance. Names of functions are depicted bold and names of parameters are italic.

```
Determine-Within-Formation-Distance-Contact(X)
For all Y
  If (
    belief(Position-Contact(X, P1),T1, S, C1)
    belief(Position-Contact (Y, P2), R1, S, C2)
    Position-Difference(P1, P2, D)
    Certainty-Handling- Difference-Between-Positions(C1, C2, D,
    C3)
    Possible-Distances(D,C3, [R])
    M = maximum-distance-relevant-for-formation
    C4 =  (number-of-[R <= M]) / (number-of-[R]) )
  Then (
    Reason-Belief-Parameter(Within-Formation-Distance- Contact
    (X, Y), determine-within-formation-distance-contact, C4) )
```

The function Position-Difference calculates the distance between two positions, Certainty-Handling-Difference-Between-Positions calculates the certainty of the distance given the certainties of the positions, Possible-Distances returns all possible distances given the calculated distance and certainty, and the rule Reason-Belief-Parameter adds the timestamp and stores the belief in long-term memory. In this example, the latest beliefs about the positions are used. In other rules, beliefs are used that ever had a specific value, or those beliefs the agent is most certain about.

## Control of Reasoning

Control is an important aspect of a cognitive agent; it determines when the agent does what. The TPCT has one main goal, which is considered a desire in the BDI model: to identify all contacts correctly. The three subtasks that the agent can perform in order to fulfill this desire are 1) processing information about contacts from the screen, 2) changing the activity of the radar system, and 3) sending the helicopter on observation missions to gain more information about a specific contact. These subtasks are the intentions of the BDI model that the agent can commit to.

A cognitive valid manner to determine when which intention becomes a commitment is to have events in the world trigger an intention. For example, when a contact suddenly changes its behavior, the attention of the agent should be drawn to this contact, regardless of the current intention. However, this type of control requires a parallel processing of all events in the world and a parallel checking of relevancy for all subtasks, which is very difficult to implement. That is why currently, we chose to implement a simpler, linear control system. To simulate parallel processing we let the agent alternately commit to one of the three intentions. Within the subtasks, the control is also kept simple, e.g., in the first subtask all contacts on the screen that are stored in a random list are monitored consecutively.

The following reasoning rule is an example part of the simple control structure. It determines when the agent starts committing to a new intention.

```
Determine-New-Intention(I)
  If (
    I = Monitor-Contacts
    belief(Number-of-Contacts-Monitored(X), T1, S, C)
    X = maximum-number-of-contacts-to-monitor )
  Then (
    Start-New-Intention-Selection(I) )
  Else if (
    I = Monitor-Contacts
    Number-of-Contacts(X)
    X < maximum-number-of-contacts-to-monitor )
  Then (
    Select-Next-Contact-To-Monitor()
    Reason-Belief-Parameter(Number-of-Contacts-Monitored(X+1),
    determine-new-intention, 1) )
```

The input parameter *I* is the current intention, the rule Start-New-Intention-Selection determines which intention is selected next depending on beliefs about contacts, and the rule Select-Next-Contact-To-Monitor selects the next contact from the list.

## Conceptual Agent Design

We have made a conceptual design of the agent capable of performing the TPCT using the DESIRE (DEsign and Specification of Interacting REasoning components, Brazier, Jonker and Treur, 2002) method. DESIRE's view of an agent is that of a composed structure consisting of interacting components. This conceptual agent model in DESIRE completes the formal model of the TPCT with the control of the different subtasks. DESIRE enables us to model the flow of information within the agent and between the agent and the external world. The agent components of the conceptual model are displayed in Figure 1, representing different kinds of tasks at different levels.

At the top level, there are two components: Warfare Officer and External World. This makes it possible to model communication between the agent and the external world. At the second level, three components are selected from the Generic Agent Model (GAM, Brazier, Jonker & Treur, 2000): (1) Own Process Control, (2) Maintenance of World Information and (3) World Interaction Management. The component Own Process Control, responsible for desire, intention and belief determination, is refined according to the addition to GAM for BDI models (Brazier et al, 2001). The component Maintenance of World Information stores the beliefs created by a subcomponent of Own Process Control. The last GAM component, World Interaction Management, manages communication with the external world via two subcomponents, Action and Observation Request Management. The fourth component at the second level, Agent Specific Task, performs tasks that require an internal decision to be made (Decision Control), and tasks that require reasoning about beliefs to derive new beliefs (Reasoner).

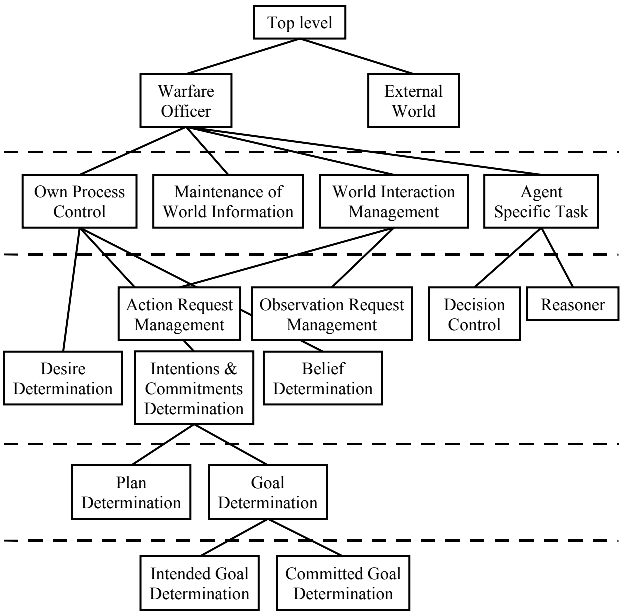

Top level
Warfare Officer
External World
Own Process Control
Maintenance of World Information
World Interaction Management
Agent Specific Task
Action Request Management
Observation Request Management
Decision Control
Reasoner
Desire Determination
Intentions & Commitments Determination
Belief Determination
Plan Determination
Goal Determination
Intended Goal Determination
Committed Goal Determination

Figure 1 Process decomposition for the agent

## Translation Process

This section first introduces ACT-R and than continues with a description of the translation process of the reasoning rules and the control of these rules into the ACT-R architecture. Finally, it introduces the environment with which the resulting agent interacts.

### ACT-R

ACT-R incorporates two types of memory modules: declarative memory and procedural memory. Declarative memory is the part of human memory that can store items; procedural memory is the long-term memory of skills and procedures. ACT-R consists of a central processing system, where the *production rules* representing procedural memory are stored and executed. The central processing system can communicate with several modules through buffers. One of those modules is the *declarative memory module* where memory items are stored. These memory items, called *chunks*, are of a specific chunk-type, which can be defined by the modeler. In a chunk-type definition, the modeler defines a number of slots that chunks of this type can assign values to. Chunks from the declarative memory module can be placed in the *retrieval buffer* if they match a retrieval request made by a production rule. A retrieval request must contain the requested chunk-type, and may contain one or more slot-value pairs that the chunk must match. The matching chunk is then placed in the retrieval buffer, so it can be read by a production rule. All buffers in ACT-R, including the retrieval buffer, can only store one chunk at a time, even when more chunks match the conditions of the request.

### Reasoning Rules

In the following paragraphs, the implementation of the reasoning rules of the formal task model in the declarative and procedural memory modules is described.

**Declarative Memory** The cognitive model of the TPCT requires the ability to retrieve beliefs with specific time and certainty constraints. The rule Determine-Within-Formation-Distance-Contact for example, requires the latest beliefs about the position of two contacts. Because beliefs are not thrown away, there may be many older beliefs about the positions of these contacts. It is therefore necessary to retrieve the latest belief of both contacts in order to determine the current distance between them. In ACT-R, creation and retrieval times of chunks are registered, but these properties are not available to the modeler. There is also no feature that enables the agent to retrieve a chunk with the highest value of a slot.

The method ACT-R provides for retrieving chunks from memory is an activation function for chunks. The activation of a chunk indicates how easy it should be to retrieve it. Every chunk that matches a retrieval request receives an activation score based on three components: a *base level activation,* a *context component* and a *noise value.* The base level activation uses the number of representations and the time since the representations. Representations are the initial entry in the declarative module and a retrieval in the declarative buffer. The context component is based on the activation of related chunks. The noise value adds a random factor to the activation value.

Our task model assumes that when the last belief is required for a reasoning process, that the last belief is always retrieved. In ACT-R, it is possible that an older chunk has been retrieved more often than the latest chunk, which results in a higher activation value. Although there are several parameters that can be set, the activation function cannot guarantee that the latest chunk is always the most active. To meet the requirements of the task model, it is therefore necessary to implement the retrieval process of those beliefs using LISP, the language ACT-R is built in.

**Procedural Memory** There are several theories about working memory (WM), most of which agree on the idea that multiple memory items can be stored at the same time in WM (Miller, 1956; Baddeley & Hitch, 1974; Hulme et al, 1995; Cowan, 2005). The number of items that can be stored in WM ranges from two to seven in these theories. ACT-R's theory of WM is that there can only be one chunk of memory stored in WM at a time. This representation of WM is not consistent with the most commonly accepted cognitive theories of WM, as well as with the formal task model based on the developed belief framework.

In the formal task model, several beliefs need to be in WM at the same time to be able to reason. For instance, different positions over time are compared in order to determine a contact's heading and speed. To implement such a rule in the ACT-R model, multiple production rules are needed to retrieve all required beliefs. Each production rule would have to draw a sub conclusion about the retrieved information so far. This method is inefficient for this task and it is inconsistent with the reasoning methods described by the naval experts. Therefore, we have tried two different methods to solve this problem. The first solution is to merge all the beliefs that need to be compared into one chunk. This solution however, still requires many production rules to retrieve all the necessary beliefs and undermines the ACT-R theory of WM being able to store only one chunk at a time.

The second method, used in the final model, consists of two parts: using the goal buffer for temporary storage and using LISP functions to retrieve beliefs. The *goal buffer* can hold just one chunk, so the different memory items need to be stored in the slots of the goal chunk. Many ACT-R modelers use this solution to be able to compare information items (see e.g., the tutorials of ACT-R 6.0). For example, in our task model several beliefs about the location of a contact at different time points form the antecedent of the logical rule that calculates the belief about the speed of a contact. These different locations can be stored in the goal buffer until all relevant locations are retrieved and the speed can be calculated. The second part of our solution is that we used LISP functions instead of production rules to access chunks in the declarative module of ACT-R. LISP functions can be called within a production rule that is firing. Then, the LISP function can retrieve multiple beliefs at the same time, compare them, and return the result to the active production rule.

The many calculations the task model requires, e.g., for calculating the certainty of a new belief from the uncertainties of the beliefs it is derived from, is a second problem we encountered during translation. In most ACT-R models of low-level tasks, calculations are modeled in production rules. The addition of three and four for example, would take ten rules: one for the initialization, two for every addition step (increment by one, remember how much has been added) and one for the finalization. It would take many more production rules to calculate the speed of a contact from its positions over time. In addition, most of the calculations are not an exact representation of the cognitive processes of a warfare officer, but are a more abstract representation. It is therefore not necessary and not efficient to model all computations in production rules. We chose to use LISP functions for those calculations.

By executing the above processes in LISP functions instead of in production rules, the amount of production rules is reduced to half of the programming code of the agent. The other half consists of supporting LISP functions. The following ACT-R production rule is an example of how we implemented the task model rule Determine-Within-Formation-Distance-Contact and how we solved the problems described above.

```
(p determine-within-formation-dist-goal1-plan6
   =goal>
      ISA        commitment1
      goal       monitor-contacts
      state      determine-formation-distance
      contact    =contact
==>
   !bind! =within-dist (calc-within-formation-distance =contact)
   !bind! =next-step (if =within-dist 'determine-formation 'determine-
   classification’)
   =goal>
      state      =next-step
      result     =within-dist )
```

The antecedent of the rule requires that the agent is currently committed to commitment1; processing information about contacts from the screen, and that the agent is going to determine whether the current contact is within formation distance of any other contact. In the consequent, the agent uses a LISP function to find all contacts within formation distance (function calc-within-formation-distance). In this LISP function, multiple beliefs with the latest timestamp are retrieved, and calculations are performed to determine the distance. If a contact exists that is close enough to the current contact, the agent starts with determining whether they actually are moving in formation. Otherwise, the next step is to determine the classification of the current contact.

## Control of Reasoning Rules

As explained above we implemented a simple, linear control structure, in which the agent alternately commits to its various subtasks. We showed the reasoning rule Determine-New-Intention, which determines that after reasoning about a number of contacts one of the other subtasks becomes the agent's intention. We did not run into any problems when implementing these task model control rules in ACT-R. In ACT-R, the control structure is also linear: only one production rule can fire at a time. The antecedent determines which rule matches the current contents of the buffers. The current intention and current step in the plan can be stored in the goal buffer. The consequence of a fired rule can influence which production rule will fire next. In our agent, this was generally done by changing the contents of the goal buffer. The following ACT-R production rule shows how we implemented the rule Determine-New-Intention.

```
(p select-next-contact-goal1
   =goal>
      ISA               commitment1
      goal              monitor-contacts
      state             next-contact
      contact           =contact1
==>
   !bind! =contact2 (determine-next-contact)
   !bind! =new-intention (= (mod *contact-counter* *max-number-of-
   contacts*) 0)
    =goal>
      Plan              read-basal-info
      state             start-step
      contact           =contact2
      new-intention     =new-intention )
```

The antecedent of the rule requires that the agent is currently committed to monitoring contacts, and that the agent is going to select a new contact. In the consequent, the agent determines whether to select the next contact or to start a new intention. The slot new-intention of the goal chunk can be true or false, influencing the next production rule that can fire.

### Coupling Environment

ACT-R has different modules for simulating visual and aural stimuli and vocal and manual actions. The visual module can be used to show letters and numbers in a window, for example to simulate the Stroop-task. However, the current task requires the visualization of a naval scenario and the features of the visual module are too limited for this. Therefore, a coupling has been made between ACT-R and an external simulation environment: Game Maker (see Game Maker). The simulation environment we developed in Game Maker consists of a screen that shows radar contacts. An additional screen displays detailed information about a contact when it is selected by a mouse click. This same window can be used to change the classification and identification value of the selected contact.

ACT-R and Game Maker communicate through two text files. In the first file, the ACT-R agent writes information and action requests, which Game Maker reads and performs. This can be, for instance, a request for detailed information, simulating a mouse click on a contact. In the other file, Game Maker writes the requested information and feedback about the performed action, which ACT-R reads and processes. The response of Game Maker to the mouse click would be to write the detailed information of the contact in the text file.

## Results and Discussion

Now that we have implemented the TPCT model in ACT-R and coupled the resulting agent with a simulation environment, we can test the behavior of the agent. In this section, we will describe and discuss the results of the translation process and the coupling with Game Maker.

Before translating the task model into ACT-R, we developed a conceptual agent model in DESIRE that gave us a structured overview of the control and information flows within the task model. We find that the structure of this conceptual agent model matches well with ACT-R. Each of the components of the conceptual model (see Figure 1) can be identified in the ACT-R code. This made the conceptual model a useful structure to base the ACT-R model on.

In addition, the serial control was easily translated from the DESIRE model to ACT-R production rules. In ACT-R, one production rule can fire at a time. This principle is the same as we chose for our agent. However, if we had chosen a more complex control, we probably would have had more trouble translating the conceptual model to ACT-R. For example, a function that determines which contact on the screen deserves attention requires calculations. Since we have already shown that calculations are difficult to model in ACT-R, a more complex control system will be more difficult to implement.

During the implementation process, we encountered some problems when translating the reasoning rules of our model to ACT-R production rules. The ACT-R theory entails that one chunk can be stored in WM, that the chunk with the highest activation value is retrieved, and that it is difficult to combine low-level calculations and high-level reasoning rules in one model. Our solution has been to implement those processes in LISP functions. However, by using so much LISP code and by not using the declarative memory module properly, the structures built into ACT-R that constitutes its theory of cognition are denied. It is likely that both theories are incomplete. ACT-R should support the storage of multiple chunks in the retrieval buffer at the same time since most researchers agree that human WM can store multiple beliefs. We should reconsider the many calculations performed in the rules of the cognitive task model.

A more practical issue we encountered during implementation concerns the number of chunks in ACT-R's memory. Every couple of minutes the agent is active, its number of chunks doubles. As a result, ACT-R becomes very slow when the agent tries to retrieve a chunk from memory. We partly solved this by creating fewer chunks during reasoning. In the original task model, every reasoning step resulted in a new belief. We identified types of beliefs that were never retrieved by the agent, and stopped adding them to ACT-R's memory. For example, only the result of a calculation is remembered, instead of all steps leading to this result. Furthermore, we deleted irrelevant chunks from the agent's memory. For instance, if the speed of a contact has been constant for a long time, the first and last beliefs about that speed provide all the information ever needed.

The coupling of the ACT-R agent and the simulation environment Game Maker consists of communication through text files; one file in which ACT-R writes its requests and. one file in which Game Maker writes its results. It is computationally impossible for both Game Maker and ACT-R to check the text file constantly for new information. The less often the text file is checked for new information, the slower the communication process becomes. However, if Game Maker and the agent would check it more often, the entire simulation would slow down.

We tried to find a balance by having Game Maker read the text file every second. The agent only reads the text file when it expects new information. Thus, when the agent communicates a request to Game Maker, it keeps reading the text file until the new information has arrived. In practice, this means that ACT-R reads the text file approximately every half second. In the future, we would prefer a different type of coupling that enables streaming of information, so the two parties do not have to actively check for new information.

## Conclusion and Future Research

We implemented a cognitive task model in ACT-R and tested how it functioned. During implementation, we found that the DESIRE control model fits well with ACT-R, but that the model's reasoning rules that are based on the developed belief framework do not fit.

ACT-R is a cognitive architecture that consists of several components. Two of those components are used for the agent: the declarative memory module and the procedural memory system. The communication with the chunks in the declarative memory module is not done by the use of the retrieval buffer, but through LISP functions. LISP functions are used to solve three issues. First, the task model requires the comparison of more than one belief, and ACT-R can only hold one belief in the representation of WM. Second, the task model requires the latest or most certain belief to be retrieved, and ACT-R does not offer a function to request beliefs with these specifics. Third, the task model describes many calculations, which are difficult to implement using ACT-R's production rules. Because only part of the theory of ACT-R matches the formal cognitive task model, this cognitive architecture is not very well suited for this kind of task model and this forced us to implement a great part of the agent in LISP code.

Overall, the implemented agent is slow. ACT-R is a software package that uses about half of the computer's CPU power, and Game Maker uses the other half. In ACT-R the slow speed is mainly due to the extensive search for specific beliefs caused by the exponential grow of chunks. To improve the performance of the agent it is necessary to extend the task model with a cognitive model for the decay of beliefs, or with a system that throws away beliefs that are not relevant for the task anymore.

In the future, we want to translate the cognitive task model to SOAR to research how well it matches with that cognitive architecture. This activity will yield a SOAR agent whose performance we can compare with the current ACT-R agent. To increase the cognitive validity of the task model we also want to focus on the development of a more cognitive plausible control system.

## Acknowledgments

The authors like to thank Tibor Bosse and Karel van der Bosch for their guidance during this research project. Furthermore we like to thank Jan Treur for commenting an earlier draft, Willem van Doesburg for developing an initial version of the training environment, and the naval instructors of the Opschool for providing domain knowledge and for taking part in the experiment.

## References

Anderson, J. R., & Lebiere, C. (1998). The Atomic Components of Thought. Mahwah, NJ: Erlbaum.

Baddeley, A.D. & Hitch, G.J. (1974). Working Memory, In G.A. Bower (Ed.), *Recent advances in learning and motivation (Vol. 8, pp. 47-90)*, New York: Academic Press.

Brazier, F.M.T., Dunin-Keplicz, B., Treur, J. & Verbrugge, R. (2001). Modelling Internal Dynamic Behaviour of BDI Agents. In Gabbay, D., and Smets, Ph. (eds.), *Dynamics and Management of Reasoning Processes.* Series in Defeasible Reasoning and Uncertainty Management Systems, 6. Kluwer Academic Publishers, 2001, 339-361.

Brazier, F.M.T., Jonker, C.M. & Treur, J. (2000) Compositional Design and Reuse of a Generic Agent Model. *Applied Artificial Intelligence Journal, 14.* 491-538.

Brazier, F.M.T., Jonker, C.M., & Treur, J., (2002). Principles of Component-Based Design of Intelligent Agents. *Data and Knowledge Engineering, 41*, 1-28.

Cowan, N. (2005). *Working memory capacity*. New York, NY: Psychology Press.

Dam, B. J. van & Arciszewski, H. F. R. (2002). *Studie Commandovoering DO-2: Beeldvorming* (FEL Technical Report FEL-02-A242). The Hague, the Netherlands, TNO FEL.

Game Maker. http://www.gamemaker.nl/

Georgeff, M.P., & Lansky, A.L. (1987). Reactive Reasoning and Planning. *Proceedings of the Sixth National Conference on Artificial Intelligence*, 677 – 682. Menlo Park, Ca: AAAI Press.

Gluck, K.A., & Pew, R.W. (Eds.) (2005). *Modeling Human Behavior with Integrated Cognitive Architectures: Comparison, Evaluation, and Validation*. Mahwah, NJ: Lawrence Erlbaum Associates.

Heuvelink, A. (2006). Modeling Cognition as Querying a Database of Labeled Beliefs. In D. Fum, F. Del Missier, and A. Stocco (Eds.), *Proceedings of the 7th International Conference on Cognitive Modeling (ICCM 2006)*, 365-366. April 5-8, Trieste - Italy.

Hulme, C., Roodenrys, S., Brown, G., & Mercer, R. (1995). The role of long-term memory mechanisms in memory span. *British Journal of Psychology, 86*, 527-536.

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review, 63*, 81-97.

# A Qualitative GOMS Approach to Evaluating Diagrammatic Interfaces

**B. Chandrasekaran and Tsviatko Yovtchev**
Department of Computer Science and Engineering
The Ohio State University, Columbus, OH 43210 USA

## Abstract

We describe an approach to evaluating diagrammatic schemes intended to support problem solving and decision-making. The methodology is in the GOMS framework in HCI, and is based on recognizing that the use of diagrams is part of a process that can be decomposed into a sequence of steps, each of which may be a Perception on the diagram, Inference, Transformation of the diagram and Visual Search. How well a diagrammatic scheme helps in a task depends on how well the human cognitive architecture can perform the actions in the various steps, and how the steps collectively contribute to performance measures such as time, error rates, and memory stress. We illustrate the approach by using it to analyze the use of some common data presentation displays in the task of discovering interesting relations between variables in a domain. Because of the current lack of quantitative empirical data about the execution of the basic operations by human architecture, the analysis is qualitative, which is nevertheless useful in providing useful insights. It also sets an agenda for empirical research to obtain the quantitative data needed, since the availability of such data would help significantly in evaluating and improving diagrammatic interfaces for decision support.

## Introduction

The distinction between informational and computational equivalence (Larkin and Simon, 1987) of representations is relevant to explain the effectiveness not only of diagrams over text under appropriate conditions, but also of one kind of diagram over another for a specific task. Some diagrammatic schemes are lauded over others as especially well suited for specific tasks. This paper discusses an approach to evaluate interactive diagrammatic interfaces to support problem solving.

A diagrammatic scheme for a problem-solving task consists of specifications for representing information in a diagrammatic form, for meaningful perceptions on the diagram, and for modifying diagrams. When such a scheme is used to assist in problem solving, the problem solver engages in a *series of steps*, each of which may be an act of *perception* on the diagram, *inference* of new information by combining background knowledge and current information, including information given by an earlier act of perception, and *transformation*, i.e., modifying the diagram in some way to facilitate further problem solving. Information obtained by perception and inference will be added to the problem solver's short-term memory (STM), or added to the diagram in some form in a transformation step for pickup in a future perception step. The sequence of steps ends when the problem solver has acquired the information corresponding to the solution of the problem of interest, or gives up for whatever reason. Characterizing the problem solving activity in terms of these acts -- perception, inference, storage in memory, and transformation – arises naturally from a high-level view of the human cognitive architecture as comprised of central cognition, various perception modules and motor components.

Recognizing that diagrams are part of such a problem solving activity – instead of viewing them as stand-alone interfaces with one-way information flow from the interface to the user – is useful when we wish to compare alternative diagrammatic schemes for solving problems of a specified type. In particular, it makes possible the use of GOMS, the well-known analytical and comparative framework (Card, et al, 1983; John and Kieras, 1996). Our goal in this paper is to develop a GOMS-inspired analysis technique that is specialized for interactive problem solving with diagrammatic representations, and to illustrate it by applying into a set of alternative diagrammatic interfaces for a decision support problem in visual data mining. Another point of comparison is the work of Peebles & Cheng (2003). The measure of complexity is the number of perceptual attention shifts during a problem solving activity in comparing two graph representations for an information extraction task. As we'll see, we keep track of a more complex set of measures.

One evaluation measure of a diagrammatic interface might be the length – in total number of steps or time – it takes to solve an average problem or the hardest problem in a class. A variation of this measure might keep a count of each *type of step*, i.e., the numbers of perceptions, inferences and transformations, if the different steps correspond to different kinds of costs. Further distinctions may be made within each of these categories, e.g., some transformations and perceptions may have different costs than others. Another dimension of evaluation is propensity for error. A poorly conceived diagram might result in errors in one or more of the perception or transformation steps, or might overload STM and cause inference errors. Alternative diagrammatic schemes for solving a task may then be multi-criterially compared in terms of such measures.

Decision support systems (DSS) are an important class of applications of diagrammatic representations. There is usually a much greater role for transformations in DSS's than is normally the case in the use of simple diagrams. DSS's may be used in widely disparate circumstances: In one situation the cost of physical interaction may be high, with a corresponding preference for interfaces that do not require many transformations. In another situation, say where a user is multi-tasking, a DSS that entails many interactions would be preferable to one that places high stress on STM. Because of such wide variations in the conditions of use, a simple uni-dimensional measure of their performance, such as total time, is usually not adequate.

We introduce PTIS, a version of GOMS tailored to our needs. We then consider a set of alternate diagrammatic interfaces for a simple problem in data understanding: deciding if there are any interesting correlations between variables in a given domain of interest. The field of data mining focuses on problems of this type, and the interfaces we consider are often proposed as good visual presentations of data. However, the interfaces and the task are used as examples of the methodology, rather than the main subjects of the paper.

## The PTIS Framework

When applying GOMS, operators are chosen that are generic enough to be applicable to the analysis of a variety of task/interface combinations of certain types. Two properties of an operator are especially important and are typically empirically obtained from studies on trained humans: the time it takes to apply the operator, and any error rates associated with application of the operator.

The operators we develop in our GOMS analysis of diagrammatic interfaces are at much higher grain sizes and complexity than the ones that GOMS research usually deals with (such as clicking buttons). For example, a basic operation in our domain is the perception of the best-fit straight line that approximates a cluster of data points. This is a common skill needed in experimental research. A person might need some training in this task, but once trained, he can visualize such a line. A corresponding motor operator is to *draw* such a line on a screen or on paper displaying such a cluster of points. Empirical data for time and error rates for operators at the grain sizes of interest to us are not yet available, so our current analysis is qualitative. However, the qualitative results are still useful in many situations, as we will demonstrate. When empirical data become available, it would be easy to convert the results to quantitative ones.

What is common in the use of all diagrammatic displays for decision support is that the user's actions belong to one of the following four types: Perception, Transformation, Inference, and (Visual) Search. The information obtained by Perception[1] or Inference is automatically placed in STM, i.e., it is not usually treated an operation. Since STM is capacity-limited, analysis should track STM load.

The operations in the Perception and Transformation categories respectively are chosen to represent basic units of the actions of the agent required for the task, but generic enough to be used as operations in a variety of tasks using diagrammatic interfaces. The best way to think of a Perception in this analysis is not as a gestalt perception act whose details the user does not access, but as a step in which the user is acquiring information from the display, and that the step has generality and reusability. Examples in data analysis are: visualizing a straight line that best summarizes a set of data; and visualizing the midpoint of a set of points. Because these Perceptions are general enough to be useful across a variety of data analysis tasks, investing in determining the timing and error rates associated with trained human perception would be worthwhile. (How these are learned would require a separate study.)

The relationship between Search and Perception needs clarification, since some of the Perceptions may also involve search. What we mean to capture in the Search category is the visual action needed to locate the objects that are the arguments for the specific Perception (and also for a specific Transformation). For example, on a display consisting of 50 labeled vertical bars whose lengths are proportional to the populations of 50 states, the Perception, `?Longer(bar x, bar y)`, would require the user to locate the two bars for the two states, and then apply the `Longer` Perception. Since comparing lengths of bars is an operation that would be useful for many tasks, and for which we can determine empirically the parameters for the human architecture, it is a good idea to separate this basic perception from searching for the items. The parameters for the perception operation would apply both to cluttered and uncluttered displays. Another reason to separate Search is that a visual search operation may also be applicable to Transformations, where a user may need to search and locate a button for a specific Transformation. In brief, we mean to include in Search visual search needed to identify the objects involved in given Perceptions and Transformations.

Inference is the name we give to the cognitive activity that processes the information in STM, by rule-based reasoning or mental imagery-like operations to obtain additional information. This process may involve additional elements brought from LTM to STM. Deciding on the next steps as well as solving the problem would typically require Inference steps.

In complexity the decision support tasks we consider occupy a place midway between using a display once to get some needed information, and open-ended interaction during which the steps are not pre-determined but require the user to engage in problem solving, e.g., to decide what Transformation should be applied next. We will assume in our framework that the user knows how to efficiently and effectively use the display for his purposes. This means that the tasks and the user's expertise are such that the next action to take is clear to the user, and that he has all the background knowledge needed for making the needed inferences. This is a fair assumption since displays for a task need to be compared based on intended optimal use of the display. (How hard it is to learn the best method for a display for a task is a separate issue, not dealt with here.)

**Precision and Accuracy of the Various Operations**. All the motor and perception related operations – Perception, Transformation and Search – are assumed to be potentially error-prone. For example, a user might make an error when required to choose the longer of two lines of almost equal length, or to distinguish between objects with very similar colors. It might also be hard to select a region with the mouse exactly within some planned coordinates, and finally, while searching among numerous items, one

[1] In the following analysis, we capitalize Perception, Transformation, Inference, etc when we intend to refer to operations that are to be taken as formally in the various sets of operations. We use lower case when we intend to refer to the general actions meant by the terms.

might miss the item that is sought or choose the wrong item. A heavy load on STM might result in loss of data, thus making the inference also unreliable. The PTIS technique allows for error rates, determined from empirical work, to be associated with the affected operation types.

## Illustrative Task

We illustrate the approach by systematically applying it to a task that is common in data mining. The domain $D$ of interest is characterized by a set of $n$ numeric-valued variables $\{x_1, x_2, ..x_n\}$, and we have a set $S$ of data about $m$ entities in $D$. Then $S = \{d_1, d_2, ..d_m\}$ where $d_i = (x_{i1}, x_{i2}...x_{in})$. We assume that the data are fully specified. Developing an understanding of the structure of D given S is a problem of great interest in data mining. A common form of such understanding is developing an account of any correlations that may exist between pairs of variables in $\{x_1, x_2, ..x_n\}$, and the ranges in which such correlations exist. The fact that correlations might exist over parts rather than the whole of the range makes visual means of hypothesizing such correlations especially useful, since standard correlation-detecting statistical algorithms might miss such correlations.

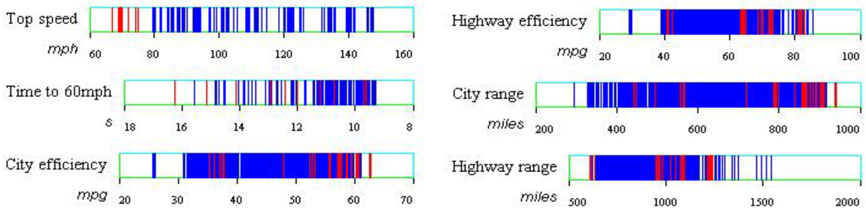


**Fig. 1.** The Spectrum display. (The figure has to be viewed in color.)

For example, in a case where there is a positive correlation over half the range and a negative correlation in the remainder, such algorithms would report no correlation at all. In contrast, a well-designed display (as we shall soon see) can help the user hypothesize such correlations easily.

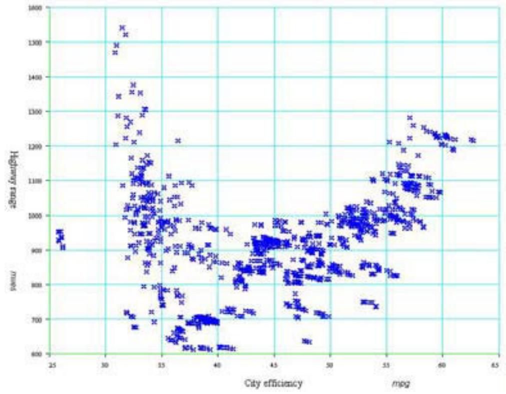


**Fig. 2.** Scatter diagram for two variables.

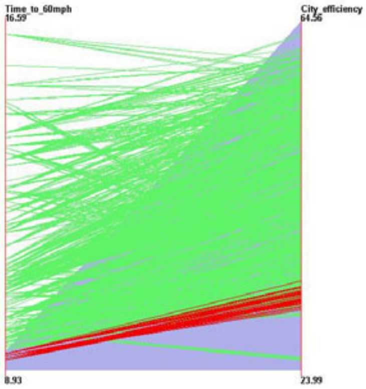


**Fig. 3.** Parallel Coordinates

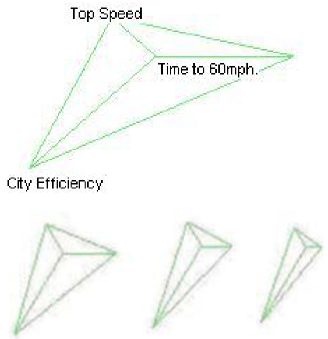


**Fig. 4.** Star Glyph; bottom, glyphs ordered by values of one of the variables

A technical caveat is in order: such visual displays can only *suggest* correlations. The significance level of the correlation and the actual correlation coefficient can only be properly computed by statistical algorithms. We assume that once the user hypothesizes such correlations and ranges using visual displays, the data, the pair of variables *and the ranges* are input to an appropriate statistical algorithm. With the added information about the ranges, the algorithm can calculate the correlation parameters accurately. In the rest of the paper, we will focus on just the hypothesizing part of correlation discovery.

Though the more complex versions of the task raise additional interesting issues (see Yovtchev, 2005, for evaluation of the displays on the various versions of the task), in the available space, we will restrict ourselves to the simple version, below:

- *Task*. Given the set of data about some domain in the form of the values that $m$ entities from that domain take on two variables $x_1$ and $x_2$, hypothesize all the subranges of the variables $x_1$ and $x_2$ where the correlation coefficient differs from 0.

## Diagrammatic Displays Considered

A number of diagrammatic forms have been proposed to represent data of the type we described[2]. In this paper, we use a subset of these displays – sufficient to introduce the approach and make the main points.

**Spectra.** In this display (Fig. 1), each variable is represented by a horizontal strip – the strips are typically normalized so that their ranges take up approximately the same length – and each of the entities in the data set is represented by a vertical stripe (the height of the stripe has no significance) in the default color, say blue, at the location corresponding to the value of the entity on that variable. More than one entity may have the same value for a variable, so the entities might be *stacked* at that location, and when entities are *dense*, i.e, many are close together, they may not be visibly distinct.

[2] For Spectra and Scatter plots, we used the Viewer (Josephson, *et al*, 1998), available from Aetion Technologies LLC, www.aetion.com. For Parallel Coordinates and Star Glyphs, we used XMDV tool (Ward, 1994).

*Transformations*. The user can select a window of variable size on any of the spectra (e.g., the window [18-16] in the “Time to 60” Spectrum in Fig. 1). This changes the color of the entities in the window (in Fig. 1 they appear in red), not only in the Spectrum where the window was selected, but on all the other Spectra as well. (As a new window is selected, the old window is automatically cleared.)

**Scatter Diagrams.** This display (example in Fig. 2) is a 2-axis Cartesian graph, with one variable on each axis. For *n* variables, a maximum of $n*(n-1)/2$ scatter plots are possible. The entities are represented as points at locations corresponding to their values on the variables. Remarks we made on stacking and density of entities in the Spectrum case apply here as well.

*Transformations*. The user can select a rectangular window in any of the scatter plots, and the entities in the window will change color, not only in that scatter plot, but in all other scatter plots that are constructed.

**Parallel Coordinates**. This display (Fig. 3) has *n* parallel axes - one per variable displayed. The *m* alternatives are displayed as *m* paths of *n-1* straight-line segments crossing the axes at positions corresponding to the entity values in the respective variables (Fig. 3 shows just two variables). Remarks on stacking and density that we made earlier also apply for this display.

*Transformations*. The user can select a range in any of the variables, and the entities in the selection window, and the lines connecting the values on other axes of each of these entities will change color. Fig. 3 shows a selection.

**Star Glyphs.** Each entity is represented as a glyph, which consists of *n* rays (for *n* variables) going out of its center whose endpoints are connected to form a polygon, as in Fig. 4 which shows an example for 3 variables. The length of a ray is proportional to the value of the entity on that variable. Making a glyph requires a minimum of three variables. The bottom of Fig. 4 shows a Star Glyph display of 3 objects, each represented on 3 variables, and ordered by the values on the variable on the ray at $0^o$. There are no Transformations available.

## Performance Analyses

*Sizes of Selection Windows*. Many of the methods call for making selections using a window, whose size the user needs to set. First, the window size needs to be large enough to capture enough samples so that the hypotheses are statistically meaningful. The size also determines the smallest range over which meaningful correlations may be hypothesized. If the window size is say 10% of the range of the variable, then any changes in correlations in ranges of the same order cannot be detected. There are precise statistical formulas available (Yovtchev, 2005) to make these estimates.

**Spectrum Display**

***Method:***

1. Transform display by making a selection window over the range of $x_1$ of size at most half as large as the smallest subrange over which any existing correlation is to be detected, where the window starts at the beginning of the range of $x_1$.

2. Perceive and store in memory the midpoint (mean) of the resulting selection in $x_2$.

3. Transform display by defining another selection window of the same size as the first one, but beginning where the first one ended. (We assume in analysis that the current window is automatically cleared.)

4. Compare the midpoint (mean) of the resulting selection in $x_2$ with previous *k* memorized ones. Infer and remember the trend that resulted from the comparison, and the beginning of the range where the trend emerged, or Infer the end of a trend that has been present so far. (Here by “trend” means whether the midpoint moves to the right or left systematically as the window moves to the right, or whether the midpoint movement has no systematic connection to the direction of the movement of window.)

5. Repeat the procedure until the end of the range of $x_1$ is reached.

***Perception:*** *Perceiving midpoint of a set of points*. In this Perception the user mentally estimates the midpoint of a given set of points on a line, e.g., the midpoint of the set of red points in the “Highway Range” Spectrum in Fig. 1. This activity has an associated error measure. It may involve sequential mental computations, but it is useful to treat it for our purposes as a reusable unit of mental activity.

***Transformation:*** The only Transformation operation is window selection. In some display versions, a window may need to be explicitly deleted; in that case, the number of Transformations will double.

***Inference:*** *Determining the trend in mid-point position*. This action can be modeled in finer detail as keeping the previous *k* midpoint locations in STM, and comparing their values, to determine if a trend, positive or negative exists, and if there was a trend, whether it continues or has stopped. The higher the value of *k*, the more reliable the estimate, but higher also the load on STM.

***Analysis:*** The sequence of operations is as follows: Select window (Transformation) in $x_1$, Perceive midpoint in $x_2$, Clear-and-Select next window (Transformation), Perceive midpoint, ..Infer trend in direction of midpoint, Transform, Perceive, Infer..

Without making finer distinctions, for a first approximation, the process takes *r/s* Perceptions, Transformations, and Inferences, where *r* is the range of $x_1$ and *s* is the size of the selection window measured in the same units as the range.

The maximum load on STM would be $(k + 3)$, the sum of the number of previous perceptions over which the trend is inferred, and a pair of locations and one sign (positive or negative) for each of the correlation ranges discovered.

*Errors*. In addition to the intrinsic error rate in the Perception of the midpoint, the display adds another potential for error: since more than one entity can occupy the same point, the user has no immediate access to the density information, and the midpoint estimate might be skewed. Because of missing density information, and due to the inherent human error in the basic perception involved, correlations may be missed, and even when detected, the starting and ending points could be off by some amount.

The load on STM, which can be quite high, can also lead to errors due to data loss. The error in tracking the direction of the movement of the midpoint due to STM load can be decreased by the user revisiting earlier window locations and repeating the movements, but this is at the cost of an increase in the number of Transformations.

*Ideas for Display Improvement*. The big source of error, *viz*., potential high stress on STM, can be minimized by changes in display design. If the user had access to a Transformation whereby beginning and end of each hypothesized correlation range can be marked on the screen along with its sign, STM load would be minimized. However, this would increase the total number of Transformations by 3 per correlation range. Further trade-offs between STM load and increase in number of Transformations are possible.

**Scatter Diagrams**

***Method:*** In the case of 2 variables, there is only one scatter diagram, as in Fig. 2. The scatter diagram is the most direct way to perceive any correlations. It calls for Perceiving correlation regions directly, as one can see in Fig. 2, that there is a negative correlation from $x_1$ value of 30 to 45, and a positive correlation from 45 to 60.

***Perception, Transformation and Inference:*** Perceiving plausible regions of correlation can be modeled as cluster detection, where the clusters are characterized by scatter around a straight line, perceiving the beginning and end points of the straight lines and the sign of their slopes. The task calls for distinguishing between clusters whose axis has a slope of 0 from those with a non-0 slope. Subject to confirmation from empirical data, it seems to us that correlation hypothesizing in this case is much less error-prone than in the Spectrum case – no stressing of STM; also faster, since it is direct and skips those inference steps that are needed for the Spectrum display. Nevertheless, in comparison with the optimal algorithms, there are bound to be some errors in the precise location of the end points, and also there are potentials to miss and mis-hypothesize correlations with a low correlation coefficient. Assuming that the hypotheses generated are to be fed to mathematical algorithm to generate quantitative information about the correlation, users might be trained to err in the direction of hypothesizing correlations when they are doubtful, with the idea that the numerical procedures might be able to reject dubious hypotheses.

There are no Transformations or Inferences needed for the 2-variable case, and thus there is little load on STM.

***Analysis:*** With the proviso that the act of Perception described above is complex, involving a sequence of mental operations, the task simply calls for one act of Perception. The temporal complexity of this Perception is approximately linear in the number of correlation regions, with a minimal part that would exist even when there are no correlations.

*Errors*. As mentioned, there are inherent errors in human perception of correlation, the error increasing as the correlation coefficient decreases. There are also errors in the locations of the end points. We hypothesize that both these errors are inversely proportional to the number of entities, i.e., human performance would have less error as the number of entities increases.

*Display Improvement*. An additional Transformation, Zoom, might help if applied for repeated Perception operations to locate the end points. Of course, this change to the display design will increase the number of Perceptions and Transformations for completing the task.

Because of its simplicity, low error rates and low stress on STM in comparison to the alternatives, the Scatter Diagram can be taken as the gold standard display for the task under consideration.

**Parallel Coordinates**

***Method:*** Depending on the density of the data, different methods seem to be appropriate.

*Relatively Sparse Data.* When the density of the data is low enough that the values that an entity takes on different axes can be distinguished, the correlation regions, if any, and the directions of the correlation are available for Perception.

*Relatively Dense Data*. In this case, since lines connecting the values of the individual entities cannot be distinguished (Fig. 3), the method is similar to that for the Spectrum display. A variation on this method is track the average slope of the lines created by the window (the average slope of the red lines in Fig. 3). If the slope starts and stays positive (negative) within a region, positive (negative) correlation may be hypothesized.

***Perception, Transformation and Inference:*** In the case of Sparse data, the basic Perception is not gestalt as in cluster recognition in the case of the Scatter Diagrams, but involves a sequence of comparisons. The user sweeps through a range of $x_1$ and visually follows the slopes of the lines connecting to $x_2$. Thus, it is likely to take longer time, and is possibly more error-prone. In the Dense case, once selection is made, the required Perception is similar to that in the Sparse case, and the same remarks apply. For Sparse Data, as in the case for Scatter Diagrams, there is no need for Transformation and Inference operations; for Dense Data, remarks made in the case of Spectrum display apply.

***Analysis:*** For Sparse*,* except for likely higher error rates in Perception, the same analysis as for Scatter Diagrams applies. For Dense, remarks similar to that in the Spectrum case apply, with possibly different values for error rates for Perception.

***Design Ideas****:* The error analysis of the cases coincides respectively with the Viewer's Spectrum and Scatter Diagram error analysis. Hence, it leads to the same design ideas -- zoom functionality, and markup operations to mark starts and ends of hypothesized correlations. The latter trades off STM overload for an increase in the number of Transformations.

**Star Glyphs**

Since the glyphs have to have a minimum of 3 variables, we add a pseudo variable whose values are the same for all the entities (or it is a real variable in the domain whose values we ignore). Let us also assume that the glyphs are ordered on their values on $x_1$ as in the bottom part of Fig. 4.

***Method:*** Scanning the glyphs in order of the value of $x_1$, for glyph *i,* compare its $x_2$ value with $k$ previous values to infer whether a trend of increase or decrease has begun, and if already begun, maintained. If the trend just began or ended, save the value of $i$ to STM. Continue until all the glyphs are scanned. The value of $k$ is set based on considerations described when we discussed inference in the use of the Spectrum display, i.e., to smooth out random local variations. As before, higher $k$ reduces statistical error, but errors due to resulting overload of STM might reduce or eliminate the advantage.

***Perception, Transformation and Inference:*** The basic Perception is one of local comparison of $x_2$ values to decide if an increase or decrease is observed. There are no Transformation operations. The issues regarding Inference are similar to our corresponding discussion for the Spectrum case. That is, the $x_2$ values of $k$ glyphs are kept in STM, and their values are compared to determine beginning, maintenance or end of positive or negative covariation trends. As before, the higher the value of $k$, the more reliable the estimate, but higher also the load on STM.

***Analysis:*** The number of basic Perception steps is $(m\text{-}1)$, the number of glyphs. The number of Inference steps is $(m - k)$. Maximum load on STM is ($k$ + 3* number of correlations ranges hypothesized), since each correlation region requires remembering 2 end points and its sign. Because no windows are used to average out behavior, the numbers of Perception and Inference steps are quite large.

*Error rates*. The basic Perception is quite reliable, except when the increase or decrease is very small, in which case the error does not likely matter much. The Inference step is error-prone because of the complexity of calculation, and the requirement on STM to keep $k$ items. Starting and end point assessments are especially likely to be error-prone because of natural statistical variations on $x_2$ values, which need to be smoothed out during the Inference step.

*Design Ideas.* As before, the load on STM may be reduced by providing Transformation operations to mark the beginnings, ends and the signs of the correlations.

### Comparing Displays

Even this level of qualitative analysis is useful in making comparisons. The Scatter Diagrams are the most direct – no Transformation operations, no Inference, and little stress on STM. The Glyphs are especially laborious to use, and the Perception and Inference steps seem prone to high error rates for both the Glyph and Spectrum displays. Whatever the general attractiveness of the Glyph displays, they are not well suited for the specific task we considered.

## Concluding Remarks

The paper outlines an approach in the GOMS framework to systematize investigating how good specific diagrammatic schemes are for specific families of tasks. Unlike earlier applications of the GOMS framework, which involved elementary operations at a relatively low level of granularity, diagrammatic interfaces used in decision support systems involve relatively complex perceptions and physical interactions. We illustrated the approach by a comparative performance analysis of several candidate diagrammatic interfaces for the task of discovering relations between variables in some domain of interest. The analysis results in estimates of the numbers of various basic operations, such as Perception, Transformation, Search and Inference, and of stress on short-term memory. For many DSS applications qualitative results as we obtained are sufficient. The precise timings about how long the entire process would take may be less important than whether the display calls for significantly more interaction compared to another display, whether perceptions are more likely to be error-prone in one than another, etc. However, empirical data about the timing and error rates of the human cognitive architecture on the basic operations can be used for more precise predictive evaluations. We expect to launch such an initiative soon.

The approach can help to identify aspects of the display that need improvement. Adding Transformations to mark partial results on the display may be considered if the analysis indicates potential for STM stress. If analysis indicates that the contribution of errors in specific Perception is significant, alternatives might be considered.

Goodness of an interface given the best method is not the same as how good it is in helping someone *learn* the best method. Our methodology can be applied to the latter task as well, and it is an important future direction of research.

## Acknowledgments

This paper was prepared through participation in the Advanced Decision Architectures Collaborative Technology Alliance sponsored by the U.S. Army Research Laboratory under Cooperative Agreement DAAD19-01-2-0009.

## References

Card, Stuart, Moran, Thomas P., and Newell, Allen (1983). *The Psychology of Human-Computer Interaction*, Lawrence Erlbaum Associates, Hillsdale, NJ.

John, Bonnie E. and Kieras, David (1996). The GOMS family of user interface analysis techniques: Comparison and contrast. *ACM Transactions on Computer-Human Interaction*, 3(4):320-351, December 1996.

Josephson, John R., Chandrasekaran, B., Carroll, Mark, Iyer, Naresh, Wasacz, Bryon, Rizzoni, Giorgio, Li, Qingyuan, and Erb, David A. (1998). An architecture for exploring large design spaces. In *Proc. of the National Conf on AI* (AAAI-98), pages 143-150. AAAI Press/The MIT Press.

Larkin, J. H., & Simon, H. A. (1987). Why a diagram is (sometimes) worth ten thousand words. *Cognitive Science*, 11, 65 -100.

Peebles, D., & Cheng, P. C.-H. (2003). Modeling the effect of task and graphical representation on response latency in a graph reading task . *Human Factors*, 45, 28-46.

Ward, Matthew O. (1994). XMDV tool: Integrating multiple methods for visualizing multivariate data. In *IEEE Visualization*, 326-333, 1994.

Yovtchev, Tsviatko (2005). PTIS - A High Level Framework for Comparative Evaluation of Decision Support Interfaces, MS Thesis, Computer Science and Engineering, The Ohio State University.

# The First Second of Symmetry: Towards a Model of Visual Search during Symmetry Verification

**Kenneth Czechowski (kentcz@cc.gatech.edu)**
**Ronald W. Ferguson (rwf@cc.gatech.edu)**
**Rudolph L. Mappus IV (cmappus@cc.gatech.edu)**
College of Computing, Georgia Institute of Technology
Atlanta, GA 30332 USA

## Abstract

This paper introduces a newly discovered pattern of eye-movements during symmetry judgments and a corresponding model of visual search. We analyze a corpus of eye movements containing over 19,000 symmetry judgments in two experiments. These eye movements show a "treble-clef-shaped" search strategy that is based on proximate feature inspection and cross-axis comparisons. We simulate this treble clef strategy and show that it accounts for the experimental data better than alternative models.

## Introduction: Symmetry Verification

Symmetry judgment is a central process in perception, supporting perceptual organization and object-centered reference frames. Symmetry detection takes place during the earlier phase of the visual process and therefore has a significant impact on object recognition and perceptual organization. It is detected quickly and accurately, even after display times as short as 50 ms.

Although symmetry judgment seems effortless and instantaneous, there is considerable evidence that it is a two-stage process. As proposed by Palmer & Hemenway (1978), symmetry judgment begins with a quick but rough symmetry estimation during an *axis detection phase* that is 50-200 ms long. Then there is a 2000-4000 ms *verification phase* that checks the axis by closely examining the stimulus.

The verification phase of symmetry judgment is important for two reasons. First, because it uses information from the axis detection phase, the pattern of verification may help us implicitly understand what axis detection initially omits. Second, the verification phase provides general clues on how visual regularity guides visual search.

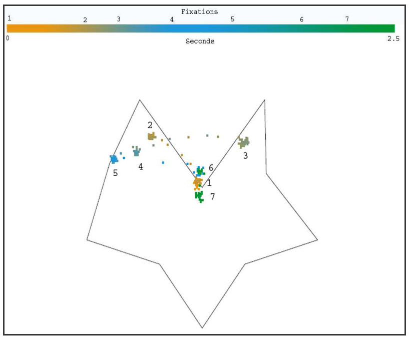


Figure 2: Although humans can detect a symmetry axis very quickly, verifying symmetry takes much longer. Here a participant uses seven fixations over 2500 ms to determine that the figure is asymmetric. The shading and color (color bar, top) indicates temporal order of samples.

Eye-tracking allows us to examine the verification phase directly to determine the pattern of fixations and saccades (Figure 2). (In contrast, the axis detection phase happens too quickly for saccades, leaving accuracy and response time as the typical dependent variables in symmetry studies.) Current eye-trackers measure eye movements with high precision and fast sample rates.

However, while eye trackers are increasingly precise, visual search paths remain highly variable, even for the same participant performing the same task on the same stimulus. Some researchers in motor control suggest that visual search uses stochastic processes to free the mind from the burden of computing the fine details of ocular control (see Mitchell 2003 for a discussion).

The variability of eye movements makes it difficult to model visual search during verification. Minimally, a large dataset is essential for analyzing underlying patterns despite this variability. In addition, general search strategies must be separated from those specific to symmetry judgment.

We address this challenge by using two large data sets of eye-tracking recordings from two previous experiments. These two experiments tracked the eye movements of 71 participants for over 19,000 symmetry judgments. We first analyze this data to determine a new model of visual search. We then use a computer simulation to test this model against alternative search strategies for eye movement. This new model of symmetry verification should allow more precise measurements of interactions between the axis detection and verification phases.

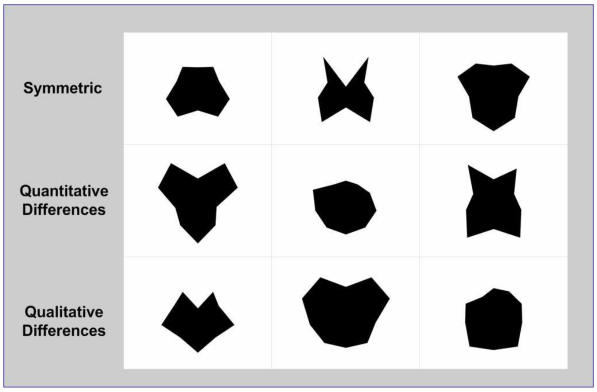


Figure 1: Example stimuli from Experiment 1, with three examples of each symmetry type.

| | Experiment 1<br>(144 stimuli) | Experiment 2<br>(480 stimuli) |
|---|---|---|
| Design Conditions | *Symmetry Types* (3): symmetry, near-symmetry with qualitative difference, near-symmetry with quantitative difference<br>*Fill Types* (2): filled / unfilled<br>*Sizes* (3): small, medium, and large | *Symmetry types* (6): symmetry, near-symmetry with qualitative difference (small and large), near-symmetry with quantitative difference (small and large), asymmetric.<br>*Complexity* (3): 10, 18, & 26 sides |
| Polygon radius | Small (50 ± 25 pixels),<br>Medium (150 ± 50 pixels),<br>Large (200 ± 100 pixels). | 150 ± 70 pixels |

Table 1: Summary of experimental designs in (Ferguson *et al.*, in preparation; Mappus *et al.*, 2005).

## Review of Corpus Experiments and Data

The eye movement data analyzed here is from two recent experiments exploring how symmetry judgment is influenced by two types of near-symmetry (Ferguson et al., in preparation; Mappus et al., 2005). One type of near-symmetry is symmetric except for a *qualitative difference* (Figure 1), where pair of corresponding parts differs relationally (e.g., a concave vertex on one side of a polygon opposite a convex one). In contrast, the other type of near-symmetry involves a *quantitative difference*, which is one of degree only (e.g., opposing concave vertices where one indentation is larger).

Previous experiments (Ferguson *et al.*, 1996) showed that participants more accurately judge near-symmetries when differences are qualitative rather than quantitative. These two experiments retest this result, but also use eye tracking to see if qualitative differences influence the number and location of visual fixations.

The experimental designs are summarized in Table 1. Experiment 1 used 10-sided polygons (Figure 1) and measured the accuracy and eye-movements of each symmetry judgment. Experiment 2 used a wider range of symmetry types (Figure 3). Nearly all stimuli were symmetric or near-symmetric (except for Experiment 2, which also had totally asymmetric polygons).

The two experiments used different display times and conditions. Experiment 1 examined symmetry judgments of unlimited duration while testing for effects of stimulus size and fill. Experiment 2, in contrast, used a fixed presentation time (1000 ms) and tested for effects of stimulus complexity and difference size.

Both experiments tracked eye movements during the symmetry judgment task using a corneal reflection eye-tracker with a temporal resolution of 8.3 ms (120 Hz). In Experiment 1, participants responded, on average, within 3000 ms using 6 fixations. Experiment 2 reduced this to 1000 ms and 3 fixations.

Overall, 96 university students with normal or adjusted-to-normal vision participated in the studies for course credit. Experiment 1 used 55 participants, but 9 were dropped from the analysis due to high error rates or eye-tracker calibration errors. Similarly, Experiment 2 used 41 participants with 16 participants dropped.

The polygonal stimuli were randomly generated by connecting points along a set of evenly-spaced radii, as in (Palmer & Hemenway, 1978). Near-symmetric polygons were generated by changing a random vertex of a symmetric polygon by a random amount (as given in Table 1). Stimuli were shown in black on a white background subtending approximately 2 visual degrees.

The results showed clear processing differences for near-symmetric figures where differences were qualitative rather than quantitative. Participants made more judgment errors when differences were quantitative (Figure 4), and also used more fixations (an average of 6.3 fixations for quantitative and 5.3 for qualitative). Experiment 2 showed similar results (Figure 5), although the effect was greatly attenuated when quantitative differences were large or when the figure was extremely complex (e.g., for 26-gons).

These experiments showed that qualitative differences influence the number and placement of fixations, but did not predict the search path. The goal on this analysis is to determine if there is an underlying search strategy that participants used.

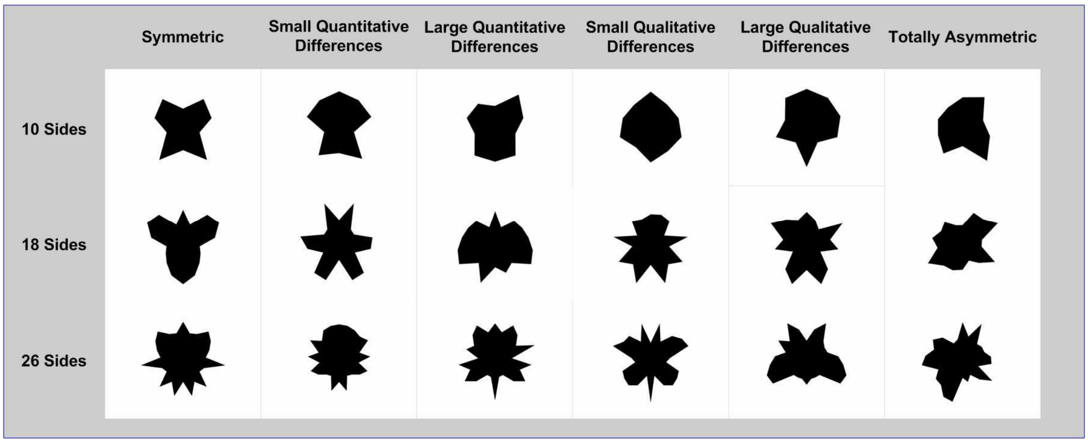

Symmetric
Small Quantitative Differences
Large Quantitative Differences
Small Qualitative Differences
Large Qualitative Differences
Totally Asymmetric
10 Sides
18 Sides
26 Sides

Figure 3: Example stimuli from Experiment 2. This chart shows one stimuli for each of the 18 different conditions.

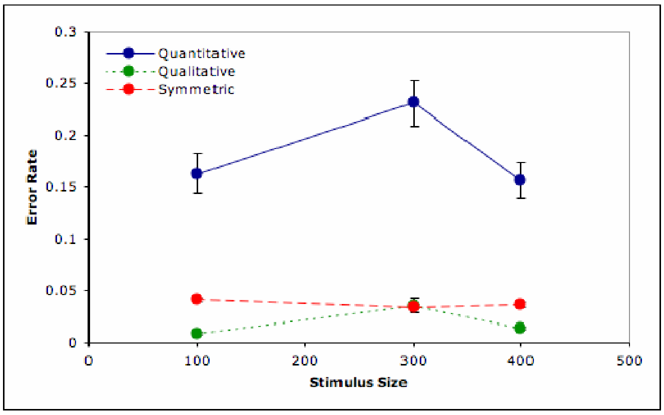


Figure 4: Error rates from Experiment 1 for symmetric figures, and near-symmetric figures with qualitative and quantitative differences.

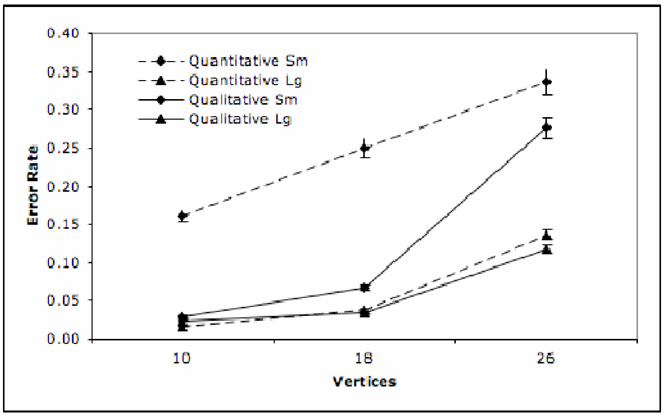


Figure 5: Error rates from Experiment 2 for near-symmetric figures with small and large qualitative and quantitative differences.

## Analysis of visual search patterns

In general, the use of near-symmetric stimuli emphasized symmetry verification. Each near-symmetric stimulus contained a single differing vertex, and so verification was difficult and required visual search.

Interestingly, our analysis shows that while individual movement paths were highly variable, the aggregate paths show a predictable pattern that we have dubbed the "Treble Clef" strategy (Figure 6). In this strategy, participants initially fixate at the center of the figure, and then move up. They then descend while moving from the left to the right.

The Treble Clef strategy has several components, which we will cover in turn, using graphs of the vertical and horizontal movement components found in Figure 7.

**Initial 200-250 ms. fixation**. Most participants began in the center of the stimulus and remain there for the first 200-250 ms (Figure 7). This is expected because experiment protocol requires each subject to fixate on the center of the screen prior to stimulus onset. The 200-250 ms duration corresponds to the time needed for the axis detection phase.

**Upward movement to tip of vertical axis**. Then for the rest of the initial 500 ms the eye movements begin to spread out and migrate to the top of the figure.

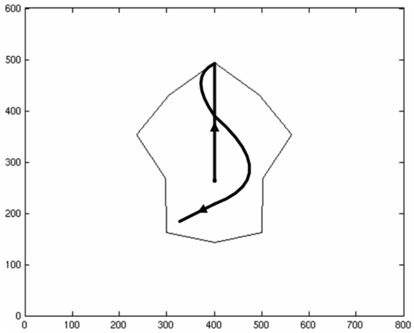


Figure 6: Diagram of Treble Clef search strategy. The diagram illustrates the general pattern of eye movements during the first 1,000 ms of a symmetry judgment task.

**Oscillating downward movement.** During the last 500 ms, the eye-tracking samples appear to spread out perpendicular to the center axis and slowly move down the figure. It is not until the end of the 1,000 ms that there are any samples at the bottom of the figure (Figure 8). This shows a clear top to bottom pattern; a pattern that has been noticed in both search tasks (Salvucci, 2000) and comparative search (Pomplun, 1998).

**Other characteristics**. There are several other characteristics of the visual search data. First, the data shows a clear undershoot bias; rather than overshooting the saccade endpoint and landing outside the figure, the vast majority of the sample points remain inside the figure.

Second, the treble clef pattern remains whether the display time is constrained or unlimited. Similarities between the mean vertical/horizontal positions for the first 1,000 ms of Experiment 1 and the mean positions from Experiment 2 suggests that search patterns are very similar for the first 1,000 ms despite the time constraints of Experiment 2.

This is also supported by the relatively similar accuracy levels for 10-sided polygons in Experiments 1 and 2 (Figures 4-5), which suggest that participants are able to maintain accurate verification even when display times are reduced from 3000 to 1000 ms. Participants are either able to optimize their search strategy for the shorter time or gain little from the visual fixations after the first 1000 ms.

Also note a slight lag in the peaks of Experiment 1's mean positions (Figure 7-A,B) relative to Experiment 2's (Figure 7-C,D). The apparent expanding of the waves in the graphs suggests that Experiment 2's time constraints may have improved the efficiency of the eye movements.

In summary, the treble clef pattern is very consistent. On average, participants start at the center, quickly move up, and then slowly move down, swerving from the left to the right. Separating the samples by stimulus type, we see a striking consistency across all symmetry categories despite other performance differences.

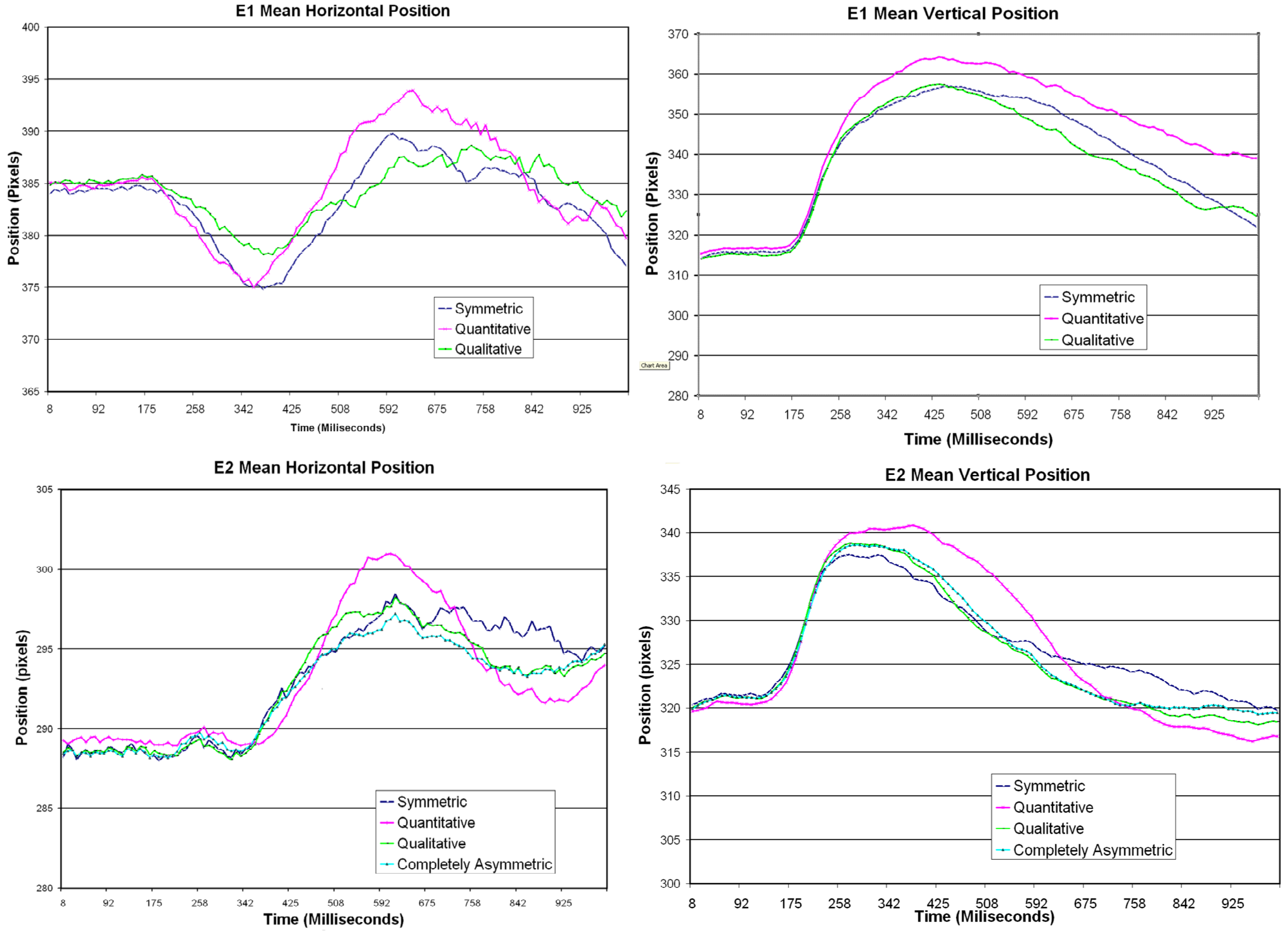


Figure 7: An overview of aggregate eye movements during Experiments 1 and 2, showing mean horizontal and vertical positions over time for all symmetry types. The Horizontal center is at 400 and the Vertical center is at 300.

## Simulation

To further test the treble clef strategy, we compared it against three alternative search strategies: Random Search, Greedy Search, and the Area Activation Model (AAM).

**Eye-movement model.** To do this, we first built an eye movement model, based on known psychological results, to serve as the basis for all four search strategies. The model groups movement into two categories: fixations and saccades. A fixation is a period of stable movements closely clustered around a point. Saccades are fast ballistic movements that propel the eye across a visual scene from one fixation to another. The model first generates a fixation at the stimulus center. At the end of each fixation the search strategy selects a new fixation location and plots a saccade. This continues until 1,000 ms has expired.

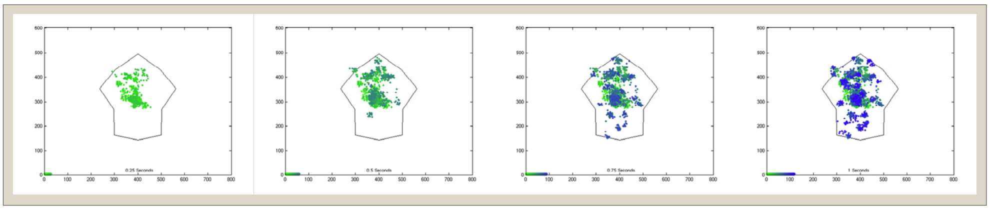


Figure 8: Time slices of eye-tracking samples for all subjects in Experiment 2. The diagram shows samples from a stimulus divided into four time slices: 0-250 ms, 0-500 ms, 0-750 ms, and 0-1000 ms.

To capture the stochastic nature of the fixations and saccades, variables such as the fixation duration and saccade duration are drawn from probabilistic distributions. In this model, the lengths of the fixation duration are sampled from a gamma distribution with a mean of 200 ms and a standard deviation of (1/3)*(200ms) (Epelboim & Suppes, 2001; Salvucci, 2000). During a fixation, locations for each time step are drawn from a 2D Gaussian distribution centered at the target with a standard deviation equal to one visual degree. To move from one fixation to another, a straight line saccade is charted with samples evenly distributed along the line. The saccade duration is 20 ms + .2 ms * visual angle. Saccade landings are not precise and instead of traveling the full distance d from one fixation to the next, saccade distance is drawn from a Gaussian distribution with a mean of d and standard deviation equal to 0.1d (Salvucci 2000). Following the undershoot bias, there is a 90% chance that the actual saccade distance will be less than d and a 10% chance that the actual saccade distance is greater than or equal to d.

The eye movement model then serves as the basis for the following four search strategies.

**Random Search Strategy.** The Random search strategy assumes that there is no underlying motivation for selecting fixations. Each vertex has an equal chance of being selected, with the constraint is that no vertex is selected twice.

**Greedy Search Strategy.** The Greedy search strategy assumes that the visual system attempts to maximize information by reducing time spent on saccades, and always selects fixation locations on the closest unvisited vertex.

**The Area Activation Model *(AAM)*.** The Area Activation Model (Pomplun *et al.*, 2000) is a generalized model of eye-movements during search tasks. Like the greedy search strategy, AAM tries to optimize information. However, instead of selecting nearby vertices to minimize saccade time, AAM finds clusters of vertices where it can maximize information in a single fixation. One particularly interesting result of this is that fixations can occur at the center of a cluster of vertices rather than directly on a single vertex.

According to the model, a 3D activation mesh is created preattentively. The mesh consists of a mixture (summation) of 2D Gaussians centered on the stimuli's items (vertices in our case). The peaks of the mesh become the candidate fixation locations and are weighted by their relative heights. The first candidate is selected using a weighted probability. Subsequent candidates are selected based on their proximity to the current fixation.

**Treble Clef.** The Treble Clef search strategy mimics the pattern discovered in the two experiments. After the initial fixation, the next fixation is at the top of the figure. The remaining fixation locations are on vertices that are lower than the current fixation. This search strategy also implements the swinging motion by alternating between vertices on the left and right of the symmetric axis.

## Comparing the Strategies

The goal of the simulation is to evaluate each search strategy's fit to the empirical data. Therefore, search strategies are tested by comparing the samples generated by the simulation (simulation samples) with the samples generated during Experiment 2 (experiment samples). The variability of the data requires the comparison of estimated sample distributions rather than a sample-to-sample comparison.

To quantitatively measure the accuracy of the simulations for a particular stimulus, we estimate the distributions of both the experimental samples and the simulation samples then calculate the divergence of the two distributions. The distributions are estimated using a mixture of Gaussians. One 2D Gaussian distribution, with a standard deviation equal to the radius of the foveal region, is placed at the coordinates of each sample point. The Gaussians are summed across the 2D space and scaled relative to the number of samples used to calculate the estimation. The percent of overlap, which we use as the measure of accuracy, is the summation of the minimum of the corresponding masses:

$$\sum_{X}\sum_{Y}\min[f(x,y),g(x,y)]$$

Where f(x,y) is the estimated probability distribution of the experimental samples and g(x,y) is the estimated probability distribution of the simulation samples. Since this calculation is specific to each stimulus, accuracy results are averaged across a random subset of Experiment 2 stimuli.

Figure 10 compares the mean accuracies of the different search strategies. Of the strategies tested, Treble Clef performs the best. Both Greedy and AAM perform with similar accuracy; this is expected because after selecting the first fixation the AAM strategy uses a greedy algorithm for selecting subsequent fixations. However, the deterministic nature of Greedy and the AMM strategies limits their ability to account for the variety of scan paths recorded from our experiment participants. The Random Strategy performs

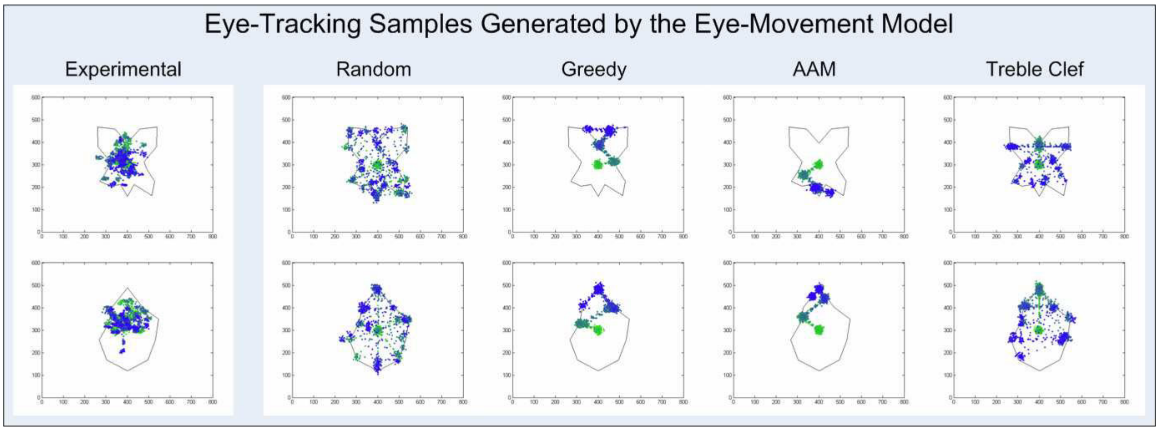


Figure 9: Eye-tracking samples generated by the simulation. The diagram shows how the different search strategies generate different sets of samples. The experimental samples from Experiment 2 are included on the left for comparison purposes.

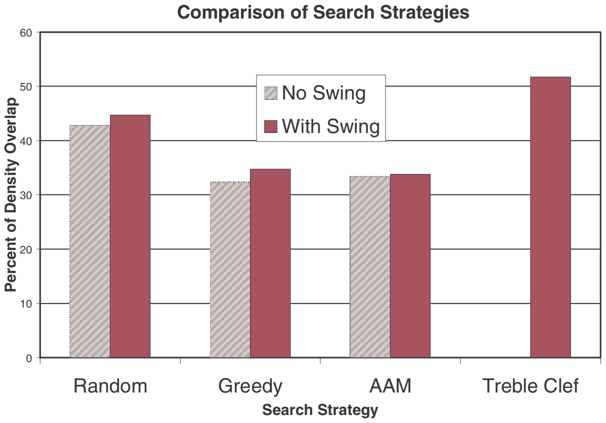


Figure 10: Mean accuracy of each search strategy. In this graph, the values are the percent of overlap with the estimated empirical density averaged across a randomly selected subset of Experiment 2 stimuli.

better than Greedy and AAM, but does not account for the vertex preferences that appear in the experimental data.

To simulate the comparative nature of the symmetry task we also tested a swing variant of Random, Greedy, and AAM. In all cases, adding the swing constraint (a constraint that successive fixations alternate between the left and right side of the symmetric axis, as is expected for symmetry judgment) improved performance.

## Discussion and Future Work

The eye-tracking data and simulation results demonstrate the existence of the Treble Clef pattern. While the variability of eye-movements disguises the underlying strategy for selecting fixation locations, the use of a large corpus has enabled us to identify elements of a common strategy. Furthermore, the use of an eye-movement simulator has shown this strategy better represents the experimental data than several alternative strategies.

The discovery of this strategy provides valuable insight into symmetry verification and symmetry perception in general. Previous work shows that symmetry type influences detection through differences in accuracy, number of fixations, response time, and scan paths (Ferguson et al 1996; Mappus et al 2005). Similarly, our eye-tracking data shows that symmetry types differ with respect to their mean vertical and horizontal positions, especially after 500 ms (see Figures 7-10). These search patterns, like accuracy and response time, are affected by the symmetry type. This suggests that studying perturbations of the Treble Clef search pattern may indicate processing differences between symmetry types. It also suggests that one could determine how a symmetry type influences the visual system by determining when the eye-movement patterns diverge.

The Treble Clef pattern also suggests that the ocular system influences search strategies. Instead of quick straight-line movements and sharp angles between fixations, the pattern shows wavy movements. This can be explained by momentum-like forces impacting the ocular muscles, which suggests that search strategies are optimized to work within the physical constraints of the ocular system.

In the future, we will refine the Treble Clef search strategy and the eye-movement model to better account for the experimental data. An experiment designed to test a wider variety of symmetry types could improve the parameters of the eye-movement model and thus simulation accuracy. This is one goal for follow-on experiments.

We have also begun work on an experiment that uses electroencephalographic (EEG) recordings to find brain activation patterns during symmetry judgment. By evaluating eye-tracking along with EEG we may gain valuable insight into the cognitive processes involved in symmetry judgment.

## Acknowledgments

This work is supported by a National Science Foundation grant under the Artificial Intelligence and Cognitive Science program and a Presidential Undergraduate Research Award to the first author. Special thanks to Paul Corballis for help with the experimental design and to the Georgia Tech Psychology Visual Attention Lab for use of their facilities. We also wish to thank reviewers' comments and suggestions.

## References

Epelboim, J., & Suppes, P. (2001). A model of eye movements and visual working memory during problem solving in geometry. *Vision Research*, 41, 1561-1574.

Ferguson, R. W., Aminoff, A., & Gentner, D. (1996). Modeling qualitative differences in symmetry judgments. Cognitive Science Conference. Hillsdale, NJ: Erlbaum.

Ferguson, R. W., Mappus, R. L., Czechowski, K., & Corballis, P. M. (in preparation). Spotting differences: Effects of geometric relations on visual search patterns. Manuscript in preparation.

Mappus, R. L., Ferguson, R. W., Czechowski, K., & Corballis, P. M. (2005). Spotting differences: How qualitative asymmetries influence visual search. Cognitive Science Conference. Hillsdale, NJ: Erlbaum.

Mitchell, J., Zipser, D. (2003) Sequential memory-guided saccades and target selection: a neural model of the frontal eye fields. *Vision Research*, 43, 2669-2695.

Palmer, S. E., & Hemenway, K. (1978). Orientation and symmetry: Effects of multiple, rotational, and near symmetries. *Journal of Experimental Psychology: Human Perception and Performance*, 4, 691-702.

Pomplun, M. (1998). Analysis and models of eye movement in comparative visual search. Göttingen: Cuvillier.

Pomplun, M., Reingold, E. M., Shen, J., & Williams, D. E. (2000). The area activation model of saccadic selectivity in visual search. Cognitive Science Conference. Mahwah, NJ: Erlbaum.

Salvucci, D. D. (2000). A model of eye movements and visual attention. International Conference on Cognitive Modeling. Veenendaal, Netherlands: Universal Press.

# Diagrammatic Reasoning: Route Planning on Maps with ACT-R

**H. A. Dye (hdye@ttocs.org)**
U.S. Military Academy/Army Research Laboratory
MADN-MATH, 646 Swift Road
West Point, NY 10566, USA

**Keywords:** Cognitive modeling; diagrams; maps

## Diagrammatic Reasoning

Diagrammatic reasoning, reasoning from graphical representations rather than from word-based representations, is pervasive in our society. Computers allow us to easily design and transmit diagrams that encapsulate a variety of information. Maps are specific instances of diagrams that are used to provide current and projected information (Chandrasekaran et al., 2002); for example, public transportation systems are displayed as color coded graphs. Other examples of diagrammatic reasoning include geometric problem solving in mathematics and free body diagrams in physics. Diagrams can offer cognitive shortcuts relative to verbal descriptions of certain kinds of information, notably relational and spatial information. Thus, diagrams can reduce the working memory load and make possible certain cognitive efficiencies.

### Challenges

The overall goal of this research project is to produce cognitively-congruent models of diagrammatic reasoning in the Adaptive Control of Thought – Rational (ACT-R) architecture (Anderson et al., 2004). This study investigated perception and reasoning during a problem solving task that utilizes a diagram. The model presented here uses the architecture's perceptual and motor modules; visual objects were created and placed in a virtual window on which the perceptual and motor modules could then act. There were two challenges in building the model: (1) ACT-R's visual module is text- rather than diagram-based and (2) previous vision modeling efforts (e.g., Fick and Byrne, 2003) have focused on target search where the target is identified rather than using the visual information for subsequent decision making. To validate the ACT-R model, participants performed two diagrammatic reasoning tasks.

### Diagrammatic Reasoning Tasks

Eighteen participants (U.S. Military Academy cadets) performed two simple tasks on a 5x5 grid-based map, consisting of labeled points, lines, and regions. The simplicity of these maps allowed both the isolation of the effect of specific changes in the maps and the extraction of relatively rich cognitive data. Specifically, the two tasks were (1) *"find,"* finding a target location (B) on the map (perception) or (2) *"plan,"* finding the target location (B) and executing a planned route from location (A) (perception plus decision making) (see Figure 1). The target location was positioned in one of the four corners and task difficulty was manipulated by limiting the number of direct paths to the target. For example, in Figure 1, one direct path to the target has been eliminated. Zero, one, or two (both) direct paths to the target could be eliminated.

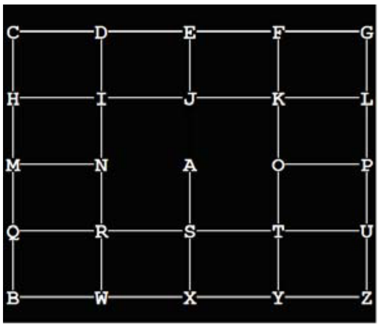


Figure 1: A sample map

Both tasks began with the participants being presented with a center-screen fixation point. In the find task, participants pressed the space bar to indicate that they had found the target location. As a check, the grid labels were then erased and the participant used the mouse to click on the target location. In the route planning task, participants found the target location and moved a red outlined box along the paths using the arrow keys to indicate the selected path. When the box was positioned over the target location, participants pressed the enter key. The search task is an assumed subtask in the route planning task. Task difficulty was manipulated by blocking direct paths to the target location (deleting paths). E-Prime was used to display and manipulate the grid and collect response data. End target location and the number and position of paths deleted were completely counterbalanced in a within-subjects design.

The average response times in msec for the find and plan tasks for target position and the blocked paths are shown in Figures 2. Find and plan times generally increased across target location according to a left-to-right upper and upper-to-lower strategy with a mixed strategy on the lower, either left-to-right or right-to-left. Plan task times increased with increasing numbers of paths blocked.

### Cognitive Modeling

Following the general strategy exhibited by participants, the ACT-R 6.0 cognitive model begins the find and plan tasks by moving visual attention away from the fixation

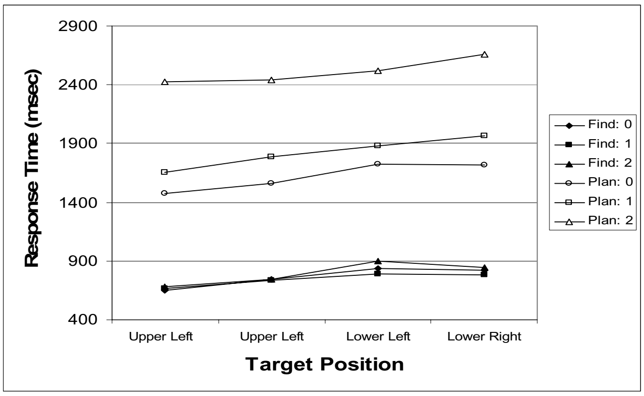


Figure 2: Averaged participant responses for the tasks with 0, 1, or 2 paths blocked and target location

point, looks at the corners of the grid, proceeding from the upper left to the upper right and then lower right and lower left until it finds the target location. In the model of the find task, the model then presses a key to indicate that it has found the target location. The model of the route planning task builds on the search model. After finding the target, the model's perception focuses on the area around the starting location (A). [Note: Visual attention in ACT-R must be directed to an object and cannot be directed to an open space. To allow the model to position visual attention on the deleted paths, these paths were colored black rather than actually deleted (Cassenti, Kelley, & Ghirardelli, 2006).] The model then plans a route to the end location (B) from the start location (A) by focusing visual attention on a path in a region defined by the current location and proximity to the target location. The first arrow key is determined by the existing edge nearest the target that captures visual attention. Next, the model selects a path with the same direction as the last traversed path, causing the same arrow key to be pressed again. If no such path exists, the model shifts strategies and attempts to find any path that advances the route towards the target position. The model does not observe the labels of the intermediate locations or non-relevant paths during the find or planning processes.

The find and the plan models were run for 144 trials each or the equivalent of one participant. The mean response times as a function of complexity (blocked paths) by end target location for the two models are shown in Figure 3. The find and plan models reproduced the participants' data fairly well (see Figure 3), with r = .93 and RMSD = 0.14 and r = .97 and RMSD = 0.13, respectively.

There are some limits to the current model, even with some post-experiment adjustments. Currently, the model does not take into account the following efficiencies and errors: a decrease in response time when a participant pressed the same key repeatedly; or errors when participants attempted to traverse paths that had been deleted or mistook the target location.

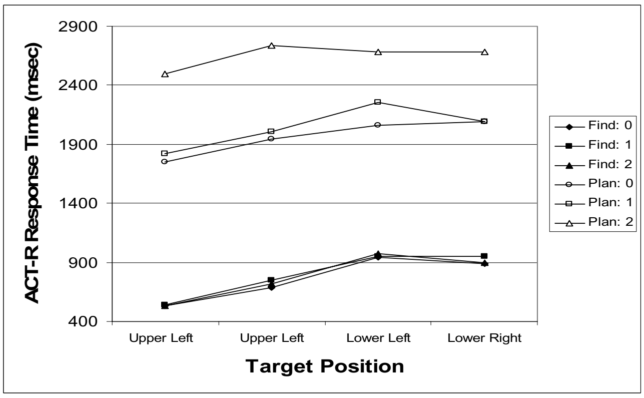


Figure 3: ACT-R model responses for find and plan tasks with 0, 1, or 2 paths blocked and target location

## Conclusions

The correspondence between the model and participants' data was reasonably high. A largely serial ACT-R model of the search and path selection process matched participants' data with respect to the find task alone and to the plan task, which subsumes the find task. Even with this simple grid, more blocked paths resulted in greater response times and with the generally used search strategy, times varied in an orderly fashion with target location. Two questions are prompted by these results: (1) What is the effect of diagram features such as missing paths or irrelevant paths on route planning? (2) At a more detailed level than explored in this model, how are finding and planning processes interrelated in the time to the first keystroke. For the participant data, the first plan task keystroke took approximately 200 msec longer than search task keystroke, pointing to a promising window in which to explore the basics of diagrammatic reasoning.

## Acknowledgments

This research was conducted while the researcher jointly held a National Research Council postdoctoral fellowship at the U.S. Military Academy and the Army Research Laboratory. I would like to thank Dan Cassenti for his input.

## References

Anderson, J. R., Bothell, D., Byrne, M. Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, *111*, 1036-1060.

Cassenti, D. N., Kelley, T. D., & Ghirardelli, T.G. (2006). Awareness yet underestimation of distractors in feature searches. In R. Sun (Ed.) *Proceedings of the 28th Annual Conference of the Cognitive Science Society* (pp. 1086-1091). Mahwah, NJ: Erlbaum.

Chandrasekaran, B., Josephson, J.J., Bannerjee, B., Kurup, U., & Winker, R. (2002). Diagrammatic reasoning in support of situation understanding and planning. *Proceedings of the Army Science Conference*, Orlando, FL.

Fick, C. S., & Byrne, M. D. (2003). Capture of visual attention by abrupt onsets: A model of contingent orienting. In F. Detje, D. Doerner, & H. Schaub (Eds.) *Proceedings of the Fifth International Conference on Cognitive Modeling* (pp. 81-86). Bamberg, Germany: Universitas-Verlag Bamberg.

# Meter based omission of function words in MOSAIC

**Daniel Freudenthal (D.Freudenthal@liv.ac.uk),**
**Julian Pine (Julian.Pine@liv.ac.uk)**
School of Psychology, University of Liverpool

**Fernand Gobet (Fernand.Gobet@Brunel.ac.uk)**
School of Social Sciences, Brunel University

## Abstract

MOSAIC (Model of Syntax Acquisition in Children) is augmented with a new mechanism that allows for the omission of unstressed function words based on the prosodic structure of the utterance in which they occur. The mechanism allows MOSAIC to omit elements from multiple locations in a target utterance, which it was previously unable to do. It is shown that, although the new mechanism results in Optional Infinitive errors when run on children's input, it is insufficient to simulate the high rate OI errors in children's speech unless combined with MOSAIC's edge-first learning mechanism. It is also shown that the addition of the new mechanism does not adversely affect MOSAIC's fit to the Optional Infinitive phenomenon. The mechanism does, however, make MOSAIC's output more child-like, both in terms of the range of utterances it can simulate, and the level and type of determiner omission that the model displays.

**Keywords:** MOSAIC, Syntax Acquisition, Optional Infinitives, Determiner Omission.

## Introduction

Child speech differs from adult speech in a number of ways. Apart from the average child utterance being noticeably shorter than adult utterances, child speech often lacks inflection where this is appropriate in the adult language. It is also rather telegraphic (i.e., is marked by the relative absence of function words). These characteristics are illustrated in utterances (1) and (2), which are plausible child utterances.

(1) He go home.
(2) I want cookie.

The lack of inflection in child speech has been the subject of considerable Nativist theorizing in recent years. Early theories suggested that utterances like (1) reflect the omission of an inflectional morpheme (-s) from a finite form. More recent theories (Wexler, 1998), however, claim, on the basis of cross-linguistic data, that such utterances actually reflect the use of a non-finite form (the infinitive) in place of a finite form (in this case the 3rd singular present tense). Following Wexler's work utterances like (1) have become known as *Optional Infinitive* (OI) errors. Wexler proposes that children have, from a very early age, correctly set all the inflectional and phrase structure parameters for their language, but are subject to a 'Unique Checking Constraint' which results in them optionally producing infinitives in contexts where a finite form is required. As a result of maturation, children will provide the correct, inflected form increasingly often as they get older, leading to a decrease in OI errors.

Alternative accounts claim that OI errors can be understood in terms of input-driven learning mechanisms without the need to assume innate knowledge. In particular, it is claimed that OI errors can be explained as compound (auxiliary/modal + infinitive) constructions with a missing modal or auxiliary (Ingram & Thompson, 1996). Thus, an utterance such as *he go home* might result from omitting the modal *will* from *he will go home*. According to these accounts, OI errors disappear as children's utterances become longer and missing modals and auxiliaries are realized more and more often.

MOSAIC is a computational model that implements the view of OI errors as truncated compound constructions. Freudenthal et al. (2006, 2007) have shown that the rates at which children produce OI errors can be closely simulated through an input-driven learning mechanism that produces partial utterances. Freudenthal et al. were able to show that MOSAIC provides a close quantitative fit to the OI data from four different languages: English, Dutch, German and Spanish. They were also able to trace the differential rates with which children produce OI errors in these languages back to characteristics of the input from these languages: the frequency of compound constructions and the position of the infinitive in compound constructions.

The particular mechanism used by Freudenthal et al., however, suffers from some weaknesses as the model only produces utterance-final phrases. That is, the model learns the input it receives by building up its representation from the right edge of the utterance. The OI errors with third singular subjects that this model produces are largely learned from questions (e.g. (*Can*) *he go?*). Since children produce OI errors as declaratives, it seems somewhat implausible they should learn such constructions from interrogative contexts. A further problem with the simulations reported by Freudenthal et al. is that child language is far more telegraphic than MOSAIC's output. That is, children will often produce utterances with many omitted constituents (e.g., *Play train*). Since such constructions do not occur (as utterance-final phrases) in the input, they cannot be produced by MOSAIC.

Freudenthal et al. (2005a) report preliminary simulations with a version of MOSAIC that alleviates this problem. This version learns from the left as well as right edge of an utterance and associates sentence-initial and sentence-final phrases. Given an utterance like *He wants to go to bed* the model is capable of associating the phrase *go to bed* with the sentence-initial word *he* resulting in the OI error *he go to bed*. This version of MOSAIC, however, is still unable to produce certain structures that children frequently produce. In particular, children often appear to omit material from multiple locations in an utterance. Thus, an utterance like (3) appears to involve the omission of both a modal or auxiliary and an article from an utterance like *he can go to the shops*.

(3) He go to shops

Since MOSAIC is capable of omitting only one sentence-internal phrase from an utterance it cannot produce an utterance like (3). Modifying MOSAIC so that it is able to produce such utterances will therefore greatly increase its credibility as a model of children's early multi-word speech.

## Omission Errors in Child Speech

It has long been recognised that omission errors are an important characteristic of child speech (Brown, 1973). Moreover, it is clear that children can make multiple errors of omission within the same sentence. Such errors have often been interpreted as resulting from performance limitations in production (Bloom, 1990; Valian, 1991). According to this view, the child is thought to have full competence (a correct underlying representation), but some elements of this representation fail to surface due to a processing bottleneck in production. In the words of Bloom, this kind of analysis is '…one way to reconcile a Nativist theory of language acquisition with the fact that most of young children's sentences are less than three words long…' (Bloom, 1990, p. 492).

An elegant performance limitations account of the pattern of omission errors in child speech is provided by Gerken (1991, 1996). Gerken's account focuses on the prosodic structure of the target utterance, in particular the occurrence of an element with respect to metrical feet. The metrical foot is a basic prosodic unit, which is described by the nature and number of syllables it contains. Gerken's account focuses on the position of unstressed syllables relative to trochaic feet (which have a strong-weak stress pattern). The majority of English (di-syllabic) words are trochaic in nature: primary stress is placed on the first syllable (e.g. PAper, TAble). Gerken argues that children have a preference for trochaic meter to the extent that unstressed syllables that are not part of a trochaic foot are more likely to be omitted. Thus, children are likely to omit the first (unstressed) syllable from *banana*, resulting in *nana*. The omission of unstressed (or weak) syllables that are not part of a trochaic foot also goes some way towards explaining the omission of elements from sentential contexts. Thus, Gerken (1996) shows that children are more likely to omit the object article *the* from sentence (4b) (where it is unfooted), than from sentence (4a) where it is part of a trochaic foot (An asterisk denotes an unfooted element, S and W stand for Strong and Weak. Dashes connect items that combine to form a foot).

```
(4) a. he KICKS the PIG
        *      S-----w   S(-w)

(4) b. he CATCHes the PIG
        *        S----w  *    S(-w)
```

Gerken (1991) also explains the finding that children are more likely to omit pronominal subjects than objects in terms of the stress pattern. Unstressed sentence-initial subjects are likely to be omitted as they are unfooted. Sentence-internal objects are less likely to be omitted as they can be part of a trochaic foot. Further support for Gerken's account comes from recent work by Demuth et al. (in press), who show that children are less likely to omit determiners from footed than from unfooted contexts.

Gerken's account is appealing, as it provides a unified explanation of function word omission in child speech that is largely independent of grammatical class. A mechanism based on this account could therefore be readily combined with MOSAIC's input-driven learning mechanism (which does not assume knowledge of grammatical categories) to simulate the pattern of sentence-internal omission in children's speech. However, since within Gerken's account, modals, like other function words, can be unfooted and therefore omitted from modal + verb structures, it is also possible that a prosody-based omission mechanism may itself be sufficient to explain the OI phenomenon.

The aim of this paper is therefore to investigate the utility of Gerken's metrical template account as a means of increasing the levels of omission that MOSAIC displays, while at the same time considering the possibility that a prosody-based omission mechanism might be sufficient to simulate the level of OI errors in children's output. To this end, prosodic structure was assigned to MOSAIC's output, and unstressed items were probabilistically deleted from the output based on their location relative to trochaic feet. As suggested by Gerken, this mechanism was implemented as a limitation in production[1]. Thus, MOSAIC's learning mechanism (association of sentence-initial and sentence-final phrases) remained unaltered and omission of unstressed elements only occurred in production. Output generated in this way was then compared with output generated by applying the prosody-based omission mechanism to the input samples on which the model was trained. This allowed us to establish whether MOSAIC's learning mechanism was necessary to simulate the rate of OI errors in children's speech.

## The Simulations

The simulations were conducted using the version of MOSAIC described in Freudenthal et al. (2005a) augmented with the chunking mechanism described in Freudenthal et

[1] Clearly this does not address the question of how this bias is acquired. However, given that, at present, MOSAIC's learning mechanism operates at the level of the word rather than the syllable, this question is currently beyond the scope of the model.

al. (2005b). MOSAIC learns from realistic input (child-directed speech) and combines a strong utterance-final bias (recency effect) with a smaller primacy effect. MOSAIC's basic learning mechanism slowly builds up a representation of the utterances it is shown by starting at the right edge of the utterance and slowly working its way to the beginning of the utterance. This mechanism is complemented by a (slower) learning mechanism that builds up its representation of the utterance by starting at the left edge of the utterance, and slowly working its way to the end of the utterance. MOSAIC associates these utterance-final and utterance-initial phrases and is therefore capable of producing utterances with missing sentence-internal phrases. This is illustrated in Fig. 1. Since the utterance-final phrases MOSAIC learns tend to be longer than the utterance-initial phrases (as utterance-final learning is faster) the omitted phrases tend to be located near the left edge of the utterance.

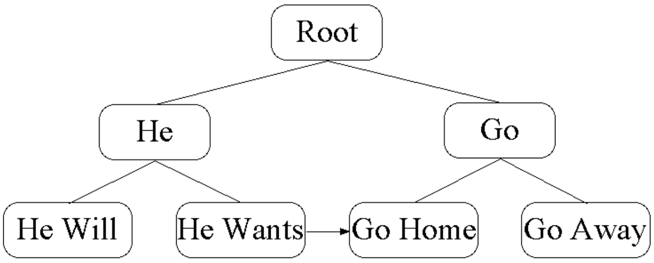


Figure 1: A partial MOSAIC network. The sentence-initial phrase *he wants*, and the sentence-final phrase *go home* have been associated, allowing the model to produce the utterance *He wants go home*.

Output is generated from MOSAIC by traversing all the branches in the model and outputting the (utterance-final) phrases they encode. Where these phrases have been associated with utterance-initial phrases a concatenation of these phrases is also produced. MOSAIC can produce output with an increasing Mean Length of Utterance (MLU), thereby simulating developmental change. Learning in MOSAIC is relatively slow, and the input is shown to the model several times. With every exposure to the input MOSAIC represents more and longer phrases that were present in the input. Output is generated from the model after each exposure to the input, which results in output files of increasing MLU. For the present simulations, models were run using the maternal speech directed at two English children (Anne and Becky). The child-directed speech for these children consists of ~33,000 and ~25,000 utterances. Where relevant the output from the model was compared to the actual speech produced by Anne and Becky.

### Determining the Stress Pattern

The input that MOSAIC learns from is transcribed in an orthographic format that does not include any prosodic information. Likewise, the output from MOSAIC consists of simple text files that lack prosodic information. In order to probabilistically delete unstressed elements the stress pattern for an output utterance thus needs to be assigned. This was done on a word-by-word basis using the stress pattern detailed in the dictionary entry for the individual words. The Unilex dictionary (Fitt & Isard, 1999) was used for this purpose. The Unilex dictionary contains some 100,000 lemmas and details their phonetic form, syllabification and stress pattern. For all utterances in MOSAIC's output, the stress pattern was determined by concatenating the stress patterns for the individual words[2]. Mono-syllabic function words (articles, determiners, pronouns etc.) as well as modals and auxiliary verbs (including the copula) were assigned weak (no) stress. All content words were considered stressed. After the stress pattern had been determined it was decided which unstressed elements were not part of a trochaic foot (i.e. were unfooted). This was done in the following manner:

1. All elements preceding the first stressed syllable in an utterance were deemed unfooted.
2. Every stressed syllable was considered the start of a new foot.
3. An unstressed element that was preceded by a stressed syllable was considered part of a trochaic foot.
4. An unstressed element that was preceded by an unstressed syllable was deemed unfooted.

This procedure results in utterances (4a) and (4b) being assigned the indicated stress pattern. In both (4a) and (4b) the subject *he* is unfooted as it precedes the first stressed syllable. The object article *the* in utterance 4a is part of a trochaic foot as it is preceded by the stressed syllable *kicks*. The article in 4b is unfooted as it is preceded by the unstressed syllable *–es*. A further example is given in (5).

```
(5) a. he can GO to the SHOPS
        *    *  S—w  *   S(-w)

(5) b. PETE can GO to the SHOPS
          S-------w  S—w  *     S(-w)
```

Once the stress pattern for an utterance was determined unstressed (mono-syllabic) words were probabilistically deleted from the utterance. The asymmetry in the omission of footed and unfooted items was modelled by setting the probability of deleting an unstressed item to different values for footed and unfooted items.

## Results

[2] In instances where a word had no entry in the dictionary, no stress pattern was applied to the utterance, and no omission from this utterance was possible. Such utterances were maintained in the analyses presented as their omission affected the MLU distribution, which precluded MLU matching across simulations.

As was mentioned earlier, the omission of unstressed elements can lead to modal omission, and thus result in OI errors. This raises the possibility that the omission mechanism itself may be sufficient to explain the OI phenomenon. This possibility was investigated by running the omission mechanism on the input files (maternal speech) for Anne and Becky, and comparing the rates of OI errors as well as simple and compound finites in the resulting output with the child data at around MLU 2.1. The results of this analysis were compared to those obtained from MOSAIC models with and without omission. This allowed us to compare the performance of the omission mechanism with the learning mechanism of MOSAIC. Comparing the performance of MOSAIC with and without omission allowed us to establish if the omission mechanism had any effects (positive or negative) on MOSAIC's output.

Running the omission mechanism on the maternal speech resulted in utterances that were considerably longer than the child speech they were compared against. For this reason, the rates of OI errors were also determined for the subset of utterances that were not longer than three words. This resulted in output files with an MLU of approximately 2.1. The results of the analyses on short and long utterances are presented in Fig. 2. These results were obtained by setting the omission probability to 0.5 for unfooted items, and 0.1 for footed items. The omission mechanism was also run with probabilities of 0.8 and 0.2 respectively. This gave very similar results.

As can be seen in Fig. 2, the omission mechanism did result in the production of OI errors, in particular when the analysis was restricted to short utterances (0.19 for Anne's input and 0.12 for Becky's input). These proportions are higher than those that occur in the maternal speech directed at these children (~ 5%). However, they are considerably lower than the rates of OI errors that the English children display early in development.

These results suggest that omission of weak elements can account for some of children's OI errors, particularly when combined with an additional mechanism that restricts the length of the utterances children produce. The mechanism implemented for these analyses (only selecting short utterances) however, is not sufficient to produce OI errors at rates comparable to the rates that actual children produce.

Fig. 2a. Data and input analysis for Anne.

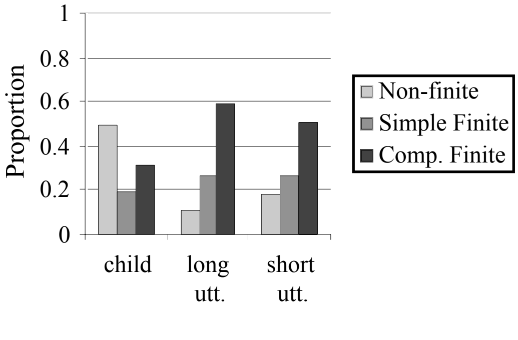


Fig 2b. Data and input analysis for Becky.

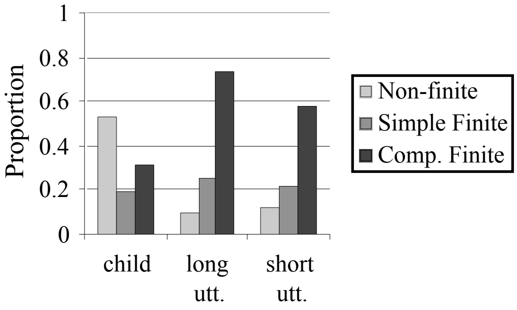


Fig. 2: Rates of OI errors, simple and compound finites for children and maternal speech with omission.

The next set of analyses was aimed at establishing if MOSAIC's mechanism for restricting the length of utterances (omission of sentence-internal material through the concatenation of utterance-initial and utterance-final phrases) is more successful in producing OI errors at rates comparable to English children. For these simulations standard MOSAIC models were run and output at an MLU of 2.1 was generated.

Fig. 3a. Data and simulations for Anne.

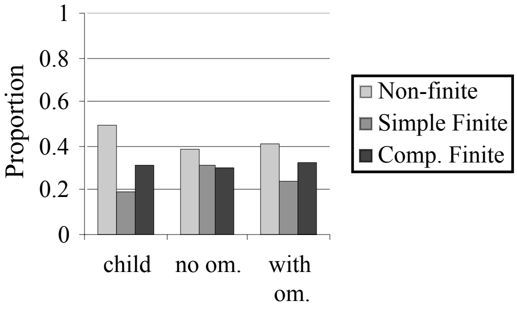


Fig 3b. Data and Simulations for Becky.

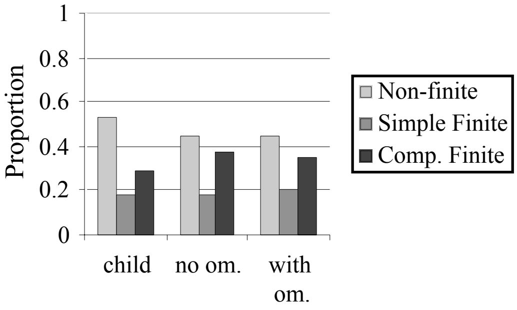


Fig. 3. Rates of OI errors, simple and compound finites for children and simulations with and without omission.

Next, the omission mechanism was run on MOSAIC's output. Where necessary, output from slightly more mature models was selected in order to match the MLUs in the simulations without omission (omission of words from utterances reduces the MLU for the output). The omission mechanism was run with an omission probability of 0.5 for unfooted syllables and 0.1 for footed syllables. Fig. 3 gives the results for these analyses. The rates of OI errors in MOSAIC's output clearly provide a closer match to the children's data than the omission of weak elements from complete utterances (both short and long ones). The prosody-based omission mechanism had very little effect on the model's fit to the data.

## Omissions errors in MOSAIC's output

Having established that the addition of the prosody-based omission mechanism does not adversely affect MOSAIC's fit to the OI data, we can now assess whether the model produces any utterances that it previously could not. The examples in Table 1 show that this is the case.

Table 1: Examples of utterances with multiple sentence-internal omissions in MOSAIC's output.

| |
|---|
| He go to shop |
| He sit on chair |
| She give you kiss |
| She go hospital |
| That going sleep |

All the examples in Table 1 constitute OI errors where the modal has been omitted through the association of an utterance-final and utterance-initial phrase. Additionally, the prosody-based omission mechanism has resulted in unstressed words like *the*, *a*, and *to* being omitted. Thus, the phrase *He go to shop* may have been learned from the input utterance *He wants to go to the shop.* During learning, MOSAIC has associated the utterance-final phrase *go to the shop* with the utterance-initial phrase *he.* The omission mechanism has resulted in the unstressed and unfooted determiner *the* being omitted. In four out of the five examples an unfooted item has been omitted. In the phrase *She go hospital* the particle *to* which forms a trochaic foot with *go* is missing.

One further measure of how well the model's output approximates children's speech relates to the levels of determiner omission. Demuth et al. (in press) provide an analysis of 5 English children which shows that 4 of these children omit determiners from unfooted contexts at higher rates than from footed contexts. In order to determine how well MOSAIC approximates this pattern we assessed the levels of determiner omission from footed and unfooted contexts in the simulations as well as in the actual speech of Anne and Becky at different MLU points. This was done in the following manner. First, a list of nouns that are predominantly used with a determiner was compiled by searching the maternal speech for nouns that are used with a determiner in at least 75% of the cases. Next, the child speech and model output were searched for utterances containing one of these nouns. Allowing for the occurrence of common adjectives, it was then decided if a determiner (*a*, *an* or *the*) was provided, and whether the context was footed or unfooted. Utterances that contained other determiners (e.g. *my*) were disregarded. Note that the assignment of the metrical pattern was done in an identical (automated) manner for the child speech and model output. That is, all function words were considered to be unstressed. For all other items, the stress pattern given by the Unilex dictionary was used. Tables 2a and b compare the child data with MOSAIC's output before the omission mechanism was run. Apart from Anne's earliest stage, the children omit determiners from unfooted contexts at higher rates than from footed contexts. This is not the case for the simulations. Provision levels in footed contexts exceed those in unfooted contexts in just 2 of the 6 simulations (by a maximum of 8 percentage points), while provision levels in unfooted contexts are higher (by 14 percentage points) in one of the simulations.

Table 2a: Determiner provision in footed and unfooted contexts for Anne and Anne's model without omission.

| | Anne | | Anne's model | |
|---|---|---|---|---|
| MLU | Footed | Unfooted | Footed | Unfooted |
| 2.2 | .13 | .14 | .50 | .42 |
| 3.0 | .70 | .47 | .65 | .66 |
| 3.5 | .80 | .62 | .76 | .76 |

Table 2b: Determiner provision in footed and unfooted contexts for Becky and Becky's model without omission

| | Becky | | Becky's model | |
|---|---|---|---|---|
| MLU | Footed | Unfooted | Footed | Unfooted |
| 2.2 | .53 | .20 | .31 | .45 |
| 3.0 | .79 | .60 | .66 | .66 |
| 3.7 | .86 | .60 | .78 | .72 |

Table 2c: Determiner provision in footed and unfooted contexts for Anne and Becky's model with omission.

| | Anne's model | | Becky's model | |
|---|---|---|---|---|
| MLU | Footed | Unfooted | Footed | Unfooted |
| 2.2 | .43 | .29 | .29 | .28 |
| 3.0 | .59 | .40 | .65 | .39 |
| 3.5 | .65 | .40 | .69 | .41 |

Table 2c presents the results of this analysis on MOSAIC's output after the omission mechanism was run. These results look much improved over the simulations without omission. Apart from the early simulation for Becky, determiner omission clearly occurs more frequently in unfooted contexts. The developmental pattern (increase in provision rates) in the models is not as pronounced as it is in the children. The simulations, however, were run with fixed omission probabilities (0.5 for unfooted items and 0.1 for unfooted items) for all developmental stages. A simple solution to this problem would be to vary these probabilities with developmental stage.

## Conclusions

This paper set out to establish the value of a prosody-based omission mechanism aimed at making the output of MOSAIC more child-like. One particular aim was to allow MOSAIC to produce utterances with multiple sentence-internal omissions. The prosody-based omission mechanism clearly increases the range of utterances that MOSAIC can produce and thus makes the model's output more child-like and increases its credibility as a model of childen's early multi-word speech. It is also apparent that the model without the omission mechanism does not simulate the differential rates of determiner omission from footed and unfooted contexts. The addition of the omission mechanism rectifies this divergence between the model output and child speech, and thereby increases the child-likeness of the model's output on this measure as well.

Obviously, it is not very surprising that the mechanism produces these results, as this is what it has been designed to do. However, future, (cross-linguistic) work may provide a more stringent test of the mechanism. In particular, the mechanism predicts that the pattern of omission of function words will be different for languages that predominantly display iambic feet (e.g. French). Some evidence for this claim is provided by Tremblay & Demuth (in press).

It could be argued that the present mechanism is somewhat crude in that all function words are considered unstressed. The mechanism could however, easily be made more sophisticated by specifying stress patterns for different types of (frequent) constructions. Some possible refinements have already become apparent as a result of the simulations reported here. Inspection of the pattern of determiner omission in the two children suggests that omission levels after pronouns with a contracted copula (e.g. *that's*) are lower than in the model's output. Such an effect could easily be incorporated in the present mechanism on the plausible assumption that a pronoun with a contracted copula receives higher stress (and therefore forms a trochaic foot with a determiner that follows it) than a bare pronoun. Another possible refinement would be to specify different stress patterns for interrogative and declarative utterances.

The analyses reported here also have theoretical implications. The simulations which determined the levels of OI errors when the omission mechanism was run on the input showed that prosody-based omission alone is not sufficient to explain the OI phenomenon even when restricting the analysis to short utterances. Thus, an (unspecified) learning mechanism which produces short complete utterances (one possible instantiation of full competence) coupled with prosody-based omission does not provide an adequate fit to the child data. In order to obtain such a fit, omission needs to be co-determined by other factors. The simulations reported here suggest that a learning mechanism that is subject to a primacy and recency effect is such a factor.

## Acknowledgements

This research was funded by the Economic and Social Research Council under grant number RES000230211.

## References

Bloom, P. (1990). Subjectless sentences in child language. *Linguistic Inquiry*, *21*, *491-504*.

Brown, R. (1973). A first language: The early stages. London: George Allen & Unwin Ltd.

Demuth, K., McCullough, E. & Adamo, M. (in press). The prosodic (re)organization of determiners. To appear in *Proceedings of the 31st Boston University Conference on Language Development*.

Fitt, S & Isard, S. (1999). Synthesis of regional English using a keyword lexicon. *Proceedings: Eurospeech 99*, Vol. 2, pp. 823-6.

Freudenthal, D., Pine, J.M., Aguado-Orea, J. & Gobet, F. (2007). Modelling the developmental patterning of finiteness marking in English, Dutch, German and Spanish using MOSAIC. *Cognitive Science*, *31*, *311-341*

Freudenthal, D., Pine, J.M. & Gobet, F. (2006). Modelling the development of children's use of optional infinitives in English and Dutch using MOSAIC. *Cognitive Science*, *30*, *277-310*.

Freudenthal, D., Pine, J.M. & Gobet, F. (2005a). Simulating optional infinitive errors through the omission of sentence-internal elements. In B.G. Bara, L. Barsalou & M. Bucciarelli (Eds.), *Proceedings of the 27th Annual Conference of the Cognitive Science Society.* Mahwah NJ: LEA.

Freudenthal, D., Pine, J.M. & Gobet, F. (2005b). On the resolution of ambiguities in the extraction of syntactic categories through chunking. *Cognitive Systems Research*, *6*, *17-25*.

Gerken, L. A. (1991). The metrical basis for children's subjectless sentences. *Journal of Memory and Language*, *30*, *431-451*.

Gerken, L. A. (1996). Prosodic structure in young children's language production. *Language*, *72*, *683-712*.

Ingram, D & Thompson, W. (1996). Early syntactic acquisition in German: evidence for the modal hypothesis. *Language*, *72*, *97-120*.

Tremblay, A. & Demuth, K. (in press). Prosodic licensing of determiners in children's early French. In A. Belikova, L. Meroni and M. Umeda (Eds.). *Proceedings of the Conference on Generative Approaches to Language Acquisition*. Somerville, MA: Cascadilla Press.

Valian, V. (1991). Syntactic subjects in the early speech of American and Italian children. *Cognition*, 40, 21-81.

Wexler, K. (1998). Very early parameter setting and the unique checking constraint: a new explanation of the optional infinitive stage. *Lingua*, *106*, *23-79*.

# Storm: A Framework for Biologically-Inspired Cognitive Architecture Research

**Douglas Pearson (douglas.pearson@threepenny.net)**
ThreePenny Software, 4649 Eastern Ave. N.
Seattle, WA 98103 USA

**Nicholas A. Gorski (ngorski@umich.edu)**
**Richard L. Lewis (rickl@umich.edu)**
**John E. Laird (laird@umich.edu)**
University of Michigan
Ann Arbor, MI 48109 USA

**Abstract**

We have developed a software framework called Storm to aid the development of cognitive architectures based on the structure and function of the brain. The goals of the framework are to make it both easy and fast to develop and experiment with alternative architectures and components of architectures. In addition, the framework supports explicitly mapping its components to structures in the brain. We demonstrate a working implementation of the framework, where we have developed a simple model of skill learning and memory management in a simple 2D grid world.

## Introduction

In cognitive modeling, there is a divide between models that attempt to capture the details of neural activity and those that attempt to model complex overt behavior. Models of complex behavior often use combinations of symbolic and non-symbolic representations of knowledge in cognitive architectures. Detailed models at the neural level posit direct mappings to structures and processes of neural systems in the brain. The achievements of neural models to date have been impressive (Munakata & Johnson, 2006); but it is very difficult to create models of the interactions of sufficient brain systems for anything approaching a complex task (e.g., see Simen et al, 2004 for a recent attempt). Conversely, the cognitive architecture approach has been successful at modeling a wide variety of complex tasks (e.g., see the models reported in Gray, 2007) but the mapping of the components of those models to structures and processes in the brain often remains unclear – although Anderson's recent work has demonstrated that it is possible to map some structures in ACT-R to specific brain regions (Anderson, 2007).

We propose an alternative between high-level cognitive architectures and low-level neural models. Our approach is to create architectures composed of models of brain structures and their interconnections (at possibly multiple levels of abstraction) – a brain-based architecture capable of cognitive behavior.

In order to pursue this approach, it behooves us to take a step back and not jump immediately into the construction of a specific architecture. Instead, the first step, and the subject of this paper, is to develop a software framework in which such models can be easily developed, tested, evaluated, and extended. More specifically, we believe a useful framework will support:

(1) Rapid prototyping of architectures composed of a heterogeneous collection of interacting components operating in parallel, with their own possibly unique time scales, processes and representations.

(2) Easy and efficient simulation of the dynamics of such architectures.

(3) Explicit and flexible mappings of architecture-to-brain structure, and easy exploration of the implications of such mappings for predictions of brain activity. The framework should also make it easy to exploit existing detailed databases on brain structure and brain connectivity (Alexander, Arbid & Weitzenfeld, 1999).

(4) Maximal flexibility in programming languages, operating systems, and parallel computation.

Furthermore, our goal is not only to develop a tool to aid our own research, but a tool that others will use for their own explorations, thereby facilitating the sharing of components between research groups.

In this paper, we describe the Storm framework, a software infrastructure intended to realize the above goals. Using the initial implementation of Storm, we developed a simple architecture that includes action selection, reinforcement learning, and simple long-term and short-term memories. This architecture is not (yet) meant to be a faithful model of brain structures, but is meant to demonstrate the capabilities of Storm.

Various features and motivations for Storm have precedent in the cognitive modeling and neural network modeling communities. The explicit goals of Storm are perhaps most closely aligned with NSL (Neural Simulation Language; Weitzenfeld, Arbib & Alexander, 2002). There are key differences, however. NSL defines a new object-oriented language that must be used for creating models. In contrast, Storm allows users to develop architectural components in standard computer languages (C++, Java), while providing support for communication between modules and scheduling model execution. In addition, Storm provides facilities for explicitly mapping model components to brain structures. Storm also differs from neural network toolkit approaches such as Leabra (O'Reilly & Munakata, 2000) and Eliasmith and Anderson (2002) because it has no a priori bias to specific models of the brain or neurons. Finally, we note that the goal of providing an appropriate abstraction layer for building event-driven simulations is also adopted in the implementations of some existing cognitive architectures, including ACT-R (Bothell, 2004) and Epic (Kieras & Meyer, 1997), though neither of these architectures embraces the general framework goals described above.

# The Storm Framework

## Defining an architecture

The framework must have a way of representing architectural components, how they interact and the computations they perform. In the Storm framework, we decompose an architecture into two types of components: *function modules* and *state variables.* Function modules perform processing while state variables hold persistent structures and provide communication between function modules. This approach naturally reflects a simple dynamical systems view of brain architecture, in which the union of state variables represents the current state of the system, and the function modules represent the dynamic relationships among those state variables.

Function modules receive inputs from a set of state variables and generate outputs to one or more state variables. Figure 1 shows a function connectivity graph for a simple architecture with two function modules (the M1 & M2) and four state variables (the circles). In this case, the state variables A, B, and C are the inputs for functional module M1, which generates outputs for A and D. Module M2 receives input from D and generates output for C. The graph is not explicitly represented in the framework as a separate data structure, but is implicitly defined by the inputs and outputs of the modules.

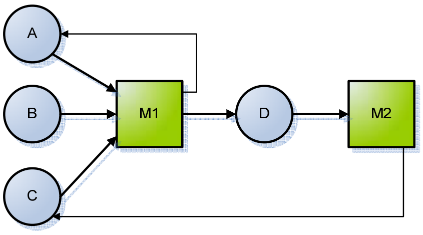


Figure 1: Functional Connectivity Graph

An architecture's decomposition into state variables and function modules represents theoretical commitments about brain architecture. It is possible to build both highly interactive and highly encapsulated systems and subsystems using the framework. The framework itself does not impose theoretical constraint, and the use of the term *module* here should not be taken to imply a commitment to, for example, Fodorian (1983) modules. Rather, a framework module is the software component that permits the specification of the dynamic relationships among state variables. The extent to which a given set of state variable and function modules realizes an encapsulated module or a fully interactive subsystem depends on the details of the connectivity between state variables. (And as we see below, the architecture-to-brain mapping need not even imply strict localization of function).

## Simulating the dynamics of an architecture

All of the components are run asynchronously with the architecture developer having complete control over when components start executing and how long they execute. The developer can specify independently for each module:

- When a module initiates execution. Examples include: periodically (such as every 50 msec), whenever inputs change, or even some delay after inputs change.
- The length of simulated time it takes for a module to execute and for data to travel between modules. This can be a fixed number, such as 10 msec or can be dependent on input parameters, such as (1 msec * number of changed inputs).

The Storm framework automatically coordinates the execution of the components (function modules and state variables), following the temporal constraints declared for each of the components, freeing the developer from writing code that schedules the execution of the modules. Thus, when an architecture runs, the framework automatically schedules all of the components, initiates their execution and provides a complete trace of the temporal activity of every module and state variable, including behavior in an external task environment. All of the scheduling is based on simulated time, which depending on the calculations performed in the modules could be much slower, or possibly even faster than real time.

This layer of abstraction thus allows the modeler to focus on the control structure of the *brain architecture* rather than the control structure of the *simulation*. Importantly, the function modules may be flexibly implemented via arbitrary code in the underlying target language, but the modules do not interact by calling each other directly, and the modeler need not worry about how to manage their parallel execution. (This abstraction away from simulation control structure is a common property of simulation environments long used in other areas of science.)

## Experimenting with an architecture

Storm's design makes it easy to quickly add or replace modules because all of the information about a module is local to that module. (A critical aspect of this locality is the distributed nature of the simulation control, above). This makes it possible for research groups to share modules as well as to have multiple implementations of a given module. For some experiments, it might be desirable to have a coarse, but efficient implementation of a module, or replace a small network of modules and variables with a single module that is extremely fast, but only approximates a given computation. Moreover, during early development a coarse model might be all that is available. In others cases, a very accurate, but slow implementation of a module can be used when detailed behavior is critical.

The individual function modules and state variables are created by the architecture developer using a standard programming language. The framework currently supports C++ and Java, but will soon support MATLAB and R. This makes integrating existing code simpler and allows a module developer to select a language that is particularly well suited to the behavior they wish to model. The framework is agnostic as to which language is used to specify modules and one could imagine supporting the use of neural modeling systems.

In order to provide maximum flexibility and efficiency, the framework is designed to run on multiple operating systems (Windows, OSX, Linux) and has underlying support for parallel execution, which supports multi-core computers and will support clusters. This is transparent to users, determined at runtime based on the available resources. The framework itself is lightweight and requires minimal computational resources.

## Mapping onto the Brain

In order to compare the processing in the architecture with what is known about the brain, the framework must support the explicit representation of processing and communication in the brain. In Storm, the *brain mapping graph* formally declares assumptions about the physical substrates of the state variable, and by implication, the function modules. State variables are mapped to different physical structures and regions within the brain, which are then mapped to physical coordinates in a normalized brain coordinate system. In the current design only state variables are explicitly mapped to brain regions. Function modules are thereby implicitly mapped to regions based on the state variables they use (Figure 2).

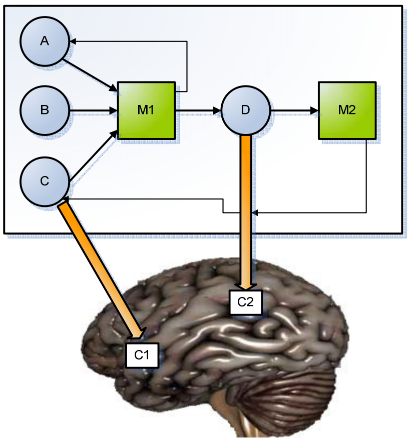


Figure 2: Brain Mapping Graph

This mapping scheme is quite flexible because there are no restrictions on what state variables might represent, and there are no restrictions on the target vocabulary of brain structures. For example, state variables might represent synaptic weights that could be changing over both the short and long-term, and such state variables might correspond to long-distance synapses in the brain that connect distal cortical areas. Or, a state variable might represent a quantity of some neurotransmitter that is fairly localized in space, or a vector of activation values representing patterns of firing activity in a particular part of the hippocampal formation, or an abstract short-term control symbol thought to be distributed over a broad area of prefrontal cortex.

Thus, this mapping scheme does *not* enforce a simple one-to-one mapping of computational function onto local structure. Rather, the mapping explicitly identifies the physical substrates of the state variables, and these physical substrates may be at any level of spatial resolution. The mapping of function to structure is then implicit in the mappings that function modules inherit from their state variables.

The functional connectivity graph together with the brain mapping graph imply a brain connectivity graph. That is, the connectivity of the state variables and function modules and their mapping to brain regions implicitly make claims about how the brain regions are connected, which can be tested against known constraints on how brain regions are actually connected. The predicted brain connectivity is derived from the connectivity of the function modules and the mappings from the architectural components to the brain as shown in Figure 3. These structures, together with the simulation provide Storm three important capabilities:

(1) Detecting inconsistencies between known brain connectivity constraints and the architecture.

(2) Predicting the time-course of activity in brain regions. This could support the automatic generation of simulated fMRI, MEG, or EEG for the modeled brain regions.

(3) Changes made to the architecture for functional reasons automatically change the biological predictions as the brain connectivity is derived directly from the functional elements of the architecture.

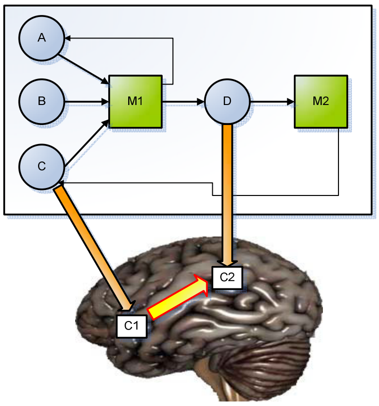


Figure 3: Brain Connectivity Graph

# An Empirical Example in Using Storm

## Example Task Requiring Learning and Memory

In order to demonstrate the Storm framework, we created a simple task that requires learning control knowledge for both internal and external actions. The example task is set in a 5x5 grid-world, shown in Figure 4. The domain contains three special locations, or *boxes,* in fixed positions: boxes A and B are *reward* boxes, while box M is an *information* box. The agent is rewarded with a positive reward when it opens one of the boxes and a negative reward when opening the other. The agent perceives a symbol when it opens the information box; this symbol is correlated with the location of the positive reward box (but does not correlate to any perceived feature of the boxes). An agent that cannot maintain the symbol in an internal memory would be unable to receive the maximum reward in every episode, making the task un-learnable.

The agent can move in the four cardinal directions, and if a box is in its current location, the agent can open the box. The agent perceives its location in the grid and any reward signal, but cannot perceive the labels on the boxes (A or B). If the agent is in the information box square and the box is open, the agent also perceives a symbol. An episode concludes when the agent opens the box containing the positive reward. The location of the rewards is randomized between episodes.

Reward is structured such that a positive reward has magnitude of +10, a negative reward is -10, and on every step that the agent does not open a reward box, it receives -1 reward.

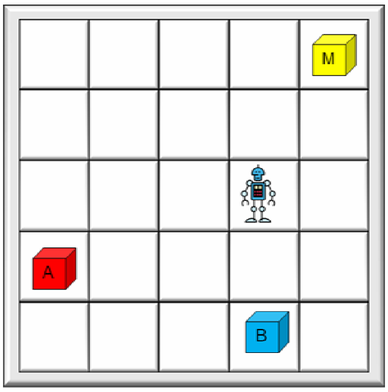


Figure 4: Information Box Task

## An Example Architecture Developed using Storm

In order to help illuminate some of the framework's capabilities, we used Storm to develop a simple architecture capable of supporting an agent that learns to perform in the example task. This architecture combines a simple long-term memory with a basic reinforcement learning mechanism that learns control knowledge for both internal and external actions.

Our example architecture is shown in Figure 5. Function modules in the figure are represented as rectangles, and state variables as ovals. In this model, the environment is represented as a function module (for convenience) which receives a motor action as input and generates sensory information as output. Sensory Input is used by both Long and Short Term memories, which in turn is used by Action Selection to choose an internal Long Term Memory retrieval as well as an external Motor action. The Reinforcement Learning mechanism uses Working Memory, the Internal Reward Signal, and selected actions to adjust the control knowledge used by Action Selection.

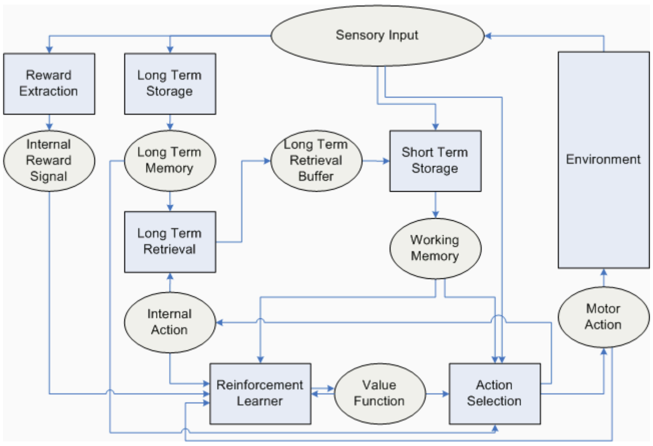


Figure 5: Simple Architecture Implemented in Storm

The details of each function module are described below. The Reward Extraction module reads the explicit reward generated by the environment in Sensory Input and stores it as an Internal Reward Signal. Long Term Storage stores any perceived symbol (i.e. the contents of the information box) to Long Term Memory. Long Term Retrieval retrieves a symbol corresponding to the Internal Action from memory and stores it in the buffer. Short Term Storage reads the agent's location from Sensory Input and the contents of the Long Term Retrieval Buffer and puts the concatenation of both in Working Memory.

The agent decides how to act in the Action Selection module, which uses Working Memory and the Value Function to select its actions (using a decaying epsilon-greedy strategy). The Value Function is a table that associates a pair of internal and motor actions with the contents of working memory and the estimated future reward of applying those actions. The Value Function is adjusted by the Reinforcement Learner with Sarsa (Sutton, 1996) based on input from Working Memory, Internal Reward Signal, and Internal and Motor Actions.

In the example architecture, function modules initiate their processing when their input state variables change, and all take a fixed amount of time to process and create results. During execution, many of the modules will execute in parallel, such as those that depend on Sensory Input. Others execute in sequence because of the dependencies of their input variables on other function modules. This parallelism enables an agent to perform internal and motor actions simultaneously.

An execution trace of the example architecture's function modules is shown in Figure 6, which is generated from the execution logs by a Storm utility. There are four different types of events logged by Storm for a function module.

- *RequestWakeup*: can occur on the time step when an input variable's value changes,
- *Wakeup*: occurs on the time step immediately following a *RequestWakeup* event (unless explicitly delayed),
- *Finished*: occurs on the time step on which the module sets its output variables and completes processing, and
- *Processing*: this is the time that a module is inferred to be processing between *Wakeup* and *Finished* events.

Multiple events that occur on the same time step are plotted as one event (e.g. all Environment events occur on the same step).

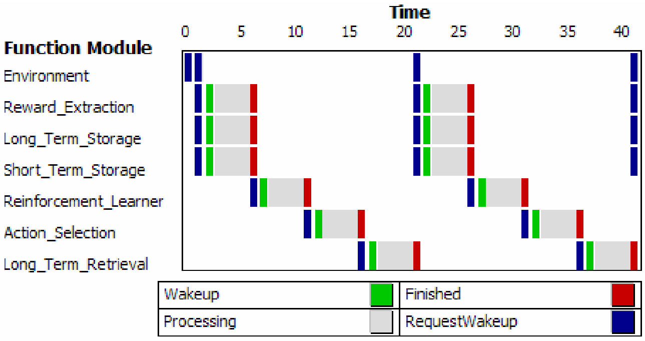


Figure 6: Sample execution trace of the example architecture as generated by a Storm utility.

Processing begins in the Environment module which sets the Sensory Input state variable (see Figure 5). When Sensory Input is set, a *RequestWakeup* event triggers the Reward Extraction, Long Term Storage, and Short Term Storage modules. All three modules then process in parallel, after which they set their respective output variables.

The Reinforcement Learner module next begins processing, as it relies on the Internal Reward Signal set by the Reward Extraction module. Similarly, the Action Selection module relies on the output of the Reinforcement Learner, and the Long Term Retrieval module on Action Selection, which explains the serial behavior seen in Figure 6. This behavior arises from the dependences of the input variables of each module, and is not explicitly timed or engineered. However, the Environment module *is* configured to process periodically, which explains why it does not begin executing at the same time as the Long Term Retrieval module even though inputs for both modules are set by Action Selection.

Although the mapping of state variables to brain regions is an important commitment made in Storm, this example architecture is so simple that we do not hypothesize a mapping. Rather, the purpose of this example is to illustrate the framework's specification and simulation capabilities.

## Results

**Example Architecture** We developed two agents in the architecture to perform the example task, one that automatically retrieves the information symbol from long-term memory (when available) and one that must learn to retrieve it. The performance of the two agents is shown in Figure 7. Asymptotically, the behavior of both agents is the same: the agent moves directly to the information box, opens it, and then simultaneously retrieves the identifying symbol from long-term memory while navigating to the positive reward box, opening it upon reaching its location. The results indicate that learning both control knowledge for an internal action in addition to an external motor action is not significantly more difficult than for an external action alone.

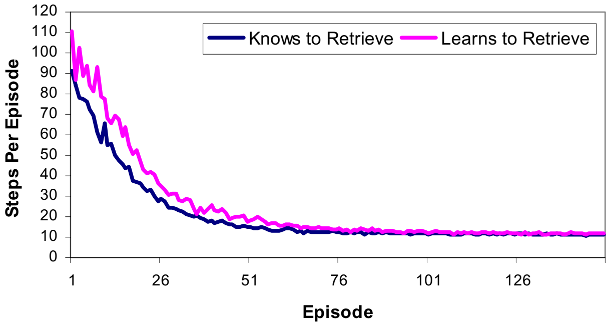


Figure 7: Learning curves for two agents performing on the simple task, average of 225 trials.

**Modified Task and Expanded Working Memory** To study the flexibility of an architecture using the Storm framework, we modified the task so that the agent had to learn to manage long-term memory retrievals:

- Instead of a single motor action to open a box, the agent now has two available actions. When the correct one is used to open the positive reward box, the standard reward is still received. However if the reward box is opened with the other action, a smaller positive reward (+1) is received.
- The information box contains an additional symbol identifying the correct action to use when opening the positive reward box. Both symbols are still automatically stored to long-term memory.

After an agent using the example architecture opens the information box, both symbols are then automatically stored to long-term memory. However, the Long Term Retrieval Buffer (and thus Working Memory) can still only store one retrieved symbol at a time. The agent therefore must learn to recall the two symbols at different times: the symbol identifying the correct box during navigation and the action symbol on the step before it will open the box.

We tested an agent using the example architecture as well as an agent with an expanded working memory that can store the two most recently retrieved symbols in working memory on the modified task. The results for both agents are presented in Figure 8. Although the agent using the architecture modified with an expanded working memory learns more quickly than the agent using the unmodified architecture, these results show that an agent using the unmodified architecture with limited working memory is still able to learn the modified task.

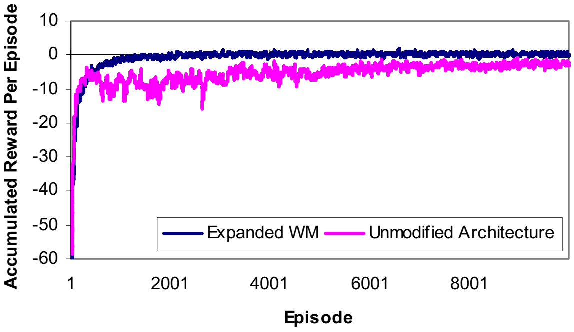


Figure 8: Learning curves for agents performing on the modified task, 25 per. moving avg. of medians for 45 trials.

In order to modify the architecture with an expanded working memory, only the Short Term Storage function module and Working Memory state variable needed to be changed – the rest of the architecture's function modules and state variables remained the same. This demonstrates an advantage to experimenting within the Storm framework: the modularized approach to development leads to architectures that can be modified quickly and easily.

**Architectural Delay** In our example architecture, all function modules took the same constant amount of time to process data (5 units of time as seen in Figure 6). In order to experiment with function modules processing at different time scales, we introduced a delay to the Long Term Retrieval module: with a delay, the module processes for 20 units of time rather than 5. This change has two effects: first, retrieved memories are available two environment steps after the Internal Action is selected; second, retrieved symbols in the buffer persist for two environment steps. Because of these changes, the agent can improve its performance by learning to make a retrieval from Long Term Memory two steps before it gets to the reward box.

The results of two agents, one modified with a delayed retrieval and the other unmodified, in the modified task are shown in Figure 9. While the agent using the unmodified architecture initially learns more quickly, the behaviors are indistinguishable after the 2000th episode.

In order to experiment with delaying Long Term Retrieval in the architecture, our implementation in Storm required only a single line of code to be changed. Storm's mechanism for scheduling the processing of function modules makes changing timing constraints to be a straightforward exercise.

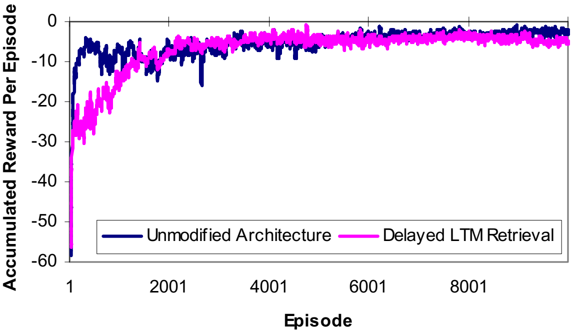


Figure 9: Learning curves for an architecture modified with delayed Long Term Retrieval compared with an unmodified architecture, 25 per. moving avg. of medians for 45 trials.

**Summary of Experiments** The Storm framework has allowed us to experiment with the example architecture in several dimensions, the results of which are not all shown in this paper:

(1) We experimented with two timing conventions: both waking function modules when input variables have been set as well as function modules processing periodically at set intervals. The example architecture implements a hybrid approach and uses both approaches in its modules.

(2) We experimented with the timing of individual modules, delaying their output such that the processing time of various modules overlaps.

(3) We explored reinforcement learning modules implementing a variety of learning algorithms with various parameter settings; switching algorithms is as simple as changing the module used by the framework.

(4) We have simulated environments in C++ function modules and interfaced to external Java environments.

When experimenting along all of these dimensions, the necessary changes to function modules were minor and no changes to the framework were necessary. In contrast, experimenting with existing cognitive architectures to modify the behavior of working memory, long-term memory, or timing constraints can often be difficult and time consuming.

## Discussion

By developing our example architecture using the Storm framework, we have had valuable experiences which begin to shed light on the advantages (and disadvantages) of using a lightweight framework to model brain function.

The Storm framework has minimal overhead so as to not impede the development of a diverse set of functional modules. The framework does, however, strictly enforce that any data shared between function modules must be contained within state variables: designers must be explicit and consistent in the organization of data into state variables.

Modeling the timing of function modules and state variables is an important aspect of the framework and is straightforward to use and experiment with. This allows a designer to focus on implementing behaviors and not be concerned with the implementation of timing constraints.

One possible disadvantage of using the framework is the strict enforcement on the organization of data into state variables. Experimental architectures may not want to make strong commitments to the separation of data; algorithms achieving high-performance may also require a high level of abstraction as realized in function modules and state variables.

In the future we plan to begin testing Storm's ability to scale by building iteratively larger and more complex architectures, as well as developing psychologically plausible models using state variables that map to brain regions and model brain function.

## Acknowledgments

The authors acknowledge the funding support of the DARPA "Biologically Inspired Cognitive Architecture" program under the Air Force Research Laboratory "Extending the Soar Cognitive Architecture" project award number FA8650-05-C-7253.

## References

Alexander, A., Arbid, M. & Weitzenfeld, A. (1999). Web Simulation of Brain Models. *Proc. of the 1999 International Conference on Web-Based Modeling and Simulation*, 29-33. The Society for Computer Simulation International, San Diego, CA.

Anderson, J. R. (2007) *How Can the Human Mind Occur in the Physical Universe?* Oxford University Press.

Bothell, D. (2004). ACT-R 6.0 implementation. http://act-r.psy.cmu.edu/actr6/

Eliasmith, C. Anderson, C. H. (2002). *Neural Engineering: Computation, Representation, and Dynamics in Neurobiological Systems.* MIT Press.

Fodor, J. A. (1983). *Modularity of Mind: An Essay on Faculty Psychology*. Cambridge, Mass.: MIT Press

Gray, W. (2007). *Integrated Models of Cognitive Systems*, Oxford University Press.

Kieras, D. & Meyer, D. E. (1997). An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. Human-Computer Interaction., 12, 391-438.

Munakata, Y., & Johnson, M. H. (Eds.) (2006). *Processes of Change in Brain and Cognitive Development: Attention and Performance XXI.,* Oxford: Oxford University Press.

O'Reilly, R. and Munakata, Y. (2000) *Computational Explorations in Cognitive Neuroscience: Understanding the Mind by Simulating the Brain*, Cambridge, MIT Press

Simen, P., Polk, T. A., Lewis, R. L. & Freedman, E. (2004). A computational account of latency impairments in problem solving by Parkinson's patients. *Proceedings of ICCM 2004*, Pittsburgh.

Sutton, R. S. (1996). Generalization in reinforcement learning: Successful examples using sparse coarse coding. D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo (Eds), *Advances in Neural Information Processing Systems: Proc. of the 1995 Conference*, 1038-1044. MIT Press.

Weitzenfeld, A., Arbib, M., and Alexander, A. (2002), *The Neural Simulation Language: A System for Brain Modeling.* MIT Press.

# Vector Generation of an Explicitly-defined Multidimensional Semantic Space

**Alex Grintsvayg** (grinta@rpi.edu)
**Vladislav D. Veksler** (vekslv@rpi.edu)
**Robert Lindsey** (lindsr@rpi.edu)
**Wayne D. Gray** (grayw@rpi.edu)

Cognitive Science Department, 110 8th Street
Troy, NY 12180 USA

Measures of Semantic Relatedness (MSRs) are a recent breed of computational models of text comprehension. MSRs have been successfully used to model human web browsing behavior (Pirolli & Fu, 2003), language acquisition (Landauer & Dumais, 1997), and text comprehension (Lemaire, Denhiere, Bellissens, & Jhean-Iarose, 2006), among other things. MSRs have also been used in the applied domain for augmented search engine technology (Dumais, 2003), ETS essay grading (Landauer & Dumais, 1997), and many other applications.

The two most common types of measures of semantic relatedness are vector-based MSRs and probabilistic MSRs. Vector-based MSRs are complex, computationally expensive algorithms that represent words as vectors in a multidimensional semantic space. They work fairly well for small corpora, but the large amount of preprocessing they require makes them unusable for very large or dynamic corpora. Probabilistic MSRs are the opposite: simple metrics that can be used on an extremely large corpus. Their only downside is that they cannot compute the similarity between groups of words (something that vector-based MSRs can do easily).

In this paper we are proposing a new MSR that combines the best features of probabilistic and vector-based approaches, while adding flexibility and broadening the range of tasks that MSRs are capable of carrying out. Specifically, this technique allows non-vector-based MSRs to represent words in vector form. This representation gives probabilistic MSRs the ability to measure large multi-word terms without requiring them to perform computationally expensive preprocessing. In addition, the proposed MSR is incremental (allowing the addition of new terms to the corpus without the need for the large-scale recalculations performed by traditional vector-based measures) and has the ability to model domain-specific expertise by explicitly defining the dimensions of the semantic space that it uses. Preliminary results show that the proposed probabilistic-to-vector-based MSR conversion produces a measure that surpasses the performance of the original probabilistic MSR.

## VGEM

In order to convert a probabilistic measure, *S*, into vector-based form, we use Vector Generation from Explicitly-defined Multidimensional semantic space (VGEM). VGEM's semantic space is explicitly defined by a set of words *d = {d1, d2, ..., dn}*, where each word defines a single dimension. To compute the vector for a word in this semantic space, VGEM uses *S* to calculate the semantic relatedness between the target word *w* and each dimension in *d*:

$$v(S,w,d) = [\ S(w,d1), S(w,d2), ..., S(w,dn)\ ]$$

For example, if *d* = {"animal", "friend"} and the word in question is "dog", then the vector for "dog" would be [*S*("dog","animal"), *S*("dog","friend")]. If *S*("dog", "animal") is 0.81 and *S*("dog","friend") is 0.84, then the vector is v[0.81, 0.84]. See Table 1, Figure 1.

Table 1: Sample VGEM Computations

| Words | Dimensions | |
|---|---|---|
| | Animal | Friend |
| Dog | 0.81 | 0.84 |
| Cat | 0.81 | 0.67 |
| Tiger | 0.79 | 0.13 |
| Robot | 0.02 | 0.60 |

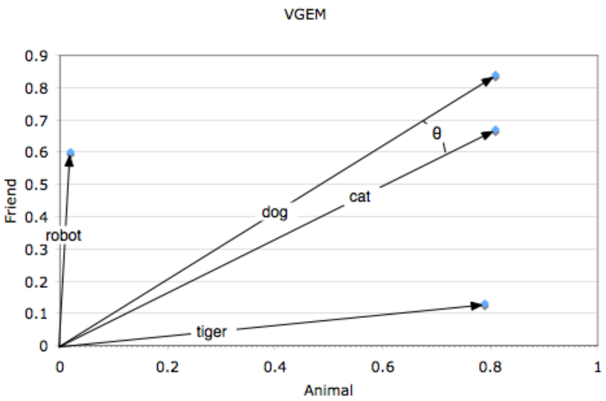


Figure 1: VGEM Semantic Space

Like all vector-based measures, VGEM defines similarity between two words to be the cosine of the angle between the vectors that represent those words. As the angle becomes smaller, and the cosine approaches 1, the words are considered more related. A value of 1 means that the two words are identical in meaning. For example, in Figure 1 the angle between “dog” and “cat” is relatively small, so the cosine of that angle will be close to 1 (.994), and the two words will be considered to be more related than any other pair of words shown.

Using this vector-based approach allows VGEM to represent a group of words as a vector sum of the words that make up the group. For example, to compute the vector for this paragraph, VGEM would create a vector representation for each word in the paragraph and add those vectors (component by component). This vector sum will represent the meaning of the whole paragraph, and its relatedness to other vectors may be measured as the cosine of the angle between those vectors. Continuing with the example in Table 1/Figure 1, the vector to represent the words "dog cat tiger" would be the sum of first three vectors in Table 1, v[2.41, 1.64].

### Advantages

The main advantage of VGEM over probabilistic MSRs is that it can compute relatedness between multi-word terms. A probabilistic MSR cannot find the similarity between two paragraphs because the probability of any two paragraphs co-occurring (word for word) in any context is virtually zero. VGEM, like other vector-based measures, can simply represent a paragraph or even a whole document as a vector, and then compare that vector to other vectors within its semantic space.

Moreover, VGEM is incremental, and does not need to pre-compute all semantic relatedness scores within the corpus before it can be used to make comparisons. Among other advantages, this lack of need for extensive preprocessing affords VGEM a larger dynamic lexicon. Other MSRs cannot handle corpora that are very large or corpora that change often (adding even a single word may require reprocessing the whole corpus).

## Performance

In addition to granting probabilistic MSRs the ability to process multi-word terms by converting them into vector form, it is important to note that this conversion preserves, or possibly improves, the representative accuracy of the original measure. Here we examine the conversion of a popular probabilistic MSR, Pointwise Mutual Information (PMI), into vector-based form called VGEM-PMI (VGEM that uses PMI as its similarity metric). PMI is a computationally inexpensive technique, and it does reasonably well on most tests of language comprehension (Turney, 2001).

For the purposes of this preliminary comparison we chose 199 random words as the dimensions for VGEM, and the World Wide Web (indexed by Google) as the corpus for both measures. To evaluate MSR performance, we compared each measure (PMI and VGEM-PMI) to human word association norms (Nelson, McEvoy, & Schreiber, 1998). The association norms database that we used contains 5017 ***cue*** words that were presented to human subjects, along with the top ***target*** words that the subjects responded with for each cue. For each of the 5017 cue words, the MSR was presented with a list containing $n$ target words that are related to the cue (based on the human data) and $n$ random words (distractors). The $2n$ words were sorted based on their semantic similarity to the cue word (as measured by the MSR). Then, the top $n$ words were compared to the original $n$ cue words to see how many of them matched. The score on each trial was $c/n$, where $c$ is the number of words that correctly matched the originals targets. The final score for each MSR was the average of the scores across all trials.

Our preliminary results show that VGEM-PMI (M=58.04%, SE=.28%) performed better than PMI (M=52.50%, SE=.28%), $t_{\text{two-tail}}=14.66, p<.001$.

## Summary and Future Work

VGEM-PMI performed better than PMI on the human word association norms test. While this result is promising, we believe that VGEM can do a lot better. In our test, we crudely defined VGEM's dimensions using 199 random words. Clearly, there are much better ways of doing this. Our future research will focus on different ways of selecting dimensions to best capture the relationships between all the words in the corpus.

Explicitly selecting VGEM's dimensions may even allow us to model domain-specific expertise. To do this, the words that constitute the dimensions could be chosen from a specific domain (e.g., politics, meteorology, or early Renaissance art). This would create an MSR that can discern the nuances of the meanings of words from the chosen domain. A modeler might create a dozen such MSRs, each proficient in a different area of expertise.

VGEM is a powerful tool for any task that could use an MSR. It is fast enough to work on any corpus, yet powerful enough to compare the meanings of whole pages of text at once. Its versatility allows it to model domain-specific expertise and learning, which might shed new light on the way in which humans acquire language.

## Acknowledgments

Many thanks to Stephane Gamard for his many contributions to this project. The work was supported in part by the Disruptive Technology Office, ARIVA contract N61339-06-C-0139 issued by PEO STRI. The views and conclusions are those of the authors, not of the US Government or its agencies.

## References

Dumais, S. (2003). Data-driven approaches to information access. *Cognitive Science, 27*(3), 491-524.

Landauer, T. K., & Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review, 104*(2), 211–240.

Lemaire, B., Denhiere, G., Bellissens, C., & Jhean-Iarose, S. (2006). A computational model for simulating text comprehension. *Behavior Research Methods, 38*(4), 628-637.

Nelson, D. L., McEvoy, C. L., & Schreiber, T. A. (1998). The University of South Florida word association, rhyme, and word fragment norms. : http://www.usf.edu/FreeAssociation/.

Pirolli, P., & Fu, W.-T. (2003). SNIF-ACT: A model of information foraging on the World Wide Web. *Lecture Notes in Computer Science, 2702*, 45-54.

Turney, P. (2001). Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL. In L. De Raedt & P. Flach (Eds.), *Proceedings of the Twelfth European Conference on Machine Learning (ECML-2001)* (pp. 491-502). Freiburg, Germany.

# Dynamic Visualization of ACT-R Declarative Memory Structure

**Andrea Heiberg[2] (Andrea.Heiberg@mesa.afmc.af.mil)**
**Jack Harris[1] (Jack.Harris@mesa.afmc.af.mil)**
**Jerry T. Ball[1] (Jerry.Ball@mesa.afmc.af.mil)**
[1]Air Force Research Laboratory / [2]L-3 Communications at Air Force Research Laboratory
6030 S. Kent St.
Mesa, AZ 85212 USA

## Introduction

We propose an automated technique for visualizing changes to declarative memory (DM) in the ACT-R 6 cognitive architecture (Anderson et al., 2004; Anderson & Lebiere, 1998). In this technique, DM chunks and the relationships between chunks are displayed graphically in a labeled tree diagram. A series of diagrams, automatically generated during a model run, allows the modeler to easily visualize how DM changes over time. The technique is potentially useful for any ACT-R model with a complex DM structure.

## Labeled Tree Diagrams

Labeled tree diagrams are commonly used in theoretical linguistics (e.g., Radford 1988) to represent constituent structure. The structure of "I increased the airspeed" may be represented as in Figure 1. Top-level SENTENCE contains constituents NOUN-PHRASE and VERB-PHRASE; VERB-PHRASE contains VERB ("increased") and NOUN-PHRASE ("the airspeed"), and so on. Figure 1 was generated from labeled bracket notation with a third-party software tool, phpSyntaxTree (Eisenbach & Eisenbach, 2006).

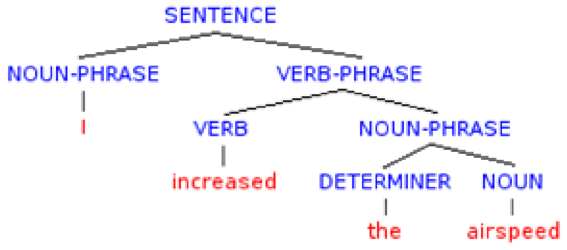


**Figure 1 Labeled Tree Diagram**

Similar diagrams have long been used for exposition of cognitive models (e.g., Anderson, 1983; Anderson, Budiu, & Reder, 2001).

## Declarative Memory Structure Visualization

The automated, dynamic visualization technique is being used in the ACT-R implementation of the Double R model of language comprehension (Ball, 2007; Ball, Heiberg, & Silber, 2007). Figure 2 shows a graphical representation of the final DM structure for "I increased the airspeed". The nodes of the tree are the names of chunks and slots. The tree structure captures the relationships between chunks. For example, chunk PRED-TRANS-VERB (transitive verb predicate) has three constituent slots, SUBJ (subject), HEAD, and OBJ (object); OBJ contains an OBJ-REFER-EXPR (object referring expression) chunk, etc.

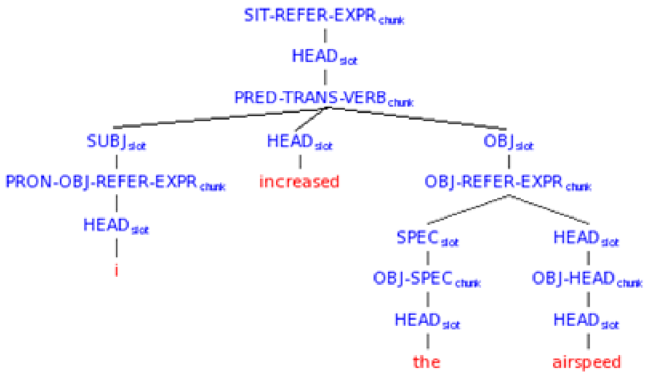


**Figure 2 Double R Model DM Structure**

The diagrams are also used to visualize changes to DM during the model run. Figure 3 shows a DM snapshot after processing "I"; Figure 4, "increased"; Figure 5, "the"; and Figure 2, the final structure after processing "airspeed".

For the development of the large-scale Double R model, the technique has proven to be greatly more efficient than examining DM by hand. Creating a series of representations takes seconds, as opposed to the minutes required to draw a single diagram by hand.

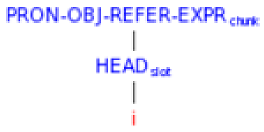


**Figure 3 DM Snapshot after Processing "I"**

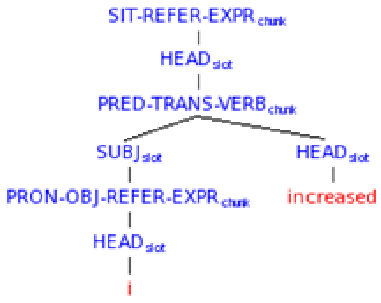


**Figure 4 DM Snapshot after Processing "increased"**

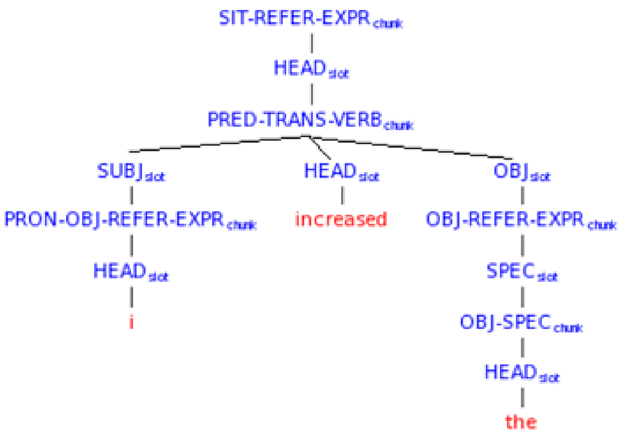


**Figure 5 DM Snapshot after Processing "the"**

The technique may also be applied to non-linguistic models, to help visualize complex DM structures. An example from the ACT-R 6 tutorial (http://act-r.psy.cmu.edu/actr6) is the Siegler child addition (Siegler & Shrager, 1984) model. The chunks for that model include:

(two isa number value 2 name "two")
(three isa number value 3 name "three")
(five isa number value 5 name "five")
(f23 isa plus-fact addend1 two addend2 three sum five)

Figure 6 shows a graphical representation of chunk f23:

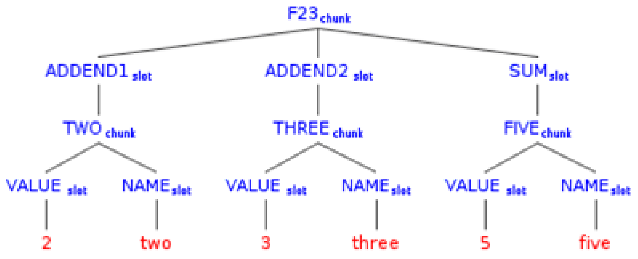


**Figure 6 Siegler Model DM Structure**

## Implementation

During a model run, snapshots of DM are created by invoking image generation from ACT-R production rules. DM is traversed from a starting chunk; slots and chunks are recursively examined to produce a labeled bracket representation, which is then input to an image generator (phpSyntaxTree) that is integrated with the system. The code is written in Lisp; ACT-R 6 functions are used to traverse DM. The implementation is generic, and may be used with any ACT-R 6 model.

A DM chunk may ultimately refer to itself. To avoid infinite processing, traversal stops at any previously visited chunk. For example, in the communication model (Matessa, 1999; Matessa & Anderson, 2000) shown in Figure 7, chunk C5 appears at the top of the tree and in the BELOW slot of chunk C6. However, C5 is expanded only once.

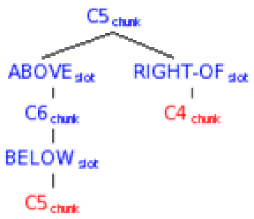


**Figure 7 Communication Model DM Structure**

## Summary

The automated, dynamic visualization technique proposed here may be used to help understand the DM structure of an ACT-R model. Relationships between chunks are displayed graphically in a labeled tree diagram. A series of diagrams is automatically created during a model run to show how DM changes over time. The technique has proven to be particularly useful for the development and exposition of a large-scale model. The implementation of the technique is general, and so may be used with any ACT-R 6 model. The clear view of DM provided by the technique helps make assumptions about a model explicit; it is hoped that this will help provide a better understanding of cognitive modeling.

## Acknowledgments

This research is funded by the Warfighter Readiness Research Division of the Air Force Research Laboratory.

## References

Anderson, J. R. (1983). *The Architecture of Cognition*. Cambridge, MA: Harvard University Press.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C. & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review* 111 (4), 1036-1060.

Anderson, J. R., Budiu, R. & Reder, L. M. (2001). A theory of sentence memory as part of a general theory of memory. *Journal of Memory and Language* 45, 337-367.

Anderson, J. R. & Lebiere, C. (1998). *The Atomic Components of Thought*. Mahwah, NJ: Erlbaum.

Ball, J. (2007). Construction driven language processing. In *Proceedings of the Second European Cognitive Science Conference.*

Ball, J., Heiberg, A. & Silber, R. (2007). Toward a large-scale model of language comprehension in ACT-R 6. In *Proceedings of the Eighth ICCM.*

Eisenbach, A. & Eisenbach, M. (2006). phpSyntaxTree tool, http://ironcreek.net/phpsyntaxtree.

Matessa, M. P. (1999). Communication in collaborative problem solving. Ms.

Matessa, M. & Anderson, J. R. (2000). An ACT-R model of adaptive communication. In *Proceedings of the Third ICCM*, 210-217, University of Groningen, Netherlands.

Radford, A. (1988). *Transformational Grammar: A First Course*. Cambridge: Cambridge University Press.

Siegler, R. S. & Shrager, J. (1984). Strategy choices in addition and subtraction: How do children know what to do? In C. Sophian (ed.), *Origins of Cognitive Skills*. Hillsdale, NJ: Erlbaum.

# A Belief Framework for Modeling Cognitive Agents

**Annerieke Heuvelink (A.Heuvelink@few.vu.nl)**
Vrije Universiteit Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
TNO Defence, Security and Safety, Department of Training and Instruction
Kampweg 5, 3769 ZG Soesterberg, the Netherlands

### Abstract

Simulation-based training in complex decision-making can be made more effective by using intelligent software agents to play key roles. For successful use in training, these agents should show representative behavior. Representative behavior may reflect expert behavior, but may also be far from optimal, especially under stress conditions. Current agent architectures hardly offer support to model cognitive properties that are essential to human decision-making. The present paper describes a framework in which agents beliefs are extended with additional arguments with which such dynamic cognitive properties can be formalized. An historic military event is used to demonstrate that the resulting framework is capable of modeling representative behavior.

## Introduction

Organizations that operate in highly uncertain and dynamic environments, such as the military, require competent staff personal. However, the very nature of their missions makes it hard to setup real-world training. Scenario-based simulator training is considered an appropriate approach for training decision-making in complex environments (Oser, 1999). A main requirement of simulator training is that it correctly represents these aspects of the real world that are necessary to achieve the learning objectives. Perhaps the most important aspect of human decision-making is the interaction with other humans, e.g., team members. In order for simulation-based training to be an alternative of real-world training, simulated entities must be able to respond naturally and validly to any emerging situation. Therefore our goal is to develop agents that are capable of generating behavior that is representative for the human they represent.

There is growing conviction and evidence that we can develop such agents by capturing the human cognitive processes in a cognitive model. The research fields of Artificial Intelligence (AI) and Cognitive Science (CS) have yielded various architectures that can be used to develop cognitive models (Pew & Mavor, 1998).

We start this paper with describing properties of architectures that are currently used for cognitive modeling. We then elaborate on various typical features of human cognition and argue that these architectures lack the mechanisms to model these features. Next, we explain how more human-like behavior can be achieved by formalized reasoning rules on beliefs with additional arguments. We illustrate the strength of this belief framework by implementing a cognitive model of a key player in a historic military incident. Finally we draw conclusions on the significance of our work and propose future research.

## Related Research

The potential and benefits of representing human behavior in (training) simulations by cognitive models of key players is generally recognized. As a result several models have been developed that can play such key roles (see e.g., Gluck & Pew, 2005). In general these models are either implemented in a cognitive architecture, like ACT-R or SOAR, or in an agent architecture such as JACK or JADEX. Cognitive architectures embody a theory of cognition, while agent architectures often encapsulate Beliefs, Desires and Intentions (BDI) (Georgeff & Lansky, 1987). For all architectures it holds that they themselves are not a model, but that they offer the constructs to build a model. The most basic construct is a declarative information entity with which the knowledge of the agent can be represented. We will refer to these knowledge entities as *beliefs*.

Since it is important for an agent to have a correct and consistent view of the world, a central issue is how an agent keeps a consistent belief database upon receiving information that is inconsistent with its current beliefs. Within AI the problem is generally solved by throwing away beliefs that cause the inconsistency. By doing so, the agent no longer has access to what it believed before, which is not very human-like. Cognitive architectures tend not to eliminate inconsistent beliefs but deal with them when they retrieve beliefs into working memory (e.g., Anderson & Lebiere, 1998; Paglieri, 2004). Mechanisms differ between architectures, but often the way they revise and retrieve beliefs is fixed. This aspect restrains the agent from having access to his old, possibly currently disbelieved, beliefs.

Research in cognitive science shows that the nature of the beliefs that form the inconsistency influence the way humans solve the inconsistency. For example, the *time* on which information is received has a large influence on the belief formation of humans. Famous temporal order effects in the updating of beliefs are primacy and recency (e.g., Anderson, 1981), which are considered to be typical human biases. Dieussaert et al. (2000) found that when a belief is deduced from a conditional statement (e.g., if A then B), that then upon receiving a categorical statement (not B) the initial *strength* of the belief in the conditional (A) is important for the revision of belief B. Another major finding is that the *source* of the beliefs is very important for how they are treated. Humans are biased to believe information that is obtained by one's own over information communicated by others. The trust of a human in the source of the information is another important factor for its believability (e.g., Mercier & Henst, 2005).

Many more cognitive biases are found in the formation of and reasoning on beliefs (Wickens & Flach, 1988). The availability bias denotes the tendency of humans to focus on the most salient outcome, which is *time* related. The confirmation bias functions on two levels; it denotes the tendency to only search for information that confirms the current hypothesis as well as the tendency to give congruent information much more weight than incongruent information. The latter strongly influences the *strengths* of beliefs. The as-if bias denotes the tendency of humans to treat *sources* 'as if' they are the same.

Cognitive biases influence the quality of human decision-making and are found to arise especially under stress conditions (see e.g., Baron, 2000). Since we want our agent to generate representative human behavior under a variety of stress conditions, we need to be able to model the before mentioned processes. Current architectures don't offer support to model (biased) reasoning over beliefs taking their initial time, their source and their certainty into account. In the next section we propose a framework with which such processes can be formalized.

We are not the first to tackle the problem of modeling biased reasoning and belief revision. However, as mentioned above most cognitive belief models adept the strengths of beliefs upon receiving new information and by doing so loose access to what was believed before. Moreover, stress is often not a factor in the revision or retrieval of beliefs.

## Belief Framework

We want to develop a decision-making agent that reflects a human in the way it acts and reasons. For this goal we develop a logical framework in which beliefs represent the agent's declarative information entities. We decide to represent beliefs in predicate logic since this format enables formal verification of global properties which is useful for training. To ensure that an agent can have an up-to-date consistent belief set without loosing access to its old beliefs, we propose to *time stamp* each belief at the time it is formed with that time. With this feature it is possible to model (biased) reasoning over beliefs over time. We found only one other paper that proposes to time-stamp beliefs. Sripada (1993) took this approach in search of a more efficient belief revision technique, but only looked at binary beliefs.

We on the other hand want our agent to have graded beliefs like a human and therefore further propose to *certainty stamp* beliefs. The certainty stamp of a belief denotes the strength of the agent's belief in its truth value at the time captured in the time stamp. Last we propose to *source stamp* each belief, by which the origin of the information is captured. Using these three extra arguments various cognitive processes can be formalized as will be shown in the next sections.

### Belief Predicate

A belief can be seen as a collection of properties that can be captured with the following *belief* predicate:

$$
\begin{aligned}
&\forall p\ \forall a\ \forall v\ \forall t\ \forall s\ \forall c\ [\quad \mathit{belief}(p, a, v, t, s, c) \leftrightarrow \\
&\qquad \exists b\ \exists e\ [\quad \mathit{beliefhasterm}(b, e) \wedge \\
&\qquad\qquad \mathit{termhaspredicate}(e, p) \wedge \\
&\qquad\qquad \mathit{termhasattribute}(e, a) \wedge \\
&\qquad\qquad \mathit{termhasvalue}(e, v) \wedge \\
&\qquad\qquad \mathit{beliefhastimestamp}(b, t) \wedge \\
&\qquad\qquad \mathit{beliefhassource}(b, s) \wedge \\
&\qquad\qquad \mathit{beliefhascertainty}(b, c)\ ]\ ]
\end{aligned}
$$

The core of a belief is a term that denotes the information that is believed, e.g., that the identity (p) of track2 (a) is hostile (v). Besides this term a belief consists of three extra arguments denoting the time it was formed (t), its source (s) and how certain the agent is of that belief (c).

To formalize relations between beliefs over time it is necessary to have a reference to time that specifies the time at which a certain belief was held by the agent. For this we introduce a two-place predicate HoldsAt. When we reify the belief predicate of the object language to a propositional term b, we can state using this meta-language predicate at which time the belief is held: HoldsAt(b, t).

For every belief(p, a, v, t, s, c) that can be found in the agent's database it can stated that HoldsAt ( belief (p,a,v,t,s,c), t), since the t of the belief denotes that it was then formed and thus logically holds.

### Formation of Beliefs

By using the aforementioned belief system we can model relevant cognitive properties and processes. The first interesting process is the transfer of information from the outside world into a belief. Research in cognitive science mentioned above pointed out that the source of the information as well as the current state of beliefs (confirmation bias) is relevant for this process. These two aspects influence the strength with which an agent ends up believing that information, i.e., the certainty of its belief. We accommodate these aspects by transferring information from the world into a belief in three stages.

First, a *presourceexpectancybelief* is formed:

$$
\begin{aligned}
&\forall p\ \forall a\ \forall v\ \forall t\ \forall s\ \forall c\ \ [\\
&\quad \mathit{HoldsAt}(\mathit{input\_from\_world}(p, a, v, s, c), t)\\
&\quad \rightarrow\\
&\quad \mathit{HoldsAt}(\mathit{presourceexpectancybelief}(p, a, v, t, s, c), t)\ ]
\end{aligned}
$$

Secondly, the influence of the source on the believability of the given information is determined, by using the agent's trust level in that source. In how far this bias occurs, i.e., how much this process moves the perceived certainty away from the actual certainty, is influenced by the current stress level of the agent.

$$
\begin{aligned}
&\forall p\ \forall a\ \forall v\ \forall t\ \forall s\ \forall c\ \forall tr\ \forall st\ [\\
&\quad \mathit{HoldsAt}(\mathit{presourceexpectancybelief}(p, a, v, t, s, c), t) \wedge\\
&\quad \mathit{HoldsAt}(\mathit{trust\_in\_source}(s, tr), t) \wedge \qquad (-1 \leq \mathrm{tr} \leq 1)\\
&\quad \mathit{HoldsAt}(\mathit{stress}(st), t) \qquad\qquad\qquad (\ 0 \leq \mathrm{st} \leq 1)\\
&\quad \rightarrow\\
&\quad \mathit{HoldsAt}(\mathit{preexpectancybelief}(p, a, v, t, s, c + tr{*}c{*}st), t)\ ]
\end{aligned}
$$

Thirdly, the current state of beliefs is taken into account. This is not done directly, but through the notion of expectancies. The *expectancy* predicate has 4 arguments, denoting the expected term (p, a, v) as well as a certainty. Expectancies differ from beliefs in that they are formed automatically and can be considered unconscious.

Expectancies are formed in two ways; each term that is currently believed gets transferred to an expectancy that will hold the next time step. The strength of the expectancy is a function of the strength of the belief and the persistence of the predicate; we will elaborate on the latter later on. Secondly, certain (combinations of) beliefs can yield new expectancies. The certainty of expectancies decays over time and the expectancy ceases to exist when its certainty becomes equal to zero.

To determine the final certainty of the belief existing congruent and incongruent expectancies are taken into account. The extent to which these expectancies bias the certainty of the agent in the final belief is influenced by the current stress level. Since multiple situations are possible multiple rules are needed to formalize this process:

$$\begin{array}{l}
\forall p\,\forall a\,\forall v\,\forall t\,\forall s\,\forall c\ \ [\\
\quad \mathit{HoldsAt}(\mathit{preexpectancybelief}(p, a, v, t, s, c), t) \wedge\\
\quad \neg\exists w\,\exists d\,[\,\mathit{HoldsAt}(\mathit{expectancy}(p, a, w, d), t)\,]\\
\quad \rightarrow\\
\quad \mathit{HoldsAt}(\mathit{belief}(p, a, v, t + 1, s, c), t + 1)\,]
\end{array}$$

$$\begin{array}{l}
\forall p\,\forall a\,\forall v\,\forall t\,\forall s\,\forall c\,\forall d\,\forall st\ \ [\\
\quad \mathit{HoldsAt}(\mathit{preexpectancybelief}(p, a, v, t, s, c), t) \wedge\\
\quad \mathit{HoldsAt}(\mathit{expectancy}(p, a, v, d), t) \wedge\\
\quad \mathit{HoldsAt}(\mathit{stress}(st), t)\\
\quad \rightarrow\\
\quad \mathit{HoldsAt}(\mathit{belief}(p, a, v, t + 1, s, c + d * st), t + 1)\,]
\end{array}$$

$$\begin{array}{l}
\forall p\,\forall a\,\forall v\,\forall t\,\forall s\,\forall c\,\forall u\,\forall d\,\forall st\ \ [\\
\quad \mathit{HoldsAt}(\mathit{preexpectancybelief}(p, a, v, t, s, c), t) \wedge\\
\quad \mathit{HoldsAt}(\mathit{expectancy}(p, a, u, d), t) \wedge\\
\quad u \neq v \wedge\\
\quad \neg\,\exists e\,[\,\mathit{HoldsAt}(\mathit{expectancy}(p, a, v, e), t)\,] \wedge\\
\quad \mathit{HoldsAt}(\mathit{stress}(st), t)\\
\quad \rightarrow\\
\quad \mathit{HoldsAt}(\mathit{belief}(p, a, v, t + 1, s, c - d * st), t + 1)\,]
\end{array}$$

Intermediate rules (not denoted) handle new (pre)beliefs whose certainties lie outside the certainty range.

An agent can also form new beliefs using *conditional statements* and its current beliefs. These rules, together with believed *categorical statements*, make up the task specific knowledge of an agent. The formation of a new belief by a conditional statement happens in two stages. First a *preexpectancybelief* is formed, which is than transferred into a belief using the mechanisms described above. A belief formed by a reasoning rule receives that rule's name as it source. An example rule is the following:

$$\begin{array}{ll}
\forall c\ \ [ & \mathit{HoldsAt}(\ \mathit{belief}(\ \mathit{weather},\ \mathit{local},\ \mathit{raining},\ t,\\
& \mathit{integratedsources},\ c),\ t)\\
& \rightarrow\\
& \mathit{HoldsAt}(\mathit{preexpectancybelief}(\mathit{status},\ \mathit{street},\ \mathit{wet},\ t,\\
& \mathit{deduce\_wet\_from\_raining},\ c),\ t)\ ]
\end{array}$$

Note that this rule requests as input a just formed belief (denoted by t) whose source is equal to *integratedsources*.

## Belief Integration

An important aspect of the belief framework is that reasoning rules request beliefs as input that have as time argument the *current time* (t) and as source argument *integratedsources* (s). The requested time argument denotes the claim that the belief should just be formed and thus holds (present in working memory) while the source denotes the claim by which rule it should be formed. The reasoning rule that produces beliefs with *integratedsources* as source argument deduces what exactly is currently believed by the agent. This rule deals with any inconsistencies in the belief set formed by beliefs from different sources or at different times. The retrieval of a belief into working memory can be seen as its human equivalent.

To implement this process we first implement the agent's memory by the following simple rule, which assumes that beliefs are never forgotten.

$$\begin{array}{l}
\forall p\,\forall a\,\forall v\,\forall t'\,\forall s\,\forall c\,\forall t\ [\\
\quad \mathit{HoldsAt}(\mathit{belief}(p, a, v, t', s, c), t)\\
\quad \rightarrow\\
\quad \mathit{HoldsAt}(\mathit{belief}(p, a, v, t', s, c), t{+}1)\,]
\end{array}$$

To facilitate the formalization of reasoning rules that use the agent's memory we introduce the *lastbelief* predicate, which denotes the most recent belief in the agent's memory for given specifications. Its definition is:

$$\begin{array}{l}
\forall p\,\forall a\,\forall v\,\forall t\,\forall s\,\forall c\,\forall n\ [\\
\qquad\qquad \mathit{HoldsAt}(\mathit{lastbelief}(p,a,v,t,s,c),n)\\
\qquad\qquad \leftrightarrow\\
\qquad\qquad [\ \ \mathit{HoldsAt}(\mathit{belief}(p,a,v,t,s,c),t) \wedge t \leq n \wedge\\
\qquad\qquad\quad \neg\exists t'\ \ [\ \ \mathit{HoldsAt}(\mathit{belief}(p,a,v,t',s,c),t') \wedge\\
\qquad\qquad\qquad\qquad t' \geq t \wedge t' \leq n\ ]\ \ ]\ \ ]
\end{array}$$

To determine what exactly is believed by the agent, it is relevant to consider that a belief's validity over time is strongly influenced by its predicate. Values of certain predicates are much more persistent than others; consider the chance that a person's sex, marital status or mood changes over time. An agent's certainty level in a belief whose predicate is very persistent does not change much over time. However, beliefs about predicates of which the values are likely to change will quickly loose certainty. The persistence level of a predicate also influences the decaying factor of expectancies about it. The rule that determines what exactly is believed, so that is responsible of deducing the current belief from old beliefs, is formalized as:

$$\begin{array}{l}
\mathit{given}\ (p, a)\\
\forall v1\,\forall t1\,\forall s1\,\forall c1\,\forall t\,\forall pd\,\forall c'\ \ [\\
\quad \mathit{HoldsAt}(\mathit{lastbelief}(p, a, v1, t1, s1, c1), t) \wedge\\
\quad \mathit{HoldsAt}(\mathit{persistence\_decay}(p, pd), t) \wedge \qquad (0 \leq \mathrm{pd} \leq 1)\\
\quad \neg\exists c''\,\exists v2\,\exists t2\,\exists s2\,\exists c2\\
\qquad [\ \ \mathit{HoldsAt}(\mathit{lastbelief}(p, a, v2, t2, s2, c2) \wedge\\
\qquad\quad c2 - pd * (t - t2)\ \ > c1 - pd * (t - t1)\,]\\
\quad \rightarrow\\
\quad \mathit{HoldsAt}(\mathit{belief}(p, a, v1, t{+}1, \mathit{integratedsources},\\
\quad \ldots \qquad\qquad\qquad\qquad\qquad c1 - pd * (t - t1)), t{+}1)\\
\quad \mathit{HoldsAt}(\mathit{belief}(p, a, v1, t{+}10, \mathit{integratedsources},\\
\qquad\qquad\qquad\qquad\qquad\qquad c1 - pd * (t - t1)), t{+}10)\,]
\end{array}$$

Also in this case there is an intermediate rule that handles beliefs whose certainties lie outside the certainty range.

Following this rule, the agent ends up believing the value of the belief whose certainty is the greatest after taking into account the time passed since it was formed and the persistence of the predicate. This might entail that an older belief with a higher certainty is believed over a newer belief from a different source or the other way around, it depends on the nature of predicate. The determination of the new certainty is currently kept straightforward; it is equal to the highest one after taking the time into account. Other sources that claim the same do not contribute to its certainty.

A belief that is consciously deduced using this rule is stated to hold for ten following time points. This reflects the fact that items retrieved by humans also stay a while in working memory. The above rule takes many aspects into account and is cognitive expensive. As mentioned on page 2 humans display a bias to treat all sources as equally likely. With this simplification a decision can be made much cheaper, for example, by simply taking the most recent one. In such cases the antecedent becomes:

$$\begin{array}{l}
\mathit{HoldsAt(lastbelief(p, a, v1, t1, s1, c1), t)} \wedge \\
\mathit{HoldsAt(persistence\_decay(p, pd), t)} \wedge \\
\neg\exists v2 \exists t2 \exists s2 \exists c2 \\
\qquad [\ \mathit{HoldsAt(lastbelief(p, a, v2, t2, s2, c2)} \wedge t2 > t1\ ]
\end{array}$$

Which rule is applied is influenced by the agent's stress level and should be determined at the control level.

## Reasoning over Beliefs over Time

With the given belief predicate we can deduce whether an agent believes something for a longer period of time. The *timecertaintyintegratedbelief* predicate denotes the time when the term of the current integratedsources-belief was believed for the first time. Furthermore it should hold that no other value was believed in the mean time and that it did not become unknown caused by the time passed and the decay of certainty:

$$\begin{array}{l}
\mathit{given(p, a, pd)} \\
\forall n\ \forall c \forall t \forall d\ \ [ \\
\quad \mathit{HoldsAt(belief(p, a, v, n, integratedsources, c), n)} \wedge \\
\quad \mathit{HoldsAt(belief(p, a, v, t, integratedsources, d), t)} \wedge \\
\quad \forall v' \forall t' \forall c'\ [ \\
\qquad \mathit{HoldsAt(belief(p, a, v', t', integratedsources, c'), t')} \wedge \\
\qquad v \neq v' \wedge t' < n \wedge t > t' \wedge \\
\qquad \neg\exists t'' \exists e\ \ [ \\
\qquad\quad \mathit{HoldsAt(belief(p, a, v, t'', integratedsources, e), t'')} \\
\qquad\quad \wedge\ t'' > t' \wedge t'' < t\ ]\ \ ] \\
\quad \forall t' \forall c'\ \ \ [ \\
\qquad \mathit{HoldsAt(belief(p, a, v, t', integratedsources, c'), t')} \wedge \\
\qquad t \leq t' \wedge t' < n \wedge \\
\qquad \exists t'' \exists e\ [ \\
\qquad\quad \mathit{HoldsAt(belief(p, a, v, t'', integratedsources, e), t'')} \wedge \\
\qquad\quad t'' > t' \wedge c' - pd * (t' - t'') > 0\ \ \ ]\ \ ] \\
\quad \leftrightarrow \\
\quad \mathit{HoldsAt(timecertaintyintegratedbelief(p,a,v,t),n)}\ \ \ \ ]
\end{array}$$

Note that this rule can be made executable by replacing the HoldsAt(b, tx) statements with HoldsAt(b, n), given that a memory system is in place. This should obviously hold for an implemented model, as presented in the next section.

This extra object predicate is useful for modeling the deduction of a belief based on the persistence of another, e.g., position stays equal → speed = 0. The predicate is also very useful to model the reasoning over belief patterns over time. E.g., to determine whether a ship zigzags the beliefs over time concerning its headings have to be integrated. The following rule depicts the principle, but should be filled with more domain specific knowledge.

$$\begin{array}{l}
\mathit{given(p, a, v1, v2)} \\
\forall t1 \forall t2 \forall t3 \forall n\ \ \ [ \\
\quad \mathit{HoldsAt(timecertaintyintegratedbelief(p,a,v1,t1),n)} \wedge \\
\quad \mathit{HoldsAt(timecertaintyintegratedbelief(p,a,v2,t2),t1')} \wedge \\
\quad t1' < t1 \wedge \neg\exists t1'' \exists v \exists t\ \ \ [ \\
\qquad \mathit{HoldsAt(timecertaintyintegratedbelief(p,a,v,t),t1'')} \wedge \\
\qquad t1'' < t1 \wedge\ t1'' > t1'\ ] \\
\quad \mathit{HoldsAt(timecertaintyintegratedbelief(p,a,v1,t3),t2')} \wedge \\
\quad t2' < t2 \wedge \neg\exists t2'' \exists v \exists t\ [ \\
\qquad \mathit{HoldsAt(timecertaintyintegratedbelief(p,a,v,t),t2'')} \wedge \\
\qquad t2'' < t2 \wedge\ t2'' > t2'\ ] \\
\quad \rightarrow \\
\quad \mathit{HoldsAt(preexpectancybelief(pp,a,vp,n,this\_rule,c),n)}\ ]
\end{array}$$

## Case Study – Iran Air Flight 655

To illustrate our approach we present an historic case for which we developed and implemented a cognitive model of a human decision maker. It concerns the Identification Designation Supervisor (IDS) aboard the combat information center of the *USS Vincennes* cruiser that in 1988 erroneously shot down an Iranian Airbus (Fogarty, 1988). This accident has been widely referred to as an example of faulty decision-making under stress (Klein, 1998). Using this case, we want to investigate whether our approach can be used to model the behavior of the IDS-officer.

We now give a short description of the sequence of most relevant events that led to the wrong identification of the airbus by the IDS, which contributed to it being shot down. This description mixes facts about the behavior of the IDS with assumptions about his reasoning. We deduced both from the formal investigation rapport (Fogarty, 1988).

**Time: 10.47 AM**

- The IDS is focused on an Iranian P-3. Since the P-3 belongs to hostile country Iran and is a patrol aircraft that can guide other aircraft on hostile missions, the IDS expects hostile aircrafts.
- The radar reports a new track of interest (track2) at a range of 47nm and bearing 025, which corresponds to the runway of Iranian airport Bandar Abbas. The IDS observes the new track and based on the fact that the track's origin is an Iranian airport also used for military aircrafts, he believes it might be hostile.
- In order to determine whether the track represents a commercial aircraft, the IDS checks the Bandar Abbas commercial airline departure times schedule. However, the time of departure and scheduled time differ too much to make the neutral identification.
- In order to obtain more information the IDS sets its remote control indicator (RCI) challenge gate at the track, so it can pick up the track's Identify Friend or Foe (IFF) Mode, a system all planes are equipped with. Based on his hostile assumption he expects to receive mode II or mode III.
- The IDS picks up the neutral IFF Mode III-6675. However, all aircrafts can emit Mode III and therefore this information is not conclusive for a neutral identification.

**Time: 10.48 AM**

- The IDS observes from its Large Screen Display (LSD) that track2 is locked on by the USS Sides, however does not react. When military aircrafts are locked on to, they tend to change behavior. Non-military aircrafts do not notice when they are locked-on and therefore is unchanged behavior an indicator of a neutral aircraft. However, the IDS keeps believing the track might be hostile.

**Time: 10.50 AM**

- The IDS sees a Mode II-1100 on its RCI-display. He expected this response from the last track he queried and simply assumes that the signal comes from that track.
- Since the IDS knows that a modeII-11XX block is used by Iranian F-14's he reports track2 as 'possible F-14'.

## Cognitive Model of the IDS

Our approach currently focuses on formalizing belief predicates and processes on beliefs with which we can model *how* humans process information. The formalization of w*hen* they do that has not yet been tackled. However, to simulate a cognitive model that demonstrates the former, an implementation of the latter is needed. To simulate human control we use a simple goal-directed reasoning strategy. For this strategy to work we abstracted the necessary in- and output of each rule, added the goal it contributes to and specified what satisfies that goal. For the example rule on page 3 two of these constructs would be:

*Input_of_rule_goal(deduce_wet_from_raining,determine_status (street), belief_tc(local,weather,raining,integratedsources))*
*satisfies_goal(determine_status(street), belief_vtsc(status,stree t))*

Furthermore we added backwards-reasoning rules as:

$$\begin{array}{l}
\forall g1\ \forall p1\ \forall a1\ \forall r1\ \forall p2\ \forall a2\ \forall s\ \forall r2\ \forall g2\ \forall t \quad [\\
\quad HoldsAt(goal(g1),\ t)\\
\quad HoldsAt(satisfies\_goal(g1,\ belief\_vtsc(p1,\ a1)),\ t)\\
\quad \neg\exists v \exists s \exists c\ [\ HoldsAt(belief(p1,\ a1,\ v,\ t,\ s,\ c),\ t)\ ]\\
\quad HoldsAt(output\_of\_rule\_goal(r1,\ g1,\ belief\_tsc(p1,\ a1)),\ t)\\
\quad HoldsAt(input\_of\_rule\_goal(r1,\ g1,\ belief\_vtc(p2,\ a2,\ s)),\ t)\\
\quad HoldsAt(output\_of\_rule\_goal(r2,\ g2,\ belief\_vtc(p2,\ a2,\ s)),\ t)\\
\quad \neg\exists v \exists c\ [\ HoldsAt(belief(p2,\ a,\ v,\ t,\ integratedsources,\ c),\ t)\ ]\\
\quad \rightarrow\\
\quad HoldsAt(goal(g2),\ t) \quad ]
\end{array}$$

The main goal of the IDS-officer is to identify each track in the environment as quickly as possible in terms of *hostile*, *neutral* or *friend*. From this main goal all other relevant sub goals are determined each time step by backtracking, using the agent's task knowledge as well as its current belief state.

The model is implemented using the LEADSTO language with which temporal dependencies between two state properties can be modeled and depicted graphically (Bosse et al. 2007). The modeled dynamic properties have the following executable format: Let α and β be state properties of the form 'conjunction of atoms or negations of atoms', and *e*, *f*, *g*, *h* non-negative real numbers. In the LEADSTO language $\alpha \rightarrow_{e,f,g,h} \beta$, means:

If state property α holds for a certain time interval with duration *g*,
Then after some delay (between *e* and *f*) state property β will hold for a certain time interval of length *h*.

In the following figures traces are shown that visualize the IDS properties (on the vertical axes) over time (horizontal axes). Dark boxes on top of a line denote that the property *HoldsAt* that time, light boxes below that it does not. In all traces the certainty and persistence decay parameters range from 0-10 instead as proposed in the text from 0-1. For displaying purposes the *integratedsources* beliefs that hold for 10 timestamps are summed up in one predicate belief_t.

We lack the space to show all the reasoning steps of the IDS model, so we focus on the events of bullet 2. Figure 1 shows a trace depicting that the IDS actively observes the altitude of the track from its screen (ownCROD) and forms a belief about its value. This trace shows how the IDS's trust in his CROD (0.8) given his stress level (0.5) influences the certainty of the final belief (7 instead of 5).

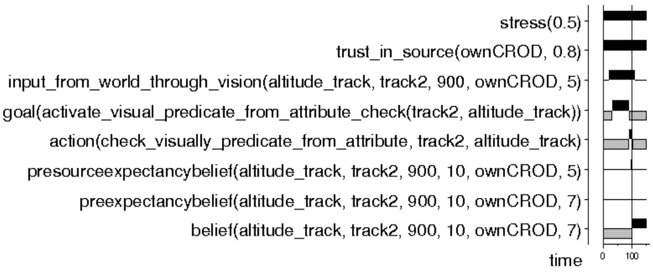


Figure 1: Observation of World and Formation of Belief

Next he reasons about the track's origin taking into account the track's position and altitude he just observed. The outcome, a belief about the airport it departed from, leads together with beliefs about the nature of that airport to a belief about the track's identity which is biased by the existing expectancy of hostile tracks (bullet 1), see figure 2.

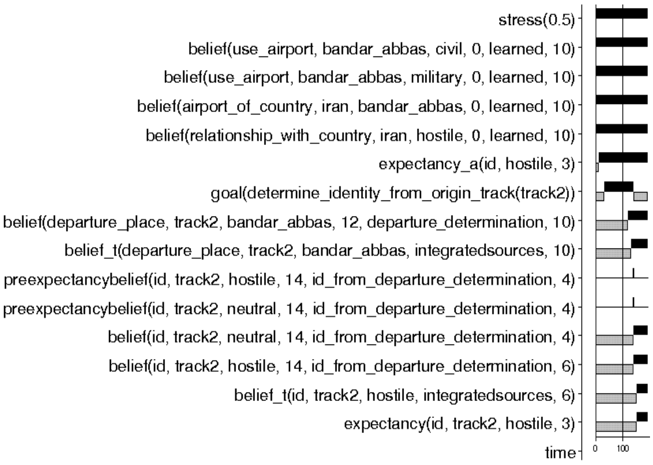


Figure 2: Formation of New Belief and Expectancy

In the following time steps the IDS performs various actions that lead to new beliefs that contribute to the reasoning about the track's identity. Unfortunately the IDS biased reasoning caused by his stress level causes him to belief he is dealing with a hostile F-16.

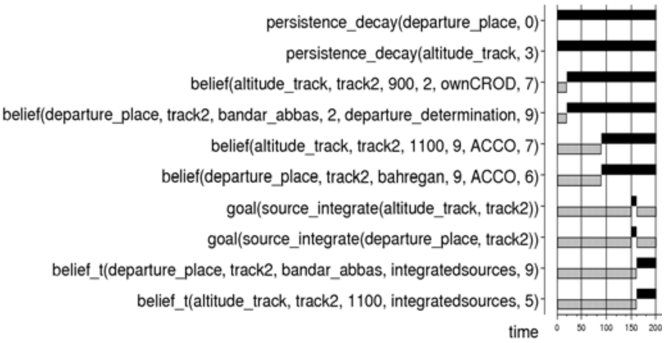


Figure 3: Source Integration on Two Predicate Types

To illustrate one important aspect of our framework a bit further we made a trace that displays the source-integration process on two different types of belief predicates: see figure 3. It can be seen that based on the nature of their predicate the beliefs are treated differently.

## Discussion and Conclusion

We developed a framework for cognitive modeling based on beliefs with a *time*, *source* and *certainty* label attached. These extra labels enable the formalization of various processes on beliefs that lie at the basis of human cognition. Interactions between the time, source and certainty of beliefs has been made explicit, which is not possible in other architectures. Moreover, the influence of these parameters on each other is made tunable by the introduction of a stress level parameter.

The model of the IDS-officer shows that the framework is capable of generating human-like behavior. Agents modeled with this framework will be capable of generating more human-like behavior than, e.g., standard BDI agents. The fact that they are able to show behavior that is more representative for humans will make the agents more believable to the trainee that interacts with them. Since the believability of a training environment influences the effectiveness of the training, the modeling of agents using our framework will contribute to the effectiveness of the training and achievement of training objectives.

Our research doesn't stop here. The current framework will be extended by adding formal specifications of other relevant cognitive processes, such as attention and trust. Although the latter is already represented in the framework the current trust of an agent in sources is static. In reality however, trust is a dynamic property which is strongly influenced by experience. An agent capable of reasoning over its experiences with sources would be able to adapt its trust in sources. Stress level is another parameter that is currently fixed and that we would like to formalize as a dynamic property. Also the persistence values of properties are currently given and static, which is reasonable assuming that humans have learned them during their lifetime. However, when an agent would be capable to determine these values based on experiences with the environment, it would be much more adaptable to new environments.

As the next research step we will tackle the control of the agent. The simple control implemented in this paper was sufficient for demonstrating the reasoning rules. However, real humans have to deal with a limited amount of attention and processing power and therefore make many decisions on the control level. We like to develop a control framework in which we can capture these, probably biased, processes.

The cognitive validity of the model is debatable. However, by incorporating more outcomes of cognitive science research in our approach, we hope to approach our goal: the modeling of agents that can correctly represent human behavior in specific task training environments.

## Acknowledgements

The author likes to thank Jan Treur for many fruitful discussions during this research and assisting on formal details, Tibor Bosse for clarifying aspects of the LEADSTO language and Karel van den Bosch for commenting an earlier draft.

## References

Anderson, J. R., & Lebiere, C. (1998). *The Atomic Components of Thought*. Mahwah, NJ: Lawrence Erlbaum Associates.

Anderson, N. H. (1981). *Foundations of Information Integration Theory*. New York, NY: Academic Press.

Baron, J. (2000). *Thinking and Deciding* (3d. edition). New York, NY; Cambridge University Press.

Bosse, T., Jonker, C. M., Meij, L. van der, & Treur, J. (2007). A Language and Environment for Analysis of Dynamics by SimulaTiOn. *International Journal of Artificial Intelligence Tools, 16(3),* to appear.

Dieussaert, K., Schaeken, W., Neys, W. de, & d'Ydewalle, G. (2000). Initial belief state as a predictor of belief revision. *Current Psychology of Cognition, 19(3),* pp. 277 – 286.

Fogarty, W. M. (1988). *Formal Investigation into the Circumstances Surrounding the Downing of Iran Air Flight 655 on 3 July 1988* (Invest. Rep. 93-FOI-0184). Department of Defense, USA.

Georgeff, M. P., & Lansky, A. L. (1987). Reactive Reasoning and Planning. In *Proceedings of the Sixth National Conference on Artificial Intelligence* (pp. 677 – 682). Menlo Park, Ca: AAAI Press.

Gluck, K. A., & Pew, R. W. (Eds.) (2005). *Modeling Human Behavior with Integrated Cognitive Architectures: Comparison, Evaluation, and Validation*. Mahwah, NJ: Lawrence Erlbaum Associates.

Klein, G. (1998). *Sources of Power: How People Make Decisions*. Cambridge, MA: MIT Press.

Mercier, H. & der Henst, J.-B. V. (2005). The source of beliefs in conflicting and non-conflicting situations. In *Proceedings of the 27th Annual Conference of the Cognitive Science Society* (pp. 1495 – 1500). Mahwah, NJ: Lawrence Erlbaum Associates.

Oser, R. L. (1999). A structured approach for scenario-based training. In *Proceedings of the 43rd Annual Meeting of the HFES* (pp. 1138 – 1142). Santa Monica, CA: Human Factors and Ergonomics Society.

Paglieri, F. (2004) Data-oriented Belief Revision: Towards a Unified Theory of Epistemic Processing. In *STAIRS 2004: Proceedings of the Second Starting AI Researchers' Symposium* (pp. 179 – 190). Amsterdam: IOS Press.

Pew, R. W., & Mavor, A. S. (1998). *Modeling Human and Organizational Behavior*. Washington, DC: National Academy Press.

Sripada, S. M. (1993). A Temporal Approach to Belief Revision in Knowledge Bases. In *Proceedings of the Ninth Conference on Artificial Intelligence for Applications* (pp. 56 – 62). Orlando, FL; IEEE Computer Society Press.

Wickens, C. & Flach, J. (1988). Information Processing. *Human Factors in Aviation*. San Diego, CA: Academic Press.

# A Formal Empirical Analysis Method for Human Reasoning and Interpretation

**Tibor Bosse**[1]**, Mark Hoogendoorn**[1]**, Catholijn M. Jonker**[2]**, and Jan Treur**[1]

[1]Vrije Universiteit Amsterdam, Department of Artificial Intelligence, De Boelelaan 1081a,
1081 HV Amsterdam, The Netherlands, {tbosse, mhoogen, treur}@cs.vu.nl

[2]Delft University of Technology, Department of Mediametics, Mekelweg 4,
2628 CD Delft, The Netherlands, catholijn@mmi.tudelft.nl

### Abstract

The study of human reasoning often concentrates on reasoning *from* an already assumed interpretation of the world, thereby neglecting reasoning *towards* an interpretation. In recent literature within Cognitive Science, means taken from the area of nonmonotonic logic are proposed to analyze the latter aspect of human reasoning. In this paper this claim is further worked out and tested against empirical material of human reasoning during critical situations (incident management). Empirical and simulated reasoning traces have been analyzed by comparing them and by automatically checking properties on them.

## 1. Introduction

In recent years, from the area of Cognitive Science, there is an increasing interest in tools originating from the area of nonmonotonic reasoning. In (Stenning and van Lambalgen, 2006) it is shown how the empirical study of human reasoning processes has been too much dominated by an emphasis on classical, deductive logic. This applies equally well to the socalled rule-based or syntactic stream (e.g., Braine and O'Brien, 1998; Rips, 1994), as to the model-based or semantic stream (e.g., Johnson-Laird, 1983; Johnson-Laird and Byrne, 1991). In their analysis of human reasoning they claim that much more important than the question whether reasoning should be considered from a syntactical or semantical perspective, is the distinction between: a) reasoning *towards* an interpretation, and b) reasoning *from* an interpretation. The latter type of reasoning is reasoning within an already unambiguously determined formalized frame, and can be analyzed by means of classical logic. The first type of reasoning, however, still has to find such a frame and has to deal with ambiguities and multiple interpretation possibilities, and does not have a unique outcome. It is at this point that they propose nonmonotonic logic as a more adequate analysis tool for human reasoning processes. Within nonmonotonic logic it is possible to formalize reasoning processes that deal with multiple possible outcomes, which can be used to model different possibilities of interpretation; see (Engelfriet and Treur, 2003) for a similar perspective. Thus, from an empirical angle, within the area of human reasoning within Cognitive Science, a new, more empirical perspective was introduced to study nonmonotonic reasoning processes.

The current paper reports research to further work out and test this empirical perspective in the context of incident management. Detailed reports are available that describe what went wrong in the management of well-known disasters, see, e.g., (Ministry of the Interior, 1996). These reports provide empirical data showing how humans reason under the pressure of a critical situation. Cases taken from them form the basis of the research reported in this paper to further detail and illustrate the use of the Stenning-van Lambalgen perspective on reasoning and interpreting. The leading example is an airplane crash.

The outline of the paper is as follows. The aircrash example is presented in Section 2. Section 3 presents an abstract formalization of a reasoning process leading to multiple interpretations, and Section 4 shows how Default Logic can be used to specify such processes. To obtain simulation of such reasoning, variants of Default Logic are considered in which control decisions can be represented. To this end, in Section 5 a temporalized form of Default Logic is chosen to simulate the possible reasoning traces for the case study. In Section 6 a number of properties of such reasoning traces are formalized and checked. Section 7 presents the conclusions.

## 2. The Incident Management Domain

Within incident management people are working under severe pressure; having incomplete information, decisions have to be made quickly, which can have a huge impact on the successfulness of the whole operation. This paper focuses on one example: that of the Hercules airplane crash at the military airport of Eindhoven in the Netherlands (Ministry of the Interior, 1996). This example is taken because it is representative for the occurrences in incident management. The plane, carrying a military brass band in the cargo room and a crew of four people, flew into a flock of birds just before landing, causing one of the engines to fail, which made the plane tilt to one side. As a result, the plane crashed on the runway and caught fire. The Air Traffic Controller (ATC) had information that a military brass band was on board of the plane. Afterwards he claimed to have informed the alarm centre operator of this fact, who in turn stated never to have received the information. As a result, the operator did inform fire fighters, but declared the wrong scenario (i.e., for merely the crew on board). After the fire fighting forces had arrived at the scene, one of them contacted the air traffic controller, asking how many people were on board of the plane. Since the air traffic controller reasoned under the assumption that the message of a military brass band being on board had been passed through to the fire fighters, he answered that this was unknown, interpreting the question as a request for the exact amount of people on board. The fire fighter therefore assumed that only the crew was on board, thus the brass band was not rescued.

## 3. Multiple Interpretations

Reasoning towards an interpretation can be formalized at an abstract generic level as follows. A particular interpretation for a given set of formulae considered as input information for the reasoning, is formalized as another set of formulae, that in one way or the other is derivable from the input information (output of the reasoning towards an interpretation). In general there are multiple possible outcomes. The collection of all possible interpretations derivable from a given set of formulae as input information (i.e., the output of the reasoning towards an interpretation) is formalized as a collection of different sets of formulae. A formalization describing the relation between such input and output information is described at an abstract level by a multi-interpretation operator. The input information is described by propositional formulae in a propositional language $\mathbf{L_1}$. An interpretation is a set of propositional formulae, based on a propositional language $\mathbf{L_2}$.

**Definition 1 (Multi-Interpretation Operator)**

a) A *multi-interpretation operator* **MI** with input language $\mathbf{L_1}$ and output language $\mathbf{L_2}$ is a function $\mathbf{MI} : \mathsf{P}(\mathbf{L_1}) \rightarrow \mathsf{P}(\mathsf{P}(\mathbf{L_2}))$ that assigns to each set of input facts in $\mathbf{L_1}$ a set of sets of formulae in $\mathbf{L_2}$.

b) A multi-interpretation operator **MI** is *non-inclusive* if for all $\mathbf{X} \subseteq \mathbf{L_1}$ and $\mathbf{S}, \mathbf{T} \in \mathbf{MI(X)}$, if $\mathbf{S} \subseteq \mathbf{T}$ then $\mathbf{S} = \mathbf{T}$.

c) If $\mathbf{L_1} \subseteq \mathbf{L_2}$, then a multi-interpretation operator **MI** is *conservative* if for all $\mathbf{X} \subseteq \mathbf{L_1}$, $\mathbf{T} \in \mathbf{MI(X)}$ it holds $\mathbf{X} \subseteq \mathbf{T}$.

The condition of non-inclusiveness guarantees a relative maximality of the possible interpretations. Note that when **MI(X)** has exactly one element, this means that the set $\mathbf{X} \subseteq \mathbf{L_1}$ has a unique interpretation under **MI**. The notion of multi-interpretation operator is a generalization of the notion of a nonmonotonic belief set operator, as introduced in (Engelfriet, Herre, and Treur, 1998). The generalization was introduced and applied to approximate classification in (Engelfriet and Treur, 2003). A reasoner may explore a number of possible interpretations, but often, at some point in time a reasoner will focus on one (or possibly a small subset) of the interpretations. This selection process is formalized as follows (see Engelfriet and Treur, 2003).

**Definition 2 (Selection Operator)**

a) A *selection operator* **s** is a function $\mathbf{s} : \mathsf{P}(\mathsf{P}(\mathbf{L})) \rightarrow \mathsf{P}(\mathsf{P}(\mathbf{L}))$ that assigns to each nonempty set of interpretations a nonempty subset: for all **A** with $\phi \neq \mathbf{A} \subseteq \mathsf{P}(\mathbf{L})$ it holds $\phi \neq \mathbf{s(A)} \subseteq \mathbf{A}$. A selection operator **s** is *single-valued* if for all non-empty **A** the set **s(A)** contains exactly one element.

b) A *selective interpretation operator for* the multi-interpretation operator **MI** is a function $\mathbf{C} : \mathsf{P}(\mathbf{L_1}) \rightarrow \mathsf{P}(\mathbf{L_2})$ that assigns one interpretation to each set of initial facts: for all $\mathbf{X} \subseteq \mathbf{L_1}$ it holds $\mathbf{C(X)} \in \mathbf{MI(X)}$.

It is straightforward to check that if $\mathbf{s} : \mathsf{P}(\mathsf{P}(\mathbf{L_1})) \rightarrow \mathsf{P}(\mathsf{P}(\mathbf{L_2}))$ is a single-valued selection operator, then a selective interpretation operator **C** for multi-interpretation operator **MI** can be defined by the composition of **MI** and **s**, i.e., by setting $\mathbf{C(X)} = \mathbf{s(MI(X))}$ for all $\mathbf{X} \subseteq \mathbf{L_1}$.

In this section some interpretations that play a role in the analysis of the plane crash accident are taken as the leading example. The part chosen focuses on the ATC and its interaction to the operator. This information was derived based on training material, see (NIBRA, 2001). An issue is the difference in opinion as to whether or not the ATC communicated to the operator that there are more than 25 people on board. Initial observations of the ATC are:

observation(plane_crash, pos),
observation(cargo_plane, pos),
observation(passengers_on_board, pos).

Focusing on the ATC, the analysis results in two interpretations that differ only in the communication to the operator, formalized as follows:

***Common part of the interpretations***

observation(passengers_on_board,pos)
observation(cargo_plane,pos)
observation(plane_crash,pos)
belief(plane_crash_occurred,pos)
belief(passenger_count(more_than_25),pos)
not belief(passenger_count(maximum_4),pos)
not belief(passenger_count(unknown),pos)
action(communicate_to(plane_crash,operator),pos)
action(communicate_to(call_backup_via_06_11,operator),pos)

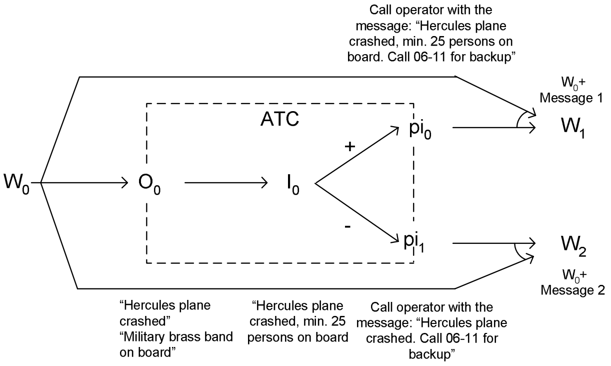

Call operator with the message: "Hercules plane crashed, min. 25 persons on board. Call 06-11 for backup"
$W_0$+ Message 1
ATC
$W_0$
$O_0$
$I_0$
+
-
$pi_0$
$pi_1$
$W_1$
$W_2$
$W_0$+ Message 2
"Hercules plane crashed" "Military brass band on board"
"Hercules plane crashed, min. 25 persons on board
Call operator with the message: "Hercules plane crashed. Call 06-11 for backup"

Figure 1 Reasoning Traces based on Interpretations

***Interpretation 1: common part +***

action(communicate_to(passenger_count(more_than_25),operator),pos)
not action(communicate_to(passenger_count(maximum_4),operator),pos)
not action(communicate_to(passenger_count(unknown),operator),pos)

***Interpretation 2: common part +***

not action(communicate_to(passenger_count(more_than_25),operator),pos)
not action(communicate_to(passenger_count(maximum_4),operator),pos)
not action(communicate_to(passenger_count(unknown),operator),pos)

Figure 1 provides an overview of ATC's first decision making. The figure shows the state of the world at time 0, $W_0$, and as a consequence of the communication to the operator, $W_1$ and $W_2$, which correspond with the two interpretations above. In the figure a difference is made between the observation ($O_0$), the internal representation made from that ($I_0$), and the interpretation of the situation in terms of actions to take ($pi_0$ and $pi_1$). Note that there are two moments of interpretation: from observations into internal representation, and from internal representation into actions to take.

## 4. Representing Interpretation in Default Logic

The *representation problem* for a nonmonotonic logic is the question whether a given set of possible outcomes of a reasoning process can be represented by a theory in this logic. More specifically, representation theory indicates what are criteria for a set of possible outcomes, for example, given by a collection of deductively closed sets of formulae, so that this collection can occur as the set of outcomes for a theory in this nonmonotonic logic. In (Marek, Treur and Truszczynski, 1997) the representation problem is solved for default logic, for the finite case. Given this context, in the current paper Default Logic is chosen to represent interpretation processes. For the empirical material analyzed, default theories have been specified such that their extensions are the possible interpretations.

A *default theory* is a pair **⟨D, W⟩**. Here **W** is a finite set of logical formulae (called the background theory) that formalize the facts that are known for sure, and **D** is a set of default rules. A default rule has the form: **$\alpha$: $\beta_1$, .., $\beta_n$ / $\gamma$**. Here **$\alpha$** is the precondition, it has to be satisfied before considering to believe the conclusion **$\gamma$**, where the **$\beta$**s, called the justifications, have to be consistent with the derived information and **W**. As a result **$\gamma$** might be believed and more default rules can be applied. However, the end result (when no more default rules can be applied) still has to be consistent with the justifications of all applied default rules. For convenience we only consider **n = 1**. Moreover, in the examples, *normal default theories* will be used: based on defaults of the form **$\alpha$: $\beta$ / $\beta$**. For more details on Default Logic, such as the notion of extension, see, e.g., (Reiter, 1980; Marek and Truszczynski, 1993). For the possible interpretations presented in Section 3, the following Default Theory has been specified.

***Set of defaults D***

```
{observation(plane_crash, pos) : belief(plane_crash_occurred, pos) /
 belief(plane_crash, pos) }
{observation(plane_crash, pos) ∧ observation(cargo_plane, pos) ∧
  observation(passengers_on_board, pos) :
 belief(passenger_count(more_than_25), pos) /
 belief(passenger_count(more_than_25), pos) }
{observation(plane_crash, pos) ∧ observation(cargo_plane, pos) ∧
  ¬observation(passengers_on_board, pos) :
 belief(passenger_count (maximum_4), pos) /
 belief(passenger_count (maximum_4), pos) }
{observation(plane_crash, pos) ∧ observation(cargo_plane, pos) ∧ \
  ¬observation(passengers_on_board, pos) :
 belief(passenger_count (unknown), pos) /
 belief(passenger_count (unknown), pos) }
{belief(plane_crash_occurred, pos) :
 action(communicate_to(plane_crash, operator), pos) /
 action(communicate_to(plane_crash, operator), pos) }
{belief(plane_crash_occurred, pos) ∧
 belief(passenger_count(PN:PASSENGER_NUMBER), pos) :
 action(communicate_to(passenger_count(PN:PASSENGER_NUMBER), operator), pos) /
 action(communicate_to(passenger_count(PN:PASSENGER_NUMBER), operator), pos) }
{belief(plane_crash_occurred, pos) :
 ¬action(communicate_to(passenger_count(PN:PASSENGER_NUMBER), operator), pos) /
 ¬action(communicate_to(passenger_count(PN:PASSENGER_NUMBER), operator), pos)}
{belief(plane_crash_occurred, pos) ∧ belief(passenger_count(more_than_25), pos) :
 action(communicate_to(call_backup_via_06-11, operator), pos) /
 action(communicate_to(call_backup_via_06-11, operator), pos) }
```

***Background theory W***

```
observation(plane_crash, pos).
observation(cargo_plane, pos).
observation(passengers_on_board, pos).
belief(passenger_count (unknown), pos) →
  ¬belief(passenger_count (maximum_4), pos) ∧
  ¬belief(passenger_count(more_than_25), pos)
belief(passenger_count (maximum_4), pos) →
 ¬belief(passenger_count (unknown), pos) ∧
 ¬belief(passenger_count(more_than_25), pos)
belief(passenger_count (more_than_25), pos) →
 ¬belief(passenger_count (unknown), pos) ∧
 ¬belief(passenger_count(maximum_4), pos)
action(communicate_to(passenger_count (unknown), operator), pos) →
 ¬action(communicate_to(passenger_count (maximum_4), operator), pos) ∧
 ¬action(communicate_to(passenger_count(more_than_25), operator), pos)
action(communicate_to(passenger_count (maximum_4), operator), pos) →
 ¬action(communicate_to(passenger_count (unknown) , operator), pos) ∧
 ¬action(communicate_to(passenger_count(more_than_25), operator), pos)
action(communicate_to(passenger_count (more_than_25), operator), pos) →
 ¬action(communicate_to(passenger_count (unknown), operator), pos) ∧
 ¬action(communicate_to(passenger_count(maximum_4), operator), pos)
```

## 5. Simulation by Temporalized Default Rules

In this section, a generic simulation model for default reasoning is specified (based on the executable temporal LEADSTO language; cf. Bosse et al., 2005), and applied to the case study. As discussed in Section 3, to formalise one reasoning trace in a multiple interpretation situation, a certain selection has to be made, based on control knowledge which serves as a parameter for the interpretation to be achieved. Variants of Default Logic in which this can be expressed are Constructive Default Logic (Tan and Treur, 1992) and Prioritized Default Logic (Brewka, 1994; Brewka and Eiter, 1999). A *Prioritized Default Theory* is a triple **⟨D,W, <⟩**, where **⟨D,W⟩** is a Default Theory and **<** is a strict partial order on **D**. *Constructive Default Logic*, see (Tan and Treur, 1992), is a Default Logic in which selection functions are used to control the reasoning process. Selection functions take the set of consequents of possibly applicable defaults and select one or a subset of them. A selection function can represent one of the different ways to reason from the same set of defaults, and thus serves as a parameter for different reasoning traces (achieving different interpretations). This knowledge determines a selection operator (see Section 3).

The generic simulation model for default reasoning described below is an executable temporal logical formalization of Constructive Default Logic, based on the temporal perspective on default and nonmonotonic reasoning as developed in (Engelfriet and Treur, 1998). The input of the model is (1) a set of normal default rules, (2) initial information, and (3) knowledge about the selection of conclusions of possibly applicable rules. The output is a trace which describes the dynamics of the reasoning process over time. Globally, the model can be described by a generate-select mechanism: first all possible (default) assumptions (i.e., candidate conclusions) are generated, then one conclusion is selected, based on selection knowledge. After selection, the reasoning process is repeated. In the executable temporal logical language LEADSTO, the generic default reasoning model can be described by the following local dynamic properties (LPs):

### LP1 Candidate Generation

If I have derived (x,s1), and I have a default rule that allows me to assume (y,s2), and I do not have any information about the truth of y yet, then (y,s2) will be considered a possible assumption.

∀x,y:info_element ∀s1,s2:sign
 derived(x, s1) ∧ default_rule(x, s1, y, s2, y, s2) ∧ not derived(y, pos) ∧
 not derived(y, neg) $\rightarrow_{0,0,1,1}$ possible_assumption(y, s2)

Note that the sort sign consists of the elements pos and neg.

### LP2 Candidate Comparison

Each possible assumption is a better (or equally good) candidate than itself.

∀x:info_element ∀s:sign
  possible_assumption(x, s) $\rightarrow_{0,0,1,1}$ better_candidate_than(x, s, x, s)

If (x,s1) is a possible assumption, and (y,s2) is no possible assumption, then (x,s1) is a better candidate than (y,s2).

∀x,y:info_element ∀s1,s2:sign
  possible_assumption(x, s1) ∧ not possible_assumption(y, s2) $\rightarrow_{0,0,1,1}$
  better_candidate_than(x, s1, y, s2)

If (x,s1) is a possible assumption, and (y,s2) is a possible assumption, and it is known that deriving (x,s1) has priority over deriving (y,s2), then (x,s1) is a better candidate than (y,s2).

∀x,y:info_element ∀s1,s2:sign
  possible_assumption(x, s1) ∧ possible_assumption(y, s2) ∧
  priority_over(x, s1, y, s2) $\rightarrow_{0,0,1,1}$ better_candidate_than(x, s1, y, s2)

### LP3 Candidate Selection

If (x,s1) is a possible assumption, and it is the best candidate among all possible assumptions, then it will be derived.

∀x:info_element ∀s1:sign
  possible_assumption(x, s1) ∧ [∀y:info_element ∀s2:sign
  better_candidate_than(x, s1, y, s2) ] $\rightarrow_{0,0,1,1}$ derived(x, s1)

### LP4 Persistence

If (x,s) is derived, then this will remain derived.

∀x:info_element ∀s:sign
  derived(x, s) $\rightarrow_{0,0,1,1}$ derived(x, s)

The generic default reasoning model described has been used to simulate the reasoning process as performed by the Air Traffic Controller in the Hercules disaster (see Section 2). An example simulation trace is shown in Figure 2. In this figure, time is on the horizontal axis, and different states are on the vertical axis. A dark box on top of a line indicates that a state property is true; a light bow below a line indicates that it is false. As shown in Figure 2, there are initially three important aspects of the world: the fact that there is a plane crash, that it involves a cargo plane, and that there are passengers on board. At time point 1, the ATC correctly observes these three information elements. Next, he starts the interpretation process: according to his default rules, he generates two possible assumptions: there is a plane crash, and the passenger count is over 25. Based on his selection knowledge, first the former assumption is derived (time point 4: derived(belief(plane_crash, pos), pos)). As the latter possible assumption does not conflict with the former, the possible assumption that the passenger count is over 25 is derived as well (see time point 11). Next, the ATC generates four possible assumptions on actions: (1) communicating that there is a plane crash, (2) communicating that the emergency number 06-11 should be called, (3) communicating that the passenger count is over 25, and (4) *not* communicating that the passenger count is over 25. The first two possible actions are translated to actions; after that, the ATC selects the conclusion *not* communicating the passenger count over the conclusion for communicating the passenger count; thus, this information does not reach the operator.

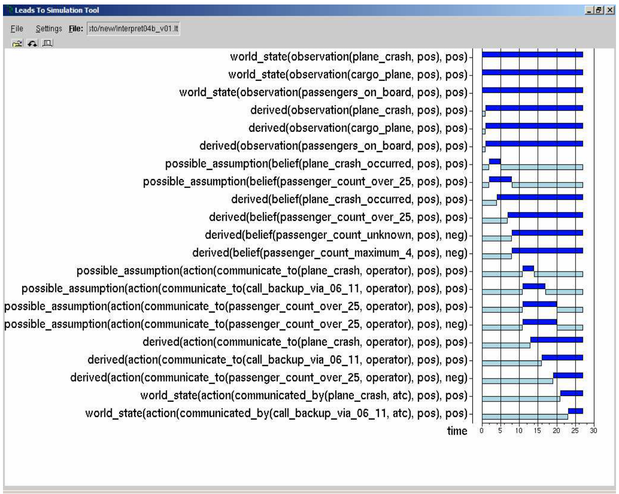

Leads To Simulation Tool
File
Settings
File: sto/new/interpret04b_v01.lt
world_state(observation(plane_crash, pos), pos)
world_state(observation(cargo_plane, pos), pos)
world_state(observation(passengers_on_board, pos), pos)
derived(observation(plane_crash, pos), pos)
derived(observation(cargo_plane, pos), pos)
derived(observation(passengers_on_board, pos), pos)
possible_assumption(belief(plane_crash_occurred, pos), pos)
possible_assumption(belief(passenger_count_over_25, pos), pos)
derived(belief(plane_crash_occurred, pos), pos)
derived(belief(passenger_count_over_25, pos), pos)
derived(belief(passenger_count_unknown, pos), neg)
derived(belief(passenger_count_maximum_4, pos), neg)
possible_assumption(action(communicate_to(plane_crash, operator), pos), pos)
possible_assumption(action(communicate_to(call_backup_via_06_11, operator), pos), pos)
possible_assumption(action(communicate_to(passenger_count_over_25, operator), pos), pos)
possible_assumption(action(communicate_to(passenger_count_over_25, operator), pos), neg)
derived(action(communicate_to(plane_crash, operator), pos), pos)
derived(action(communicate_to(call_backup_via_06_11, operator), pos), pos)
derived(action(communicate_to(passenger_count_over_25, operator), pos), neg)
world_state(action(communicated_by(plane_crash, atc), pos), pos)
world_state(action(communicated_by(call_backup_via_06_11, atc), pos), pos)
time
0
5
10
15
20
25
30

Figure 2. Simulation trace of the reasoning of the ATC.

It is important to note that the trace shown in Figure 2 corresponds to one possible course of affairs. This means that it corresponds to one path through Figure 1, which is in this case the path $W_0$ - $O_0$ - $I_0$ - $pi_1$ - $W_2$. In default reasoning terms, the trace eventually results in one extension for the set of default rules shown in Section 3. By changing the selection knowledge, different extensions are generated. Although in this paper only one partial example is shown (due to space limitations), the complete reasoning processes of four different parties involved in the Hercules disaster have been modeled. Moreover, for all of these reasoning processes, all different settings of selection knowledge have systematically been selected. This way, a large number of traces have been generated, which together cover all possible reasoning traces based on multiple interpretations for this domain, including the (non-optimal) ones reported in the empirical material.

## 6. Verification of Properties for Traces

section addresses the automated verification of properties against two types of traces. First of all, traces that include full information are addressed. In these traces, the interpretation of the particular agent under analysis is available as well as the observations and actions performed by the agent. The second type of trace addressed is a trace merely consisting of the external information (i.e. observations and actions). Note that all of these properties are specified independent of the specific case study, and can therefore easily be reused.

### 6.1 Analysis of Complete Traces

Verification of a simulated or empirical default reasoning trace including complete information can address a number of aspects. First it can address whether all conclusions in the trace are grounded by justified application of default rules. Next it can be verified whether the process has been exhaustive, i.e., whether for all applicable default rules the conclusion occurs. These properties have been given a temporal form (in the spirit of Engelfriet and Treur, 1998), and specified in the temporal predicate logical language TTL cf. (Bosse et al., 2006). All of these properties have been checked automatically and shown to be satisfied for traces as the one presented in Figure 2, using the TTL Checker environment.

**groundedness(γ:TRACE):**
  ∀t:TIME, i:info_element, s:sign
  [state(γ, t) |= derived(i, s) ⇒  grounded (γ,i,s,t) ]

**grounded(γ:TRACE, i:info_element, s:sign, t:TIME):**

```
  [follows_from_default(γ,i,s,t) ∨  follows_from_strict_constraint(γ,i,s,t) ∨
   world_fact(γ,i,s,t)]
```

**world_fact(γ:TRACE, i:info_element, s:sign, t:TIME):**

```
  ∃t2:TIME < t  state(γ, t2) |= world_state(i, s)
```

**follows_from_strict_constraint(γ:TRACE, i:info_element, s:sign, t:TIME):**

```
  ∃C:CONJUNCTION, t2:TIME < t  [ state(γ, t2) |= strict_constraint(C, i, s) &
  ∀i2:info_element,s2:sign [ element_of(i2, s2, C) ⇒
                                   state(γ, t2) |= derived(i2, s2) ] ]
```

Note that elements of the sort CONJUNCTION refer to conjunctions of <info_element, sign> pairs.

**follows_from_default(γ:TRACE, i:info_element, s:sign, t:TIME):**

```
  ∃t2:TIME < t, C:CONJUNCTION
  [state(γ, t2) |= default_rule(C, i, s, i, s) &
   ∀i1:info_element,s1:sign
    [ element_of(i1, s1, C) ⇒  state(γ, t2) |= derived(i1, s1) ]
   & ∀t3≥t ∀s'≠ s  not state(γ, t3) |= derived(i, s')
```

**consistency(γ:TRACE):**

```
  ∀i:info_element, s:sign, t:TIME
  [ state(γ,t) |= derived(i, s) ⇒
   ¬∃t2:TIME, s2:sign [s ≠ s2 & state(γ,t2) |= derived(i, s2)] ]
```

**exhaustiveness(γ:TRACE):**

```
  ∀t:TIME, i:info_element, s:sign, C:CONJUNCTION
  [state(γ, t) |= default_rule(C, i, s, i, s) &
   ∀i2:info_element,s2:sign [ element_of(i2, s2, C) ⇒
                                   state(γ, t) |= derived(i2, s2) ] &
  ¬∃t2:TIME, s3:sign [s ≠ s3 & state(γ, t2) |= derived(i, s3)]
  ⇒ ∃t3:TIME [state(γ, t3) |= derived(i, s)]
```

**derived_persistency(γ:TRACE):**

```
  ∀t1, t2 [ state(γ, t1) |= derived(i, s) & t1<t2 ⇒ state(γ, t2) |= derived(i, s) ]
```

These verification properties assume that all information is fully available, including the interpretation that has been derived. In empirical traces however, such information might not be present. Such information could be obtained by interviews and added to the traces, but this does not always give an adequate representation of reality, since people tend to avoid admitting mistakes in incident management. The following section shows how properties can be verified for empirical traces, without having knowledge on the interpretation. In addition, it specifies properties on correctness of interpretation based upon selection of the most specific default rule.

## 6.2 Analysis of Externally Observable Traces

In this section verification properties are specified assuming traces that merely consist of the observations received by the agent, and the actions that have been performed by the agent. Note that conflicting observations at the same time point are not allowed. Several different properties are identified. First of all, a derivable interpretation is defined, which is simply an interpretation that can be derived based upon the observations received, and a default rule:

**derivable_int(γ:TRACE, t:TIME, C:CONJUNCTION, i:info_element, s:sign):**

```
  state(γ, t) |= default_rule(C, i, s, i, s) &
   ∀i2:info_element, s2:sign
   [ element_of(i2, s2, C) ⇒  ∃t':TIME ≤ t
     [ state(γ, t') |= observation(i2, s2) &
   ¬[∃s3:SIGN, t'':TIME ≤ t & t'' ≥ t'
       [ state(γ, t'') |= observation(i2, s3) & s2 ≠ s3 ] ] ] ]
```

An interpretation is considered to be correct if it follows from the most specific default rule that can be applied:

**most_specific_int(γ:TRACE, t:TIME, i:info_element, s:sign):**

```
∃C:CONJUNCTION
  [ derivable_int(γ, t, C, i, s) &
  ∀C2:CONJUNCTION ≠ C, s2:SIGN
     [ derivable_int(γ, t, C2, i, s2) & s ≠ s2 ⇒
       size(C2) < size(C) ] ]
```

Based upon such most specific interpretations, actions to be performed can be derived:

**derivable_ac(γ:TRACE, t:TIME, C:CONJUNCTION, i:info_element, s:sign):**

```
  state(γ, t) |= default_rule(C, i, s, i, s) &
   ∀i2:info_element, s2:sign
   [ element_of(i2, s2, C) ⇒ most_specific_int(γ, t, i2, s2) ]
```

An action is considered to be correct in case it follows from the most specific default rule that is applicable:

**most_spec_ac(γ:TRACE, t:TIME, i:info_element, s:sign):**

```
∃C:CONJUNCTION
  [ derivable_ac(γ, t, C, i, s) &
  ∀C2:CONJUNCTION ≠ C, s2:SIGN
     [ derivable_ac(γ, t, C2, i, s2) & s ≠ s2 ⇒
       size(C2) < size(C) ] ]
```

Given the fact that it can now be derived what the correct actions are, properties can be verified against empirical traces to investigate the performance shown in that empirical trace. A first property which can be verified is whether the correct actions have been performed in the empirical trace without taking to much time to start the performance of this action (i.e. within duration d):

**correct_action(γ:TRACE, t:TIME, i:info_element, s:sign, d):**

```
 [ most_spec_ac(γ, t, i, s) &
  [ ¬∃t':TIME < t  most_spec_ac(γ, t', i, s) ] &
  [ ¬∃t'':TIME > t & t'' < t + d  ¬most_spec_ac(γ, t'', i, s) ] ]
  ⇒ ∃t''':TIME ≥ t & t'''≤ t + d [ state(γ, t''') |= world_state(i, s) ]
```

Of course, things do not necessarily run so smoothly, therefore, detection of errors is of crucial importance. An error first of all occurs when an action is not performed that should have been performed according to the correct interpretation:

**missing_action(γ:TRACE, t:TIME, i:info_element, s:sign, d):**

```
 most_spec_act(γ, t, i, s) &
 [ ¬∃t':TIME < t  most_spec_ac(γ, t', i, s) ] &
 [ ¬∃t'':TIME > t & t'' < t + d  ¬most_spec_ac(γ, t'', i, s) ] &
 [ ¬∃t''':TIME ≥ t  & t'''≤ t + d [ state(γ, t''') |= world_state(i, s) ]
```

Furthermore, an error occurs when an action can be performed that is not derivable from the correct interpretation:

**incorrect_action(γ:TRACE, t:TIME, i:info_element, s:sign, d):**

```
state(γ, t) |= world_state(i, s) &
¬∃t':TIME ≤ t  & t' ≥ t – d [ most_spec_ac(γ, t', i, s) ]
```

The properties specified above have been automatically verified against the empirical trace of the Hercules disaster. The analysis shows that the correct_action property is not satisfied for the Hercules disaster trace, due to the fact that the trace does not show that the ATC has passed the information on the number of people on board of the plane. As a result, the missing_action property holds. Finally, the incorrect_action property is not satisfied, as only missing actions occur in the trace. These results comply to the human analysis of the Hercules disaster.

## 7. Conclusion

This paper shows how a number of known techniques and tools developed within the area of nonmonotonic logic and AI can be applied to analyze empirical material on human reasoning and interpretation within Cognitive Science; cf. (Stenning and van Lambalgen, 2006). The formal techniques exploited in the empirical analysis approach put forward are:

(1) multi-interpretation operators as an abstract formalization of reasoning towards an interpretation,

(2) default logic to specify a multi-interpretation operator,

(3) a temporalized default logic to specify possible reasoning traces involved in a multi-interpretation process,

(4) an executable temporal logical language to specify a generic executable default reasoning model to simulate such reasoning traces, and

(5) an expressive temporal logical language to specify and verify properties for reasoning traces

It has been shown how indeed these techniques and tools obtain an adequate formalization and analysis of empirical material on human reasoning in critical situations in incident management. Simulated traces have been generated, compared to the given empirical traces and found adequate. Relevant properties of both simulation as well as empirical traces have been verified and results were shown of this verification process. The properties and default rules presented in this paper have all been specified in a generic fashion, such that they can easily be reused for studying other cases.

The presented approach can be used to enable automated detection of interpretation errors in incident management. Such detection could potentially avoid unwanted chains of events which might result in catastrophic consequences. As a first case study to investigate the suitability of the presented approach for this purpose, the Hercules disaster has been used, showing promising results. The disaster is representative for many of the disasters that occur. It is however future work to perform a more thorough evaluation, using a variety of cases.

Note that the executable temporal logical language LEADSTO, which was used for simulation in Section 5, is not the only language that can be used for this purpose. Also other languages and tools are suitable, such as SModels, a system for answer set programming in which a specification can be written in (an extended form of) logic programming notation, see (Niemelä et al., 2000).

## References

Bosse, T., Jonker, C.M., Meij, L. van der, Sharpanskykh, A., and Treur, J. (2006). Specification and Verification of Dynamics in Cognitive Agent Models. In: Proceedings of the Sixth International Conference on Intelligent Agent Technology, IAT'06. IEEE Computer Society Press, 2006, pp. 247-254.

Bosse, T., Jonker, C.M., Van der Meij, L, and Treur, J. (2005) LEADSTO: a Language and Environment for Analysis of Dynamics by SimulaTiOn. In: Eymann, T., et al. (eds.), *Proc. of MATES'05*. Lecture Notes in AI, vol. 3550. Springer Verlag, pp. 165-178. Extended version in: *Int. J. of AI Tools*, 2007.

Braine, M.D.S., and O'Brien, D.P. (eds.) (1998). *Mental Logic*. Lawrence Erlbaum, London.

Brewka, G. (1994), Adding priorities and specificity to default logic. In: MacNish, C., Pereira, L., and Pearce, D., (eds.), *Proc. of JELIA'94*, LNAI, vol. 838. Springer Verlag, pp. 247-260.

Brewka, G., and Eiter, T. (1999), Prioritizing Default Logic: Abridged Report. In *Festschrift on the occasion of Prof.Dr. W. Bibel's 60th birthday*. Kluwer.

Engelfriet, J., Herre, H., and Treur, J. (1998), Nonmonotonic Reasoning with Multiple Belief Sets, *Annals of Mathematics and Artificial Intelligence,* vol. 24, pp. 225-248.

Engelfriet, J., and Treur, J. (1998), An Interpretation of Default Logic in Minimal Temporal Epistemic Logic. *Journal of Logic, Language and Information*, vol. 7, pp. 369-388.

Engelfriet, J., and Treur, J. (2003), Multi-Interpretation Operators and Approximate Classification. *Int. Journal of Approximate Reasoning*, vol. 32, pp. 43-61.

Johnson-Laird, P.N. (1983). *Mental Models*. Cambridge: Cambridge University Press.

Johnson-Laird, P.N., and Byrne, R.M.J. (1991). *Deduction.* Hillsdale, NJ:Erlbaum.

Marek, V.W., and Truszczynski, M. (1993), *Nonmonotonic Logics*. Springer Verlag.

Marek, V.W., Treur, J., and Truszczynski, M. (1997), Representation Theory for Default Logic. *Annals of Mathematics and AI*, vol. 21, pp. 343-358.

Ministry of the Interior (1996), *Airplane Crash Airbase Eindhoven 15th July 1996* (in Dutch), SDU Publishers, The Hague.

NIBRA (Netherlands Institute for Fire Service and Disaster Management) (2001), *Fire Watch First Class: Airplane Fire Fighting (in Dutch).*

Niemelä, I., Simons, P., and Syrjänen, T. (2000). Smodels: a system for answer set programming. In: Proceedings of the 8th International Workshop on Non-Monotonic Reasoning, Breckenridge, Colorado, USA, April 2000.

Rips, L.J. (1994). *The Psychology of Proof: Deductive reasoning in human thinking*. MIT Pres, Cambridge, Mass.

Stenning, K., and van Lambalgen, M. (2006), A working memory model of relations between interpretation and reasoning. *Cognitive Science Journal*, Elsevier Science Inc., Oxford, UK. In press.

Reiter, R. (1980) A logic for default reasoning. Artificial Intelligence, 13:81-132.

Tan, Y.H., and Treur, J. (1992) Constructive Default Logic and the Control of Defeasible Reasoning. In: B. Neumann (ed.), *Proc. ECAI'92*. Wiley and Sons, 1992, pp. 299-303. Extended version: Allis, V.E., Tan, Y.H., and Treur, J., Meta-level Selection Techniques for the Control of Default Reasoning. *Future Generation Computer Systems*, vol. 12, pp. 189-201.

# Towards Human-Like Robustness in an Intelligent Tutoring System

**Hameedullah Kazi (hameedullah.kazi@ait.ac.th)**
**Peter Haddawy (haddawy@ait.ac.th)**
Computer Sciences and Information Management Program
Asian Institute of Technology
PO Box 4, Klong Luang, Pathumthani 12120, Thailand

**Siriwan Suebnukarn (ssiriwan@tu.ac.th)**
School of Dentistry
Thammasat University
Khong Luang, Pathumthani 12121, Thailand

## Abstract

Intelligent tutoring systems are no different from other knowledge based systems in that they are often plagued by brittleness. Intelligent tutoring systems for problem solving are typically loaded with problem scenarios for which specific solutions are constructed. Solutions presented by students, are compared against these specific solutions, which often leads to a narrow scope of reasoning, where students are confined to reason towards a specific solution. Student solutions that are different from the specific solution entertained by the system are rejected as being incorrect, even though they may be acceptable or close to acceptable. This leads to brittleness in tutoring systems in evaluating student solutions and returning appropriate feedback. In this paper we discuss a few human-like attributes in the context of robustness that are desirable in knowledge based systems. We then present a model of reasoning through which a tutoring system for medical problem-based learning, can begin to exhibit human-like robust behavior in evaluating solutions in a broader context using UMLS, and respond with hints that are mindful of the partial correctness of the student solution.

## Introduction

While traditional knowledge based systems often work well for narrowly defined tasks within specialized domains, they lack the meta-cognition and human-like common sense to deal with unforeseen situations. Many systems suffer from brittleness and they are often unaware of their own limitations (McCarthy, 1984). It is normal for a complex system to fail at some point, however what makes a system brittle is that it shows sudden failure beyond a certain point. Human beings also fail, however they are able to establish some self recovery before their failure leads to catastrophe (Nielsen et al., 2002). Thus the failure humans exhibit is often soft and gradual rather than being hard and sudden.

The need to emulate human-like behavior in intelligent systems has often led to an examination of how the human mind works. Minsky (1986) describes a possible explanation of how in the event of damage to some parts of the brain, significant functionality is still maintained, by the delegation of tasks to other parts that have not suffered damage. In other words, the failure of some sub-systems leads to a task delegation to other sub-systems, thereby resulting in some degree of robustness.

Sloman (1996) has argued that the human mind employs a combination of rule based and heuristic methods for reasoning, where rule-based methods are characterized as systematic and logical set of laws, while heuristic methods are based on principles of association, similarity and contiguity. Some researchers have advocated the use of heuristic methods as a solution to the problem of brittleness in knowledge based systems. Accurate results may not be achievable where factual knowledge is found to be insufficient or the knowledge base is known to contain gaps, in which case heuristic methods can be employed to achieve partially correct, if not fully accurate results (Paritosh, 2006). These heuristic methods should be able to exploit the knowledge structure of the knowledge based system to provide reasonable answers.

In the next few sections we describe how the issues of gradual failure, self analysis of limitations, self recovery, task delegation and the use of multiple modes of reasoning in the context of robustness, can be applied to an intelligent tutoring system for medical problem-based learning (PBL) using UMLS (U.S. National Library of Medicine, 2007), which is a collection of various medical ontologies.

## Robust Output Quality

A knowledge based system is designed to respond to input which has a specific format and is confined to a certain scope of knowledge. If the input happens to fall outside this scope, the output quality is expected to deteriorate. Groot, Teiji & Harmelen (2005) describe how a quantitative analysis of the robustness of knowledge based systems can be achieved. They outline a few definitions of robustness, one of which is that the output quality of a knowledge based system should decrease monotonically with decrease in input quality. They mention that while this demand may be practically too strong, a system that exhibits somewhat monotonic output may be considered robust. They also argue that the rate of output quality change in a robust knowledge based system, should be slow. A knowledge based system that is brittle, will exhibit abrupt degradation in its output quality as the input quality deteriorates beyond a certain point. However, a robust system will show a smooth degradation in its output quality as the input quality deteriorates beyond the edge of the system knowledge as shown in Figure 1.

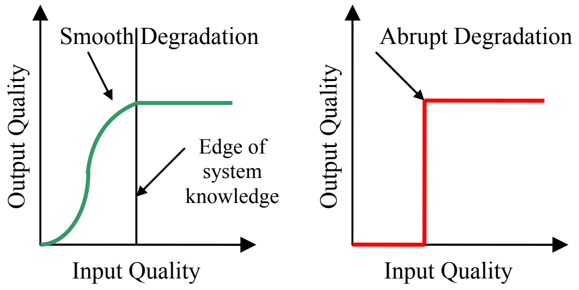


Figure 1: Smooth vs. Abrupt Degradation

## Reasoning Scope in Medical Tutoring Systems

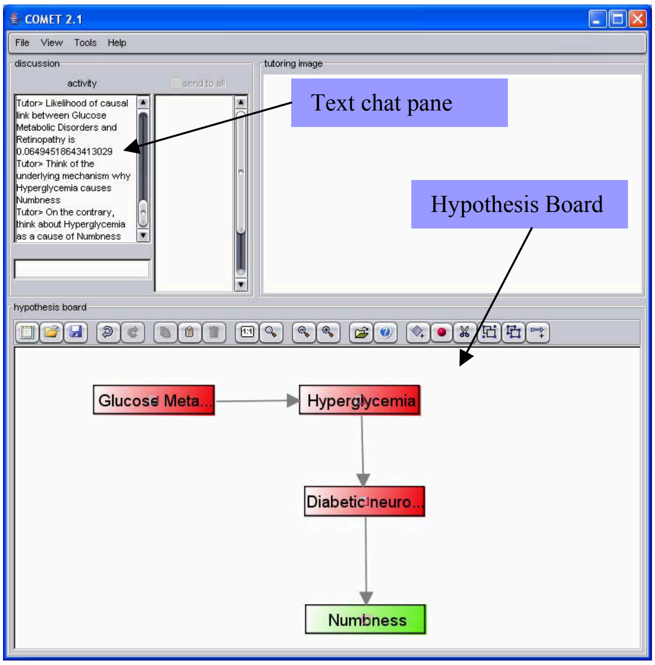
Intelligent tutoring systems can be considered knowledge based systems whose problem solving activity is to evaluate student solutions to a posed problem and provide feedback to the students in the form of hints. The task of generating intelligent hints that are suited to the knowledge level of the student, is addressed in many tutoring systems (Kabassi, Virvou & Tsihrintzis, 2006; Suebnukarn & Haddawy, 2006) as part of student modeling. However the task of evaluating student solutions in a broad scope of reasoning is yet to be addressed in sufficient depth. Tutoring systems that offer some latitude in accepting differing solutions often confine students to a narrow scope of solution representation. Crowley & Medvedeva, (2006) accept a broad range of solutions for a given problem, but students are restricted to a local and customized ontology for choosing their solution concepts. Lulis, Michaels & Evens, (2004) emphasize the need for qualitative reasoning in tutoring systems and provide a mechanism through which students are able to present qualitative responses, however the response is only confined to assigning values to a small set of variables. The COMET system (Suebnukarn & Haddawy, 2006) provides an interface through which students can construct their hypothesis (solution) in the form of a directed acyclic graph. It evaluates a student hypothesis by comparing it against a specific expert solution. Nodes in the hypothesis that are not found in the expert solution are simply deflected and the system responds with the hint "<Node> is beyond the scope of this problem".

The responses of such tutoring systems in unanticipated situations are quite contradictory to how a human tutor would normally respond. If the student response happens to fall outside the scope of the tutoring system's knowledge, the system responds with a premeditated hint that is often oblivious of the partial correctness of the student response. At the same time these tutoring systems are devoid of the meta-cognitive ability to assess their own capability in order to inform the student of the system's limitations or to attempt self recovery.

This motivates the need to have a medical tutoring system that offers students a broad scope of hypothesis representation and at the same time offers an assessment of the student hypothesis that describes the quality or degree of correctness. The tutoring system should be able to respond with certainty when the knowledge base is found to be sufficient. However when the knowledge base is not found to be sufficient, the system should be able to exploit its knowledge structure to achieve partial if not complete results. Thus the system should exhibit a gradual deterioration in quality when its knowledge limit is reached. Such a tutoring system should also have the ability to assess its own limitations and be able to inform the students about these limitations, which can help the students to reason accordingly.

## Robustness Vis-à-Vis Tutoring Systems

The proposed tutoring system is designed to cover PBL in the medical domain. A PBL session typically comprises of a group of 6-8 students, who are given a problem to solve within a period of about two hours. Based on the description of the problem scenario posed to the students, they are expected to form their solution in the form of a hypothesis graph, where graph nodes represent medical concepts and directed edges indicate the cause effect relationships between respective nodes.



Tutor> Likelihood of causal link between Glucose Metabolic Disorders and Retinopathy is 0.06494518643413029
Tutor> Think of the underlying mechanism why Hyperglycemia causes Numbness
Tutor> On the contrary, think about Hyperglycemia as a cause of Numbness

Figure 2: System Prototype

We have developed a system prototype using java. The problem representation in our system is the same as that in COMET (Suebnukarn et al., 2006) of a directed acyclic graph to describe a hypothesis. The hypothesis graph is based on the Illness Script (Feltovich & Barrows, 1984), where nodes are enabling conditions, faults or consequences. The knowledge base of our system is formed by combining UMLS tables with an additional table that represents causal links between concepts. The system interface provides students with a workspace as a hypothesis board to form their hypothesis, along with a text chat pane through which the system returns feedback in the form of hints, as shown in Figure 2. For purposes of forming their hypothesis, students choose concepts from the diverse and widely available UMLS Metathesaurus (U.S. National Library of Medicine, 2007), as hypothesis nodes. For example, students are presented a problem scenario related to diabetes:

> "A 45-year-old woman came to the clinic with following symptoms: tiredness, always thirsty, voided frequently with large amount of urine for 4-5 months. She voided approximately 10 times during the day and 4-5 times during the night. She was hungry quite often but lost 5 kgs body weight during the past 4 months. She also had numbness, leukorrhea and delayed wound healing".

A student hypothesizes that *hyperglycemia* is a cause of *diabetic neuropathy* which is shown to be a cause of *numbness*, in Figure 2.

| | True Link | False Link |
|---|---|---|
| Accepted | ▲ | ● |
| Rejected | ● | ▲ |

| | True Link | False Link |
|---|---|---|
| Accepted with Reservation | ■ | ◆ |
| Rejected with Suggestive Hint | ◆ | ■ |

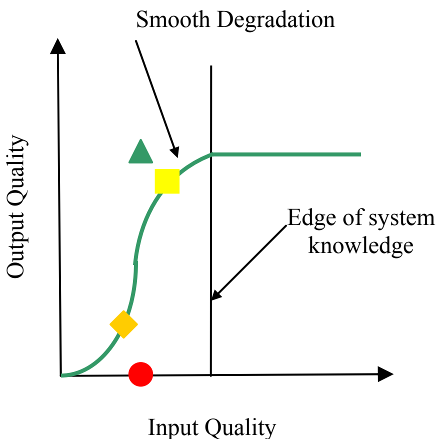


Figure 3: Desired Degradation Curve

Each hypothesis causal link drawn by the students needs to be evaluated against the knowledge base to determine whether the link drawn by the student should be accepted or rejected along with a hint to provide feedback. The output quality of an intelligent tutoring system is essentially comprised of two main components: evaluating student hypothesis and returning intelligent feedback as in the form of hints.

A causal link that is considered by a human tutor to be correct is henceforth referred to as a *true link*, whereas a causal link considered by a human tutor to be incorrect, is referred to as a *false link*. For all links that lie beyond the edge of the system knowledge, the output quality will be high if a *true link* is accepted or a *false link* is rejected by the system, as shown in Figure 3. However, if a *false link* is accepted without reservation or a *true link* is rejected without suggestive feedback that recognizes the partial correctness of the link, the output quality will be very low, as shown in Figure 3. Thus the output quality, without reservation or suggestive feedback in the hints, will be marked by fluctuating highs and lows. A system which produces fluctuating output quality as a result of deteriorating input quality is less predictable (Groot et al., 2005) and is considered less robust.

Therefore, for all hypothesis links that lie beyond the edge of the system knowledge, accepted links need to be supported with hints that show some form of reservation and suggest improvement to the causal link. Likewise rejected links need to be supported with suggestive hints that acknowledge partial correctness of the link or the closeness of the link to a true causal link, to result in somewhat smooth degradation as shown in Figure 3. The exact gradient of the curve shown in Figure 3 will be dictated by the nature of hints, as they vary from one situation to another.

## Three Tier Model for Robustness

Robustness in our system is made possible through the use of a broad and widely available medical knowledge source such as the UMLS. The system design towards maintaining human-like robustness comprises of a three tier model, as shown in Figure 4. The tiers are successively applied in order of necessity. The first tier is a rule-based expert knowledge base, while the second tier is a heuristic method of computing semantic distance using knowledge structure within UMLS, whereas the third tier is based on a probabilistic Bayesian model.

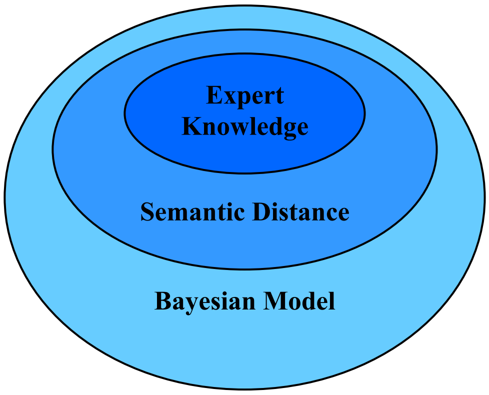


Figure 4: Three Tiers of Robust Reasoning

The system makes it a matter of priority to first employ the rule-based tier which contains sure knowledge for reasoning purposes. If the first tier fails to deliver, the system employs the heuristic mechanism in the second tier. If the second tier fails too, the system uses the robust but not so accurate, third tier of probabilistic Bayesian model. Thus the system applies a step wise fallback approach of employing multiple modes of reasoning that are designed to provide self recovery and smooth degradation in output quality with deteriorating input quality.

## Rule-Based Expert Knowledge Base

This knowledge base is in the form of a database table that comprises of sure knowledge which contains causal links such as:

*Diabetes* → *Hyperglycemia*
*Hypoinsulinism* → *Hyperglycemia*
*Glucose Metabolism Disorder* → *Hyperglycemia*
*Hyperglycemia* → *Diabetic Neuropathy*
*Diabetic Neuropathy* → *Numbness*

This knowledge base is formed through the collation of causal links found in expert solutions to various problems, and the causal links found in student solutions that are certified by the domain experts to be correct.

While evaluating a causal link between two concepts in the student hypothesis, the system first attempts to find the respective link in this knowledge base, as an attempt to use rule based certain knowledge. If the system finds the hypothesis link in this knowledge base, the link is accepted, knowing that this comes from part of the system's rule-based certain knowledge. Additionally, the system also checks to see if an indirect link between the two concepts is found or if there is a reverse link that exists between the respective concepts. However, if the link is not found in this knowledge base, the system resorts to the heuristic method in an attempt to achieve a partial if not completely accurate assessment of the link under evaluation.

## Heuristic Measure of Partial Correctness Using Semantic Distance

In this mode of reasoning, the system exploits the knowledge structure within UMLS to evaluate partial correctness of the causal link under evaluation, thereby attaining some degree of robustness. The node, from which the causal edge in the student hypothesis is emanating, is henceforth referred to as the *source node*, whereas the node, towards which the causal edge is leading to, is referred to as the *target node*. The system checks if either the *target node* or *source node* is found in any of the acceptable solutions to the given problem. If the *target node* is found, the system measures the semantic distance between the *source node* and each of the nodes that are known to cause the *target node*. Thus the system measures the closeness of the *source node* to nodes that are known to cause the *target node*, thereby obtaining a measure of partial correctness of the hypothesis link under evaluation.

The semantic distance is measured by employing a modified version of the method described by Al-Mubaid & Nguyen (2006). Parent-child relationships from the UMLS Metathesaurus are used to construct the parental hierarchy of both nodes between which semantic distance is to be measured. An appropriate hint indicating the partial correctness or the closeness of the link to a plausible one is returned to the students.

However, if the *target node* is not found in the acceptable solutions, the system checks if the *source node* is found, in which case the comparison is made between the *target node* and each of the nodes that are known to be caused by the *source node*.

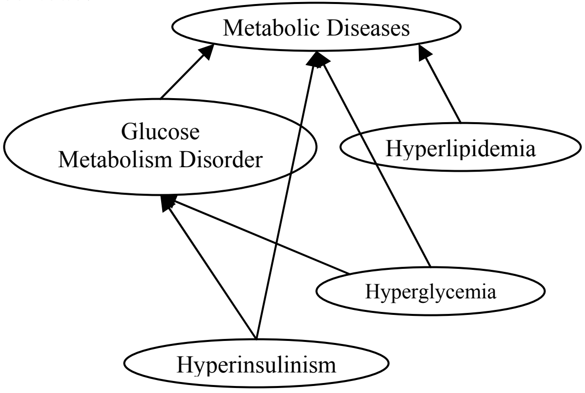


Figure 5: Parental Trees of Two Concepts

The semantic distance is only computable if the parental trees of both concepts, between which distance is to be measured, are actually connected. For example, based on the connected parental trees of *hyperlipidemia* and *glucose metabolism disorder* (*GMD*) shown in Figure 5, the semantic distance between *GMD* and *hyperlipidemia* is 2.83, whereas the semantic distance between *GMD* and *metabolic diseases* is 1.09. However, if the parental trees from both concepts happen to be disjoint, semantic distance is not computable. In this situation, the system resorts to the method of estimating likelihood of the *source node* causing the *target node* through the Bayesian model.

## Bayesian Model of Causal Links

Work done in extracting causal relationships between medical concepts in UMLS (Burgun & Bodenreider, 2001; Mendonca & Cimino, 2000) inspires us to use the Bayesian Network shown in Figure 6. This Bayesian network is used to determine the likelihood of a causal relation between nodes representing concepts *A* and *B*. *Causal Relation* is a Boolean node, where a true value indicates causal relation between nodes *A* and *B*, while a false value indicates the lack of a causal link between the respective nodes. *Semantic Type A* is the semantic type of concept *A* as defined in UMLS, and *Semantic Type B* is the semantic type of concept *B*. Each concept in the UMLS Metathesaurus is categorized under at least one semantic type from a list of 135 semantic types in the UMLS semantic network (U.S. National Library of Medicine, 2007). *Co-Occurrence Frequency* gives the frequency with which the two concepts are known to have co-occurred in medline citations, and is extracted from the UMLS table *mrcoc*. *Co-Relation Radius* is the radius distance within which concept *A* is known to be related to concept *B*. *Co-Relation Radius* is assigned a value of *zero* if the concepts are found to be directly related in the UMLS Metathesaurus, *one* if there is one intermediate node between *A* and *B*, and *two* if the relation radius is greater than one or if the concepts are not related at all.

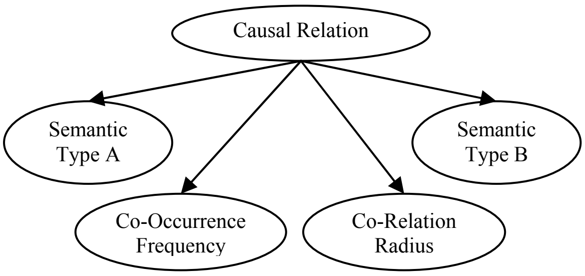


Figure 6: Bayesian Network for Causal Relationship

In order to estimate the likelihood of the causal link between two concepts *A* and *B*, the semantic types of both concepts, their co-occurrence frequency, and their relation radius is fed to the Bayesian network as evidence. The updated belief for true value of *Causal Relation* is examined to get the probability of causal relation between *A* and *B*. Based on the retrieved probability value, appropriate hints are returned to the student.

## Examples of Pedagogical Strategy Based on Step-Wise Fallback

While evaluating hypothesis links, only those links that are found in the expert knowledge base are accepted without any kind of feedback, explanation, or reservation. Links, for which the semantic distance is found to be below a certain threshold, are accepted with reservation. All other links are rejected, and appropriate feedback is returned based on the reasoning tier that was applied.

For purposes of illustration, we present a few examples of how the three tiers are applied in a step-wise fallback fashion while evaluating hypothesis links and how the tutor responds with appropriate hints. Consider the problem scenario described earlier of a patient with diabetic symptoms. While solving the case, the student draws causal links between various concepts and receives corresponding feedback from the tutoring system.

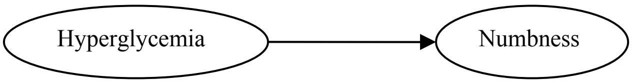


Figure 7: Student Hypothesis Link

For the hypothesis link in Figure 7, the system detects an indirect link, rejects this link and responds with the hint: “Think of the underlying mechanism why hyperglycemia causes numbness”. However if the student tries to draw a link from numbness to hyperglycemia, the system detects a reverse link and responds with the hint: “On the contrary, think of hyperglycemia as a cause of numbness”.

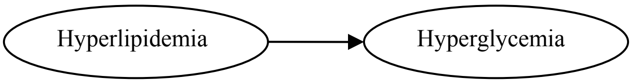


Figure 8: Student Hypothesis Link

For the hypothesis link in Figure 8, the system does not find a corresponding link in the knowledge base, so it checks the semantic distance between *hyperlipidemia* and *GMD*, rejects the link, and responds with the hint: “Hyperlipidemia is fairly close to a known cause of hyperglycemia. Instead of hyperlipidemia, think more specifically about other metabolic diseases”.

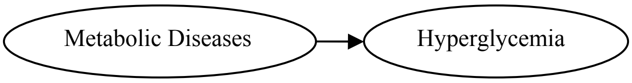


Figure 9: Student Hypothesis Link

For the hypothesis link in Figure 9, the system does not find a corresponding link in the knowledge base, but since the semantic distance between *metabolic diseases* and *GMD* is found to be very small, it accepts the link with reservation and responds with the hint: “Metabolic diseases is very close to a known cause of hyperglycemia. Metabolic diseases may be acceptable. However, think more specifically about kinds of metabolic diseases”.

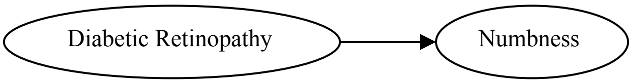


Figure 10: Student Hypothesis Link

For the hypothesis link in Figure 10, the system does not find the link in the knowledge base, and semantic distance is not computable. The system rejects the link and responds with the hint: “Diabetic retinopathy is not known to be a cause of numbness. Likelihood of causal relation between diabetic retinopathy and numbness is very low”.

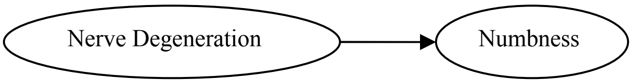


Figure 11: Student Hypothesis Link

For the hypothesis link in Figure 11, the system does not find the link in the knowledge base, and semantic distance is not computable, so the system rejects the link. However, since the Bayesian likelihood is high, the system responds with the hint: “There may be a causal relation between nerve degeneration and numbness”.

As shown above, the hints inform the student about the closeness of the hypothesis link to a plausible link. If this information is not available, the system provides information about the likelihood of the causal link. At the same time, the language of the hints generated by the system, informs the student of the tutor's reasoning limitations, which is likely to lead to improved reflective thinking and hence better learning.

## Initial Evaluation

The initial evaluation of our system was based on the agreement ratings of a collection of 15 causal links along with their respective hints, which were presented to an experienced human medical tutor at Thammasat University. The causal links comprised of five links each from three cases, for which we have already collected human expert solutions. The three cases are based on disorders such as diabetes, heart attack and pneumonia. On an agreement scale ranging from 1 (Strongly Disagree) to 5 (Strongly Agree), the human tutor was asked to rate various hints for each causal link. The average score of hints based on our measure of partial correctness and causal likelihood was 4.13, whereas the average score of the hints without the partial correctness and causal likelihood feedback was 2.13.

## Conclusions

In this paper we have described a multi tier approach in an intelligent tutoring system towards exhibiting human-like robust behavior in evaluating student hypotheses and responding in the form of hints. We have also discussed how the notion of gradual and smooth degradation in the output quality as a result of deteriorating input quality, applies to intelligent tutoring systems. Our approach towards incorporating robustness is innovative in employing a combination of rule-based, heuristic and probabilistic approaches applied successively in order of necessity, incorporating the notions of self recovery and task delegation. We have presented illustrative examples of how such human-like gradually deteriorating output quality can be observed in the responses of a medical tutoring system for PBL.

The initial assessment of our approach and feedback from human domain experts seems to indicate that the proposed methods can be useful in helping medical students acquire clinical reasoning skills. We have started to collect samples of student hypotheses for three different problem scenarios covering diseases and disorders such as diabetes, heart attack and pneumonia. We intend to conduct sub-system evaluations of the method of computing semantic distance and the method of estimating likelihood of a causal link between two concepts using the Bayesian model. Finally, we plan to measure the effectiveness of our generated hints compared with human tutors and perform quantitative evaluations of the pedagogical strategy incorporated in our system.

## References

Al-Mubaid, H., & Nguyen, H. A. (2006). A Cluster Based Approach for Semantic Similarity in the Biomedical Domain. *Proceedings of the 28th IEEE EMBS Annual International Conference, New York, USA, Aug 30-Sept. 3, 2006.*

Burgun, A., & Bodenreider, O. (2001). Methods for exploring the semantics of the relationships between co-occurring UMLS concepts. *MedInfo, 2001, 10*(Pt 1), 171-175.

Crowley, R., & Medvedeva, O. (2006). An Intelligent Tutoring System for Visual Classification Problem Solving. *Artificial Intelligence in Medicine, 2006, 36* (1), 85-117.

Feltovich, P. J., & Barrows, H. S. (1984). Issues of generality in medical problem solving. In H. G. Schmidt and M. L. De Volder (Eds.) *Tutorials in problem-based learning: A new direction in teaching the health professions.* The Netherlands: Van Gorcum.

Groot, P., Teije, A. T., & Harmelen, F. V. (2005). A Quantitative Analysis of the Robustness of Knowledge-Based Systems Through Degradation Studies. *Knowledge and Information Systems, 7* (2), 224-245.

Kabassi, K., Virvou, M., & Tsihrintzis, G. A. (2006). Requirements Capture for a Personalized Medical Tutor. *Proceedings of International Special Topic Conference on Information Technology in Biomedicine, Ioannina, Greece, October 26-28, 2006.*

Lulis, E., Michael, J., & Evens, M. (2004). Using Qualitative Reasoning in the Classroom and in Electronic Teaching Systems. *Proceedings of the 18th International Workshop on Qualitative Reasoning, Northwestern University, Evanston, IL August.*

McCarthy, J. (1984). Some expert systems need common sense. In H. Pagels (ed.), *Computer Culture: The Scientific, Intellectual, and Social Impact of the Computer.* Annals of the New York Academy of Sciences, Vol. 426, 129-137.

Mendonca, E. A., & Cimino, J. J. (2000). Automated Knowledge Extraction from MEDLINE Citations. *Proceedings of AMIA 2000 Fall Symposium*, 575-579.

Minsky, M. (1986). *The Society of Mind*. New York: Simon & Schuster.

Nielsen, P., Beard, J., Kiessel, J., & Beisaw, J. (2002). Robustness in Modeling Behavior Overview. *Proceedings of 11th CGF-BR Conference, May, 2002.*

Paritosh, P. K. (2006). The Heuristic Reasoning Manifesto. Proceedings of QR'06, Hanover, New Hampshire, July 10-12, 2006.

Sloman, S. A. (1996). The empirical case for two systems of reasoning. *Psychological Bulletin, 119*, 3-22

Suebnukarn, S., & Haddawy, P. (2006). Modeling individual and collaborative problem-solving in medical problem-based learning. *User Modeling and User Adapted Interaction, 16* (3), 211-248.

U.S. National Library of Medicine, (2007). *2007AA Introduction*. Retrieved April 19, 2007 from the World Wide Web:
http://www.nlm.nih.gov/research/umls/umlsdoc_intro.html

# Using Simulations to Model Shared Mental Models

**William G. Kennedy (bill.kennedy@nrl.navy.mil)**
**J. Gregory Trafton (trafton@itd.nrl.navy.mil)**
Navy Center for Applied Research in Artificial Intelligence
Naval Research Laboratory, 4555 Overlook Avenue SW
Washington, DC 20385 USA

## Introduction

Good team members seem to have the ability to simulate what others on the team will do in different situations. Team researchers have long studied what makes an effective team. Their methodology has been to examine how high and low performing teams accomplish team-related tasks. They have suggested that a good team-member has three knowledge components (Cannon-Bowers, Salas, & Converse, 1993):

(1) Knowledge of own capabilities [meta-knowledge],
(2) Knowledge of the task, and
(3) Knowledge about the capabilities of their teammates.

Most researchers have suggested that these three components are deeply inter-related; without any one of these, a person is not a good team member. However, without a computational theory, these claims can be difficult to examine empirically.

The focus of this paper is the third component, the cognitive modeling done of a teammate's cognitive processes. This shared understanding of teammates is frequently called a shared mental model (Mathieu, Heffner, Goodwin, Salas, & Cannon-Bowers, 2000). We start with the premise that humans use themselves as an initial model of their teammate, and then refine it as the team (and individuals within the team) gains experience. Our primary research goal is to create a computational theory of teamwork by modeling the individuals within the team so that we can eventually build plausible robots for teamwork and human-robot interaction.

## Method

To explore teamwork at the individual level, we implemented a simple cognitive model of shared mental models in a desk-top simulation of a robotic member of a team. The scenario used to test the value of the robot cognitively modeling a teammate was a two-agent security guard force made up of one human and one robotic agent, patrolling a warehouse with two separated guard stations. They begin in positions approximately across the warehouse from each other and move around the perimeter. When an alarm sounds, they must "man" the two security stations as soon as possible. The performance measure was simply the time (in steps) it takes for the team to fill both stations after the alarm sounds.

We used ACT-R (Anderson et al., 2004; Anderson & Lebiere, 1998) to model the robot's reasoning, including its modeling of the human. ACT family of theories ("ACT-R Research Group") has a long history of integrating and organizing psychological data. It has also been broadly tested in psychological and computational terms.

This project builds on our embodied robotic systems (Kennedy et al., in press; Trafton, Schultz, Bugajska, & Mintz, 2005; Trafton et al., 2006). To make the project tractable, we modeled both the human and robot as having the same movement capabilities such that both would take the same number of steps to cover the same distance. This assumption will clearly need to be revisited when we add the models to physical robots. The necessary spatial representations and reasoning capabilities were already included in the system: the cognitive model had the use of a 10-by-10, 2-D cognitive map from which the security stations closest to each agent could be determined. However, to simulate human's general weakness in accurately estimating distances outside the grasping range (Previc, 1998), the system could not always determine which station was closer and the model had to deal with that ambiguity. Finally, the robot's cognitive model of the human presumed the human would reason and behave the same as the robot would.

For this project, we have initially modeled two simple cases. The first case represents neither agent having any model of the other agent and simply doing what is best for each agent independently, i.e., going to their nearest station. If both agents arrive at the same station, one must go to the other station and this is inefficient in both time and safety. The second case represents a leader-follower shared mental model where the leader, typically the human, goes to her closest station, and the follower, typically the robot, must go to the other station. This avoids the conflict of both going to the same station.

### Simulating others in ACT-R

The robot could "see" the environment and used rule-based behavior to patrol the perimeter of the warehouse prior to an alarm and, with no shared mental model, what to do after the alarm. To decide what to do when the alarm sounded when using a shared mental model, the robot needed to model the behavior of the human. The robot modeled the human by explicitly taking information about how the robot would deal with the alarm and spawning that knowledge off as a simulation of the human teammate's decision making. The simulation decided what action the human would take in the current situation. Hence, the robot simulated what the human would do by explicitly modeling what it would do itself in a similar situation.

To run a spawned ACT-R model, a new model needs to be initialized with its own declarative memory, productions, and initial goal. This capability is part of the current version of ACT-R (R1.2-370]). To model how the human would react to the current situation, the robot's cognitive model spawned a cognitive model of the human using declarative memory of the current situation appropriately modified to place the robot in the human's situation, and provided the productions the robot itself used to decide what to do for the first, i.e., self-centered case. The simulation's initial goal was to determine which station the robot would go to if it was in the human's situation.

With the results of the simulation of what the human would do, the robot then decided to go to the other station, in accordance with the shared mental model that the human will lead and the robot follow. Figure 1 shows traces of a run in which both agents arrive at the same station and one then goes to the other station and a run in which, through having a shared mental model, they avoid the collision. The human began in the top line at position "a" and the robot began at "a" in the bottom. The sequential letters mark their steps counterclockwise prior to the alarm. The alarm sounded at "p" and the run ends when both guard stations "1" and "2" are filled.

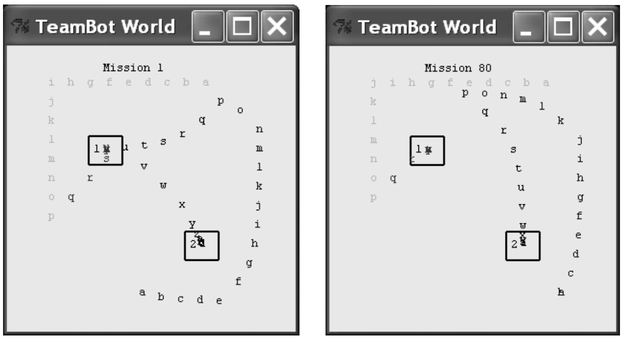


Figure 1. Traces of agents colliding (left) and avoiding collision (right) based on a shared mental model.

## Results and Discussion

We found that for even this simple scenario, the useful, shared mental model significantly improved the team's performance: with 25 simulated runs each, the system that used a shared mental model and cognitive modeling of its teammate took 3.28 fewer steps than the system that did not, t(27.7) = 8.1492, p < .001 with the Welch correction for unequal variances.

By basing the robot's cognitive model of the human on what it would do in the human's place, the task required creating only the declarative memory to simulate the robot taking the human's place and one additional production to terminate the simulation.

This work demonstrates that the impact of one agent's cognitive modeling of another agent can be effective even in a simple scenario. We expect that there are many aspects of teamwork and cognitive modeling of shared mental models that can be explored using similar techniques. As an example, the flexibility of specifying the declarative memory and productions that will be used by the spawned cognitive model, allows cognitive models to consider the effects of hypothetical declarative knowledge and productions.

## Acknowledgments

We wish to thank Dan Bothell for modeling help. This work was performed while the first author held a National Research Council Research Associateship Award at the Naval Research Laboratory and was partially supported by the Office of Naval Research under job order numbers 55-8551-06, 55-9019-06, and 55-9017-06. The views and conclusions contained in this document should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U. S. Navy.

## References

ACT-R Research Group. *ACT-R* Retrieved October 13, 2006, from http://act-r.psy.cmu.edu/

Anderson, J. R., Bothell, D., Byrne, M. D., Douglas, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of mind. *Psychological Review, 111*(4), 1036-1060.

Anderson, J. R., & Lebiere, C. (1998). *The Atomic Components of Thought*. Mahwah, NJ: Erlbaum.

Cannon-Bowers, J. A., Salas, E., & Converse, S. (1993). Shared mental models in expert team decision making. In N. J. Castellan (Ed.), *Individual and group decision making* (pp. 221-246). Hillsdale, NJ: Lawrence Erlbaum Associates.

Kennedy, W. G., Bugajska, M. D., Marge, M., Fransen, B. R., Adams, W., Perzanowski, D., et al. (in press). *Spatial Representation and Reasoning for Human-Robot Collaboration.* Paper to be presented at the National Conference of the Association for the Advancement of Artificial Intelligence, Vancouver, BC.

Mathieu, J. E., Heffner, T. S., Goodwin, G. F., Salas, E., & Cannon-Bowers, J. A. (2000). The influence of shared mental models on team process and performance. *Journal of Applied Psychology, 8*(2), 273-283.

Previc, F. H. (1998). The neuropsychology of 3-D space. *Psychological Bulletin, 124*(2), 123-164.

Trafton, J. G., Schultz, A. C., Bugajska, M. D., & Mintz, F. E. (2005, 13-15 August 2005). *Perspective-taking with robots: experiments and models.* Paper presented at the International Workshop on Robot and Human Interactions.

Trafton, J. G., Schultz, A. C., Perzanowski, D., Adams, W., Bugajska, M. D., Cassimatis, N. L., et al. (2006). *Children and robots learning to play hide and seek.* Paper presented at the 2006 ACM Conference on Human-Robot Interactions, Salt lake City, UT.

# Investigation of Procedural Skills Degradation from Different Modalities

**Jong W. Kim (jongkim@psu.edu)**
**Richard J. Koubek (rkoubek@psu.edu)**
Department of Industrial and Manufacturing Engineering

**Frank E. Ritter (frank.ritter@psu.edu)**
College of Information Sciences and Technology
The Pennsylvania State University, University Park, PA 16802 USA

## Abstract

Can we help people forget less by knowing how they learn? Can we decrease forgetting by modifying what they learn? These have been long-standing questions in applied cognitive psychology. This paper reports a study designed to investigate procedural skills degradation in a set of spreadsheet tasks. The task can be taught and performed as knowledge and skills that are declarative or procedural, and perceptual-motor or cognitive. To examine the effect of these characteristics on learning and forgetting, one group of participants used key-based commands to complete the task, and the other group used a novel mouse and menus to do the task. Participants were able to learn the task well in four learning sessions. Retention intervals (6-day or 18-day) showed clear effects on the amount of forgetting. This paradigm can measure forgetting in terms of modalities and skill types. We found evidence that the menu mode was not better than keystrokes. Furthermore, the modalities showed different effects on forgetting in terms of retention intervals.

**Keywords:** forgetting; procedural skills; modalities

## Introduction

Disuse or infrequent use of knowledge and skills can produce poor human performance. A generally observed human characteristic is that learning is often forgotten. Knowing how people forget in various tasks can help to produce better performance.

Previous forgetting studies have measured degradation of declarative knowledge. For example, Pavlik and Anderson (2005) investigated forgetting of a paired-associates of declarative knowledge.

To help learning by decreasing forgetting, there are several approaches already proposed and studied such as maximizing the amount of original learning, using refresher training, or optimizing retention intervals (e.g., Farr, 1987; Healy, 1995; Sabol & Wisher, 2001; Wisher, Sabol, & Ellis, 1999).

In this study of procedural tasks, we investigated an approach to decrease forgetting by modifying the modality, knowledge, and skills in that task, based on a theory based on the learning and forgetting equations in ACT-R (Anderson & Lebiere, 1998).

### Types of Knowledge and Skills

Surveys have shown that personnel in technical jobs perform mostly procedural tasks (Tarr, 1986). For example, in an emergency situation, the most important knowledge and skills would be procedural, such as cardiopulmonary resuscitation (CPR) or a decontamination task of biological/chemical agents. A procedural task includes several decision-making points as cognitive tasks. Konoske and Ellis (1991) noted that many procedural tasks can be viewed as an ordered sequence of steps or operations performed on a single object or in a specific situation to accomplish a goal.

Several theoretical views support this view of procedural and declarative knowledge. Anderson and his colleagues have proposed representations of declarative and procedural knowledge and their corresponding memories (Anderson & Lebiere, 1998). They note that declarative knowledge indicates factual information and procedural knowledge indicates knowledge representing our behavior.

Another terminology, “how-to-do-it” knowledge, has been recognized in the literature (Kieras, 1997). A user performing a task uses “how-to-do-it” knowledge that can be typically modeled using production rules. This “how-to-do-it” knowledge is also viewed as procedural knowledge because it also describes our behavioral aspects of performance with goals, operators, methods, and selection rules.

### Forgetting Procedural Skills

Humans acquire knowledge and skills from training. The acquired knowledge and skills can be forgotten with the passage of time. Forgetting can cause decreased performance.

Particularly, procedural skills may not be always well retained over time and they need to be. For example, non-medical trainees on a space flight may need to rapidly perform an advanced cardiac life support task during space flight missions (Ramos & Chen, 1994). Cardiopulmonary resuscitation (CPR) is an emergency medical procedure for restoring normal heartbeat and breathing to victims of heart failure. McKenna and Glendon (1985) studied skill retention of CPR. They had 120 occupational first responders as experimental subjects. They reported that less than a quarter of their trained personnel were skillful at performing the CPR task six months after training.

Interestingly, as shown in the study, procedural skills are not always well retained. The CPR task is a procedural task that includes several decision-making points as cognitive tasks. Therefore, it is presumed that there may exist different relationships of forgetting between different types of knowledge and skills.

Hagman and Rose (1983) mentioned that the best predictor of forgetting is the number of steps required in the procedural tasks. There is a supporting study of skill retention by the US Army Research Institute during the mobilization of the Individual Ready Reserve (Sabol & Wisher, 2001; Wisher, Sabol, Sukenik, & Kern, 1991). However, it seems that we rarely forget how to ride a bicycle or how to swim after learning these skills. These are perceptual-motor control skills. This aphorism and their investigations suggest that procedural (discrete) skills might be forgotten much more rapidly than perceptual-motor (continuous) skills.

## A Way to Test Forgetting

Research of text editing skills has provided important findings on human performance and information processing. For example, Card, Moran, and Newell (1983) studied how a user's skills would interact with computer-based systems focusing on text editing tasks. Singley and Anderson (1989) investigated the transfer of cognitive skills in text editing tasks by providing an in-depth theory of learning through the ACT* architecture.

In our study, as an extension of text editing tasks, a set of spreadsheet tasks were examined to measure degradation of procedural knowledge and skills. A spreadsheet task provides cognition-demanding task characteristics for the experimental study, and it can be modified to support different types of inputs. It also provides a task with some ecological validity. Our spreadsheet task is done with a tool that allows us to examine two sets of knowledge and skills, that is, procedural or declarative, and cognitive or perceptual-motor skills.

## Can a Cognitive Model Predict a Forgetting Curve?

Cognitive architectures provide a framework to build a model. The architectures not only support but also confine modeling capabilities to provide models that match possible human behavior (Taatgen, Lebiere, & Anderson, 2006).

These architectures have started to be used to examine forgetting. Pavlik and Anderson (2005) investigated the spacing effects of a paired-associates memory and optimization of learning based on the ACT-R architecture (see Anderson & Lebiere, 1998). ACT-R is a cognitive architecture and has been validated to model human behavior and learning in a variety of tasks (Anderson & Lebiere, 1998).

Chong (2004) stated that the existing set of mechanisms from several architectures could not model forgetting procedural skill. For example, EPIC does not provide a rule learning mechanism. In Soar, as a rule learning mechanism, chunking is used to model learning but not skill degradation. ACT-R is limited to learning and forgetting of declarative knowledge. Thus, it is worth exploring and extending the existing architectural mechanisms to model procedural skill degradation.

We are using ACT-R for this project. For the first step of our investigation, we delved into the utility theory in ACT-R 5 and report findings in this paper. In the meanwhile, the current version of ACT-R 6 introduced a new utility learning mechanism. The utilities of productions change in terms of the rewards they receive based on the difference learning equation. It is necessary to note that the current version of ACT-R 6 no longer uses the PG-C formulation.

ACT-R 5 selects one production to fire among competing productions, to achieve the model's goal. The mechanism allows a model to learn problem-solving strategies from experience based on the probability of success and the relative cost of different strategies in a production. Each production rule is associated with a utility value indicating how much the production is able to achieve the model's current goal ($U_i = P_iG - C_i + \varepsilon$). $P_i$ is the expected probability to successfully achieve the model's current goal. The probability is decomposed to $q$ and $r$ ($P = qr$, where $q$ is the probability that a production will achieve its intended next state, and $r$ is the probability that the production achieves its goal when it arrives at the intended next state). $C_i$ is the expected cost to achieve the model's objective. $G$ is the value of the goal. $\varepsilon$ is noise.

The probability of success is calculated by the number of successes divided by the number of successes and failures, as shown in equation 1.

$$P = r(t) = \frac{Successes(t)}{Successes(t) + Failures(t)},\ q = 1 \qquad (1)$$

This is the probability learning equation in ACT-R 5. Lovett (1998, p. 265) proposed time-based decay in ACT-R's production parameter learning. This mechanism discounts past experience and adjusts the timing of successes and failures. Similar to the ACT-R's base level activation, each success and failure experience in a production is decayed in terms of a power function.

$$Successes(t) = \sum t^{-d} \qquad (2)$$

$$Failures(t) = \sum t^{-d} \qquad (3)$$

We simply applied the time-based decay mechanism of ACT-R's production parameter learning to modeling of skill degradation. Successes can be considered as learning sessions and failures can be knowledge retention without learning. Figure 1 shows the probabilities ($r(t)$) of achieving a goal over time using equations from Lovett (1998). From Day 1 to 4, the probabilities increase indicating learning. Then, without learning, the values of $r(t)$ decrease, and increase again with learning.

To test the general results of this theory (that knowledge is learned and forgotten) and how a different interaction modality may help decrease forgetting we trained and tested learning and forgetting on a set of spreadsheet task with new input modality, which we present next. We also found an interesting effect in this aspect of ACT-R, which we will need to take up in later work why the model predicts poorer performance at day 22 than when it started.

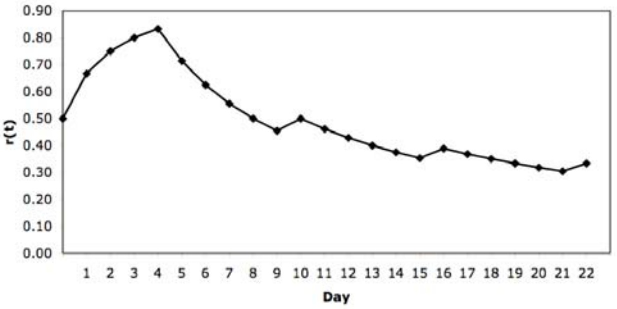


Figure 1: Time-based decay of ACT-R's production parameter learning.

## Method

### Participants

Nineteen undergraduate and graduate students at Penn State were recruited and completed the task. None had prior experience with the Dismal spreadsheet or using a vertical mouse. They were randomly assigned to conditions.

### Materials

The Dismal[1] spreadsheet was implemented to gather and analyze behavioral data (Ritter & Wood, 2005). Dismal extends the GNU Emacs editor using its extension language, Emacs Lisp. Dismal is useful here because it is novel to the participants. Figure 3 shows an example Dismal spreadsheet. We have used a vertical mouse, shown in Figure 2, because it provides new motor skills to learn (and to forget). The vertical mouse is ergonomically designed to reduce stress on a user's wrist. Instead of a palm-down position of a regular mouse, this vertical mouse requires different hand and forearm postures. None of the participants had used a vertical mouse so we could minimize participants' previous knowledge and skills. Keystrokes, mouse clicks, mouse movements, and task completion time were recorded by the Recording User Input (RUI) system (Kukreja, Stevenson, & Ritter, 2006).

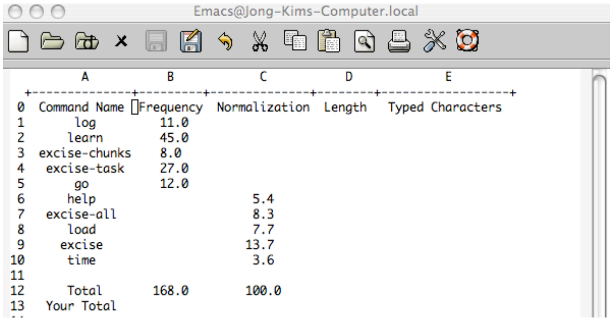
### Design

The experiment was constructed by two independent variables, modality and retention interval. Modality consists of two levels including menu-based command users with a vertical mouse (M) and key-based command users with a keyboard (K), representing two different types of skills in the task.

For the key-based command users, ten participants performed the procedural spreadsheet task using only a keyboard (K). For instance, the key-based command for "Open (or find) a file" is `C-x C-f`. (`C` indicates holding down the control key while pressing `x`).

For the menu-based command users, nine participants performed the same task using a vertical mouse (M). For instance, to open a file, they moved the mouse pointer to `File` on the menu bar, then clicked `Open File`.

Figure 2: A vertical mouse from the Evolent company.



```
          A            B            C            D            E
  +--------------+---------+-------------+--------+-----------------+
0  Command Name  Frequency  Normalization  Length   Typed Characters
1       log         11.0
2      learn        45.0
3  excise-chunks    8.0
4   excise-task     27.0
5       go          12.0
6      help                       5.4
7   excise-all                    8.3
8      load                       7.7
9     excise                     13.7
10     time                       3.6
11
12     Total       168.0        100.0
13  Your Total
```

Figure 3: A screenshot of the spreadsheet task in Dismal.

### Procedure

The experiment consisted of 115 sessions (76 for learning and 39 to measure forgetting) with nineteen participants. A learning session was constructed from a study and a test task. A forgetting session was constructed only by a test. A study task is when a participant uses the study booklet (Users Guide for the Dismal Spreadsheet) to learn. Each study task was limited to 30 minutes of study. A test task is when participants perform the given tasks with the booklet during learning sessions and without the booklet during forgetting sessions.

In the first week, four consecutive learning sessions were held. On Day 1, participants had a maximum of 30 minutes to study the given spreadsheet task, and then performed the task. On Days 2 to 4, participants were allowed to refresh their acquired knowledge and skills from Day 1, using the study booklet, and then performed the tasks.

After the four learning sessions in the first week, participants returned for forgetting sessions as part of one of two types of retention interval. Retention interval (R) indicates a time period of skill disuse.

Participants had a 6- or 18-day retention interval. For the group with 6-day retention intervals (R6), participants returned back to be measured every 6-days for three times on Day 10, 16, and 22. For the group with an 18-day retention interval (R18), participants returned back to be measured 18 days after the learning session on Day 22.

Participants performed a set of two novel spreadsheet tasks. The spreadsheet (Figure 3) consists of five columns

[1] http://acs.ist.psu.edu/dismal/dismal.html

(A to E). Column A had ten different names of computer commands. Column B had frequencies of each command listed from row 1 to 5. Column C had normalized frequencies listed in rows 6 to 10. There are five blank cells in B and C columns (e.g., B6 to B10, and C1 to C5). Column D and E had ten blank cells.

The set had 14 steps. First, they opened a Dismal spreadsheet, saved the file as another name, and completed the complex spreadsheet manipulation by calculating and filling in the blank cells, basically using these two equations with commands (e.g., summation or multiplication).

$$\text{Normalization} = \frac{(\text{Frequency} \times 100.0)}{\text{Total frequency}} \qquad (4)$$

$$\text{Frequency} = \frac{(\text{Normalization} \times \text{Total freqeuency})}{100.0} \qquad (5)$$

The steps had multiple sub-steps, including five data normalization calculations, five data frequency calculations, ten calculations of length, ten calculations of total typed characters, four summations of each column, and an insertion of the current date using a Dismal command, (*dis-current-date*).

## Results

All nineteen participants that completed learning sessions were able to complete forgetting sessions—one participant could not complete the series due to a scheduling conflict (i.e., job interview) that arose in the course of the experiment. This resulted in a cell distribution of 5, 5, 4, and 5 participants. We report all these participants here. In the R6 group, there were ten participants, five mouse users and five keyboard users. In the R18 group, there were nine participants, four mouse users and five keyboard users.

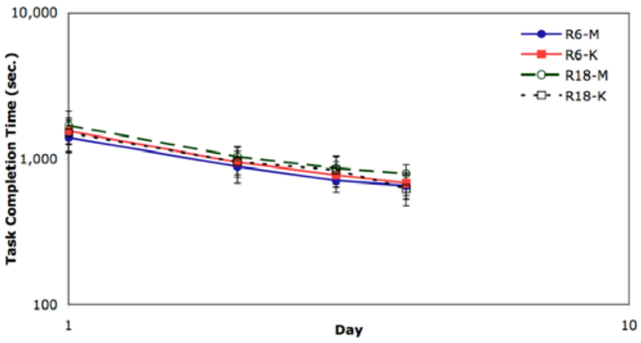


Figure 4: The log-log plot of learning curves for the four groups in the learning sessions.

### Learning Procedural Skills

Figure 4 shows the log-log plot of learning curves of the four study sessions. This figure shows that the groups all learned over the four learning sessions. They all performed at pretty much the same level. Their average time to perform the set of tasks decreased from an average of 1,396 to 654 s for R6-Mouse, 1,549 to 680 s for R6-Keyboard, 1,690 to 791 s for R18-Mouse, and 1,504 to 625 s for R18-Keyboard. Independent samples t-tests were conducted for mouse (M) and keyboard (K) users for each study session. There were no significant differences, for all comparisons, t(17) < 1.1, p ≥ .33. These results suggest that the input/manipulation style factor, keystroke or mouse driven, did not lead to significant differences in learning on this task over this time range and for this population.

### Power Law of Learning: Mouse vs. Keyboard

Figure 5 shows the average time for the two modalities (M or K) over the four consecutive days of learning. The averages of the task completion time of the mouse (M) group were 1,527+374 s on Day 1, 950+160 s on Day 2, 775+149 s on Day 3, and 714+135 s on Day 4. The averages of the task completion time of the keyboard (K) group were 1,527+321 s on Day 1, 949+212 s on Day 2, 803+182 s on Day 3, and 653±132 s on Day 4. The learning curves of the mouse (M) and keyboard (K) groups follow the Power law of learning:

$$y = 1477x^{-0.56},\ R^2 = 0.98 \text{ for the Mouse group}$$
$$y = 1503x^{-0.59},\ R^2 = 0.99 \text{ for the Keyboard group}$$

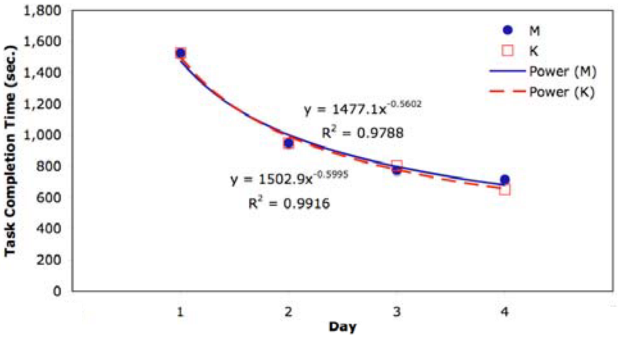


Figure 5: The Power curves of learning: M vs. K.

### Forgetting Procedural Skills

Figure 6 shows overall performance on the spreadsheet task over the learning and forgetting sessions. There are four groups in terms of two retention intervals (R6 and R18) and two modalities (M and K). All of four groups similarly learned the spreadsheet task. However, participants in each group forgot in a different manner.

### Forgetting: 6-Day Retention Interval

Participants (n=5) using menu-based commands with 6-day retention intervals (R6-M) showed a 42% increase in task completion time at the first forgetting measure on Day 10. The task completion time on Day 4 is 654±127 s. The task completion time at the first return on Day 10 is 930±252 s. However, paired samples t-test revealed that there is no significant difference between Day 4 and Day 10 performance, $t(4) = -1.77, p > 0.05$.

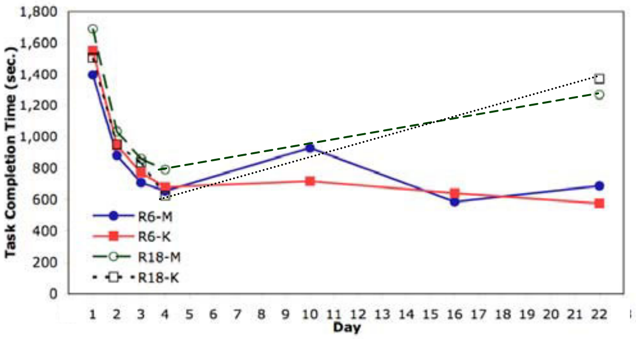


Figure 6: Learning and forgetting curves of four different groups, R6-M, R6-K, R18-M, and R18-K.

Participants (n=5) using key-based commands with 6-day retention intervals (R6-K) showed a 5% increase in task completion time at the first forgetting measure on Day 10. The task completion time on Day 4 is 680±124 s. The task completion time at the first return on Day 10 is 716±169 s. Paired samples t-test revealed no significant performance difference between on Days 4 and 10, t(4) = -.66, $p > 0.05$.

**Forgetting: 18-Day Retention Interval**

Participants (n=4) using menu-based commands with an 18-day retention interval (R18-M) showed a 60% increase in the task completion time at the first forgetting measure on Day 22. The task completion time is 791±116 s on Day 4 and 1268±177 s on Day 22. Paired samples t-test revealed that there is significant difference between Day 4 and Day 22 performance, t(3) = -3.60, $p < 0.05$.

Participants (n=5) using key-based commands with an 18-day retention interval (R18-K) showed a 119% increase in the task completion time at the first forgetting measure on Day 22. The task completion time is 625±149 s on Day 4 and 1371±329 s on Day 22. Paired samples t-test revealed that there is significant difference between Day 4 and Day 22 performance, t(3) = -4.30, $p < 0.05$.

**Forgetting: Retention Interval and Modalities**

It is of interest how much knowledge and skills can be retained with respect to retention intervals (R6 and R18) and modalities (M and K). We compared the two data points, which are based on the last day of learning sessions and the first return day after any given retention intervals. Table 1 provides the task completion time of those two data points.

Under a 6-day retention interval, participants in the menu-based modality, M, took 276 s more to complete the task after the retention interval. Participants in the key-based modality, K, took 37 s more to complete the task after a 6-day retention.

On the contrary, under an 18-day retention interval, participants in the key-based modality, K, took 746 s more to complete the task after the retention interval. Participants in the menu-based modality, M, took 478 s more to complete the task.

Table 1: Increase in task completion time

| | | Time | | Difference | % Increase |
|---|---|---|---|---|---|
| | | Day 4 | Day 10 | | |
| R6 | M | 654 s | 930 s | 276 s | 42 % |
| | K | 680 s | 716 s | 37 s | 5 % |
| | | Day 4 | Day 22 | | |
| R18 | M | 791 s | 1268 s | 478 s | 60 % |
| | K | 625 s | 1371 s | 746 s | 119 % |

In this spreadsheet task, knowledge and skills in the menu-based modality (M) are more susceptible to decay than those of the key-based modality for a 6-day retention interval. However, for longer retention interval (18-day), knowledge and skills in the key-based modality (K) are more susceptible to forgetting that those of the menu-based modality (M), although these results are not yet statistically reliable.

## Discussion and Conclusions

We showed that the approach of using Dismal and RUI supports exploring learning and forgetting in a procedural task, an office work task that has some external validity. Participants were able to learn it well in four learning sessions of less then an hour each.

The results suggest that the learning curve applies to this relatively large cognitive task (cf. Newell & Rosenbloom, 1981). The data in this study are very similar in how fast each interaction style group learns and in how fast they perform the task during learning.

Two different retention intervals (R6 and R18) showed clear effects. The 18-day retention interval was much worse than the 6-day retention interval, and performance on Day 22 after an 18-day forgetting period is still better than Day 1 (cf. Figure 6).

This forgetting rate needs to be compared to the ACT-R theory that has been started. In Figure 1, the probability that a production achieves its goal increases and decreases in terms of time and experience. For further work, this mechanism is to be fixed to model forgetting over time. Also, it is necessary to consider a new utility learning in ACT-R 6 to study how it plays a role in modeling forgetting. It appears that like many others we have found that forgetting is not a linear function.

The effect of modifying the interface modality has produced some interesting effects. During learning the keystroke and mouse driven interfaces were equally easy to learn and equally fast. This is slightly surprising, as many interface designers have argued for the superiority of menu driven interfaces over keyboard driven interfaces (e.g., Shneiderman, 1983). However, in this study, the two interfaces to the same task, one driven by keystrokes and one driven by mouse movements, basically took the same time to learn and the same time to perform. We are running further participants to clarify this effect (or lack of it).

More interestingly, we see that there may be some differences between retention intervals and these skills. The data in Figure 6 suggest that there may be a difference in the forgetting curves of these two types of skills. More participants will have to be run before we can say more about this, but it looks promising that there may be an interaction between modalities on forgetting.

Further work remains. The number of participants in this study is somewhat small, and some differences are perhaps not emerging because of the small sample size. We will be running more participants as time goes on. We will also examine the missing retention interval of 12 days. This will help explain how fast forgetting occurs and when it occurs. On Day 10 we saw very little forgetting, and on Day 22 we saw a lot of forgetting. The middle ground of Day 16 will help fill in the curve.

In addition, we will investigate the knowledge attributes of various subtasks in the set of tasks here to provide implications on forgetting. For example, there could be differences of learning and forgetting between the subtask of calculations using a normalization equation and opening a file. The former is more a cognition-demanding task than the latter that is simple declarative knowledge retrieval.

Also, we will need to investigate how the keystroke and mouse move times changed with forgetting. Did, for example, the Fitts' law constant change with forgetting? Did the simple keystroke level times that can be derived from an ACT-R model on this task increase with the forgetting interval?

## Acknowledgments

The College of IST provided partial support for participants' compensation. Sue Kase provided helpful comments.

## References

Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: Erlbaum.

Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale, NJ: Erlbaum.

Chong, R. S. (2004). Architectural explorations for modeling procedural skill decay. In *Proceedings of the Sixth International Conference on Cognitive Modeling,* Mahwah, NJ: Erlbaum.

Farr, M. J. (1987). *The long-term retention of knowledge and skills: A cognitive and instructional perspectives*. Arlington, VA: Springer.

Hagman, J. D., & Rose, A. M. (1983). Retention of military tasks: A review. *Human Factors, 25*(2), 199-213.

Healy, A. F. (1995). *Optimizing the long-term retention of skills: structural and analytic approaches to skill maintenance* (ARI Research Note 95-16): The U.S. Army Research Institute for the Behavioral and Social Sciences.

Kieras, D. E. (1997). A guide to GOMS model usability evaluating using NGOMSL. In M. G. Helander, T. K. Landauer & P. V. Prabhu (Eds.), *Handbook of human-computer interaction* (2nd ed., pp. 733-766). Amsterdam: North-Holland.

Konoske, P. J., & Ellis, J. A. (1991). Cognitive factors in learning and retention of procedural tasks. In R. F. Dillon & J. W. Pellegrino (Eds.), *Instruction: Theoretical and applied perspectives* (pp. 47-70). New York: Praeger.

Kukreja, U., Stevenson, W. E., & Ritter, F. E. (2006). RUI: Recording user input from interfaces under Window and Mac OS X. *Behavior Research Methods, 38*(4), 656-659.

Lovett, M. (1998). Choice. In J. R. Anderson & C. Lebiere (Eds.), *The atomic components of thought* (pp. 255-296). Mahwah, NJ: Erlbaum.

McKenna, S., & Glendon, A. (1985). Occupational first aid training: Decay in cardiopulmonary resuscitation (CPR) skills. *Journal of Occupational Psychology, 58*, 109-117.

Newell, A., & Rosenbloom, P. S. (1981). Mechanisms of skill acquisition and the law of practice. In J. R. Anderson (Ed.), *Cognitive Skills and Their Acquisition* (pp. 1-55). Hillsdale, NJ: Erlbaum.

Pavlik, P. I., & Anderson, J. R. (2005). Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect. *Cognitive Science, 29*, 559-586.

Ramos, M. A. G., & Chen, J. J. G. (1994). On the integration of learning and forgetting curves for the control of knowledge and skill acquisition for non-repetitive task training and retraining. *International Journal of Industrial Engineering, 1*(3), 233-242.

Ritter, F. E., & Wood, A. B. (2005). Dismal: A spreadsheet for sequential data analysis and HCI experimentation. *Behavior Research Methods, 37*(1), 71-81.

Sabol, M. A., & Wisher, R. A. (2001). Retention and reacquisition of military skills. *Military Operations Research, 6*(1), 59-80.

Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. *IEEE Computer, 16*(8), 57-69.

Singley, M. K., & Anderson, J. R. (1989). *The transfer of cognitive skill*. Cambridge, MA: Harvard.

Taatgen, N. A., Lebiere, C., & Anderson, J. R. (2006). Modeling paradigms in ACT-R. In R. Sun (Ed.), *Cognition and Multi-Agent Interaction: From Cognitive Modeling to Social Simulation*. New York, NY: Cambridge.

Tarr, R. (1986). Task analysis for training development. In J. A. Ellis (Ed.), *Military contributions to instructional technology*. New York: Praeger.

Wisher, R. A., Sabol, M. A., & Ellis, J. A. (1999). *Staying Sharp: Retention of Military Knowledge and Skills* (ARI Special Report 39): The U.S. Army Research Institute for the Behavioral and Social Sciences.

Wisher, R. A., Sabol, M. A., Sukenik, H. K., & Kern, R. P. (1991). *Individual Ready Reserve (IRR) Call-Up: Skill Decay* (Research Report 1595): The U.S. Army Research Institute for the Behavioral and Social Sciences.

# Information Seeking in Complex Problem Solving

**Xiaohui Kong (xik2@pitt.edu)**
Intelligent Systems Program, University of Pittsburgh

**Christian D. Schunn (schunn@pitt.edu)**
Learning Research & Development Center, University of Pittsburgh
3939 O'Hara Street Pittsburgh, PA 15260 USA

## Abstract

Information seeking behavior in human complex problem solving has rarely been well studied. In this paper we studied the information seeking behavior of eye-movement during human complex problem solving in the case of traveling salesman problem. A new model of human TSP solving is proposed to explain the effect of limited amount of visual working memory on the trade-off between local/global information processing and the human information seeking behavior in complex problem solving.

## Introduction

When solving problems, information seeking behavior serves as an interface between the world (external information) and cognition (internal information). Hypotheses have been proposed and argued to explain human information seeking behavior in problem solving (Gray & Fu, 2005; Gray, Sims, & Fu 2006). However, most of previous studies on information seeking behaviors are based on experiments either with relatively simple problems and/or with manifested high cost of information seeking, because natural information seeking behavior is hard to measure in the setting of complex problem solving. A recent study in modeling the behavior of human traveling salesman problem solving (Kong & Schunn, 2006) and advanced eye-tracking technology, however, gave us an opportunity to exam the information seeking behavior of human complex problem solving in the case of the traveling salesman problem solving.

The (Euclidean) traveling salesman problem is to find a path of minimum Euclidean distance between points in a plane, which includes each point exactly once and returns to its starting point. As an NP-hard combinatory optimization problem, the traveling salesman problem (TSP) is believed to be “intractable” in computer science for large inputs as long as exact optimal path is concerned.

## Experiment

### Participants

Six undergraduate students from University of Pittsburgh participated in the experiment.

### Materials and Methods

In this experiment, we used the same set of 20 TSP problems as in the experiment described in Kong and Schunn (2006). Ten of them are real world problems borrowed from TSPLIB (http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html) ranging in size from 16 points to 100 points. The remaining ten of them were randomly pre-generated according to a uniform distribution ranging in size from 10 points to 80 points. All participants saw the exact same 20 TSP problems, which allow us to examine how well the models predict the influence of particular TSP problems rather than just general trends for the effect of number of points. The experiment was conducted on a Tobii 1750 eye-tracker with a 17” screen. The resolution was set to 1024*768 pixels. During the experiment, participants were 550 to 650 pixels away from screen as recorded by the eye-tracker, measuring by the corresponding screen size and resolution. Participants were asked to find the shortest path possible by indicating the path with mouse-clicks on the screen. A Matlab program recorded all the click data and the eye-tracker recorded all the eye-movement data. The participants were paid 5$~20$ based on their performance.

### Results

Optimality of the solution is defined as our measurement of performance. Optimality (OPT) of a solution is calculated as the ratio of the optimal path length over the solution path length. So the optimality is a value smaller or equal to 1. The closer the value is to 1, the better performance the participant makes. As figure 1 shows, all the participants found close to optimal paths for all problems (OPT>0.8, MEAN = 0.95, STD = 0.037).

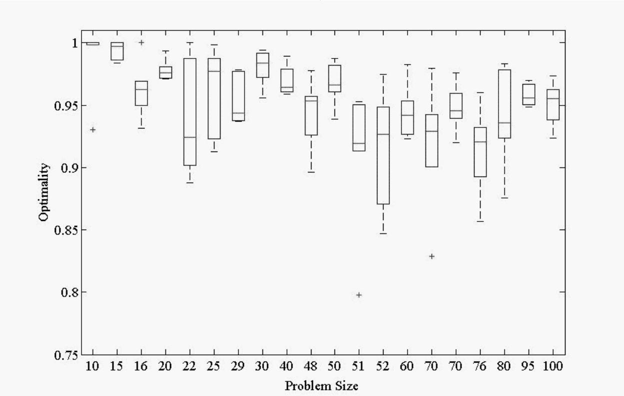


Figure 1: Participants’ performance

The eye-movement data recorded by the eye-tracker were used to analyze the information seeking actions. Distance of information seeking of each fixation was defined as the minimum between the following two values:

1. The distance of the fixation to the last visited point
2. The distance of the fixation to the next point to be visited.

The frequency of information seeking follows an exponential distribution along the distance of information seeking ($R^2 = 0.989$).

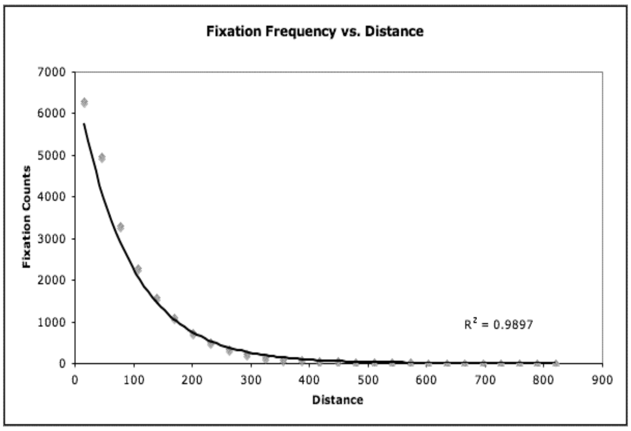


Figure 2: Frequency and distance of information seeking behaviors

We define the global information seeking actions as fixations whose information seeking distances are greater than 200 pixels (about 18 degree of eye-movement in this experiment setting). As an opposite to what was reported in Best (2005), only 23 percent of all the global information seeking actions were made in the beginning of the each trial before 10 percent of points were connected. The rest global fixations distributed through the entire problem solving procedure as shown in figure 3.

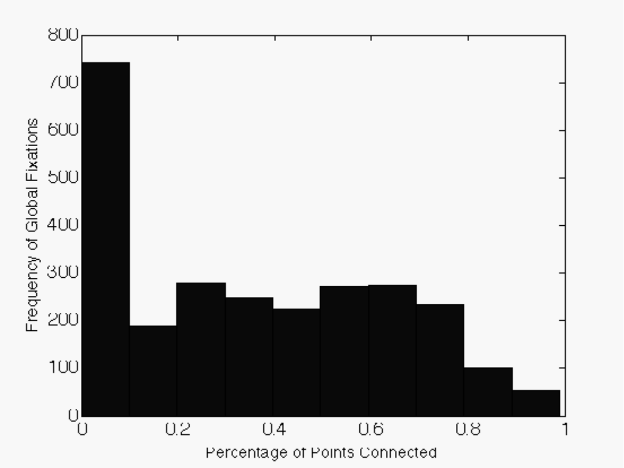


Figure 3: Number of global fixations in each stage of problem solving when part of the points had been connected

## Discussion

It is not very surprising that global information seeking actions are biased toward the beginning in the experiment described in Best (2005), since the cost of information seeking changed from low (eye-movement) to high (mouse-movement) when the experiment stage transits. This could be well explained by the soft-constraint hypothesis (Gray & Fu, 2004). But in our experiment, the cost of the information seeking had been low (eye-movement) through the entire problem solving procedure, which we would argue to be a more natural experiment setting. However, the exponential distribution of the frequency vs. distance of information seeking may not be easily explained by the soft-constraint hypothesis as a tradeoff between information seeking cost and its utility. First, the costs (measured by time) by eye-movements of different distances are not significantly different. Second, the utility of information is hard to define in this scenario, since global and local information must interplay with each other to generate a good TSP solution (Kong & Schunn 2006). Our hypothesis is that the limited size of the visual working memory (VWM) could explain this pattern of information seeking behavior. Our intuition is that people do not seek for more global information than they could actually handle in visual working memory. Since the amount of VWM is limited to several chunks, the exponential pattern of information seeking behavior helps to keep both the necessary global and local information in VWM.

## Our model

To support our hypothesis, we built a model to simulate the human TSP solving and the information seeking behavior during the process.

To account for the information seeking behavior of human TSP solving, our VWM-Reference TSP model is based on the following two hypotheses and consists of four steps:

First, the VWM only contains a constant number of chunks, which can be set as a parameter in the model. Second, the model only makes constant (in average) number of fixations near the centroids of clusters when they are generated into the VWM to serve as reference points.

Step 1. Initialization
The current working set includes all points. The current point is set to be the starting point.
Step 2. Information Seeking
Points in the current working set are grouped into K clusters according to the K-Means clustering algorithm, (MacQueen, 1967) where K is the size of the VWM in the first iteration and square root of the number of points in the current working set afterwards. The K-Means Clustering Algorithm clusters N data points into K disjoint subsets Sj containing Nj data points so as to minimize the sum of squares criterion:

$$J = \sum_{j=1}^{K} \sum_{n \sum S_j} | x_n \sum \mu_j |^2$$

where $x_n$ is a vector representing the nth point and $\mu_j$ is the geometric centroid of the points in Sj. It is assumed that people are capable of noticing clumps of points relatively quickly and easily with their basic perceptual system. This K-Means clustering algorithm is used to proximate the cluster identification, because it is available in standard programming tools and provides the centroids of the clusters as a standard output.

All the centroids are added into the collection of reference points, which was passed from previous iteration.

We then use a spline-curve to connect the current point and all the reference points to sketch a path in a rough scale. The spline-curve is hypothesized to be a general smooth route through the centroids, which captures a general tendency of a globally sketched path.

Step 3. Identify current cluster and refine local information

All the points in the current working cluster are projected to its nearest points on the spline curve. If the number of points projected onto the part of the spline curve between the current point and the first reference point is more than 2, let the current working set to be this set of points, then go back to step 1 and the next iteration begins. When N is the size of the VWM, only the first N reference points, sorted by their projection order on the spline curve, are passed to the next iteration. The rest of them are discarded.

Step 4. Move and rehearse global information

If the number of points projected between the current point and the next reference point is less than two, move from current point to those points according to the sequence they projected onto the spline curve. Set the current working set to be the points projected onto the part of spline between the first and the second reference points. Discard the first reference point from the VWM.

If the number of reference points in the VWM is less than 2, re-identify clusters at the most global level and bring in those centroids back into the VWM.

Repeat this procedure until the number of unvisited points is less than the size of the VWM. Then find the best path for the rest few points.

Figures 4a-e illustrate the steps of our model when solving a 70-points TSP.

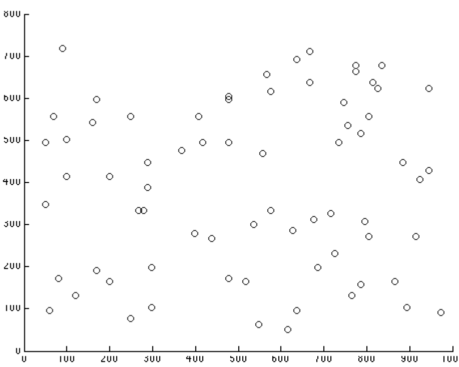


Figure 4a: Original ETSP problem

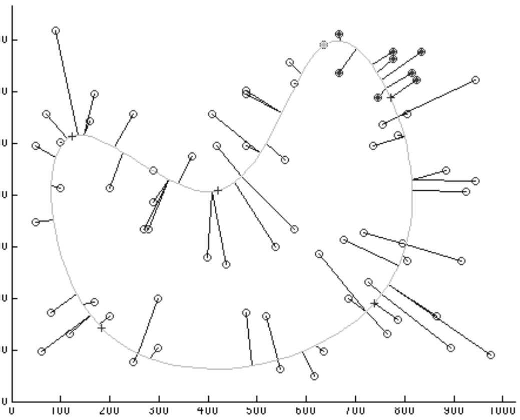


Figure 4b: Seeking global information
'+'s are the locations of the K-Means centroids which are served as reference points in VWM, and the '*' points are those to be served as the current working set in the next iteration

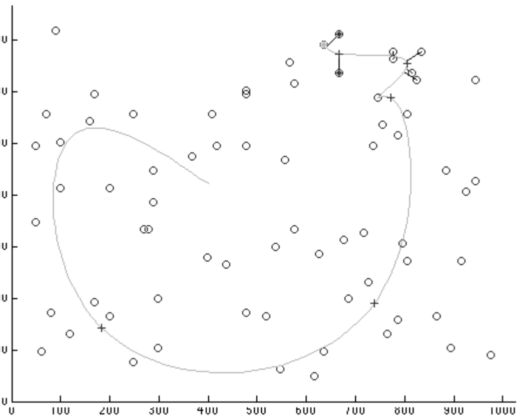


Figure 4c: As the local part of information is refined, some information in global level is discarded. '+'s are the reference points in VWM.

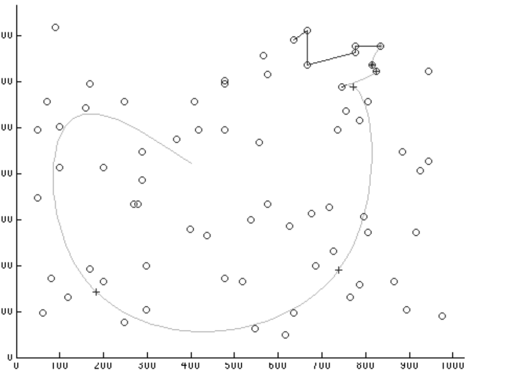


Figure 4d: When there is enough local information, make a move and rehearse the global information

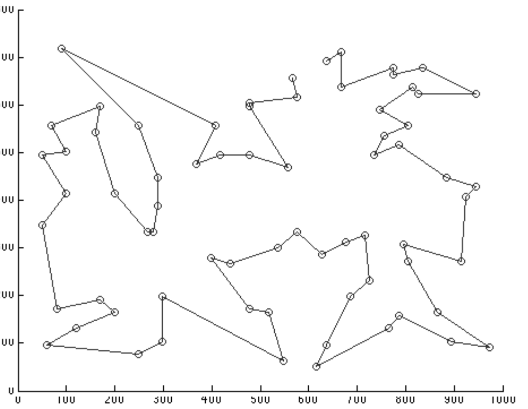


Figure 4e: The final path generated by the model

## Model Evaluation and Comparison

### Existing models of human TSP solving

*Convex Hull*

The next simplest model of TSP is the Convex Hull model, which assumes that people compute a traversal around the perimeter points, including inner points opportunistically along the way using a minimal insertion rule. The global information used by this model is the Convex Hull contour, which may be rather complex, and thus require significant working memory. The minimal insertion rule is applied globally at each point during path computation, and points that cause the smallest increase in total path length are inserted. It is somewhat implausible that people would be able to compute these minimal insertions (a local processing task) at a global level.

*Sequential Convex Hull Model*

MacGregor et al. (2000) adapted the Convex Hull model to a more plausible incremental local search version. This adaptation was base on their finding that humans perform better on problems with fewer interior points within the convex hull (MacGregor & Ormerod, 1996). Second, their experiments provided support for their hypothesis that human participants are sensitive to global information (Ormerod & Chronicle, 1999). We would call this model sequential convex hull model. The outline of the model is as follows (MacGregor et al., 2000):

1. Sketch the connections between adjacent boundary points of the convex hull.

2. Select a starting point and a direction randomly.

3. If the starting point is on the boundary, the starting node is the current node. The arc connecting the current node to the adjacent boundary node in the direction of travel is referred to as the current arc. Proceed to Step 4 immediately. If the starting point is not on the boundary, apply the insertion rule to find the closest arc on the boundary. Connect the starting point to the end node of the closest arc, which is in the direction of travel. This node becomes the current node.

4. Apply the insertion criterion to identify which unconnected interior point is closest to the current arc. Apply the insertion criterion to check whether the closest node is closer to any other arc. If not, proceed to Step 5. If it is, move to the end node of the current arc. This becomes the current node. Repeat Step 4.

5. Insert the closest node. The connection between the current node and the newly inserted node becomes the current arc. Retaining the current node, return to Step 4 and repeat Steps 4 and 5 until a complete tour is obtained.

*Pyramid Model*

Graham et al.'s model (2000) of traveling salesman problem was inspired by a hierarchical architecture of human visual and spatial perception. Their model first Gaussian-blurs the original set of points into a variety of degrees and stores those blurred images in different layers of hierarchy with the most blurred image on the top. The more blurred images serve as the global information for the less blurred images. Each layer directly guides the next layer below it each time the model develops a node into the path. So layers in the hierarchy change in a repeatedly cascaded process. The Pyramid model computes TSP solutions in the following steps:

1. Gaussian-blur the original n-points TSP image into k-1 different degrees and store them in a k-layer pyramid with the original TSP image on the bottom and the most blurred image on the top.

2. Calculate $L_i$ modes of the image in each layer i. Consider those modes in each layer as nodes in a reduce-sized TSP problem. The top layer has 3 nodes and the bottom layer has n nodes. Layer k has $n/b^k$ nodes. (The parameter b is the reduction ratio. Bottom layer is layer 1.)

3. Layer n (top layer) has 3 nodes and forms a unique tour.

4. Generate a tour of the TSP in each layer by inserting them into the tour on the previously higher layer with the following rules: (a) Sort the intensity level of the mode locations in each layer. (b) Insert these modes into the tour in descending order of their intensity, so as to produce the minimum increase in tour length. Repeat step 4 until the algorithm generates a tour in the bottom layer.

*K-Means TSP model*

The K-Means TSP model (Kong & Schunn, 2006) is based on the following three steps:

1. Clusters are identified.

In this step, points are grouped according to visual density. Points constructing a higher visual density are more likely to be grouped together. K-Means clustering algorithm was used to generate the $2\times\sqrt{N}$ clusters, where N is the number of points in the problem.

2. A sketch of the path is conceived.

A spline-curve is drawn through all the centroids and back to the start one.

3. Connect all the points along the sketched path.

All the points are projected to the nearest point on the spline-curve. Then we construct the final solution by connecting all the points in the same order as their projection on the spline-curve.

## Mean optimality

First we compared the performance of the VWM_TSP model in term of solution optimality with human data and some existing models of human TSP solving including: Convex Hull, Pyramid (Graham, Joshi, & Pizlo 2000; Pizlo, et al. 2006), Kmeans (Kong & Schunn 2006), CHSQ (MacGregor, Ormerod, & Chronicle 2000). We set the size of VWM to be 5 chunks in our VWM_TSP model for this evaluation of optimality, based on existing VWM theories (Pylyshyn, 1989).

The performance of models and human data were plotted in figure 5. Pearson correlations and average signed errors between models and human data were shown in table 1. The VWM_TSP displayed a fairly good correlation and only generated a small amount of error. Though, CHSQ has a better fit to the performance data. VWM_TSP was built under the constraint that the VWM is constant in size. This constraint made our model more theoretically plausible, where CHSQ could have arbitrarily many chunks (invisible lines in its case) in VWM in the extreme case. (Kong & Schunn 2007)

Table 1: Correlation and average signed error of model fits to human accuracy performance

| | VWM _TSP | NN | Convex Hull | Pyramid | Kmeans | CHSQ |
|---|---|---|---|---|---|---|
| Correlation | 0.62 | 0.37 | -0.02 | 0.13 | 0.24 | 0.69 |
| Ave Signed Error | -0.02 | -0.13 | 0.01 | -0.02 | -0.03 | 0.00 |

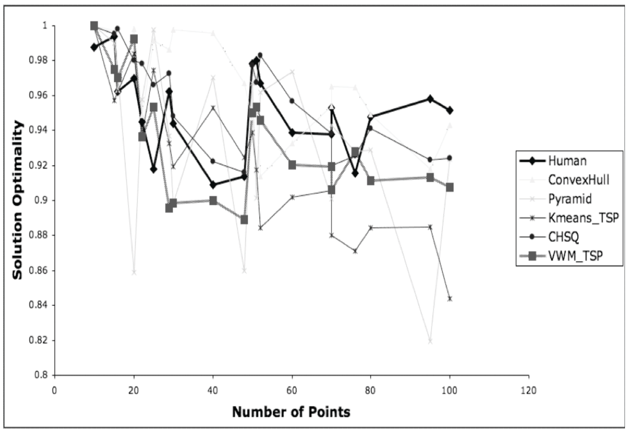


Figure 5: Mean accuracy for models and humans

## Number of information seeking actions

Assuming that the VWM_TSP model takes a constant number of information seeking actions around each cluster centroids to generate clusters, we plotted the histogram (# of bins = 30, min = 15 pixs, max = 906 pixs) of VWM_TSP's information seeking distance in figure 6 when VWM size is 5.

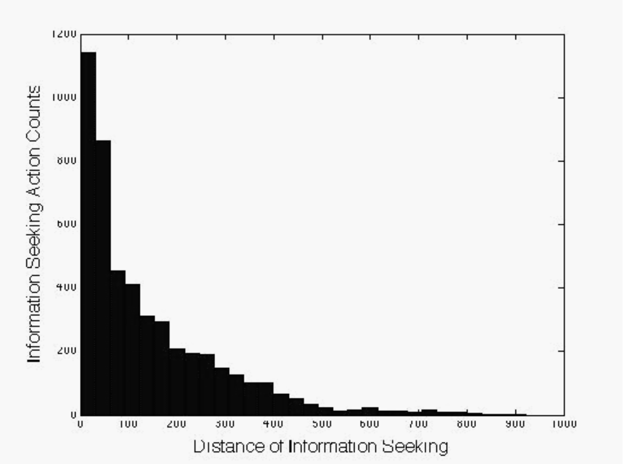


Figure 6: VWM_TSP model's information seek behavior (VWM=5)

As we can see, the count of information seeking actions decreases exponentially with the distance. To study the effect of VWM size on the information seeking behavior, we ran VWM_TSP model with different VWM size parameters (VWMSize = 2,3,4,5,6,7,8,10,15) and plotted the normalized histogram counts of each VWM size as smoothed lines in figure 7. When VWM size is too small (2, 3), the model seeks for global information much more often. In this setting, there wouldn't be much room in VWM to keep global information as soon as local information was developed. So global information had to be re-attended almost on each move. Figure 8 plotted each model setting's $R^2$ fits to exponential distribution. When VWM size is around 5, the information seeking behavior demonstrated by the model has the best fit to the exponential distribution. When the VWM size is too small or too large, the model's information seeking distance distribution deviates from the exponential distribution. This result is consistent with the existing theories of working memory that the VWM size is around 5 (Pylyshyn, 1989).

To further exam our model, we also looked at how global information seeking behavior varies along time during the problem solving procedure. In figure 9, we plotted the frequency distribution of global information seeking actions during each temporal phase of the problem solving procedure, when part of the points were connected. The model displayed a similar pattern with human data in figure 3, which again supports our hypothesis.

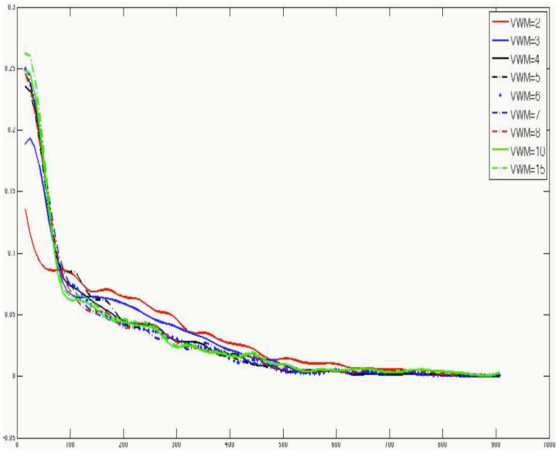


Figure 7: VWM Size vs. Information Seeking Distance Distribution

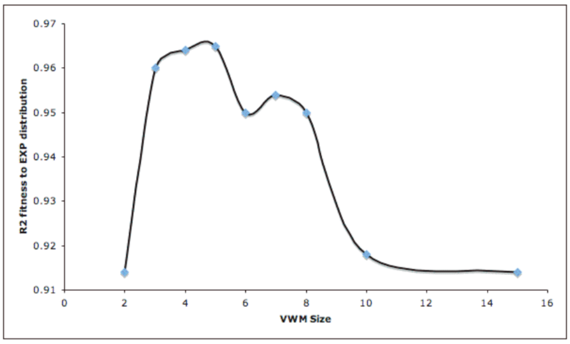


Figure 8: Effect of VWM size on model's fit to exponential distribution

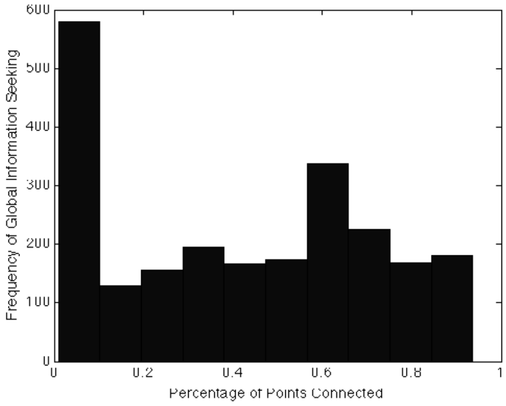


Figure 9: Frequency of global information seeking in each phase of problem solving

## Conclusion

The new experimental evidence and simulation results suggested while the cost of information seeking is low and the information utility is hard to define, the limited size of visual working memory plays an important role in the information seeking behavior while solving complex problems. Although the VWM is limited to only several slots, by keeping a good ratio of global information and local information in VWM, human is still capable in solving complex problems to its near optimal solution. Our model while having a good fit to the performance of human TSP solving, also predicts the information seeking behavior during the problem solving procedure. Our model also explored on the question of how different VWM size affects the information seeking behavior during problem solving.

## References

Best, B. (2005). A model of fast human performance on a computationally hard problem. *Proceedings of the 27th Annual Conference of the Cognitive Science Society.*

Graham, S., Joshi, A., & Pizlo, Z. (2000). The traveling salesman problem: A hierarchical model. *Memory & Cognition, 28(7)*, 1191-1204.

Gray, W. D., Sims, C. R., Fu, W.-T., & Schoelles, M. J. (2006). The soft constraints hypothesis: A rational analysis approach to resource allocation for interactive behavior. *Psychological Review*, 113(3) 461-482.

Gray, W. D., & Fu, W.-t. (2004). Soft constraints in interactive behavior: The case of ignoring perfect knowledge in-the-world for imperfect knowledge in-the-head. *Cognitive Science*, 28(3), 359-382.

Kong, X & Schunn, D. C. (2006). Global vs. Local Information Processing in Problem Solving: A Study of the Traveling Salesman Problem. *Proceedings of 7th International Conference on Cognitive Modeling*

Kong, X & Schunn, D. C. (2007) Global vs. Local Information Processing in Visual/Spatial Problem Solving: The Case of the Traveling Salesman Problem. Accepted by *Cognitive Systems Research*

MacGregor, J.N., Ormerod, T.C., & Chrinicle, E.P. (2000). A model of human performance on the traveling salesperson problem. *Memory & Cognition, 28*, 1183-1190.

MacQueen, J. B. (1967). Some Methods for classification and Analysis of Multivariate Observations. In the *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability* (pp. 281-297). Berkeley, CA: University of California Press.

Newell, A., & Simon, H. A. (1972*). Human problem solving.* Englewood Cliffs, NJ: Prentice-Hall.

Ormerod, T.C., & Chronicle, E.P. (1999). Global perceptual processes in problem solving: The case of the traveling salesperson. *Perception & Psychophysics, 61*, 1227-1238.

Pylyshyn, Z.W. (1989). The role of location indexes in spatial perception: A sketch of the FINST spatial-index model. *Cognition*, 32, 65-97.

Zygmunt Pizlo, Emil Stefanov, John Saalweachter, Zheng Li, Yll Haxhimusa, and Walter G. Kropatsch (2006) Traveling Salesman Problem: A Foveating Pyramid Model, *http://docs.lib.purdue.edu/jps/vol1/iss1/8*.

# Modeling Memories of Large-scale Space Using a Bimodal Cognitive Architecture

**Unmesh Kurup (kurup@cse.ohio-state.edu)**
**B. Chandrasekaran (chandra@cse.ohio-state.edu)**
Computer Science and Engineering, 2015 Neil Ave
The Ohio State University, Columbus, OH 43210

## Abstract

We present an extension to biSoar, a bimodal version of the cognitive architecture Soar, by adding a bimodal version of *chunking,* Soar's basic learning mechanism. We show how this new biSoar is a useful tool in modeling cognitive phenomena involving spatial or diagrammatic elements by applying it to the modeling of problem solving involving large-scale space, such as way-finding. We suggest how such models can help in identifying variables to control for in human subject experiments.

## Introduction

Cognitive architectures are of central interest in cognitive modeling since such architectures are directly useful in building cognitive models. The advantages of a general purpose architecture such as Soar or ACT-R to model and explain a variety of cognitive phenomena are well-known. However, these architectures are all based on a view of the cognitive state being symbolic or more precisely, predicate-symbolic. In this view, the agent's knowledge, goals etc are represented in terms of symbol structures that describe the world of interest in terms of properties of and relations between individuals in the world. We have argued that this view of cognitive state is too restrictive and fails to take adequate account of the role played by perceptual representations in thinking (Chandrasekaran, 2006). We have proposed that cognitive state should be viewed as multi-modal where, in addition to the traditional symbolic component, the cognitive state has several perceptual components and a kinesthetic one. The multi-modal view proposes a more involved role for perception where the perceptual systems, in addition to their role as transducers, also provides representations and processes to the cognitive process. Such a multi-modal state can support an agent experiencing the world multi-modally such as when having mental images in one or more modalities. One of the tasks of a research program that is based on this multi-modal view is to explore the consequences of multi-modal cognitive state for all components and mechanisms of a cognitive architecture.

In particular, we need to examine the implications of multi-modality for components such as working memory, LTM and I/O and for control and learning mechanisms. As a first step towards constructing such a multi-modal architecture, we built biSoar (Kurup & Chandrasekaran, 2006), which is a bimodal augmentation of the Soar architecture, where the two modes are the traditional symbolic component and a visual (diagrammatic) component. This limitation to bimodality has several advantages. First, intuitions about various issues related to multi-modality may be honed by investigating this limited version. Second, in problem solving, the most common and useful perceptual mode is the limited visual version involving diagrams. Soar was chosen for reasons of convenience but we think that many of the ideas in biSoar can be extended to other symbolic architectures such as ACT-R. However, Soar also has unique mechanisms such as chunking as a core learning mechanism, an issue that will be a focus of the current paper. As an aside, the visual component does not represent all aspects of mental imagery such as the visualization of faces but is restricted to diagrams. In addition to diagrams being common in problem solving, the focus on diagrams has the advantage of simplicity while retaining several of the challenges of bimodality that we wish to address.

Currently, both working and long term memories are bimodal in biSoar. biSoar agents are able to create, modify and delete diagrammatic objects from WM as well as extract various relations that exist between objects in this memory. However, among the issues not addressed is how diagrammatic information gets into long term memory. Phenomenologically, it seems clear that memory is capable of recalling perceptual knowledge and experience to a more or less degree of fidelity. It seems plausible that in the course of learning, learning mechanisms transfer to long term memory not only symbolic information from working memory but diagrammatic information as well. In traditional Soar, there is only one learning mechanism, chunking. So it seemed natural to us to investigate how chunking can be expanded to learn bimodally. An empirically observed feature of many spatial memories is that spatial details are often simplified (Tversky, 2000). So, a challenge for bimodal chunking is the degree to which such simplification is an intrinsic architectural feature.

A satisfactory account of bimodal learning could make an architecture with such a capability an effective medium for modeling cognitive phenomena involving spatial or diagrammatic elements. We explore the possibilities of biSoar for such modeling by applying it to the task of modeling phenomena involving the representation of and reasoning about large-scale space. We build biSoar models of problem solving in two spatial reasoning tasks that have been well studied: simplification in recalled routes and distortions in geographic recall. Such modeling can be a valuable tool for exploring the space of explanations for spatial phenomena. For each task, we create multiple models and describe how each one suggests a different explanation for the observed phenomena. We indicate how a candidate explanation can in turn suggest variables to control for in human subject experiments.

## Multi-modal Cognitive Architectures

The traditional approach to cognition and problem solving can be best described "predicate-symbolic"; that is, the knowledge and goals of an agent are represented as a set of entities, and relations (predicates) that hold between these entities. Problem solving proceeds by the application of rules of inference to these predicates. The role of the perceptual system is to give the agent information about the external world, and the role of the action system is to make changes to the world. The output of the perceptual systems, in this view, is in the form of predicate-symbolic representations. Our alternative proposal calls for a much greater role for an agent's perceptual system in cognition. Here, the agent has representations and processes that are characteristic to the individual modalities and cognition is an activity that involves all of them. The perceptual system as whole still give information about the external world, but aspects of the system are part of central cognition, independent of input from the external world.

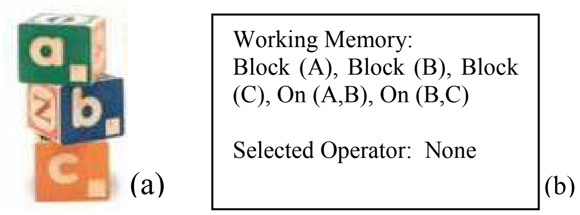
To create biSoar (Kurup & Chandrasekaran, 2006), a general-purpose cognitive architecture, Soar (Laird *et al.*, 1987) was augmented with the Diagrammatic Reasoning System (DRS) (Chandrasekaran *et al.*, 2004), a domain independent system for representing diagrams. In DRS, diagrams are represented as a configuration of points, curves and regions. That points may refer to the location of cities or that regions represent states in a map, is task-specific knowledge that is part of Soar. This allows DRS to be used in multiple task domains without any modifications. DRS also provides a set of perceptual and action routines that allows Soar to create, and modify a diagram and to extract relations between diagrammatic objects from the diagram. By the addition of the capabilities of DRS, Soar's cognitive state and long-term memory that were exclusively predicate-symbolic, now become bimodal.

### Cognitive State in Soar

Soar's representations are predicate-symbolic. The cognitive state in Soar is represented by the contents of Soar's WM and operator, if any, that has been selected. Fig 1(b) shows Soar's cognitive state representation of the blocks world example in 1(a).

### Cognitive State in biSoar

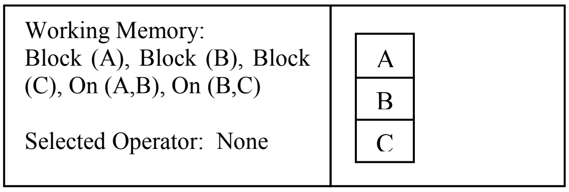
The cognitive state in biSoar is bimodal – it has both symbolic and diagrammatic parts. Fig 2 shows the bimodal representation of the world depicted in Fig 1(a). Working memory is biSoar is represented as a quadruplet, with each Identifier, Attribute, Value triplet augmented with a diagrammatic component in DRS that represents the visualization (metrical aspect) of the triplet. Since not all triplets need to be (or can be) visualized, the diagrammatic components are present only as needed. States represent the current or potential future state of interest in the world and the symbolic and the diagrammatic part may represent related or distinct aspects of the world. However, the diagrammatic representation is "complete" in a way that the symbolic representation is not. For example, from the symbolic representation alone it is not possible to say without further inference whether A is above C. But the same information is available for pick up in the diagram with no extra inference required. This has advantages (for instance in dealing with certain aspects of the Frame Problem) and disadvantages (over-specificity).



Working Memory:
Block (A), Block (B), Block (C), On (A,B), On (B,C)

Selected Operator: None

*Fig 1: (a) Blocks World and (b) Soar's representation of the world in (a).*



Working Memory:
Block (A), Block (B), Block (C), On (A,B), On (B,C)

Selected Operator: None

*Fig 2: biSoar representation of the world shown in 1(a)*

### From External Representation to Working Memory

When an agent makes use of an external diagram, such as a map, for a specific problem solving task, what he attends to or observes is only relevant parts or aspects of the diagrammatic elements. This selective attention results in simplified versions of the corresponding diagrammatic elements to be present in WM. The mechanism that transforms an external diagrammatic element into a simplified version in WM is part of human perceptual machinery and is needed as an adjunct to biSoar as well. In this paper, we refer to this attention-controlled mechanism as the simplification mechanism. This mechanism is implemented as an A*ttend* method that is part of any routine that interacts with an object in the external world. The A*ttend* method produces the equivalent of the product of attention on aspects of the diagrammatic object. One way to think of A*ttend* is that it is as if parts of the diagrammatic object on which attention is not focused is at a very low resolution resulting in the loss of many of the finer details while still preserving the general spatiality of the object. Fig

3(b) is the output of the A*ttend* operator on the curve in 3(a) where the attention has been focused on just the beginning and end points. Fig 3(d) is the result of A*ttend* on Fig 3(c) where the attention has been focused on the region's broad shape. The result of A*ttend* does depend upon the requirements of the task because that determines the aspects to which attention was paid to in the diagram. But simplification in this manner is architectural because it happens irrespective of the task or the domain.

### Bimodal LTM

There are two questions that have to be answered in an implementation of Long Term Memory (LTM) – how are elements put into LTM (i.e., learned) and how are elements retrieved from LTM. In the case of Soar the answers to these two questions are chunking for learning and a matching process that matches the LHS of a LTM rule to WM for retrieval.

**Chunking** - Chunking simply transfers the relevant contents of WM to LTM. In the case of biSoar, chunking transfers to LTM the simplified versions of the relevant external diagrammatic elements present in WM.

**Matching** - In the case of Soar the retrieval process is straightforward because matching (or even partial matching when variables are present) symbols and symbol structures to each other is an exact process; either they match or they don't. When the cognitive state is bimodal, WM has metrical elements in addition to symbols. Matching metrical elements to each other (say a curve to another curve) is not an exact process since two metrical elements are unlikely to be exactly the same. Matching metrical elements would require a different approach like a non-exact process that can match roughly similar elements in a domain-independent manner (since the matching should be architectural). It may also turn out that only calls to perceptual routines are present in LTM while matching metrical elements is a more low-level cognitive process present only in stimulus-response behavior. For now we take the latter approach where the LHS of biSoar rules contain only perceptual calls to the DRS that return symbol structures in addition to symbol structures. We think that this approach can account for many of the diagrammatic learning capabilities that are required in models of cognition except in cases where goal specifications contain irreducible spatial components, such as might be the case in the problem solving of a sculptor. The RHS of a biSoar rule can modify either symbolic or diagrammatic parts of WM.

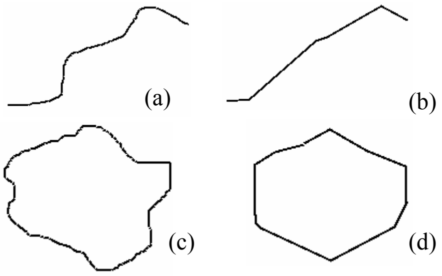


*Fig 3: (b) and (d) show the result of applying the visualize operator to (a) and (c) respectively*

## Representation of Large-Scale Space

In 1948, Tolman (1948) proposed that animals have an internal representation of large-scale space which he called the cognitive map. In 1960, Lynch (1960) produced his seminal study of the environment in which he identified *Landmarks, routes, nodes*, *districts* and *edges* as the features that are important in building a cognitive map. Since then there have been a number of models, both computational and cognitive, that have been proposed to account for a number of phenomena associated with the representation of space. A variety of behavioral/psychological studies have also aided the development of these models by providing a set of characteristics or behaviors that a model should posses.

We believe the use of a general-purpose cognitive architecture such as biSoar can be beneficial in the area of modeling spatial phenomena for three reasons – First, it restricts spatial information to be learned, represented and used within the constraints of a general cognitive architecture. Second, it allows the modeler to be flexible in the strategies and knowledge that they use to model phenomena. Third, it is easier to identify the nature of the explanation (architectural vs. content) because these are explicitly distinguished in such a framework. We use biSoar to model two commonly observed phenomena in spatial reasoning - simplification in recalled routes and distortions in the recall of relations between geographic entities.

### Sources of Map Knowledge

Knowledge of large-scale space can come from multiple sources. The most common, of course, being personal experience of navigation in space. We automatically build representations of our environment as we traverse them. A second, and important, source is maps. Our knowledge of large environments, such as the spatial extent and geographical locations of the fifty states, originated from our use of maps. Representations, originating from either source, are combined and modified in various ways during problem solving for various purposes. In this paper, we focus on phenomena involving maps.

## The Space of Explanations

When models are implemented in a cognitive architecture as possible explanations for a phenomenon, the behavior of interest can arise from one, or a combination of, two influences:– Architectural and Content where Content can be further sub-divided into Strategy and Knowledge.

An architectural explanation appeals to the specifics of the architecture of the agent to explain the phenomenon of interest. The phenomenon is produced as the result of a process that is automatic and arises out of the architecture, not a deliberative decision by the agent. A phenomenon can also emerge as a result of a particular strategy employed by the agent to solve the given task. This is different from an architectural explanation because the phenomenon is unique to the current task. An agent's behavior can also be seen as arising from its knowledge (or lack thereof) of the task domain and the world. During problem solving, an agent may learn to solve the problem one way due to the knowledge it has at the time. □iven more knowledge, the agent might have learned to solve the problem in a different way resulting in different observable phenomena.

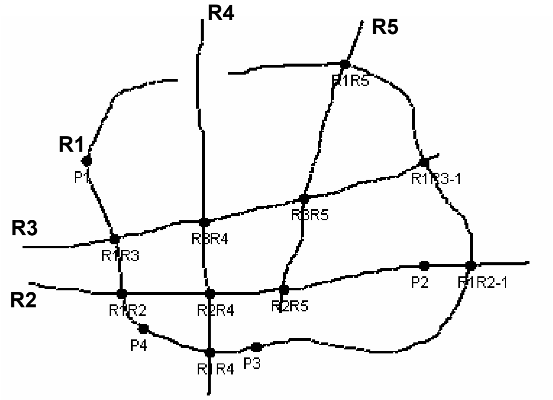


*Fig 4: The map for models Simp1 and Simp2. Routes are found from P1 to P2, P4 to R1R3-1 and R1R4 to R1R5*

In general, a phenomenon can have more than one explanation and it is difficult for an outside observer to decide if the reason for the phenomenon is architectural, strategic or knowledge related without further experimentation. Also, due to the number of free variables and tunable parameters in cognitive architectures, and the fact that they are essentially Turing machines, the ability (or inability) to build a model in the architecture cannot be taken as the final word on whether the explanation offered by the model is correct (or incorrect). Under certain circumstances, however, the inability to build a model in this framework can be taken as a sign that the approach (or strategy) is flawed. More importantly, building cognitive models help us identify the possible sources of a phenomenon. This can in turn be used to develop a series of controlled experiments to decide between the sources.

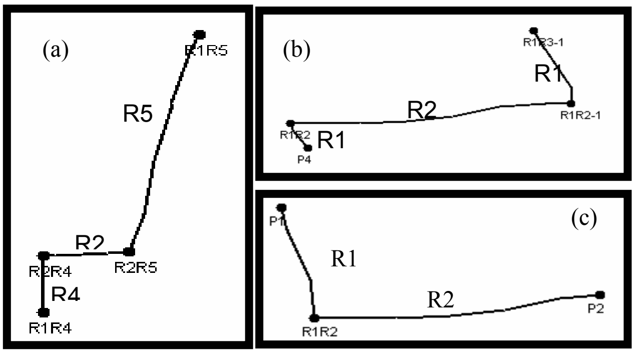


*Fig 5: Routes found by Simp1 from the map in Fig 4. (a) R1R4 to R1R5. (b) P4 to R1R3-1. (c) P1 to P2*

## Task 1 – Simplification in Route Recall

Curves recalled from spatial memories, whether they are rivers in Paris or routes by cab drivers rarely preserve their exact curvature or their orientation to each other and to other landmarks (Tversky, 2000). Details in a curve such as the actual angles at intersections are lost and route curvature is usually straightened. In this paper, we refer to this phenomenon as *simplification*. We explore how this phenomenon can arise from the architectural features of biSoar. In particular, we explore whether the chunking of the simplified diagram in WM (represent only that to which attention was paid) is enough to explain the emergence of simplification in recalled maps.

### Model 1

The agent (referred to as Simp1) is given the task of finding various routes in the map shown in Fig 4. Fig 5 shows the result of route-finding for certain locations from the map. The route-finding strategy used is a simple one in which the agent finds the routes on which the current point lies, finds the next point along all possible directions, calculates the Euclidean distance to the destination from each point and picks the one with the lowest value. The critical step in the strategy is the step where, once the next point has been selected, the agent notes the route from the current point to the selected next point paying attention to only the starting and ending points of the route. This results in a representation of the route that is simplified according to the attentional demands of the task, in WM. When Soar's chunking mechanism learns from the resolution of the sugboal, it learns this simplified representation from WM.

### Model 2

A new agent (Simp2) is created and given the same task as Simp1. Simp2's strategy is the same as Simp1's except that Simp2 chooses to pay attention to only the locations of important intersections and the names of the routes they lie on. During recall, Simp2 recalls these locations and connects them using straight lines. Fig 6 shows routes recalled between the same locations as in Fig 5.

### Discussion

The two models (represented by the two agents Simp1 and Simp2) indicate two different explanations for the simplification phenomenon. The simplified routes recalled by Simp1 are the result of an architectural feature of biSoar – bimodal chunking. Depending on which aspects of the routes that attention was paid to, Simp1 chunks a simplified version of the original route. Simp2 on the other hand, does not even bother trying to chunk the spatiality of the routes. Instead, it learns the locations of important intersections and the routes they are on and connects them with straight lines during recall. As mentioned before, the ability to create these models does not automatically suggest that either (or both) explanation is the definitive source of the simplification phenomenon. There could be other as yet unwritten models that might turn out to be, in fact, right. However, these models do suggest that one variable to control for is whether subjects are recalling only locations or both locations and routes. One way to do this would be to have a particularly attention grabbing feature on one of the curves (maybe a loop or sudden change in direction).

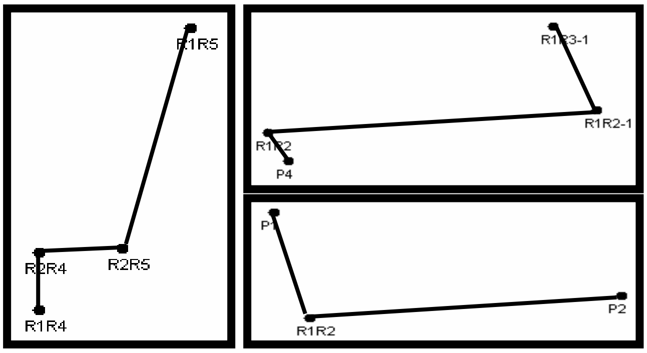


*Fig 6: Routes found by Simp2 from the map in Fig 4. (a) R1R4 to R1R5. (b) P4 to R1R3-1. (c) P1 to P2*

## Task 2 – Distortion in Geographic Recall

According to Stevens and Coupe (1978), when subjects were asked about the relation between San-Diego and Reno, most answered that San-Diego was to the west of Reno even though in reality, Reno is west of San Diego. They go on to suggest that this result indicated two things – one, that the cognitive map was unlikely to be a faithful metrical representation and two, that the representation was hierarchical in nature, the hypothesis being that since the subjects did not have any information about the relationship between SD and Reno they went up the hierarchy and compared the containing regions – California and Nevada. Since California is to the West of Nevada, it followed that SD was to the west of Reno.

We built three different models of problem solving for this task. Model 1 is of an agent that has a single simplified metrical representation of California and Nevada in LTM (and WM) like in Fig 7 (a). In this particular example San Diego to the West of Reno, but an agent that paid particular attention to these cities may have a metrical representation with the cities in their correct relationship to each other. Model 2 has symbolic information in LTM that San Diego is South of San Francisco and that Reno is East of San Francisco. It constructs a diagram (Figure 7(b)) in WM using this information and extracts the (wrong) answer from the diagram. Model 3 has symbolic information in LTM that San Diego is in California, Reno in Nevada and that California is to the West of Nevada. This information is used to construct a diagram (Figure 7(c)) and the (wrong) answer extracted from it.

### Discussion

The variety of models in Task 2 exhibit biSoar's flexibility in modeling spatial phenomena. Each model provides a different explanation and, in essence, suggests a separate control variable. For example, in Model 2, the explanation is that subjects use a specific strategy – that of comparing the location of the target cities to a common city and inferring the relationship from that knowledge. This strategy can be controlled for by using artificial maps (as Stevens and Coupe do in their original paper) that do not provide this extra information. Thus, models in biSoar have a straightforward mapping to issues to control for and building these models provides a natural way of discovering these issues. Of course, the experimenter is free to simply think of various explanations without modeling in biSoar, but the advantage is that it provides additional constraints and restricts the experimenter to those explanations that are cognitively possible. The disadvantage is that we do not know of any systematic way of generating these models/variables. Certain heuristics such as "look for at least one explanation from each possibility in the explanation space" can suggest lines along which the model builder/experimenter may approach the problem.

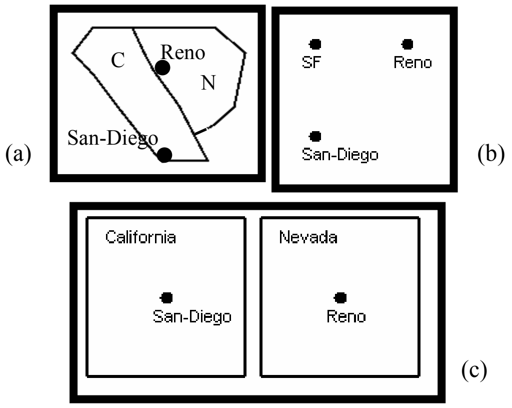


*Fig 7:(a) Map of SW U.S. in LTM (& WM) of Model 1. (b) & (c) are diagrams in WM constructed by Models 2 & 3*

## Related Work

**Soar –** Lathrop and Laird (2006) report on progress in their work on expanding Soar to include a perceptual representation and reasoning system. There is at least one important theoretical distinction between their work and ours. Our work is based on the assumption that all aspects of the agent's architecture including the cognitive state, memory, learning etc, are multi-modal and that during problem solving Soar can seamlessly access representations across all modalities. Lathrop and Laird take a different approach, one in which the perceptual system is part of the total cognitive system, but outside of high-level cognition. This means that perceptual representations can be accessed only through the input/output system and access to them is restricted to the input and output phases of Soar's decision cycle. In practice, the implementations are very similar and we believe their system can model most of what we do, including the visualization of information and subsequent extraction of the desired spatial relationship as in Model 3. However, they do not as yet have a theory of automatic learning (what we refer to as bimodal chunking) for the visual part, which provides the basis for an architectural explanation of phenomena such as simplification.

**ACT-R** – ACT-R or Adaptive Control of Thought – Rational (Anderson *et al.*, 2004) is a general cognitive architecture whose goal is to model all aspects of high-level human cognitive activity. However, there are no reports on any work in augmenting ACT-R's cognitive state to be multi-modal. Certain related work such as ACT-R/S (S for spatial) (Harrison & Schunn, 2002) augment ACT-R with representations for immediate space and object shapes for manipulation but there is no claim to a diagrammatic component that unifies experience whether from memory or perception.

**Other Work** – There have been a number of non-cognitive architecture oriented proposals for spatial representation and learning, notably the Spatial Semantic Hierarchy or SSH (Kuipers, 2000). The SSH is a multi-layered theory, that represents its knowledge of space at multiple levels – control, causal, topological and metrical, with the information at one level building on what was learned at the next lower level (except in the case of the metrical level.) In its current avatar, biSoar encompasses the topological and metrical levels of SSH. The representational and problem solving capabilities of biSoar and SSH with regards to topological information are similar. The real difference is at the metrical level. SSH proposes a few ways in which 2-D metric information may be represented but biSoar, and in particular, DRS provide a concrete representational format for metric information. Further, biSoar creates, modifies and inspects this information during problem solving making DRS an integral part of the problem solving process.

Other models include Absolute Space Representation (ASR) (Jefferies & Yeap, 2001) and MIRA□E (Barkowsky, 2001). Both combine models of representation with a metrical representation that has aspects of DRS.

Since SSH, ASR and MIRA□E are all intended to model spatial representation and reasoning, they lack the flexibility of a general cognitive architecture that biSoar provides.

## Concluding Remarks

We have presented a proposal for bimodal learning within the existing learning mechanism in Soar, chunking, and shown how building models of spatial representation and reasoning within this architecture can help in the design of experiments. We believe that a bimodal architecture augmented with bimodal chunking can be an useful vehicle in exploring the nature of the human cognitive map.

Two additional details need to be satisfactorily addressed for a measure of closure in this direction of research. The first relates to biSoar's rule matching process mentioned earlier, where elements in LTM rules are matched against structures in WM. It is not yet clear how to match diagrams on the LHS of rules to the diagrammatic part of WM. Second, the processes involved in composing diagrammatic elements from different LTM rules in WM according to the needs of the current goal. For example, subjects may remember the border of Texas using multiple diagrams - one consisting of a simplified overall view, another representing the "top hat" part and a third representing the coastline. During recall, the diagram is constructed by integrating these overlapping or locally inconsistent images with the aid of task-specific knowledge.

We believe that the idea of simplification that is presented extends naturally to other memories such as semantic and episodic memories. Further, even though it is presented in the context of Soar, the general ideas are likely to be applicable to other symbolic architectures like ACT-R.

## Acknowledgements

Advanced Decision Architectures Collaborative Technology Alliance sponsored by the U.S. Army Research Laboratory under Cooperative Agreement DAAD19-01-2-0009

## References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111*(4), 1036-1060.

Barkowsky, T. (2001). Mental processing of geographic knowledge. In D. R. Montello (Ed.), *Spatial information theory - foundations of geographic information science* (pp. 371-386). Berlin: Springer.

Chandrasekaran, B. (2006). *Multimodal cognitive architecture: Making perception more central to intelligent behavior.* Paper presented at the 21st National Conference on Artificial Intelligence, California.

Chandrasekaran, B., Kurup, U., Banerjee, B., Josephson, J. R., & Winkler, R. (2004). *An architecture for problem solving with diagrams.* Paper presented at the *Diagrammatic Representation and Inference conference.*

Harrison, A. M., & Schunn, C. D. (2002). *Act-r/s: A computational and neurologically inspired model of spatial reasoning.* Paper presented at the 24th Annual Meeting of the Cognitive Science Society, Fairfax, VA.

Jefferies, M. E., & Yeap, W. K. (2001). *The unity of global representations in a cognitive map.* Paper presented at the International Conference on Spatial Information Theory: Foundations of □eographic Information Science.

Kuipers, B. J. (2000). The spatial semantic hierarchy. *Artificial Intelligence, 119*, 191-233.

Kurup, U., & Chandrasekaran, B. (2006). *Multi-modal cognitive architectures: A partial solution to the frame problem.* Paper presented at the 28th Annual Conference of the Cognitive Science Society, Vancouver.

Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence, 33*(1), 1-64.

Lathrop, S., & Laird, J. E. (2006). *Incorporating visual imagery into a cognitive architecture: An initial theory, design and implementation* (No. CCA-TR-2006-01).

Lynch, K. (1960). *The image of the city.* Cambridge: MIT Press.

Stevens, A., & Coupe, P. (1978). Distortions in judged spatial relations. *Cognitive Psychology, 10*, 422-437.

Tolman, E. C. (1948). Cognitive maps in rats and man. *Psychological Review, 55*, 189-208.

Tversky, B. (2000). Levels and structure of spatial knowledge. In R. Kitchin & S. Freundschuh (Eds.), *Cognitive mapping: Past, present, and future* (1 ed., pp. 24-43): Routledge.

# Learning and Decision Model Selection for a Class of Complex Adaptive Systems

**Tei Laine(tei@iki.fi)**
Department of Computer Science
P.O.Box 68 (Gustaf Hällströmin katu 2b)
00014 University of Helsinki FINLAND

## Abstract

Computer modeling is gaining popularity in the study of systems whose underlying processes are difficult to observe and measure directly, or their controlled experimentation is not an option. Since real-world phenomena, for instance psychological or ecological, are often hugely complicated, and the models trying to capture their essence relatively complex, validation of the models and selection among the candidates is a challenge. Furthermore, not all computer models are used merely for explanatory purposes or to test theories, but some are used to support decision making. Therefore, it is critical which model the decision makers put their confidence on. In this article I discuss a pragmatic method for selecting between classes of models that are designed to increase understanding in the most significant single factor behind the global climate change, namely human land-use. My focus is on agent-based land-use and land-cover change models, and particularly models of learning and decision making. The proposed method fills the void left by traditional statistical model selection methods that are not applicable due to the nature of the model class of interest.

**Keywords:** Agent-based modeling; model selection; minimum description length principle; decision making.

## Introduction

These days Earth's land-cover is going through changes at faster pace than ever, and most of these changes are human initiated. Pervasive land-use and consequent land-cover changes, occurring in different time scales and spatial extent, have had and continually have adverse impact on local, regional and global level by destroying natural ecosystems and causing irreversible changes in global climate. In order to understand the impact land-use change has on ecological systems, not only its consequences but also the underlying mechanisms and forces driving land-use decisions need to be explained.

Empirical measurements are not sufficient to understand the combination of the factors behind the change (Parker, Manson, Janssen, Hoffman, & Deadman, 2003). On the other hand, experimental manipulation of landscapes is often impractical if not impossible (Baker, 1989). Combined with other methods, for instance household surveys and analysis of census data, computer models offer a relatively effortless method for testing alternative theories and formulating new hypotheses, analyzing implications of environmental policies, predicting changes and exploring interactions between, for instance, social, psychological, economical, bioecological, and even political and historical factors behind land-use.

A number of different techniques have been used in modeling the *land-use and land-cover change (LUCC)* (Parker et al., 2003), for instance equation-based models, logistic regression models based on suitability maps (Schneider & Pontius, 2001), system dynamic models, statistical methods, symbolic or rule-based systems combined with qualitative expert knowledge, and evolutionary models, such as genetic algorithms. Perhaps perhaps the most common methods are cellular automata (CA) and Markov chain (MC) models, or combinations of them (Brown, Riolo, Robinson, North, & Rand, 2005; Jenerette & Wu, 2001; Parker et al., 2003).

Most of the early modeling efforts have concentrated in biophysical processes rather than human actions (Itami & Gimblett, 2001), even if the majority of the land-use change is initiated by humans. On the other hand, mathematical and statistical methods ignore the spatial aspect of LUCC (Manson, 2000). Therefore, in this article I consider a type of models that still is an emerging approach, namely a combination of a cellular model representing the biophysical landscape, and an *agent-based* component representing the decision makers, either individuals, households or institutions. Land-use is then what links the agent to the landscape (Parker et al., 2003; Evans, Sun, & Kelley, 2006).

Since computer models are often used to inform decision makers in the process of designing environmental programs and policies, and the direct or indirect consequences of these decisions may be consequential, models' plausibility and adequacy to the task needs to be rigorously assessed, i.e., it is pivotal to have a right model to the task. Models may generate seemingly plausible outcomes even if the generating mechanism is quite arbitrary. On the other hand, proper tweaking of parameter values may make them produce any results the decision maker would like to see. The lack of adequate tools often makes it difficult to compare and choose between alternative models on a fair basis without relying on their face value, i.e., how well the model behavior confirms to the decision maker's ideals. Therefore, it is important that the choice of the model that decision makers put their confidence on is based on sound principles. In other words, the evaluation, validation and selection methods are as crucial as the models themselves.

Several different model selection methods, such as *Akaike's Information Criterion (AIC)* (Akaike, 1973), *Bayesian Information Criterion (BIC)* (Schwarz, 1978), and the *Minimum Description Length (MDL)* principle (Grünwald, 1998), particularly its enhanced version *Normalized Maximum Likelihood (NML)* distribution (Rissanen, 1999), apply to probabilistic model classes. However, LUCC models do not lend themselves easily to probabilistic interpretation but can be best characterized as *complex adaptive systems (CAS)*. Moreover, land-use change data is not always readily available in quantities warranting use of cross-validation or bootstrap methods (Lendasse, Wertz, & Verleysen, 2003).

In this article I study a model selection method based on a practical interpretation of the MDL principle. In the next chapter I review the agent-based framework for LUCC modeling. Discussion on the model selection criterion follows. The criterion was originally introduced and extensively evaluated with a set of artificial data in Laine (2006). Here its properties are addressed in the context of real-world data.

## Agent-based Models of Land-use and Land-cover Change

Two fundamental ideas behind *agent-based models (ABMs)* are: first, the decision making is distributed among autonomous actors, which either operate individually or may communicate and cooperate, and secondly, the heterogeneity of actors is captured by characteristics that may be unique or shared by agents. The focus is on the macro-level patterns in collective behavior emerging from agents' individual characteristics and micro-level phenomena, such as local behavior and interaction between agents.

ABMs come in multiple disguises but here I am particularly interested in models in which agents inhabit a simulated environment, so that they are 'physically' tied to a specific location and have a fixed neighborhood. The models of land-use and land-cover change fall into this category of models.

The agent-based approach has been used to study various land cover change related processes in several areas of the world: for instance agricultural land-use decision making by colonist households in Brazilian Amazon (Deadman, Robinson, Moran, & Brondizio, 2004), migration and deforestation in Philippines (Huigen, 2004), agricultural household land-use decision making in the US Midwest (Evans & Kelley, 2004; Laine & Busemeyer, 2004), reforestation in the Yucatan peninsula of Mexico (Manson, 2000), and ex-urban development in Maryland, US (Irwin & Bockstael, 2002).

### Land-use Framework

The conceptual assumptions behind the land-use framework were adapted from Cioffi-Revilla & Gotts (2003). The most important ones are listed below:

1. The *landscape* is an abstract rectangular area divided into *cells* of equal size, which serve as the decision-making units.

2. Each cell has various biophysical properties that remain constant over time.

3. The main actors in the model are autonomous *agents*. They have a potentially infinite existence, although they can perish. All agents are of the same type (e.g., households), but their individual characteristics may vary.

4. Agents control a region, called *parcel*, which is a set of adjacent cells on the two-dimensional landscape. Agents have exclusive access to this region, and there is no property exchange between the agents.

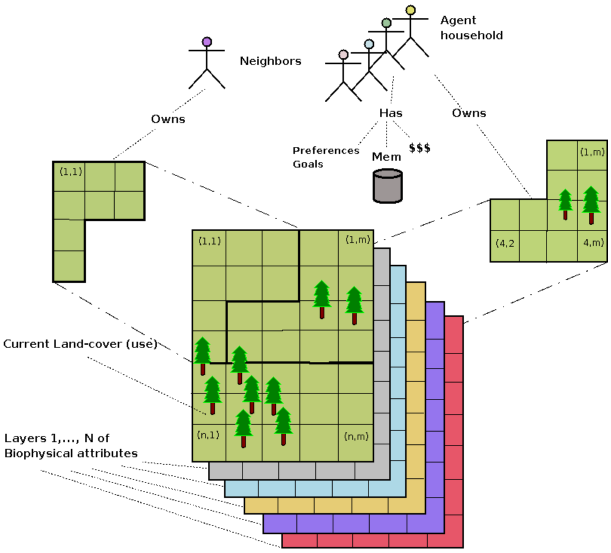


Figure 1: Main components of the land-use framework.

5. Agents make resource allocation decisions on their parcel in order to satisfy their goals. Agents have a limited set of available actions, i.e., options to which to allocate their resources. Agent actions change the use of the cells on their parcel.

6. All agents have the same learning and decision strategy.

7. The global environment consists of external conditions that are common to all parcels. These conditions may change over time.

The architecture of the system is depicted in Figure 1.

### Decision Models

At each decision round agents observe the state of their land, and make a decision about its use in the next round. They make the decision for each cell separately; they either decide to keep the old use or select another use from the given alternatives. After making the decision for each cell, they observe the payoff earned from different uses. This payoff is then used as a basis for the next decision.

In this study I am primarily interested in agents' learning and decision processes. Thus, the alternative model classes in selection consist of different decision and learning strategies. In addition to a random and a null model (which never makes any changes), other model classes chosen for the study constitute a set of relatively straightforward reinforcement-based strategies, familiar from psychology and economics literature. These are a model that makes locally greedy changes, Q-learner (Watkins & Dayan, 1992), and two versions of the *experience-weighted attraction (EWA)* model (Camerer & Ho, 1999): one that only considers its individual payoff (iEWA), and one that also takes its neighbors' payoff into account (sEWA).

## Model Selection Framework

Characteristic to the class of LUCC models, as opposed to more traditional cognitive models, is that they are often validated against land-use data instead of comparing the model's behavior to experimental human data. The modeling task then is to find out what kind of decision processes may have generated the observed land-use change patterns. This indirect derivation of agent behavior from the landscape poses another range of challenges to the validation process. Yet another validation technique emerging in LUCC modeling is field experiments, in which the researcher takes her laboratory to the stakeholders and makes them play a role game that mimics the real-world decision making context (Olivier Barreteau & Attonaty, 2001; Carpenter, Harrison, & List, 2005).

### Challenges to the Model Selection Criterion

So, which method should be used to select between agent-based LUCC models? There is no straightforward answer, but several inherent characteristics of the modeling domain needs to be taken into consideration. These challenges, more thoroughly discussed in Laine (2006), are reviewed next.

First, with the exception of some simple cases[1], it is dangerous to assume that some 'true' model exists, and design a system so that it tries to approximate this 'truth'. After all, model parameters and functions are not inherent properties of the system we want to model but theoretical constructs we use to describe the system. We impose the properties to the system. Again, there is no way to verify that a 'true model' exists, and consequently the task of estimating something that does not exist becomes quite impossible.

Secondly, existing model selection methods most commonly penalize for model complexity[2], i.e., its propensity to overfit, by taking the number of free parameters into account. A typical LUCC model is a collection of multiple autonomous components and processes that interact at multiple spatial levels and temporal scales. Thus, free parameters are not equally easy to identify in this class of models as they are in probabilistic or polynomial model classes.

Thirdly, the data available for the validation of CAS are not plenty and always not random samples. Sometimes it is even hard to make a distinction between the data and the model.

These considerations make it particularly clear that most of existing model selection methods, for instance penalized maximum likelihood methods, such as AIC or BIC, are inapplicable. Nevertheless, the MDL principle, and especially its refined formulation, the NML distribution, have some nice theoretical properties, but for many practically interesting model classes they cannot be calculated (Rissanen, 1999). Finally, in many cases the scarcity of data does not allow for adequate generalization tests.

### Normalized Minimum Error Principle

Here I propose a selection criterion that overcomes some of these challenges. It makes the following assumptions:

- No 'true' model exists.
- Measure of flexibility is based on the model's performance with respect to data, not some predetermined structural property.
- A model itself does not determine its fit to data, but an error function is required.

While the last two points address the trade-off between goodness-of-fit and the model class flexibility, the first one takes a more ideological standpoint on what is tried to achieve with the model selection criterion, namely that the goal is to find the best model to explain the data rather than a model that approximates some 'true' state of the world. We need to estimate the model's fit in order to quantify how well it captures the essential properties of the data.

The fit is not enough, since too flexible model is prone to overfit. *Two-part code*, also called a *crude version* of the MDL principle trades off flexibility to superior fit by choosing the model $H$ in class $\mathcal{M}_i$ that minimizes the sum $L(D|H,\mathcal{M}_i)+L(H|\mathcal{M}_i)$, where $L(\cdot)$ is the description length in bits. The underlying idea is that regularities in the data can be used to compress it, and the best model to explain the data is one that compresses the data most efficiently. In other words, the model using the least number of bits in describing the data most likely captures its underlying regularities. These regularities can then be used to gain insight on the structures and processes that generated the data.

The two-part code formulation still uses the maximum likelihood parameters to account for the model class flexibility (the second term in equation). We are not interested in the best-fitting model, but a well-fitting model in a class that is not overly flexible. In other words, we want to find a model that can reveal interesting patterns in the data, not a model that captures mere noise. This is where the error function comes into play. Next, I will present a method how to treat the trade-off between fit and flexibility adequately using errors.

If we want to explain an observed data sample $x^n$ from the set of all data samples $X^n$ with the help of the model class $\mathcal{M}_i$, ideally we want $\mathcal{M}_i$ to

1. contain a model $H$ that makes a small error on $x^n$, and
2. contain models $H'$ that do not make small errors on most $y^n$ belonging to $X^n$.

[1] Simple cases such as the model of the average height of six graders, or presidential candidate's approval rate.

[2] Following the terminology adopted in Laine (2006), I substitute the term 'flexibility' for 'complexity' for two reasons; first, the latter is heavily burdened, meaning different things for different people, and secondly, the LUCC model class and the modeled domain are inherently complex systems, so it would be misleading to imply that complexity is necessarily problematic.

This can be achieved by minimizing the following ratio, called *Normalized Minimum Error (NME)* (Laine, 2006):

$$NME(x^n, \mathcal{M}_i) = \frac{ER(x^n|\hat{\theta}(x^n, \mathcal{M}_i)}{\sum_{y^n \in X^n} ER(y^n|\hat{\theta}(y^n, \mathcal{M}_i))},$$

where $ER(\cdot)$ is the error model class $\mathcal{M}_i$ makes on $x^n$ using the parameter values $\hat{\theta}(x^n)$ that minimize the error, and $y^n$ are 'all possible data samples'. By normalizing each error this way we obtain a relative measure for fit and flexibility, which we can use as a model selection criterion.

The MDL principle is a general method of doing inductive inference, and the NME criterion is one way of implementing it. Yet another interpretation of the principle is the NML distribution, which selects a model class $\mathcal{M}_i$ whose *universal model* $H$, not necessarily in $\mathcal{M}_i$, minimizes the worst case regret. Regret of model $H$ with respect to class $\mathcal{M}_i$ is the extra number of bits that are required to describe the data sample $x^n$ using $H$ instead of using $x^n$'s maximum likelihood model in $\mathcal{M}_i$. $H$ is called a universal model, since it tries to mimic all models in the class $\mathcal{M}_i$. It has been proved (Rissanen, 1999) that the NML criterion defines a unique model that minimizes the maximum regret.

The NME criterion uses errors as measure of fit, whereas the NML criterion uses probabilities. The term in the denominator is the most crucial aspect of both criteria, since it accounts for their ability to penalize for excess flexibility. The relationship between these two was demonstrated in Laine (2006).

## Evaluation of the Criterion

The proposed criterion has been extensively tested with artificially generated data in Laine (2006). In this section I discuss some of its properties in the light of a representative case of real land-cover change data.

### Review of Experiments with Artificial Data

Acquisition of multiple samples of accurate land-cover data with a good resolution is difficult or at least time consuming. Therefore, the preliminary experiments were conducted with data generated by an artificial system, i.e., the same model classes that were used as candidate models were also used as data generating classes. This is a common practice when comparing multiple model selection methods (Busemeyer & Wang, 2000; Pitt, Myung, & Zhang, 2002). The experiments were run in several conditions by varying the biophysical and agent characteristics, and the error function.

The main findings in the first set of experiments are:

1. The criterion tends to select the generating class if it is among the candidates.

2. The criterion predominantly selects model classes with fewer free parameters, and never chooses a class more flexible than the generating class.

3. For no data set it strongly prefers any single class, but the selected model depends on the error function.

### Case-study

The data used in the second set of experiments comes from the state of Indiana in the Midwestern United States. The forest cover of the state of Indiana has undergone drastic changes during the last couple of hundred years; from almost 100% of the state being forest before the first settlers entered and cleared the land for agricultural production, down to 5-10% in the early 1900's, and then up to the current day's 20%, which mostly resides on the rolling hills of the South-central part of the state.

This study concentrates on deforestation and reforestation between 1940 and 1998 in two rural townships, Indian Creek and Van Buren, both about $10km \times 10km$ in size. The available data indicates that the forest cover has undergone a significant increase within the first 15 years of the study period and after that a modest but gradual increase. The overall increase of forest cover is around 20% in both townships. The change has not been unidirectional nor uniform; for instance, both deforestation and afforestation can be seen in the both townships, as pictured in Figure 2.

### Data

Data used in these experiments consists of land-cover maps covering the study period, slope and soil data, and ownership data. In addition to these, economic data (prices and wages), and forest growth data were imported as exogenous forces. The land-cover is represented as a grid of cells of size $50m \times 50m$ that records the land-use for each cell. Ownership, slope, and soil data is recorded per cell in similar grids.

### Experimental Conditions

The experiments were divided into a number of conditions by varying:

1. **Agent characteristics** Homogeneous vs. heterogeneous agents by household size, initial wealth and the number of neighbors.

2. **Fitting method** Landscape level vs. individual parcel level fit of parameters.

3. **Error function** (1) Mean absolute difference, (2) composition, (3) edge length, and (4) mean patch size. The first one measures the point by point difference between two landscapes, whereas the latter three calculate a squared difference between forest percentages, forest border lengths or mean forest patch sizes of two landscapes.

### Results

The proposed model selection criterion cannot be analyzed in isolation of the error function it uses. The current study uses four different error functions three of which are so called summary statistics; they characterize a single aspect of the land-cover, whereas the fourth one, mean absolute difference, is a location by location measure. This metric uses more information of the landscapes than the other three that do not consider location.

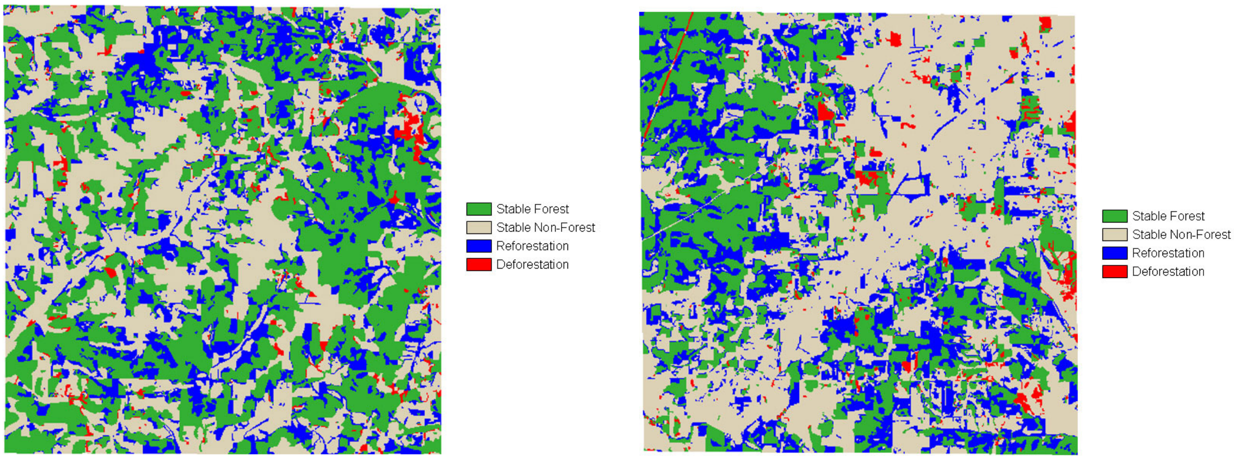


Figure 2: Deforestation, afforestation and stable forest cover in Indian Creek (left) and Van Buren (right) townships from 1940 to 1998.

| | Indian Creek | | Van Buren | |
|---|---|---|---|---|
| Error: | Selected | NME ($\mu$) | Selected | NME ($\mu$) |
| (1) | sEWA (c) | .25 (.413) | random | .499 (.588) |
| (2) | iEWA (c) | .05 (.415) | Q (c) | .12 (.585) |
| (3) | sEWA (i) | .35 (.463) | sEWA (c) | .05 (.537) |
| (4) | sEWA (c) | .103 (.406) | iEWA (c) | .49 (.594) |

Table 1: Selected model classes and their NME scores for homogeneous agents with landscape level fit (mean scores in parenthesis, c=collectively fitted, i=individually fitted).

| | Indian Creek | | Van Buren | |
|---|---|---|---|---|
| Error: | Selected | NME ($\mu$) | Selected | NME ($\mu$) |
| (1) | sEWA (i) | .193 (.249) | iEWA (c) | .59 (.752) |
| (2) | greedy (c) | .04 (.180) | Q (c) | .39 (.820) |
| (3) | Q (i) | .154 (.205) | sEWA (c) | .67 (.795) |
| (4) | greedy (i) | .674 (.778) | Q (c) | .03 (.222) |

Table 2: Selected model classes and their NME scores for heterogeneous agents with parcel level fit (mean scores in parenthesis, c=collectively fitted, i=individually fitted)).

Summary statistics are supposedly easier to fit, since there are several possible ways to get them right, for instance, several different land-cover configurations may have the same composition. Consequently, there are fewer ways of getting them wrong, too. However, there are very few ways, actually only one, of getting the location-by-location comparison correct, and a considerable number of ways of getting it wrong.

The selected models together with the respective NME scores and their means are presented in Tables 1 and 2 for homogeneous and heterogeneous agents, respectively, using different error functions. The number of decimal points is determined by how many decimals are needed to distinguish between the NME scores.

For homogeneous agents only one time out of eight is the individually fitted model class selected, whereas for heterogeneous agents three times out of eight. This is roughly what can be expected; when there is more variation in the agent population, there is potentially something to be gained by fitting the agents individually. In other words, the benefit attained in better fit outweighs the cost in extra flexibility.

In general, the selection criterion selects simpler models, i.e., collectively fitted classes, for homogeneous agents with both landscapes. However, with heterogeneous agents it predominantly selects individually fitted classes for Indian Creek, but collectively fitted for Van Buren. This indicates that either agent heterogeneity plays a bigger role in Indian Creek and some of the models classes are able to capture it, or the larger number of agents in Van Buren is hard to fit, and the selection criterion resorts to making a safe decision of selecting simpler model classes.

Finally, null and random model classes are seldom selected. This supports the fact that the real landscapes are dynamic, and undergo very characteristic changes which cannot be captured either by a chaotic or a stationary process.

## Discussion and Future Work

The literature provides us with evidence that, somewhat counter-intuitively, location-by-location comparison is not that difficult after all. Pontius *et al.* (2004) argue that not a single model has been reported that is able to predict the location of land-cover changes better than a null model, a model that predicts no change. The proposed selection criterion is looking for a model class that is simple and contains a model that fits the data well. Since the changes over time in the real landscapes are usually small, a model that predicts few changes should perform well. Why does not the NME criterion select the null model?

In the current experiments 'all possible data' was replaced by 'all available data' for practical reasons. This decision has detrimental consequences. For instance, even if both Indiana landscapes exhibit some idiosyncrasies, nevertheless they can be assumed to be generated by 'the same process'; they are physically linked, subject to the same weather conditions and under the same county rules.

However, the NME criterion penalizes a model class, possibly the null model class, that fits well both of these data samples, as if it fitted 'all data' well, and never chooses the same model class for both landscapes. There is no outstanding solution to this dilemma yet. Thus, the very first theoretical and practical challenge is to circumscribe the actual meaning of 'all possible data' in order to fully understand the relation between theoretical underpinnings of the proposed criterion and the underlying practical issues inherent to the modeled domain.

Finally, although a common agreement in the field of LUCC modeling is that model validation is crucial, this study represents one of the first attempts to introduce model selection methodology to this complex spatial domain. The goal of model selection is to find a model that helps us gain insight into the processes underlying the — natural, psychological or economic — phenomenon of interest. Although the proposed criterion penalizes for excess complexity, simplicity is not the end in itself, but prevents us from becoming overconfident in more complex models when there is not enough data to support them. On the other hand, a considerable reflection should be involved when choosing the candidate models: too simplistic models to start with do not bring us any closer to understanding complex natural phenomena.

## Acknowledgments

This study was funded by the Biocomplexity grant (NSF SES0083511) for the Center for the Study of Institutions, Population, and Environmental Change (CIPEC) at Indiana University, and was also supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778.

## References

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In B. Petrox & F. Caski (Eds.), *Second International Symposium on Information Theory* (p. 267-281). Akademiai Kiado, Budapest, Hungary.

Baker, W. L. (1989). A review of models of landscape change. *Landscape Ecology*, *2*(2), 111-133.

Brown, D. G., Page, S., Riolo, R., Zellner, M., & Rand, W. (2005). Path dependence and the validation of agent-based spatial models of land use. *International Journal of Geographical Information Science*, *19*(2), 153-174.

Brown, D. G., Riolo, R., Robinson, D. T., North, M., & Rand, W. (2005). Spatial process and data models: Toward integration of agent-based models and GIS. *Journal of Geographical Systems*, *7*, 25-47.

Busemeyer, J. R., & Wang, Y.-M. (2000). Model comparisons and model selections based on generalization criterion methodology. *Journal of Mathematical Psychology*, *44*, 171-189.

Camerer, C., & Ho, T.-H. (1999). Experience-weighted attraction learning in normal form games. *Econometrica*, *67*(4), 827-874.

Carpenter, J. P., Harrison, G. W., & List, J. A. (Eds.). (2005). *Field experiments in economics* (Vol. 10). Elsevier.

Cioffi-Revilla, C., & Gotts, N. M. (2003). Comparative analysis of agent-based social simulations: Geosim and FEARLUS models. *Journal of Artificial Societies and Social Simulation*, *6*(4).

Deadman, P., Robinson, D., Moran, E., & Brondizio, E. (2004). Colonist household decisionmaking and land-use change in the Amazon rainforest: An agent-based simulation. *Environment and Planning: Planning and Design*, *31*, 693-709.

Evans, T., & Kelley, H. (2004). Multi-scale analysis of a household level agent-based model of landcover change. *Journal of Environmental Management*, *72*, 57-72.

Evans, T., Sun, W., & Kelley, H. (2006). Spatially explicit experiments for the exploration of land-use decision-making dynamics. *International Journal of Geographical Information Science*, *20*(9), 1013-1037.

Grünwald, P. (1998). *The minimum description length principle and reasoning under uncertainty*. Doctoral dissertation, University of Amsterdam.

Huigen, M. G. A. (2004). First principles of the MameLuke multi-actor modelling framework for land-use change, illustrated with a Philippine case study. *Journal of Environmental Management*, *72*, 5-12.

Irwin, E. G., & Bockstael, N. E. (2002). Interacting agents, spatial externalities and the evolution of residential land use patterns. *Journal of Economic Geography*, *2*, 31-54.

Itami, R., & Gimblett, H. (2001). Intelligent recreation agents in a virtual GIS world. *Complexity International Journal*, *08*.

Jenerette, G. D., & Wu, J. (2001). Analysis and simulation of land-use change in the central Arizona - Phoenix region, USA. *Landscape Ecology*, *16*, 611-626.

Laine, T. (2006). *Agent-based model selection framework for complex adaptive systems*. Doctoral dissertation, Indiana University.

Laine, T., & Busemeyer, J. (2004). Comparing agent-based learning models of land-use decision making. In C. L. Marsha Lovett Christian Schunn & P. Munro (Eds.), *Proceedings of the Sixth International Conference on Cognitive Modeling* (p. 142-147). Lawrence Erlbaum Associates.

Lendasse, A., Wertz, V., & Verleysen, M. (2003). Model selection with cross-validation and bootstraps — application to time series prediction with RBFN models. In (p. 573-580). Berlin, Germany: Springer-Verlag.

Manson, S. M. (2000). Agent-based dynamic spatial simulation of land-use/cover change in the Yucatan peninsula, Mexico. In *4th International Conference on Integrating GIS and Environmental Modeling (GIS/EM4): Problems, Prospects and Research Needs.* Banff, Alberta, Canada.

Olivier Barreteau, F. B., & Attonaty, J.-M. (2001). Role-playing games for opening the black box of multi-agent systems: method and lessons of its application to senegal river valley irrigated systems. *Journal of Artificial Societies and Social Simulation*, *4*(2).

Parker, D. C., Manson, S., Janssen, M., Hoffman, M., & Deadman, P. (2003). Multi-agent system models for the simulation of land-use and land-cover change: A review. *Annals of the Association of American Geographers*, *93*(2), 316-340.

Pitt, M. A., Myung, I. J., & Zhang, S. (2002). Toward a method of selecting among computational models of cognition. *Psychological Review*, *109*(3), 472-491.

Pontius, R. G., Huffaker, D., & Denman, K. (2004). Useful techniques of validation for spatially explicit land-change models. *Ecological Modelling*, *79*, 445-461.

Rissanen, J. (1999). Hypothesis selection and testing by the MDL principle. *The Computer Journal*, *42*(4), 260-269.

Schneider, L. C., & Pontius, R. G. (2001). Modeling land-use change in the Ipswich watershed, Massachusetts, USA. *Agriculture, Ecosystems and Environment*, *85*, 85-94.

Schwarz, G. (1978). Estimating the dimension of the model. *The Annals of Statistics*, *6*, 461-464.

Watkins, C., & Dayan, P. (1992). Q-learning. *Machine Learning*, *8*(3/4), 279-292.

# Be Wary of What Your Computer Reads: The Effects of Corpus Selection on Measuring Semantic Relatedness

**Robert Lindsey** (lindsr@rpi.edu)

**Vladislav D. Veksler** (vekslv@rpi.edu)

**Alex Grintsvayg** (grinta@rpi.edu)

**Wayne D. Gray** (grayw@rpi.edu)

Rensselaer Polytechnic Institute, 110 8th Street
Troy, NY 12180 USA

## Abstract

Measures of Semantic Relatedness (MSRs) provide models of human semantic associations and, as such, have been applied to predict human text comprehension (Lemaire, Denhiere, Bellissens, & Jhean-Iarose, 2006). In addition, MSRs form key components in more integrated cognitive modeling such as models that perform information search on the World Wide Web (WWW) (Pirolli, 2005). However, the effectiveness of an MSR depends on the algorithm it uses as well as the text corpus on which it is trained. In this paper, we examine the impact of corpus selection on the performance of two popular MSRs, Pointwise Mutual Information and Normalised Google Distance. We tested these measures with corpora derived from the WWW, books, news articles, emails, web-forums, and encyclopedia. Results indicate that for the tested MSRs, the traditionally employed books and WWW-based corpora are less than optimal, and that using a corpus based on the New York Times news articles best predicts human behavior.

**Keywords:** Measures of Semantic Relatedness, semantic similarity, training corpus, corpus comparison, Pointwise Mutual Information, PMI, Normalised Google Distance, NGD, computational linguistics, natural language processing.

## Introduction

Adding a text-comprehension component to cognitive models is a worthy goal, but it is a goal with many obstacles standing in its way. Although grammar parsing is still a major problem in computational linguistics, we are close to being able to accurately approximate relative meanings of words and documents. Using statistical techniques known as Measures of Semantic Relatedness (MSRs), we can automatically extract word definitions and relationships from large text corpora.

MSRs have been used in modeling language acquisition (Landauer & Dumais, 1997), human web-browsing behavior (Fu & Pirolli, 2007), text comprehension (Lemaire et al., 2006), semantic maps (Veksler & Gray, 2007) and many other modeling applications. In more applied domains, MSRs have been used to develop a wide variety of applications such as augmented search engine technology (Dumais, 2003) and automated essay-grading algorithms for the Educational Testing Service (Landauer & Dumais, 1997). MSRs have a wide range of practical applications and are potentially useful to any cognitive model or AI agent dealing with text (Veksler, Grintsvayg, Lindsey, & Gray, Submitted).

MSR performance depends on the corpus on which it is trained. Imagine if a child learning the English language were only allowed to read Shakespeare. Although the child would certainly learn English, he or she would undoubtedly encounter a number of communications problems. A conversation with this child would be difficult because they learned a very out-of-style form of English. Many of the words in the text the child learned from are used less often nowadays, some of those words are used more often, and some maybe not at all. Moreover, many of the words would have acquired new meanings, or would be used in different contexts than in Shakespeare's day. In addition, a child exposed exclusively to Shakespeare might be able to converse about love and war, but not about how to hail a taxi or how to reboot a computer. All in all, the choice of a set of learning material, or text corpora, for children has a profound impact on how well they comprehend English. The same concept applies to MSRs.

MSRs try to learn word relations the same way children do (Landauer & Dumais, 1997; Newport & Aslin, 2004; Newport, Hauser, Spaepen, & Aslin, 2004), and consequently their effectiveness is dependent on the text corpus from which they glean information. Whereas children's exposure to speech and text may, to some degree, be considered open to many sources, MSRs are strictly bound by their training corpora. MSRs calculate the probability of the co-occurrence of two query words in order to ascertain their semantic relatedness value. This probability varies greatly from one corpus to another, so the output of MSRs trained on different text corpora also varies greatly. There are many corpora commonly used to train MSRs and each produces different semantic relatedness values.

Landauer and Dumais (1997) claim that because children do not hear most of their lexicon, they must gain their vocabulary through books. Consequently, MSRs are often trained on books in the hopes of gaining knowledge from the same source as children. To our knowledge, this corpus choice has never been objectively validated and rigorously examined in comparison with other corpus types.

Certain MSRs may take as their corpus the entire World Wide Web (Turney, 2001). A naive assumption might be that such an overwhelmingly large amount of text will result in properly trained MSRs, and that although some web pages will not accurately represent the semantic relations of our language, those few unhelpful websites are statistically insignificant. To our knowledge, the use of the World Wide

Web as a training corpus is just as unfounded as the use of any given corpus of books.

Using books, the internet, or any other text corpus that has not been studied rigorously for its effect on the performance of MSRs may seriously compromise MSR-based applications. Just as children must be trained on proper material, an MSR must likewise be trained on the proper corpus in order to accurately model human lexical knowledge. In this paper, we examine the impact of corpus selection on the performance of two popular MSRs, Pointwise Mutual Information and Normalised Google Distance. We tested these measures in combinations with WWW, books, news articles, emails, web-forums, and encyclopedia-like corpora. All MSR-corpus pairs were evaluated as to their ability to represent human lexical knowledge based on data from a large-scale free-association psychological study (Nelson, McEvoy, & Schreiber, 1998).

## The Evaluation Challenge

Deciding how to evaluate the goodness of MSRs presents a daunting challenge. First, there are at least a dozen MSRs in the published literature and more are being invented each year. Second, as we will show, the goodness of an MSR depends at least partially on the corpus on which it is trained. Third, it may well be that different MSRs capture human semantic relatedness more so in some tasks (e.g., deciding what link to click on next) than in others (e.g., deciding if the content of a paragraph provides the answer to a sought-after question).

Clearly, we do not have room in this small paper to exhaustively explore the problem space implied by the combination of these three factors. Rather, as discussed below, we choose two MSRs, a small set of large corpora, and one criteria task on which reliable and valid measures of human performance exist. However, our work is ongoing, and we intend this paper to be an exemplar, not an exhaustive, evaluation of MSRs.

## Measures of Semantic Relatedness

MSRs give computers the ability to quantify the meaning of text. MSRs define words in terms of their connection strengths to other words, and they define connection strengths in terms of word co-occurrence. In other words, two terms are related if they often occur in the same contexts. Two terms are synonymous if their contexts are identical.

### Pointwise Mutual Information (PMI)

PMI is a well-established and successful measure for approximating human semantics (Turney, 2001). PMI is based on the probability of finding two terms of interest (t1 and t2) within the same window of text versus the probabilities of finding each of those terms separately:

$$PMI(t1,t2) = \log_2 \left| \frac{P(t1,t2)}{P(t1) \mid P(t2)} \right|$$

where P(t1) and P(t2) are the probabilities of finding a window of text in the corpus containing the term t1 or t2 respectively; and P(t1,t2) is the probability of finding a window of text in the corpus containing both t1 and t2. Please see Turney (2001) for a more expanded discussion of PMI.

Window-size is a free parameter in PMI and most-all other MSRs. For web-based corpora window size is typically set to be a webpage; however, it can also be any grouping of text – a sentence, an email, a webpage, or some other organizational group.

### Normalised Google Distance (NGD)

NGD is another popular MSR (Cilibrasi & Vitanyi, 2007) that measures the similarity between two terms by using the probability of co-occurrences as demonstrated by the following equation:

$$NGD(t1,t2) = \frac{\max\{\log f(t1), \log f(t2)\} - \log f(t1,t2)}{\log M - \min\{\log f(t1), \log f(t2)\}}$$

where M is the total number of searchable Google pages, and *f(x)* is the number of pages that a Google search for *x* returns.

Although NGD was originally based on the Google search engine, this formula may be used in combination with other text corpora just as well. That Google's entire document-base is a better text corpus for this MSR is exactly the premise that we wish to challenge in the current work.

In order to use NGD as a relatedness measure, rather than a measure of distance, we convert NGD scores into similarity scores by subtracting NGD from 1 (1 being the maximum NGD score). From this point forth we will refer to the similarity score based on the NGD formula as the *Normalized Similarity Score* (*NSS*).

### Corpus Issues

A text corpus used to train an MSR may suffer from a variety of problems that impair its effectiveness. The content may be too old to accurately represent the semantic relatedness of words, as modern language uses words more or less frequently than in the past. Thus, classic literature may not be the ideal training corpus for MSRs.

Text corpora may also be too biased to be useful. For example, a corpus comprised of writings from a single political party will likely lead an MSR to calculate an overly strong relatedness between words like “axis” and “evil”. Likewise, a biased corpus may calculate a weak relatedness between words in situations where it should be higher. We may find that the internet has a commercial (or some other) bias and thus will not make a good overall training corpus.

Additionally, text corpora may be too impoverished or contain bad examples of language. A log of instant messaging conversations, for example, may provide a poor source of the English language. Using poorly written text as the training material would be similar to learning English from someone who does not speak English.

Text corpora may be too structured. For example, a dictionary or an encyclopedia may turn out to be a poor training source for MSRs.

By the same token, text corpora may be too unstructured. We presume that web forums contain conversational English and would thus make a great MSR training source, but the lack of structure in such corpora may make these suboptimal, as well.

Additionally, a text corpus may be computationally expensive to use. If it is excessively large, many MSRs will take a long time to produce a result, and some MSRs will not be able to produce the result at all.

## Corpus Evaluation

In order to select an optimal training corpus for an MSR, many corpora must be tested and have their performances compared. We studied two MSRs, PMI and NSS, and evaluated their performance on six unique corpora. The following sections describe the method by which we performed our evaluations.

### MSRs

PMI and NSS were the two MSRs used in this study. These are two popular MSRs that can handle all of the corpus types that we were considering in our research. Other MSRs, e.g. LSA (Landauer & Dumais, 1997), GLSA (Matveeva, Levow, Farahat, & Royer, 2005), ICAN (Lemaire & Denhiére, 2004), simply cannot handle large corpuses (e.g. WWW).

For five of the corpora the text-window size was a webpage. For the sixth corpus, the Enron Email Corpus, the text-window size was an email.

### Corpora

**Google Corpus**

This corpus is an extremely large collection of text (the World Wide Web), and is a popular choice for a training corpus. One major advantage of this corpus is that MSRs run extremely fast on it. Counting the number of hits returned by a search takes an inconsequential length of time.

**Wikipedia Corpus**

Wikipedia is the largest, free-content encyclopedia on the internet. We chose to study this corpus because it represents a great wealth of human knowledge. In order to use this corpus, we count the hits returned by a Google search for the terms after restricting our results to "site:wikipedia.org".

**New York Times Corpus**

New York Times is a news source that we chose to study as a corpus because of their large collection of online articles. We access this corpus the same way we access Wikipedia, by restricting Google searches to "site:nytimes.com".

**Project Gutenberg Corpus**

Books make a popular choice as an MSR training corpus (e.g. Landauer & Dumais, 1997). Project Gutenberg is an online collection of over 20,000 books. This corpus represents one of the largest online collections of books available. In order to use this corpus, we count the hits returned by Google searches restricted to "site:gutenberg.org/files".

**Google Groups Corpus**

Google Groups is a subdivision of Google's website that hosts online discussions and forums. This corpus was chosen because it represents a large collection of informal conversational language. We use this corpus in the same way we use Wikipedia, New York Times, and Project Gutenberg – by restricting our searches on Google to "site:groups.google.com".

**Enron Email Corpus**

Some time ago, a large collection of emails from Enron Corporation's top management personnel was released to public-domain. We chose to study this collection of emails as a training corpus because it is one of the largest collections of emails available. The hypothesis is that emails may make for an excellent corpus choice because they contain modern conversational language. In order to use this corpus, we imported all email bodies into a database, and ran queries on this database to find out the probability/frequency information for each PMI/NSS request.

### Limitations

Each of the corpora we chose represents a sampling from the set of all possible corpora. It is unclear to us how representative or non-representative our selection is of this larger set. Indeed, it is unclear to us how to formally characterize our selected corpora or the larger set of corpora so as to answer this question. Hence, our only claim for our current work is that we compare each of our two selected MSRs on each of our six corpora. These comparisons should allow us to begin to characterize the ways in which these two MSRs predict human performance when provided with equal training. (We view our effort as the first study of its kind, not the last.)

### Evaluation

Our evaluation method is based on a comparison between the performance of an MSR trained on particular text corpus and semantic relatedness data collected from a large-scale free-association experiment (Nelson et al., 1998). In this experiment, subjects were given a stimulus word, *cue*, and were then asked what word first came to mind, *target*. The target word that first came to mind is considered to be the most semantically related word for that cue, for that participant. More than 6,000 participants produced nearly three-quarters of a million responses to 5,019 cue words.

In order to find out whether the MSRs, trained on the six provided corpora, agreed with human judgments of word relatedness, we checked that the MSRs picked target words for each cue as more relevant to that cue than other random words. To do this, we added a list of *n* random nouns to the list of *n* target words for each cue, resulting in a list of *2n* words (*n* was limited to a maximum of 5 for cue words that were associated with more than 5 targets); this list of words was then sorted by MSRs according to word-cue relatedness. If a given MSR perfectly agreed with human judgments, the top *n* words in the sorted list would be all of the human-picked targets for that cue. If half of the targets were found by the MSR to be less relevant to the cue than half of the random words, the MSR performance on that cue would be considered 50%. The average percentage of targets found in the top *n* MSR-sorted words was used as the overall MSR performance score.

## Results & Discussion

We evaluated PMI and NSS on the following corpora: Project Gutenberg, Google Groups, Google, Enron, Wikipedia, and New York Times. PMI performed best on the New York Times corpus with an average score of 67.3%. PMI performed the worst on the Project Gutenberg corpus, the massive online collection of books, with an average score of 43.2%. NSS performed best on the Wikipedia corpus with an average score of 65.4%. NSS performed worst on the Project Gutenberg corpus with an average score of 54.2%. A two-factor ANOVA revealed a significant main effect of Corpus, F(5,25080) = 824.45, $p < .001$, a significant main effect of MSR, F(1,5016) = 1094.67, $p < .001$, and a significant effect of the Corpus by MSR interaction, F(5,25080) = 229.80, $p < .001$. PMI's performance showed a high dependence on the text corpus used, while NSS varied less from corpus to corpus.

NSS performed better than PMI on all but the New York Times corpus (mean NSS performance = 60.2%; mean PMI performance = 55.0%), and the overall performances of the two MSRs were highly correlated across the six corpuses (r-square = .82).

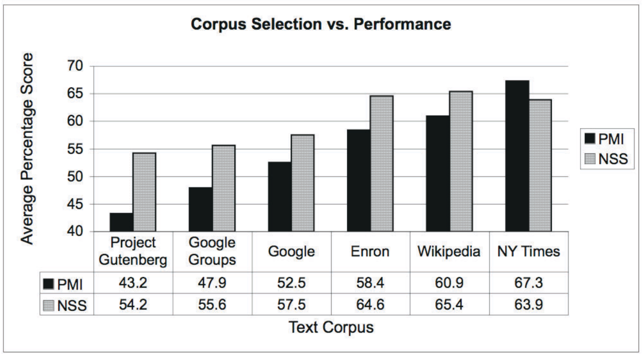


**Figure 1.** Corpus comparison for PMI and NSS. Standard error bars are too small to be displayed.

We were surprised that the New York Times corpus performed the best out of all the corpora we tested on PMI. It is not nearly as extensive as the Google corpus, nor as structured as Wikipedia, nor does it contain as much conversational English as the Enron Email Corpus or Google Groups. Yet it clearly had the highest score. Also surprisingly, Project Gutenberg, which is a large collection of online books, was the worst of these corpora. These findings have serious implications for the significant portion of MSR research and applications using books as the training corpus.

Our results show that corpus selection has a significant impact on an MSR's performance. One need look no further than at the difference in average scores between PMI using the Project Gutenberg corpus and PMI using New York Times corpus to see this fact. The fact that NSS scores do not vary nearly as much as PMI across different corpus selections indicates the presence of an MSR by corpus interaction effect. Further evidence of this effect lies in the fact that the New York Times corpus, our best corpus for PMI, did not perform as expected on NSS, our best-performing MSR. This MSR by corpus interaction effect is something in need of further investigation.

Another question that inevitably arises is why the Google corpus, which gives access to the World Wide Web as a corpus, is a suboptimal choice. Both of the MSRs that we tested, PMI and especially NSS, were designed to account for the format of the World Wide Web, and rely on its abundance of information (Cilibrasi & Vitanyi, 2007; Turney, 2001). According to our results, however, it appears that both PMI and NSS may be better served by a smaller corpus.

## Summary & Conclusions

How "good" a text corpus is for an MSR is not an intuitive matter. We found that the Project Gutenberg corpus, a large collection of books, did a poor job of modeling the human lexicon. Had we been intending to use PMI or NSS in an application such as a cognitive model and had chosen the Project Gutenberg corpus, we would have selected the worst choice possible and our cognitive model's ability to understand text as humans do would have been seriously impaired.

Our study is still ongoing. Rather than evaluating just one or two MSRs trained on a variety of corpora, we would like to test many more MSRs, on many more corpora, using various evaluation techniques (Veksler & Gray, 2006). Ultimately, we would like to find a text corpus that would be the optimal choice for all MSRs. If we knew the optimal choice for a text corpus, when using a semantic relatedness component in ACT-R (Anderson et al., 2004), C-I (Kintsch, 1988), or some other cognitive architecture, we could tell researchers exactly what corpus to train their MSR on. Researchers could rest easy knowing that their semantic relatedness component was performing at the highest level possible. Rather than worrying about the details of their MSR, we hope to allow researchers to be able to focus their attention on the actual MSR-based applications and cognitive models.

## Acknowledgments

We would like to thank Stephane Gamard for his contributions to our research. We would also like to thank Dr. Wallace of RPI for providing with the Enron email corpus. The work was supported in part by the Disruptive

Technology Office, ARIVA contract N61339-06-C-0139 issued by PEO STRI. The views and conclusions are those of the authors, not of the U.S. Government or its agencies.

## References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglas, S., Lebiere, C., & Quin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111*(4), 1036-1060.

Cilibrasi, R., & Vitanyi, P. M. B. (2007). The Google similarity distance. *Ieee Transactions on Knowledge and Data Engineering, 19*(3), 370-383.

Dumais, S. (2003). Data-driven approaches to information access. *Cognitive Science, 27*(3), 491-524.

Fu, W.-T., & Pirolli, P. (2007). SNIF-ACT: A cognitive model of user navigation on the World Wide Web. *Human-Computer Interaction, in press*.

Kintsch, W. (1988). The role of knowledge in discourse comprehension: A construction-integration model. *Psychological Review, 95*, 163–182.

Landauer, T. K., & Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review, 104*(2), 211-240.

Lemaire, B., & Denhiére, G. (2004). Incremental construction of an associative network from a corpus. In K. D. Forbus, D. Gentner & T. Regier (Eds.), *26th Annual Meeting of the Cognitive Science Society, CogSci2004*. Hillsdale, NJ: Lawrence Erlbaum Publisher.

Lemaire, B., Denhiere, G., Bellissens, C., & Jhean-Iarose, S. (2006). A computational model for simulating text comprehension. *Behavior Research Methods, 38*(4), 628-637.

Matveeva, I., Levow, G., Farahat, A., & Royer, C. (2005). *Term representation with generalized latent semantic analysis.* Paper presented at the 2005 Conference on Recent Advances in Natural Language Processing.

Nelson, D. L., McEvoy, C. L., & Schreiber, T. A. (1998). The University of South Florida word association, rhyme, and word fragment norms. : http://www.usf.edu/FreeAssociation/.

Newport, E. L., & Aslin, R. N. (2004). Learning at a distance I. Statistical learning of non-adjacent dependencies. *Cognitive Psychology, 48*(2), 127-162.

Newport, E. L., Hauser, M. D., Spaepen, G., & Aslin, R. N. (2004). Learning at a distance - II. Statistical learning of non-adjacent dependencies in a non-human primate. *Cognitive Psychology, 49*(2), 85-117.

Pirolli, P. (2005). Rational analyses of information foraging on the Web. *Cognitive Science, 29*(3), 343-373.

Turney, P. (2001). Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL. In L. De Raedt & P. Flach (Eds.), *Proceedings of the Twelfth European Conference on Machine Learning (ECML-2001)* (pp. 491-502). Freiburg, Germany.

Veksler, V. D., & Gray, W. D. (2006). *Test case selection for evaluating measures of semantic distance.* Paper presented at the 28th Annual Meeting of the Cognitive Science Society, Vacouver, BC.

Veksler, V. D., & Gray, W. D. (2007). *Mapping semantic relevancy of information displays.* Paper presented at the CHI 2007, San Jose, CA.

Veksler, V. D., Grintsvayg, A., Lindsey, R., & Gray, W. D. (Submitted). *A proxy for all your semantic needs.*

# Queueing Network Modeling of Mental Architecture, Response Time, and Response Accuracy: Reflected Multidimensional Diffusions

**Yili Liu (yililiu@umich.edu)**
Department of Industrial and Operations Engineering, University of Michigan
1205 Beal Avenue, Ann Arbor, MI USA

## Abstract

Response time (RT) and response accuracy are two of the most commonly used performance measures in cognitive psychology and studies of cognitive architecture. This paper examines the relationship and establishes a bridge between two currently separated groups of mathematical models of RT: models of RT and mental architecture and models of RT and accuracy. The bridge, called QN-RMD, is established by extending the queueing network (QN) architecture model of RT (Liu, 1996), which has successfully integrated a large number of RT-architecture models as special cases, and by representing the state changes in a mental QN as Reflected Multidimensional Diffusions (RMD). More specifically, the "state" of a K-server QN mental architecture is represented as a reflected diffusion space of K dimensions, in which "reflecting barriers" represent and reveal architectural constraints, while "absorbing barriers" represent accuracy-related response criteria, analogous to diffusion models of RT. This approach moves beyond the current 1-D diffusion models that have successfully accounted for but are limited to single-stage fast responses. 1-D diffusions can only represent the "state" of a single server system in stochastic information accumulation, not multi-server architectures. This approach extends the architectural RT models to account for accuracy, brings the diffusion/accumulator models to the architectural domain, and unifies RT/accuracy/mental architecture modeling in a larger framework.

## Introduction

*Response time* (RT) is arguably the most commonly used performance measure in cognitive psychology research; it is regarded as a reflection of the dynamic activities of an underlying *mental architecture* that transforms stimulus into response; and it is known to have a close relationship with *response accuracy*.

The large majority of existing mathematical models of RT can be classified into two groups—models of RT and mental architecture and models of RT and response accuracy. The first group of models (called RT-architecture models in this paper) focuses on using RT to infer the possible temporal and architectural structures of the underlying mental system that transforms stimulus into response. This paper uses "architecture" to refer to "macro-architecture" of processing stages. "Micro-architecture" neural network models are important but beyond of the scope of this paper. The second group of models (the large majority of which belong to the family of sequential sampling or stochastic information accumulation models) focuses on modeling the relationship between RT and accuracy. Each group has made great progress in modeling the aspects of RT it focuses on. There is, however, a substantial gap between the two groups of RT models. The architectural models have not made great progress in revealing and modeling the intrinsic relationship between RT and accuracy, while the sequential sampling models have been relatively silent about the architecture of the cognitive "black/mystery box" in which the samplings (such as random walks or diffusions) occur.

This paper describes our research that (1) extends the queueing network (QN) architectural model of RT to cover accuracy; (2) establishes a natural link between the QN and the sequential sampling/diffusion models through a modeling approach called Reflected Multidimensional Diffusions (RMD); (3) develops QN and RMD methods to use RT and accuracy together for revealing mental architecture. In short, mental architecture is represented as a QN, whose state changes can be analyzed as a RMD. More specifically, the "state" of a K-server queueing network of mental architecture is represented as a reflected diffusion space of K dimensions, in which "reflecting barriers" represent and reveal architectural constraints, while "absorbing barriers" represent accuracy-related subject-adopted response criteria, similar to diffusion models. This approach moves beyond the current 1-D diffusion models that have successfully accounted for but are limited to single-stage fast responses. 1-D diffusions can only represent the state of information accumulation of a single server, not multi-server architectures.

## Mathematical Models of RT and Mental Architecture

As shown on the left side of Figure 1, RT/architecture models have focused on two issues that are central to RT modeling and theory in cognitive psychology. One is a temporal dimension distinguishing discrete from continuous information transmission models, and the other is an architectural arrangement dimension distinguishing serial stage models from network models. All of the models assume that the psychological activity that transforms stimulus into response is composed of a system of mental processes. Discrete information transmission models assume that a mental process transmits its processing output in an indivisible unit and will not make its output available to other processes until it is completed. Therefore, a process cannot begin until all of its preceding processes are

Mathematical Models of RT and Mental Structure Classified
in terms of Discrete versus Continuous Information Transmission
and Serial versus Network Architecture

(from Liu, 1996, "Queueing network modeling of
elementary mental processes," Psychological Review,
103(1), pp. 116-136).

Mathematical Models of RT
and Response Accuracy
(sequential sampling models)

|  | Architectural arrangement of mental processes |  |  |  |
|---|---|---|---|---|
| Temporal Transmission | Serial Stages | Network Configurations |  | State Transitions |
| Discrete | Subtractive<br>Additive factors<br>General Gamma | Critical Path Network |  | Counter/accumulator<br>Random-walk |
|  |  |  |  | Accumulator |
| Continuous | Cascade |  |  | Diffusion |
|  | Queueing series | Queueing Network (QN) | ↔ | Reflected Multidimensional Diffusions<br>(RMD for state of QN) |

Figure 1: Mathematical Models of RT and Mental Architecture (left side) and Mathematical Models of RT and Response Accuracy (right side)

completed. Continuous information transmission models, in contrast, assume that each process transmits it s partial output to other processes continuously as soon as they are available rather than waiting for the full completion of processing, and thus a process can begin even though its preceding processes are still active. Serial stage models assume a serial arrangement of mental processes, whereas network models assume a network configuration. The two dimensions jointly define four classes of models as shown on the left side of Figure 1 (Liu, 1996). As described in detailed in Liu (1996), a class of queuing network models for RT and mental architecture was proposed which, in its most general form, represents continuous-transmission-network models and they include the existing models in the other three cells as special cases, and thus unify them in a larger modeling framework. Liu (1996) also reexamined the logic and conclusions of the previous models. It turns out that many of the conclusions based on the previous models are open to alternative explanations. All the QN models in Liu (1996) were discussed in relation to empirical data. Furthermore, it was shown that QN models allow us to cover a broader range of possible mental structures that mental system might assume but had not been modeled by previous models, such as feedback or non-unidirectional information flow, information "overtaking and bypassing", and process dependencies or non-selective influence of factor effects, and can be subjected to well-defined empirical tests. The QN approach to RT modeling published in Liu (1996) focuses on the use of RT to infer mental architecture and is able to broaden the scope of thinking about the possible configurations of mental systems and the possible causes for certain RT phenomena. However, some important questions remained open including how the QN models deal with response accuracy and what their relation is to the sequential sampling models described below.

## Mathematical Models of RT and Accuracy

The importance of examining RT and accuracy together in RT analysis and modeling has been emphasized by many researchers (e.g., Audley, 1960; Corbett and Wickelgren, 1977; Dosher, 1979; Meyer et al., 1988; Pachella, 1976; Pew, 1969; Ratcliff, 1978; Wickelgren, 1977). A crucial requirement is that they both arise naturally from common processing mechanisms (Ratcliff, 1978; 1985). The class of mathematical models that have achieved the greatest success in this regard is the class of sequential sampling (also called stochastic information accumulation) models, including random-walk models and related diffusion models, and counter or accumulator models. Sequential sampling models have been applied most extensively to model RT and accuracy data in choice RT experiments (e.g., Audley, 1960; Audley and Pike, 1965; Ashby, 1983; Edwards, 1965; Gronlund and Ratcliff, 1991; Heath, 1981; Laberge, 1962; Laming, 1968; Link and Heath, 1975; Pike, 1966; Ratcliff, 1978, 1981, 1985, 1988; Ratcliff and McKoon, 1982; Ratcliff and Rouder, 1998,2000; Ratcliff, Van Zandt, and McKoon, 1999; Stone, 1960; Van Zandt, Colonius, and Proctor, 2000; and Vickers, 1970). They have also been applied to model simple RT data patterns (e.g., Diederich, 1995; Schwarz, 1994; Smith, 1995), and recently, in modeling decision making (Aschenbrenner, Albert, and Schmalhofer, 1984; Busemeyer and Townsend, 1993; Busemeyer and Diederich, 2002; Diederich, 1995, 1997, 2003a, 2003b; Diederich and Busemeyer, 2003; Roe, Busemeyer, and Townsend, 2001) and classification (Ashby, 2000; Cohen and Nosofsky, 2003; Nosofsky and Palmeri, 1997). All sequential sampling models share the notion that the human information processing system accumulates information over time until a preset response criterion is reached and this accumulation process evolves stochastically.

The random walk models assume that in a two-choice response situation, the information accumulation process "randomly walks" in discrete steps between two decision boundaries (also called "absorbing barriers") based on the value of a *cumulative evidence variable*, each boundary representing one of the two choices. The process generally walks to the positive or the negative boundary depending on whether the value of the evidence variable is positive or negative. The time for the process to reach one of the two boundaries for the first time (immediately terminating the process) is called the *first passage time*, which determines RT. The probability that the process terminates at one or the other boundary is called *first passage probability*, which determines the probability of the associated response. The continuous versions of the random walk models are called diffusion models, which assume that the corresponding stochastic process drifts continuously toward the positive or the negative boundary, depending on whether the mean rate of information accumulation is positive or negative.

The term "counter models" has been used to refer to models that assume discrete counting increments, while "accumulator models" are used broadly to refer to both discrete and continuous evidence accumulation. The idea of using a counter to model RT can be traced back to McGill (1963, 1967), Laberge (1962), and Audley and Pike (1965). Usher and McClelland's (2001) leaky, competing accumulator model represents the state of the art in accumulator modeling.

These sequential sampling models are very successful in modeling RT-accuracy relations for single stage fast binary responses, but relatively silent on multistage architecture issues. The challenge is to bridge the gap between the two groups of models summarized above.

## Queueing Network Modeling of Mental Architecture, RT, and Accuracy: Reflected Multidimensional Diffusions

The QN architecture model of RT presented in Liu (1996) adopts the following assumptions similar to those commonly made in the RT literature: A stimulus is composed of several types of stimulus components (called customers), who arrive at various nodes of the processing network to request for service and that the sequence of customer arrival times, the sequence of customer service times, and the sequence of customer departure times are all stochastic processes. Presently, similar to all major RT models, QN models for RT assume that there is a separate response unit at the end of the processing network (after the "last" or the "exit" node), which is responsible for the actual response. QN models assume that a response is made when the response unit has accumulated N signal components, delivered from the "exit" node. RT is defined and determined by the network sojourn time of the Nth signal customer who completes all its network service requests and departs from the network.

To extend the 1996 QN-RT model to cover response accuracy, the QN-RMD research makes two extensions to the 1996 QN definition of RT: First, we assume that RT is defined and determined by the Nth "response activating" customer (rather than solely by the Nth "signal customer" in the 1996 model, which only elicits a correct response). In a binary RT task (e.g., Yes or No), the Nth response activating customer is the Nth Yes customer for a Yes response or the Nth No customer for a No response. In a RT task involving K alternatives, the Nth response activating customer refers to the Nth i-type customer for a trial with an i-type response. Second, we treat N as a parameter that is analogous to the setting of "counts" in accumulator models and the setting of boundary positions in random walk or diffusion models. A larger N is analogous to a higher preset count or wider boundary. The largest useful N could mean the "limit on the number of useful observations" (Swensson, 1972; p. 30; also in Usher and McClelland, 2001; p. 551-552).

In QN-RMD, the relationship between RT, accuracy, and mental architecture is studied by analyzing the departure process at the network exit node (again, its Nth departure is the Nth accumulation at the "dummy" response node) and examining how this departure process is affected by network architecture and subject-adopted response criterion. We adopt the common assumption that the departure process at each QN node i, $D_i(t)$, is a continuous stochastic process that has independent and stationary increments. Mathematically, this is equivalent to assume, $D_i(t_1) - D_i(t_0)$, …, $D_i(t_n) - D_i(t_{n-1})$ are independent for any $n \geq 1$ and $0 \leq t_0 \leq \ldots \leq t_n \leq \infty$ and the distribution of $D_i(t) - D_i(s)$ depends only on t-s, for all i. This assumption is the most commonly made in the QN literature. Formal mathematical limit theorems that justify the use of these assumptions for analyzing various types of queueing systems and thus the use of Brownian approximations have been proved for various types of flow system models (see e.g., Chen, 1996; Williams, 1998). The K-vector departure process of a K-server queueing network of mental architecture, $\boldsymbol{D}(t) = [D_i(t), i=1,\ldots, K]$, can be represented as a Reflected Diffusion in K dimensions, in which "reflecting barriers" represent and reveal architectural constraints, while "absorbing barriers" represent accuracy-related response criteria. Informally, "reflecting barriers" define the space in which the diffusions can occur. Diffusion occurs "normally," in the interior of the space, but is instantaneously "pushed back" into the space when it hits one of the reflecting barriers. The reason "reflecting barriers" exist for QN is because departure processes cannot take on any arbitrary values. For example, in a 2-node serial QN system, if K1 is in front of K2, we must have $D_1(t) \geq D_2(t)$ (all departures at K2 must have departed from K1 first). Thus, the diffusion of the 2-vectored process $[D_1(t), D_2(t)]$ can only occur in the region in which $D_1(t) \geq D_2(t)$ (i.e., a reflecting barrier exists at $D_1(t) = D_2(t)$, with a reflecting direction pointing toward the allowed region). Reflecting barriers can be defined similarly if K2 is in front of K1, or for other situations. Thus reflecting barriers reveal the architectural arrangement of the mental system. "Absorbing barriers" are defined in the same sense as current diffusion models of RT, such as those of Ratcliff. Due to space limit, this paper chooses two representative cases as illustrations.

## 1. Single–server QN for binary responses

The simplest case of a QN is a single server system. The binary response case called binary, single-step responses, has been modeled most extensively and successfully by existing diffusion RT models. "The diffusion model was designed to explain fast, *single step*, as opposed to multistep, decision processes, …" (Ratcliff, et al., 1999; p. 262.). This case serves two purposes: important by itself to show a concrete link between QN and RW/diffusion models, and as the base or starting point for modeling more complex architectures. In this QN, binary response (Yes/No) are triggered by two types of customers, A and B, who arrive at the server with arrival rates of $\lambda_a$ and $\lambda_b$, respectively, in accordance with a renewal process having an arbitrary interarrival distribution and are served by the server with service rates of $\mu_a$ and $\mu_b$, respectively.

This method assumes that each type-A customer departing from the server carries an information amount of +1, while each type-B departing customer carries -1. This assumption is similar to, e.g., the classical RT diffusion models of Ratcliff et al., (1999) and the "two-barrier single channel model" of Smith (2000). In the following I show how this single server QN is mathematically identical to the classical RT diffusion models, but it offers a QN interpretation to it. Let $S_n$ denote the total amount of information carried by the first N departing customers. We have, $S_n = \sum_{i=1}^{N} X_i$ , i ≥ 1;

where $X_i$=1 for a departing customer of type A and Xi = -1 for a departing customer of type B.

Clearly this departure process $S_n$ is a random walk (RW), and its relation to existing RW models of binary RT becomes at least intuitively apparent. If we speed up the departure process by considering smaller and smaller time intervals and letting Δt go to 0, then the total amount of information carried by all departed customers by time t, D(t), follows a diffusion process. In fact, this comes naturally also as a consequence of our general assumption of independent stationary increments mentioned earlier. In the queueing literature it is commonly regarded as a harmless assumption to treat fast discrete customer departures as a continuous flow (see, e.g., Harrison, 1985).Mathematically, D(t) can be characterized with the Kolmogorov backward equation, as follows:

$$\frac{\partial}{\partial t}p(t,x,y) = (\frac{1}{2}\sigma^2\frac{\partial^2}{\partial x^2}+\mu\frac{\partial}{\partial x})p(t,x,y)$$

where p(*t*, *x*, *y*) is the transition density, *x* is the starting state, *y* is the ending state, of time period *t*. This equation is called the Kolmogorov backward equation because the differentiation is with respect to the backward variable (the initial state) *x* on the right side of the equation above. This equation is identical to Ratcliff et al's (1999, p.299) diffusion equation, with difference only in the notations.

When μ≠0 (diffusion with a drift), then as shown in Harrison (1985) in his analysis of diffusion approximation of stochastic flows in queueing systems, we have, $P_x\{X_t=b\}$, the probability that the process first crosses the barrier b before crossing the barrier at 0, when the starting position is at *x*, as $P_x\{X_t = b\} = \frac{1-\xi(x)}{1-\xi(b)}$, $0 \leq x \leq b$,

$$\text{where } \xi(z) \equiv \exp(\frac{-2\mu z}{\sigma^2})$$

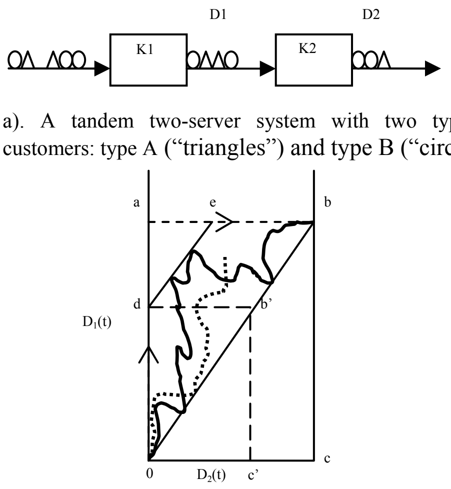
This result is the same as that in Ratcliff et al (1999, p.299), who presented $P_x\{X_t=0\}$, the probability that the process first crosses the absorbing barrier 0 before crossing the barrier at b: $Px\{Xt=0\} = 1 - Px\{Xt=b\} = 1- \frac{1-\xi(x)}{1-\xi(b)} = \frac{\xi(b)-\xi(x)}{\xi(b)-1}$

This is identical to Ratcliff et al (1999, p299), with the difference found only in the symbol notations.

The convergence of the results of the queueing literature and the RT-diffusion models shown above offers a queueing architectural explanation to the RT diffusion modeling. Since our diffusion representation of the departure process of the single-server queueing system has converged precisely with the diffusion model of Ratcliff, all the related results of Ratcliff apply. One intuitive interpretation of the diffusion parameters with the QN parameters is: The mean drift rate, μ, is determined by the difference between the two mean arrival rates (λa-λb). This is intuitive, since an RT trial with stimulus A would carry more customers (features) of A, thus produce a greater arrival rate of type A customers than a B-type RT trial. This is similar in spirit to, e.g., Ratcliff's (1978) work of using stimulus relatedness to decide drift rate. The boundary positions can be interpreted as the minimum amount of total positive or negative information carried by all the departed customer to elicit an A or B response, respectively (as in Ratcliff, we will also use b and 0 as the two boundaries). The starting point is the subject's response bias, which can be assumed as "preloaded departures" or "information preloaded in the system"—thus called a "bias." A discussion of the relationship between this single server 1-D diffusion model and the corresponding accumulator model can be found in Liu (2005). Due to space limit, we elect to discuss 2-D cases next to illustrate how to consider architecture issues in this QN-RMD framework.

## 2. A tandem 2-Server QN for binary responses

A basic, fundamental, and illustrative case involving all three issues: RT, accuracy, and architecture is a tandem 2-server QN as shown in Figure 2, which goes beyond single-stage RT-accuracy modeling and demonstrates the importance of considering "reflecting" barriers, in addition to "absorbing" barriers of the conventional 1-D diffusion models.



a). A tandem two-server system with two types of customers: type A ("triangles") and type B ("circles")

b). For customers' departure processes $D_1(t)$ and $D_2(t)$, if K1 and K2 form a discrete processing series, then we have two 1-d diffusions in a row, first along the border from 0 to a, then along the border from a to b. If K1 and K2 form a continuous-flow series, then we have one 2-d reflected diffusion. See text for more details.

**Figure 2 A tandem 2-Server QN and its Reflected Diffusion Space**

In this tandem 2-server QN shown in Figure 2, we consider a pair of 2-vectored departure processes, $\{D_{1A}(t), D_{2A}(t)\}$ and $\{D_{1B}(t), D_{2B}(t)\}$, corresponding to the type-A and type-B departures from K1 and K2, respectively. Similar to the diffusion cases for a single server, we assume each customer departing at K2 carries the same amount of information to contribute to its type of response only. A response is made when $D_{2A}(t)$ or $D_{2B}(t)$ first reaches its criterion value. Let us consider two cases, corresponding to the debates between discrete and continuous information transmission between stages.

**Case A. A series of 2 discrete processing stages corresponds to a sequence of 2 1-d diffusions**

The mental architecture theories of discrete processing series (e.g., Donders' and Sternberg's theories, upper-left cell of Figure 1) assume that K1 must complete all its work before K2 can start. In other words, all departures from K1 must complete before K2 starts its departure process, i.e., $D_1(t)$ must complete before $D_2(t)$ starts. In Figure 2, this can be visualized as a sequence of two 1-d diffusions for each type of customers, first along the line from 0 to a for $D_1(t)$, and then along the line from a to b for $D_2(t)$. For the simplest case of "no bias" RT tasks, both types of customers go through the same diffusion sequence (first 0-to-a, and then a-to-b).

**Case B. A continuous-flow 2-server queue-series corresponds to a reflected 2-d diffusion.**

The mental architecture assumption of continuous flow (e.g., McClelland's Cascade, Miller's and Liu's queue series, lower-left cell of Figure 1) does not require K1 to complete its processing before K2 can start. In other words, $D_1(t)$ and $D_2(t)$ may occur concurrently, subject to certain constraints in a queueing system. Specifically, the joint distribution of $D_1(t)$ and $D_2(t)$ can be characterized as a reflected 2-d diffusion: If K1 is in front of K2, then diffusion occurs in the upper region above the reflecting barrier shown as the diagonal line (0—b) in panel b of Figure 2. If K2 has finite waiting space, s, (in front of K2), then diffusion is further bounded by a reflecting barrier shown as line (d—e), whose vertical distance to line 0—b is s. A reflecting barrier (line a—b) exists if $D_1$ has an upper limit. Absorbing barrier is shown as line b—c. For simplicity of presentation and analysis, we continue to focus on the no-bias RT situation for now, meaning that the two types of customers "race in the same diffusion space," shown as a pair of trajectories (a solid and a dotted curve, where the solid one wins in this illustration) in panel b of Figure 2. Several testable performance predictions can be made with regard to RT-accuracy relation in this situation, including:

1). When K1's service rate is much larger than K2 (e.g., K1 is a super fast perceptual server) and K2 has unlimited waiting space (i.e., $s=\infty$), then the probability is almost 1 that $D_1$ is much greater than $D_2$, thus the probability of hitting the reflecting barrier at (0—b) is almost 0. In this situation, in terms of its effect on RT/accuracy, the 2-d diffusion would behave as if it is a 1-d diffusion of $D_2(t)$ along the line of (0—c), similar to the classical diffusion RT models and to the single server case discussed earlier. Informally, if K1 is so powerful, we don't have to worry about it; we just need to consider K2.

2). When K1 is not a super fast server or when an experimental factor increases the sojourn time at K1 (thus decreases the departure rate at K1), RT/accuracy will not show the same type of exponential relationship predicted by classical RT-diffusion models or single-server QN models, since now we have a true 2-d reflected diffusion, bound by the reflecting barriers, whose effect can not be ignored. Quantitatively, it would take a longer time to achieve the same level of accuracy obtained in the 1-d diffusion case.

3). Further, when there is a very low limit in the departure process of K1 (e.g., in the so-called data-limited tasks in which impoverished or significantly degraded stimuli are used), RT/accuracy will not show the same type of exponential relationship predicted by classical RT-diffusion models or single-server QN models. This can be visualized in panel b of Figure 2: when a is smaller than b (slide the line a-b downward to, say, d-b'), the 2-d diffusion will never be able to reach the absorbing barrier b-c (i.e., subject never responds) UNLESS the subject reduces the absorbing barrier, by moving it to the left to b'c', by willing to make less accurate responses. This offers an alternative explanation to the classical "infinite-RT" problem (Ashby, 1982).

4). The order of server arrangement is an architectural research question itself. The analysis above assumes K1 is in front of K2. When K2 is in fact in front of K1, diffusions would occur in the lower region. Thus, a method of using RT/accuracy together to reveal the order of K1 and K2 is to see whether an upper- or a lower-region diffusion best fits the data.

## 3. Other network cases

The single-server and the 2-server QNs and their diffusion representations described above are the simplest network cases. Concrete testable predictions can also be made about more complex network arrangements. Two examples are listed below:

1). A series of K discrete processing stages correspond to a series of K 1-d diffusions. This is an extension of Case A of the 2-server QN to a general series of discrete stages.

2). A series of K continuous flow servers correspond to a reflected K-d diffusion with a "lower-triangular reflection matrix." Characteristics of this reflection matrix reveal the layout of the series. This is an extension of Case B of the 2-server QN to a general queueing series.

Additional network cases and related discussions can be found in Liu (2005).

In summary, the QN-RMD (Queuing Network-Reflected Multidimensional Diffusions) represents mental architecture as a QN, whose state of operation can be represented as a multidimensional diffusion space. QN-RMD extends the QN architectural RT models to account for accuracy, brings the Random Walk/diffusion models of RT and accuracy to the multi-server architectural domain, and unifies the two currently separated schools of approaches in a larger framework. The work helps reduce the "fragmentary nature of the results" (Luce, 1986, p. 491), by "synthesizing what we know" (Newell, 1990; p. 16).

## Sample References

Audley, R. J., and Pike, A. R. (1965). Some stochastic models of choice. *British Journal of Mathematical and Statistical Psychology*, 18, 207-225.

Busemeyer, J. R., and Townsend, J. T. (1993). Decision field theory: A dynamic-cognitive approach to decisions making in an uncertain environment. *Psychological Review*, 100(3), 432-459.

Cohen, A. L., and Nosofsky, R. M. (2003). An extension of the exemplar-based random-walk model to separable-dimension stimuli. *Journal of Mathematical Psychology*, 47, 150-165.

Dai, J.G., and Harrison, J. M. (1992). Reflected Brownian motion in an orthant: Numerical methods for steady-state analysis, *Annals of Applied Probability*, 2(1), 65-86.

Harrison, J. M. (1985). *Brownian motion and stochastic flow systems*. New York: Wiley.

Liu, Y. (1996). Queueing network modeling of elementary mental processes. *Psychological Review*, 103(1), 116-136.

Liu, Y. (2005). *Queuing network modeling of mental architecture, response time, and response accuracy: Reflected multidimensional diffusions.* Tech Report 05-11 of the Dept of Industrial and Operations Engineering, University of Michigan.

Luce, R. (1986). *Response Times: Their role in inferring elementary mental organization*. New York: Oxford University Press.

Newell, A. (1990). *Unified theories of cognition*. Harvard University Press.

Ratcliff, R. (1985). Theoretical interpretations of speed and accuracy of positive and negative responses. *Psychological Review*, 92, 212-225.

Ratcliff, R., and McKoon, G. (1997). A counter model for implicit priming in perceptual word identification. *Psychological Review*, 104, 319-343.

Ratcliff, R., Van Zandt, T., and McKoon, G. (1999). Connectionist and diffusion models of reaction time. *Psychological Review*, 106(2), 261-300.

Smith, P. L. (2000). Stochastic dynamic models of response time and accuracy: A foundational primer. *Journal of Mathematical Psychology,* 44, 408-463.

Townsend, J. T., & Ashby, F. (1983). *The Stochastic Modeling of Elementary Psychological Processes.* Cambridge: Cambridge University Press.

Usher, M., and McClelland, J. L. (2001). The time course of perceptual choice: The leaky, competing accumulator model. *Psychological Review,* 108(3), 550-592.

# Are Simpler Strategies less Error-prone in Inference from Memory?

**Rui Mata (ruimata@umich.edu)**
University of Michigan, Dept. of Psychology, 530 Church Street
Ann Arbor, MI 48109 USA

## Inferences from Givens vs. Memory

The adaptive toolbox approach to decision-making holds that people possess a repertoire of strategies and that they adapt to the characteristics of the task environment by selecting the appropriate one from that repertoire (Gigerenzer, Todd, & the ABC Research Group, 1999). For example, Gigerenzer et al. have suggested that people rely on simpler strategies when cognitive costs of information search are high. Bröder and Schiffer (2003) tested this hypothesis by observing people's strategy selection in a condition in which people could search for information on a computerized display, information from givens, or alternatively had to retrieve information from memory – inference from memory. Bröder and Schiffer found more participants relied on the simpler, noncompensatory strategy Take the Best (TTB; Gigerenzer et al., 1999) compared to other more information-intensive strategies in the inference from memory compared to inference from givens condition. Bröder and Schiffer argued that given "the assumption that retrieving pieces of information sequentially from memory causes cognitive costs in terms of time, effort, and *error proneness*, the tendency to use noncompensatory heuristics like TTB seems fairly adaptive" (emphasis added; p. 289). The present work aims to test the assumption that the simple TTB is less error-prone compared to more information-intensive strategies when used in inference from memory.

## Simulation

I used the elementary information process (EIP) framework (Payne, Bettman, & Johnson, 1993) to model two simple inference strategies, TTB and Tally (Gigerenzer et al., 1999). The EIP framework is a production-system theory which allows constructing different strategies using the same basic EIP building blocks, such as READ (read a value of an option into working-memory), COMPARE (compare the values of two options in working-memory), ADD (add a value to a tally concerning an option), and DECIDE (choose an option). The rationale for using this framework is it allows changing the efficiency of basic decision components independent of differences between strategies. The two strategies combine EIPs differently. For each decision, TTB searches for information on the options concerning the most valid cue (READ), compares the options on that cue (COMPARE), and chooses the option for which the cue speaks (DECISION). If the cue does not discriminate the process is repeated with the second most valid cue, and so on until a decision is made. Tally looks up information on each cue (READ) and computes a tally for each option (ADD), it then compares the two tallies (COMPARE) and makes a decision (DECISION).

I modelled the effect of making inferences from memory by increasing the error in EIPs shared by TTB and Tally: READ and COMPARE. The rationale underlying this manipulation is that retrieving information from memory may lead to failed retrievals (READ) or errors in comparing cues in working-memory (COMPARE). I simulated 1000 decisions of TTB and Tally between all paired-comparisons of 16 options possessing 4 binary cues. The average results can be observed in Figure 1. The manipulation of errors in READ suggests Tally is a more robust strategy particularly for large proportions of error. However, the opposite prediction resulted from the manipulation of COMPARE. An empirical study asking participants to execute TTB and Tally in givens and memory conditions is currently under way to test these predictions.

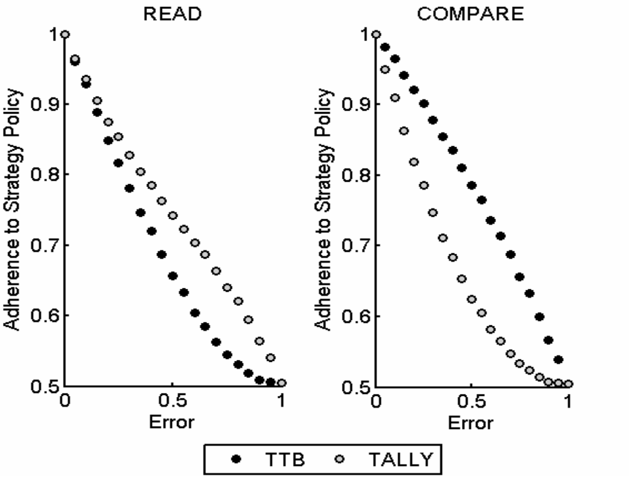


Figure 1: Proportion of errors of TTB and Tally as a function of errors in EIP (READ, COMPARE).

## References

Bröder, A. & Schiffer, S. (2003). "Take The Best" versus simultaneous feature matching: Probabilistic inferences from memory and effects of representation format. *Journal of Experimental Psychology: General, 132*, 277-293.

Gigerenzer, G., Todd, P. M., & the ABC Research Group. (1999). *Simple heuristics that make us smart*. New York: Oxford University Press.

Payne, J. W., Bettman, J. R., & Johnson, E. J. (1993). *The adaptive decision maker*. Cambrige, UK: Cambridge University Press.

# The RecMap Model of Active Recognition Based on Analogical Mapping

**Georgi Petkov (gpetkov@cogs.nbu.bg)**
Central and East European Center for Cognitive Science,
New Bulgarian University
21 Montevideo Str., Sofia 1618, Bulgaria

**Luiza Shahbazyan (ltsavak@abv.bg)**
Central and East European Center for Cognitive Science,
New Bulgarian University
21 Montevideo Str., Sofia 1618, Bulgaria

## Abstract

We propose an object recognition model based on analogical mapping and transfer. The objective of our model is to be able to generate and bind structural representations; and to recognize objects from a small set of primitives. The input is mapped to the associative memory and activation is spread upwards. Anticipations are generated through local mappings and transferred to be tested serially in the order of their relevance. Due to these mechanisms, our model is able to simulate phenomena such as object priming and global precedence effect. Additionally, it provides a framework for integrating visual perception and other higher-order cognitive processes.

**Keywords:** Recognition, Analogy-making, Anticipation, Attention, Binding.

## Theoretical Framework

It is a textbook assumption that object recognition is a heuristic process. Rather than passive recipients of sensory data, we actively form hypotheses and make anticipations. Fortunately, we are provided with memory and rich environmental context, which constrain the number of plausible anticipations, thus enabling the fast and reliable recognition of virtually infinite number of objects, people, and events. Committed to these beliefs, we started developing a model of object recognition based on the cognitive architecture DUAL (Kokinov, 1994a). There are several premises fundamental to this model: (i) recognition is a heuristic process constructing structural representations, (ii) anticipations are generated by analogy, (iii) context supports (and sometimes hinders) recognition. These assumptions, as well as their empirical support and computational implementations, will be discussed in details in the following sections.

Object recognition is one of the most rigorously debated topics in the field of vision sciences. Although the issue has been approached from numerous paradigms and theoretical perspectives, the task of understanding and modeling vision is still far from being accomplished (see Peissig & Tarr, 2007 for a recent review). A central question is what are that type of representations on which recognition operates. There are two basic approaches: view-based and object-based theories and respectively, models. The basic idea behind the view-based approach is that there is a stored template for each of the previously seen visual patterns. Recognition involves matching the stimulus input to an existing template. Most of these models implement some kind of normalization process in order to reduce the number of the necessary templates (Poggio & Edelman, 1990).There are plenty of computational models committed to this paradigm and they are capable of simulating a vast range of psychological phenomena. However, they show weak performance in some domains such as class-level recognition, matching known to unknown viewing conditions as well as generalization (Tarr & Bulthoff, 1998).

In contrast, object-centered theories assume that recognition involves matching of view-point independent descriptions of spatial arrangements among parts of the object (Marr & Nishihara, 1978; Biederman, 1987). A classic example of this type is the recognition-by-components theory (RBC; Biederman, 1987). Its core premise is that recognition consists of extracting invariant structural representation of the object in terms of spatial relationships among basic shapes or components, the so-called geons, which are then matched to stored object representations. Furthermore, Biederman and Gerhardstein (1993; 1995) argue that all perceptual objects are decomposable and each object has a unique configuration of parts. Structural computational models were developed by Hummel and Biederman (1992) and Hummel and Stankiewicz (1996).

One question that is still open is how spatial inter-part relations originate. Because such relations are present at linguistic level, they can be readily available in the memory storage. So, the possibility remains that these relations are actively participating in object recognition through top-down mediation of the binding process. Nevertheless, it is not clear how this process is realized.

One plausible mechanism is analogy making, in particular mapping, which has already been granted the status of core principle for many cognitive processes (Gentner, Holyoak, & Kokinov, 2001). Several analogy-making computational models exist: COPYCAT (Hofstadter, 1984), ACME (Holyoak & Thagard, 1989), SME (Falkenhainer, Forbus, & Gentner, 1990), TABLETOP (French & Hofstadter, 1991), LISA (Hummel & Holyoak, 1997, 2003), AMBR (Kokinov & Petrov, 2001; Kokinov, 1994b) among others. Besides simulating analogy-making, these models demonstrate excellent performance in modeling a number of other cognitive processes such as memory retrieval (Forbus, Gentner, & Law, 1994), judgment (Petkov, 2006), and infant categorization (Kuehne, Gertner, & Forbus, 2000). Although the incorporation of perception and high-order cognition is not a new idea (Chalmers, French, & Hofstadter, 1992), none of the models, which pursues this endeavor, explores the possibility that the mapping of objects in two domains, based on the relational structure, which serves other cognitive processes, can readily account for the top-down influence of these relational structures on the binding of sensory information.

More specifically, we propose that the visual information is mapped to the information stored in memory. By analogy, anticipations are generated about the spatial relations between the elements at the input and how they can be bind to each other. Later, these anticipations are transferred and verified with the actual data. The implementation of this mechanism has one additional implication: the generated anticipations refer to all levels of the semantic hierarchy, so that binding and recognition of a higher level can precede the recognition of the lower level if the level of activation of these anticipations is higher. The global precedence phenomenon has been extensively documented (Navon, 1977; Kimchi, 1992), but it poses difficulty for the structural theories, which argue for part-base recognition (Biederman, 1987). Although our model is intrinsically structural, it overcomes this limitation due the specificity of its architecture. When the activation from the target spreads, it propagates upwards to the superordinate levels of the semantic network allowing anticipation formation at any level. As a result, the level of recognition in not fixed to a particular location within the hierarchy either.

Another important factor in object recognition is the available contextual information. Recognition performance is more accurate when the object is primed with consistent scene and dropped when the prime is inconsistent (Palmer, 1975). In addition, object detection is more accurate and naming is facilitated when the object appears in a consistent setting (Biederman, Mezzanotte, & Rabinowitz, 1982; Boyce & Pollatsek, 1992). Our model is context sensitive because the activation level of a particular bit of information basically represents its relevance to the current context. More active elements are anticipated more rigorously and verified with priority, thus speeding up their recognition

Major advantage of the proposed models is its potential for integration of visual perception with other cognitive processes such as reasoning on the basis of common mechanism, that is, mapping. Similar principle has been incorporated in COPYCAT (Hofstadter, 1994), although it focuses on higher-order perception of events and analogical reasoning. Our task is to determine the potential of mapping ability to support the cognitive system from the very beginning of visual processing, through fast, implicit recognition, to relatively slower, explicit analogy making.

# RecMap Model of Active Recognition Based on Analogical Mapping

In RecMap model, the recognition involves (i) mapping of limited input information onto structurally organized memory traces; (ii) creation of anticipations on the basis of these mappings; (iii) sequential checking of the anticipations. The model is based on the cognitive architecture DUAL (Kokinov, 1994a), and builds up on the AMBR model (Kokinov & Petrov, 2001, Kokinov, 1994b) The RecMap model uses all mechanisms of AMBR and proposes new mechanisms for anticipation-forming, binding, and recognition, which are integrated with the old ones, thus allowing the mapping process to guide recognition as well.

## The AMBR Model

AMBR model consists of a huge number of interacting with each other hybrid micro-agents with symbolic and connectionist part. The permanent agents (concepts and some of the instances) constitute the system's long-term memory, a semantic network with merged representation of the declarative and episodic knowledge. Each agent represents bits of information, but even small pieces of knowledge are represented by a coalition of many agents. At the same time, each agent has an activation level depending on its relevance to the ongoing context, and only active agents participate in symbolic operations.

The AMBR agents that represent the environment (source node) and the task (goal node) serve as a source of activation, which spreads with decay. Each active *instance-agent* emits a marker. This marker is sent to its parent *concept-agent* (representing type) and then upwards in the class hierarchy. When two markers meet, a *hypothesis-agent* for correspondence between the two marker-origins is created. The structural correspondence mechanism in turn creates new hypotheses on the basis of old ones. For example, if two relations are analogical, their respective arguments should also be analogical, etc. Thus, dynamically, a constraint satisfaction network of interconnected, competing with each other hypotheses emerges. Once a hypothesis maintains leading activity long enough and reaches a critical value, it is promoted to a winner, representing the analogy performed by the model.

In AMBR, the process of analogy-making is not separated into sub-processes - retrieval and mapping overlap and interact with each other. As a result, the structural constraints, crucial for the mapping process, influence the retrieval as well.

## Innovations

In comparison to AMBR, RecMap is equipped with several new mechanisms. These are the creation and maintenance of hypotheses for recognition, anticipatory mechanism, and attention.

After a correspondence hypothesis emerges, a structural correspondence mechanism creates a *hypothesis for* *recognition*. For example, suppose that a certain line from the environment happens to be mapped to a particular line from memory. If the second line is a part of a square, then a recognition-hypothesis that the first line is also a part of a square emerges.

Simultaneously, the *anticipatory mechanism* (Petkov, Naydenov, Grinberg, & Kokinov, 2006) is operating. The memorized instance-agents inform the relevant relations in which they participate for all their hypotheses. If a certain relation collects the hypotheses for all its arguments, it creates an *anticipation-agent*, representing the expectation that the same relation is present in the environment. The anticipation-agents are copies of their mentor-relations but all their arguments are replaced with the respective analogical elements from the target situation.

The *attention* mechanism monitors all anticipation-agents, sorts them by their activation (i.e. relevance), and at fixed time intervals asks a simulated perceptual system to check the relation represented by the most active one. In the current version of the model the perceptual system is just a pre-defined list of the relations, which are currently present. Another role of attention is to *bind* together the hypotheses for recognition for the respective relation and their arguments. For example, because all operations are performed locally, the relational arguments may have hypothesis that are parts of a wall, but not necessarily one and the same wall. So, the binding mechanism is responsible to bind all hypothesized walls into one.

Thus, various hypotheses for recognition emerge locally, support or suppress each other, and are merged by the attentional binding mechanism. The recognition hypotheses in turn emit markers upwards in the conceptual system, participate in new hypotheses and anticipations, and create even more abstract recognition hypotheses. As a result of the relaxation of the network of hypotheses, the most active of them are promoted to winners.

## Experimental Simulations

The domain for these simulations is hierarchical objects consisting of vertical or slopped lines and ovals, organized by spatial relations in figures (see Figure 1).

There are several concepts in the *long-term memory*, named 'house', 'lorry', 'tree'[1], etc., as well as their parts, and the parts of these parts and so on. For example, a particular instance-agent for a house is linked to its parts – 'roof', 'wall', and to the relation 'above' between them. In turn, the agent 'wall' is represented with a head-agent, linked to four lines, two of them horizontal, two vertical, as well as to several relations between these lines. The 'roof' consists of one horizontal, one left-right slopped, and one right-left slopped line, etc. When a target object is given to the model, it is represented only by these lines (called primitives), without any relations between them.

The model's task is to organize the square and the triangle[2] through the anticipatory and attentional mechanisms, and to create hypotheses that the square and triangle are respectively wall and roof (in competition with many other hypotheses), to anticipate possible relations between the square and the triangle, to organize them in a single object, and finally to recognize the house. Note that these processes overlap and the given order is very rough.

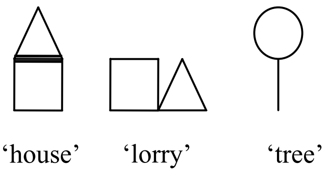


Figure 1: Example of the objects used in the simulations.

There are several instances for each concept from the *long-term memory*, and each concept is randomly linked to some of these instances, thus ensuring that the activation would propagate from the semantic (conceptual) to episodic (concrete instances) memory.

In a series of five simulations the main properties of the model are demonstrated. In the first simulation, a single object is recognized, thus the integrated work of all mechanisms is tested. In the second simulation, an object that shares all but one relation with the first one is recognized. In the third simulation, various priming effects are simulated. In the fourth and fifth simulations, the ability of the model to deal with more complex scenes and situations is tested.

### Simulation 1: Recognition of a House

The task of the model in the first simulation was to recognize a single object – a 'house'. There are only seven primitives on the input, representing the straight contours – three horizontal lines, two vertical, one left-to-right slopped, and one right-to-left slopped.

The activation spreads from these primitives to the parent concepts of these lines and then back to some of their instances. Many hypotheses for correspondence emerge. For example, each of the horizontal lines creates its own hypotheses with various horizontal lines, which participate in various objects. In turn, these correspondences create hypotheses for recognition. For example, if 'line-b1' is a part of a wall, and the target 'line-1' is analogical to 'line-b1', then the respective correspondence creates and supports a hypothesis for recognition that 'line-1' is also a part of a wall.

The more instances of a particular concept are relevant, the more support the hypotheses for recognition about the

[1] All names, used in the simulations are arbitrary. The model would work as well if the agents were named ag01, ag02… The choice of the set of primitives is also arbitrary, without any psychological validation.

[2] Note, there are not any squares or triangles in the memory. The objects are organized as hypotheses for recognition as 'wall', 'cabin' (of a lorry), etc. The terms square and triangle are used only for better description.

respective concept would receive. Thus the pressure for top-down priming influence is presented. Note, however, that because of the pressure for one-to-one mapping, the hypotheses for correspondence between the target line and the stored lines inhibit each other. Thus, the contextual top-down influence is limited.

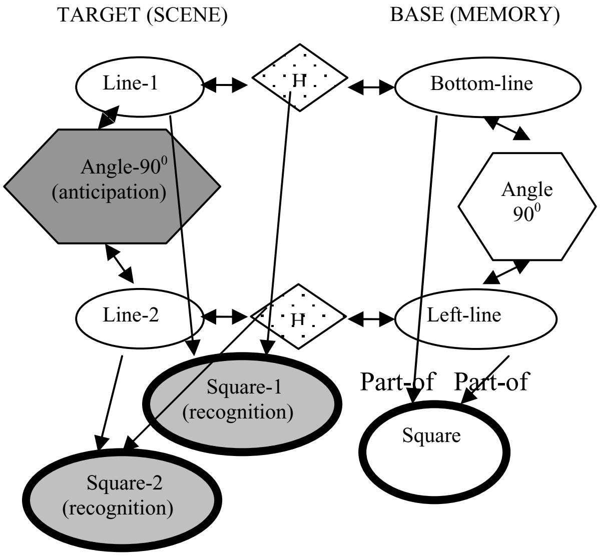


Figure 2: The hypothesis for correspondence H2 creates the recognition-hypothesis ‘square-2’. Independently, H3 creates ‘square-3’. In turn, the anticipation ‘angle-90’ is created because each of the arguments of the memorized relation ‘angle-90’ has hypotheses for correspondence.

At the same time, numerous anticipations about possible relations between the target lines are created (see Figure 2 for an example).

In turn, the new agents (hypotheses and anticipations) influence the spread of activation making some elements more relevant than others. The attention mechanism checks sequentially the anticipations. If certain anticipation is rejected, it just ‘died’. If it is confirmed, the respective anticipation turns into an instance-agent. Thus, the description of the scene is enriched a bit. At the same time, the hypotheses for recognition of the relational arguments are bound with each other (see Figure 3). Thus, new relations are involved in the competition between the recognition-hypotheses.

**Results** As a result of the simulation, at time 59.24 (hundreds cycles of the program) a recognition-hypothesis ‘house’ becomes a winner; recognition-hypotheses for ‘roof’ and ‘wall’ become winners respectively at times 99.84 and 104.96. Interestingly, the whole object was recognized before its parts consistent with the global precedence effect demonstrated by Navon (1977). Actually, before the whole object is recognized, there is no reason to recognize the square as a wall or as a cabin for example. The resolve of the puzzle starts after confirming the anticipation ‘above’ with the two parts as arguments. Thus, the already created instance-agent ‘above’ add the decisive support for the ‘house’ and later on, the parts of the house are recognized as well.

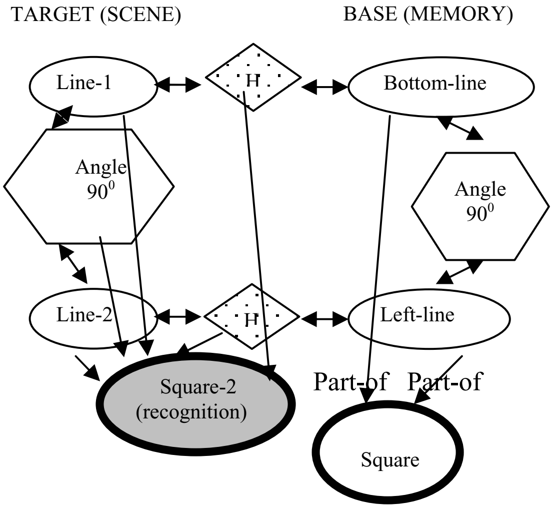


Figure 3: If the anticipation ‘angle-90’ is confirmed, it is transformed into an instance-agent. At the same time, the recognition-hypotheses of its arguments are bound to each other (compare to Figure 2).

### Simulation 2: Recognition of a Lorry

Everything, including the seven input lines, was the same, as in the first simulation except that in the list of predefined relations the relation ‘above’ is replaced with ‘in-touch’, thus the correct response of the model was ‘lorry’.

**Results** At time 60.20 a ‘lorry’ was recognized; a ‘cabin’ and a ‘trailer’ were recognized respectively at time 107.66 and 110.32.

### Simulation 3: Priming

The role of the third simulation was to simulate the impact of context priming on the recognition process.. Actually, simulation 3 consists of three separate runs of the program. In all three runs, the model’s task is to recognize a single lorry (the simulation 2 is repeated). However, at the first run, additional instance of a ‘road’, associatively linked to ‘lorry’ is attached to the input, thus supplying the concept ‘lorry’ with extra activation.

The prediction was that the higher activation of the instances of ‘lorry’ would facilitate the recognition process. In the second run, the concept ‘fence’, associated with ‘house’ is activated, expecting to hinder the recognition. Finally, it the third run, again ‘fence’ is pre-activated but with extremely high stimulation, thus simulating abnormal fixation.

**Results** At the first run, RecMap recognized ‘lorry’ at time 58.92; ‘cabin’ at time 86.58; ‘trailer’ at time 94.08, thus fully confirmed our expectations (compare with the respective results without priming from the Simulation 2 –

60.20, 107.6,6 and 110.32). At the second run the respective times were 62.22, 11.508, and 130.72. At the third run the model made wrong recognition – at time 103.12 a 'roof' was recognized, at time 139.28 – a 'house, at time 178.34 – a 'cabin'.

These results in agreement with the effects of consistent and inconsistent contextual priming demonstrated in the psychological literature (Palmer, 1975, Biederman, et al., 1982).

### Simulation 4: Recognition of Two Objects

In the fourth simulation 14, instead of 7 lines were attached to the input of the model. A situation with two different objects was simulated (see Figure 4, left panel). There were not any relations between parts or primitives of different objects (i.e. anticipations for such relations were created but later rejected).

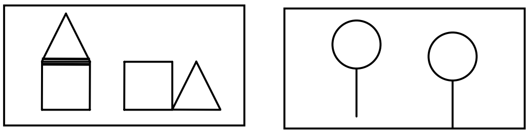


Figure 4: Left panel: Stimuli, presented to the model in simulation 4. Right panel: The base situation, used in simulation 5.

**Results** The overall recognition time was slowed down but not considerably. The times for recognition of a 'lorry', 'cabin', 'trailer', 'house', roof', 'wall' were 58.98, 69.38, 93.04, 97.56, 122.38, 136.14, respectively. This is evidence that the model can operate on more complex scenes with a little increase of computational resources. Thus, the model's ability to scale up is demonstrated.

### Simulation 5: Integration of Recognition and Analogy-making

Finally, a whole base episode was added to the long-term memory and the capability of the RecMap model to perform the whole cycle from perception to complex analogy was tested. The base situation consists of two trees with a relation 'left-of' between them (see Figure 4).

**Results** The model successfully recognized the house and the lorry (just as in the Simulation 4) and continued with the analogy-making process.

Because all concepts for 'house', 'lorry', and 'tree' are sub-classes of the superordinate concept 'neighborhood', the respective markers from the target and base objects cross, and new hypotheses for correspondence between the target objects and the trees are created. In turn, the RecMap mechanisms created anticipations that the lorry is in left of the house, and vice versa. The former anticipation is rejected, the latter one is confirmed, and thus the right spatial analogy was settled.

## Conclusions

The RecMap model for recognition, based on the DUAL architecture and the AMBR model for analogy-making is presented. The main assumptions of the model are that the analogy-making is very basic human ability, and that the recognition is an active process of dynamical creation and verification of various hypotheses and anticipations. The model is based on an associative organization of the memory; on high context sensitivity; on basic mechanisms for analogy-making and hypotheses creation; on anticipatory behavior; and on attentional mechanism for sequential testing of anticipations.

Our model was successful at stimulating several effects that are considered characteristic for human object recognition. To begin with, the model manages to anticipate, verify, and construct hierarchical structural representations of objects by analogy, which reveals the potential of this mechanism to support low as well as high-level recognition. Even more, it is able to recognize as different two objects that share all but one relation.

However, structural does not always mean part-based as we demonstrated. The recognition may start from the whole and then proceed to the parts of the objects. Even more interesting is the chronology of events when the model is presented with two objects. When the recognition began with a particular object, it continued with its parts only on the basis of the competition between the active anticipations.

Furthermore, we showed that the influence of context in the priming simulations can be modeled in an ecological manner. The model is not only able to simulate facilitative effects when the priming is consistent, but also slows down and is prone to mistakes when the context is leading.

Finally, we demonstrated that our model is able not only to bind objects and recognize them, but also to perform analogical reasoning. The novel in our approach is that all processes are guided by one and the same underlying principle - the ability of analogical mapping.

Nevertheless, there are several limitations of the current model. Although we assumed some kind of attentional mechanism, future work is needed to develop one with higher psychological plausibility. Another shortcoming is the type of the information that is used. Although the top-down construction of structural descriptions is important, it is unlikely that it is the sole pressure in recognition. Continuous metric information should be added as well as other bottom-up pressures such as salience. Finally, the model's ability to recognize and reason about more variable and complex objects and events should be tested.

The greatest future challenge is to implement the same principles of binding on more realistic stimuli and to use both structural and metric information.

## Acknowledgments

This work is supported by the Project ANALOGY: Humans – the Analogy-Making Species, financed by the FP6 NEST Programme of the European Commission. (Contr. No 029088).

## References

Biederman, I. (1987). Recognition-by-components: A theory of human image understanding. *Psychological Review, 94,* 115–147.

Biederman, I., & Gerhardstein, P. C. (1993). Recognizing depth-rotated objects: Evidence and conditions for three-dimensional viewpoint invariance. *Journal of Experimental Psychology: Human Perception and Performance, 19,* 1162–1182.

Biederman, I., & Gerhardstein, P. C. (1995). Viewpoint-dependent mechanisms in visual object recognition: Reply to Tarr and Bülthoff (1995). *Journal of Experimental Psychology: Human Perception and Performance, 21,* 1506–1514.

Biederman, I., Mezzanotte, R. J., & Rabinowitz, J.C. (1982). Scene perception: Detecting and judging objects undergoing relational violations. *Cognitive Psychology, 14,* 143–177.

Boyce, S. J., & Pollatsek, A. (1992). Identification of objects in scenes: The role of scene background in object naming. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 18,* 531–543.

Chalmers, D. J., French, R. M., & Hofstadter, D. R. (1992). High-level perception, representation, and analogy: a critique of artificial intelligence methodology. *Journal of Experimental and Theoretical Artificial Intelligence, 4,* 185-211.

Falkenhainer, B., Forbus, K. D., and Gentner, D. (1990). The structure-mapping engine. *Artificial Intelligence*, *41*, 1-63.

Forbus, K. D., Gentner, D., & Law, K. (1994). MAC/FAC: A model of similarity-based retrieval. *Cognitive Science, 19*, 141-205.

French, R. M., & Hofstadter, D. R. (1991). Tabletop: A stochastic, emergent model of analogy-making. *Proceedings of the 13th Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Lawrence Erlbaum.

Gentner, D., Holyoak, & K., Kokinov, B. (Eds.). (2001). *The analogical mind: Perspectives from cognitive science*. Cambridge, MA: MIT Press

Hofstadter, D. R. (1984). The Copycat project: An experiment in nondeterminism and creative analogies. *AI Memo 755*, Cambridge, MA: MIT Artificial Intelligence Laboratory.

Holyoak, K. J. & Thagard, P. (1989). Analogical mapping by constraint satisfaction. *Cognitive Science*, 13, 295-355.

Hummel, J. E., & Biederman, I. (1992). Dynamic binding in a neural network for shape recognition. *Psychological Review, 99,* 480–517.

Hummel, J. E., & Holyoak, K. J. (1997). Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review, 104,* 427–466.

Hummel, J. E., & Holyoak, K. J. (2003). A symbolic-connectionist theory of relational inference and generalization. *Psychological Review, 110,* 220–264.

Hummel, J. E., & Stankiewicz, B. J. (1996). An architecture for rapid, hierarchical structural description. In T. Inui & J. McClelland (Eds.), *Attention and performance XVI: Information integration in perception and communication*. Cambridge, MA: MIT Press.

Kimchi, R. (1992). Primacy of wholistic processing and global/local paradigm: A critical review. *Psychological Bulletin, 112*, 24-38.

Kokinov, B. (1994a). The DUAL cognitive architecture: A hybrid multi-agent approach. In A. Cohn (Ed.), Proceedings *of the Eleventh European Conference on Artificial Intelligence*. London: John Wiley & Sons

Kokinov, B. (1994b). *A hybrid model of reasoning by analogy.* In K. Holyoak & J. Barnden (Eds.), *Advances in connectionist and neural computation theory: Vol. 2. Analogical connections*. Norwood, NJ: Ablex

Kokinov, B., & French, R. M. (2003). Computational models of analogy making. In L. Nadel (Ed.), *Encyclopedia of Cognitive Science*. London: MacMillan.

Kokinov, B., & Petrov, A. (2001). Integration of Memory and Reasoning in Analogy-Making: The AMBR Model. In D. Gentner, K. J. Holyoak, & B. Kokinov (Eds.), *The analogical mind: Perspectives from cognitive science*. Cambridge, MA: MIT Press.

Kuehne, S. E., Gentner, D. & Forbus, K. D. (2000). Modeling infant learning via symbolic structural alignment. *Proceedings of the Twenty-second Annual Conference of the Cognitive Science Society*, 286-291.

Marr, D., & Nishihara, H. K. (1978). Representation and recognition of the spatial organisation of three dimensional structure. *Proceedings of the Royal Society of London, Series B (Biological Sciences)*, *200*, 269–294.

Navon, D. (1977). Forest before trees: The precedence of global features in visual perception. *Cognitive Psychology, 9,* 353-383.

Palmer, S. E. (1975). The effects of contextual scenes on the identification of objects. *Memory & Cognition, 3*, 519-526

Peissig, J. J. & Tarr, M. J. (2007). Object recognition: Do we know more today than we did twenty years ago? Chapter for *Annual Review of Psychology ,58*, 75-96

Petkov, G. (2006). Modeling Analogy-Making, Judgment, and Choice with Same Basic Mechanisms. In D. Fum, F, Missier, & A, Stocco (Eds.), *Proceedings of the Seventh International Conference on Cognitive Modeling*. Trieste, Italy: Edizioni Goliardiche.

Petkov, G., Naydenov, Ch., Grinberg, M., & Kokinov B. (2006). Building robots with analogy-based anticipation. *In: Proceedings of the KI 2006, 29th German Conference on Artificial Intelligence,* Bremen, in press.

Poggio, T., & Edelman, S. (1990). A network that learns to recognize 3-dimensional objects. *Nature, 343,* 263–266.

Tarr, M. J., & Bülthoff, H. H. (1999). Image-based object recognition in man, monkey and machine. In M. J. Tarr & H. H. Bülthoff, *Object recognition in man, monkey and machine*. Cambridge, MA: MIT Press.

# Modeling the Range of Performance on the Serial Subtraction Task

**Frank E. Ritter**[1] **(frank.ritter@psu.edu), Michael Schoelles**[3]**,**
**Laura Cousino Klein**[2]**, and Sue E. Kase**[1]

[1]College of Information Sciences and Technology, and [2]Biobehavioral Health Department
The Pennsylvania State University, University Park, PA 16802 USA
[3]Cognitive Science Department, Rensselaer Polytechnic Institute, Troy, NY 12180 USA

### Abstract

We present a model of serial subtraction, a task where subjects repeatedly subtract a 1- or 2-digit number from a 4-digit number. The model performs 4 min. blocks of these subtractions like subjects do. The current model replicates part of the pace and % correct for group data. Because performance on this task varies widely between subjects, we explore what it means to match the data distribution. We find that our model represents individual subjects better than group means. We can start to model a distribution of performance and illustrate some of what this approach will entail.

## Introduction

Serial subtraction, repeatedly subtracting a 1- or 2-digit number from a 4 digit number is part of the Trier Social Stressor Task (TSST, Kirschbaum, Pirke, & Hellhammer, 1993). This is an interesting task for two reasons. One reason is that it has been used over 100 times in published articles to study the effects of stress on physiology (e.g., Kudlielka, Buske-Kirschbaum, Hellhammer, & Kirschbaum, 2004; Nater et al., 2006; Taylor et al., 2006; Tomaka, Blascovich, Kelsey, & Leitten, 1993). It is a cognitive task used to cause stress, but we don't know how it's performed—there is only one report on how well it is performed (Tomaka, Blascovich, Kelsey, & Leitten, 1993), and this report only provides data on one 4-min. block.

The second reason it is interesting is that subtraction is an interesting task in its own right and as a component task to many other tasks. It involves many cognitive mechanisms making it a good task to study cognition, not just the biobehavioral effects of laboratory stress. Real world tasks that use subtraction include air traffic control, navigation, and piloting the wide range of vehicles that use angular directions.

It would be useful to have a cognitively plausible model of performance of subtraction. This model would serve as an explanation and summary of task performance, helping to summarize regularities, and a model would also be the starting point of a theory of how cognition changes with stress. Because the task requires not only executive control and memory but interaction with the verbal system as well, a model will be able to quantify the constraints that these subsystems of cognition impose on the task. These requirements suggest that the model be constructed on an embodied cognitive architecture (Anderson, in press).

Previous work with an earlier model has shown that the general pattern of high level results (i.e., number of attempts per 4-min. block and percent correct) with serial subtraction can be predicted (Ritter, Reifers, Klein, Quigley, & Schoelles, 2004), and we have used this approach to describe how popular theories of stress could influence performance on this task (Ritter, Reifers, Schoelles, & Klein, 2007). The next steps, presented here, are to create more detailed predictions of performance and compare these predictions to more detailed subtraction performance data than has been previously presented.

The remainder of this paper presents a serial subtraction experiment, the architecture and model, subtraction data, and a comparison of the model with human data. The model's predictions match the individual data fairly well, and provide lessons for understanding how serial subtraction is performed. The model-data comparison also makes suggestions for the further development of cognitive architectures.

## The Serial Subtraction Experimental Data

As part of a larger project on the biobehavioral effects of stress in men and women, serial subtraction was administered as part of the TSST. Several aspects of serial subtraction performance were recorded. We present several of them here as an initial summary of performance on serial subtraction. They are taken from a more complete report (Ritter, Bennett, & Klein, 2006).

## Subjects

Thirty-six healthy women and 20 men, 18-30 years of age (μ=21.1) were recruited to participate in a study examining hormonal responses to stress.

## Method

All subjects participated in the same protocol, which consisted of a baseline rest period, the TSST challenge period (approximately 30 min.), and a recovery period.

Following informed consent and a baseline rest period, participants were asked to complete the TSST which consisted of: (a) preparing a 3.5-min. speech on a personal failure, which they were told would be recorded for later observation, and then (b) completing two blocks of serial subtraction across a 15-min. period. The first subtraction set required counting backwards from a 4-digit number by 7's; the second set required counting backwards by 13's.

Subjects' serial subtraction answers were corrected against a list of answers from the starting 4-digit number. When an incorrect answer was given, the subject was told to "Start over at <the last correct number>". At 2 min. into each 4-min. session, subjects were told that "2-minutes remain, you need to hurry up".

Performance on the first block of 7's and first block of 13's were recorded on the experimenter's scoring sheets. Part way through the study a mark to indicate where the 2-min. warning occurred was added to measure pace of the subtractions. Subjects were paid $30 for their time at the end of the study.

## Results

All 56 subjects completed the task. Table 1 shows the subtraction rates. Overall performance was generally accurate. The proportion correct was not different across problem types (t(56)=1.7, ns).

Table 1. Serial subtraction performance on 4-min. blocks of 7's and 13's, means, (SD), and [ranges].

| (N=56) | 7's | 13's |
|---|---|---|
| **Attempts** | 47.0 (17.1) [8-106] | 36.3 (15.1) [9-78] |
| **%Correct** | 82% (14) [43-100] | 78% (17) [31-100] |

These results are fairly comparable to Tomaka et al.'s (1993) data of 61 attempts per block of 7's for their subjects that saw the task as challenging and 46 for their subjects who saw the task as threatening. While we do not know the variance in Tomaka's data, we can compare it to this data assuming that the variance in each case is equivalent. If we do so, for number of attempts and number correct there is not a reliable difference between this data and Tomaka et al.'s (1994) threatened condition t(36)<1, however, there is a reliable difference between this data and his challenged condition t(36) > 4, p<0.05. There is also a reliable difference for proportion correct, with Tomaka's subjects being correct more often (91% and 92% correct, respectively).

A wide range of performance is found. Figure 1 shows that for the first block of 7's the number of attempts ranged from 8 to 106 attempts, and the number correct and error rates had similar variance. The second block, the 13's, had similar variance. The range of these scores suggests more individual variability than implied by Tomaka et al.'s values or the means in Table 1.

Error rates by sub-blocks were computed for subjects where the scoring sheet was marked with the location of the 2-min. warning. These scores are shown in Table 2. Line 3 in the table shows that subjects made many more errors in the second half of the experiment than in the first half (e.g., 6% of the 7's errors were in the first half, 94% in the second). This trend appeared to be consistent across problem types: On the 7's problems, 33 of the 34 subjects increased their errors in the second sub-block; on the 13's, 30 subjects increased their errors.

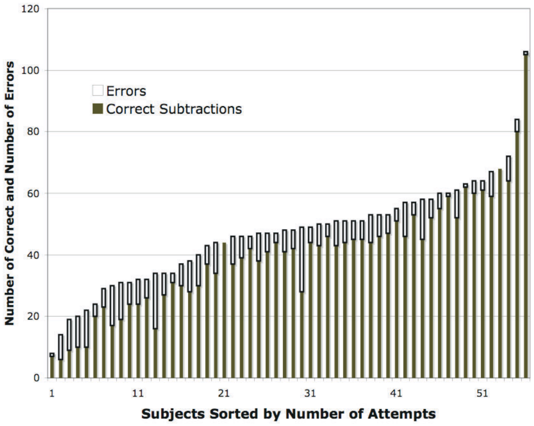


Figure 1. Histogram of attempts and errors for the 7's block.

Table 2. Serial subtraction performance before and after the 2-min. warning.

|  | 7's |  | 13's |  |
|---|---|---|---|---|
| **(N=34)** | **Pre-2-min.** | **Post-2-min.** | **Pre-2-min.** | **Post-2-min.** |
| **Errors** | 0.94 (1.3) | 6.65 (3.7) | 1.15 (1.9) | 6.58 (4.1) |
| **Min/max** | 0/5 | 1/20 | 0/8 | 1/26 |
| **Error %** | 6 (12) | 94 | 13 (21) | 87 |

These errors could have occurred either because of fatigue, the cumulative effects of stress, memory effects such as proactive interference, or perhaps due to the interruption. Or, it could be due to a combination of these effects. This effect is not surprising, in that many theories of stress predict that one starts out good and gets worse as time progresses (Ritter et al., 2007). More fine-grained human data, which we are preparing from another study, will be required to see where and how the increase in errors occurs.

### Summary of Study

The data from this study extend the details of how serial subtraction is performed. We provide further details on this task, including means, SDs, and ranges on subtraction attempts, correct subtractions, and errors by sub-block. This study also provided data on another problem size (13's). The 13's problems appear to be slightly more difficult than the 7's, which might be expected (13's problems have about 38% more simple subtractions, and this ratio here is 30%).

The results confirm the rate of subtractions by 7's for 4-min. blocks previously found, but the rates found here are slightly slower than Tomaka et al. (1993, exp. 2) found. There may be several reasons for the lower number of subtractions per 4-min. block here than in the Tomaka et al. study. The subjects in this study may have been in a more threatening condition. While Tomaka et al.'s subjects were connected to an EKG, subjects in this study were connected to a blood pressure machine, had an indwelling catheter in their arm, and their subtraction attempts were preceded by a talk to a video camera on "an embarrassing incident." Tomaka's subjects had a longer and more relaxing break between sessions than did these subjects (5 min. rest vs. a word problem set that took about 4 min.).

The results show a trend to increasing errors with time. Nearly all subjects made most of their errors in the second half of the tasks. While we cannot see exactly where the errors occurred, it does appear that either the warning or the time on task eventually leads to errors.

## The Serial Subtraction Model

ACT-R 6.0 (Anderson et al., 2004) is a useful architecture to model this task for three reasons: (a) It provides a subsymbolic level to implement changes in processing; (b) it permits the parallel execution of the verbal system with the control and memory systems, which appears to be important for this task; and (c) ACT-R has been used for other models of addition and subtraction developed by other researchers. Therefore, the representation of integers and mathematical rules can be transferred from these to other math models.

### Overview of ACT-R

ACT-R is a two layer modular architecture based on the production system framework. One layer contains symbolic representations and has a serial flow in that only one production can fire at a time. The second layer is a sub-symbolic layer with numeric quantities as representations that are the result of computations performed as if they were executed in parallel.

Figure 2 shows ACT-R's modular architecture. The ACT-R modules communicate through buffers, which can hold a single copy of a declarative memory chunk. The default set of modules can be partitioned into Perceptual, Motor, Control, Memory and Representation Modules. The model presented in this paper exercises the Declarative, Procedural, Goal, Imaginal, and Speech modules. This section describes the details of these modules at the level necessary for understanding our model.

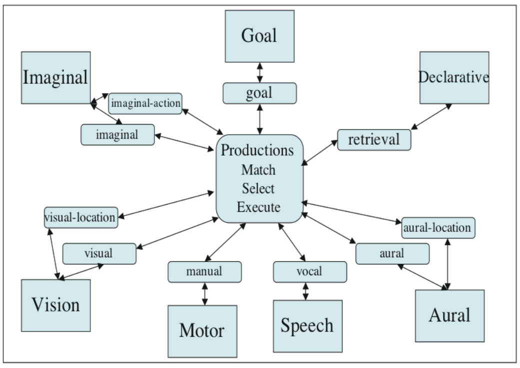


Figure 2. The ACT-R 6 Architecture.

The Declarative Module and the Retrieval buffer make up the declarative memory process. Declarative Memory contains chunks that are typed slot-value objects representing facts. At the sub-symbolic level chunks have a numerical activation value, which quantifies memory operations. Activation in this model is determined by the recency and frequency of use of the chunk plus a component that reflects retrieval system noise. Productions request the retrieval of the chunk from Declarative Memory that has the highest activation among all chunks that match a specified retrieval pattern above a retrieval threshold. Activation represents the degree to which the chunks have been learned and decays over time. Chunks are either created initially or are created during processing. For initially created chunks, the activation can be set as if the chunk had been created at some date in the past and had been previously used. Chunks created during processing are created with a specified base level of activation.

The Procedural Module contains Procedural Memory that consists of condition-action rules (productions). The productions represent procedural knowledge. At each cycle, the conditional constraints specified in the productions are matched against the contents of the buffers. All matching productions are entered into the Conflict Set. The production to execute is determined at the sub-symbolic level by calculating a utility value for each matched production. The production with the highest utility is executed, which consists of performing the operations specified in its actions.

The Imaginal Module buffer implements a problem representation capability. In the Serial Subtraction Model the Imaginal Buffer holds the current 4-digit number being operated on (i.e., the minuend) and the subtrahend. The Goal module and goal buffer implement control of task execution by manipulation of a state slot.

The Speech Module and Buffer speak the result of each subtraction. The rate of speech is a parameter that specifies the rate in seconds per syllable.

## The model

Our model of serial subtraction described starts with a main goal to perform a subtraction and a borrow goal to perform the borrow operation when needed. Both types of goal chunks contain a state slot, the current column indicator, and the current subtrahend (i.e., the number being subtracted). The current problem is maintained in the Imaginal Buffer. This buffer is updated as the subtraction is being performed.

The model starts out with an integer minuend (i.e., the number being subtracted from) of 4-digits. All numbers in the model are chunks of type integer with a slot that holds the number. The model also contains subtraction and addition fact chunks whose slots are the integer chunks described above. This representation of the integers and arithmetic facts has been used in many ACT-R arithmetic models and therefore is a good example of reuse.

The model outputs the answer by speaking the 4-digit result. It has two strategies for answering. The *calc-and-speak* strategy speaks the result in parallel with the calculation of the answer. That is, if the current problem is subtract 7 from 8195 the model would have the speech module speak "eighty one" while the operation of subtracting chunk seven from chunk five was being performed. The other strategy is a basic strategy where the answer is spoken only after the entire subtraction has been performed. All results here are obtained using the calc-and-speak strategy.

The model determines if a borrow operation is required by trying to retrieve a comparison fact that has two slots, a greater slot containing the minuend and a lesser slot containing the subtrahend. If the fact is successfully retrieved then no borrow is necessary, otherwise a borrow subgoal is created and executed.

Borrowing is performed by retrieving the addition fact that represents adding ten to the minuend. The subtraction fact with the larger minuend is retrieved. The model then moves right one column by retrieving a next-column fact using the current column value as the cue. If this retrieval fails then there are no more columns so the borrow subgoal returns back to the main task goal. If there is a next column and its value is not zero then one is subtracted from it by retrieval of a subtraction fact. If the value was 0 then the problem is rewritten in the Imaginal Buffer with a 9 and the model moves to the next column and repeats the steps discussed above, returning to the main task when there are no more columns. If the answer is incorrect, the problem is reset to the last correct answer.

In the main task when the subtraction is complete, the problem is rewritten in the Imaginal buffer and the model speaks the answer using one of the speaking strategies.

There appear to be three important parameters for this model. The rate that the model speaks is controlled by the syllables-per-second parameter (SYL). The retrieval time is controlled by the base level constant (BLC) and decay parameters. The error rate for retrievals in this model is due to the activation noise parameter (ANS). In collecting the model data these parameters (except the decay parameter) were varied to produce outcomes discussed in the results section.

## The Model and Data: Matching the Range of Human Performance

The model's average performance with values of SYL=0.15 s/syllable (ACT-R default), ANS=0.1, and BLC=1 (ACT-R default) was 77 attempts with 83% correct, with no values below 68 attempts. This does not match the distribution of human data. Thus, we started to search for parameter values and parameter value sets to match our subjects' performance.

Figure 3 shows a summary of a parameter sweep on these parameters (ANS: 0.01 to 0.71 by 0.035, SYL: 0.01 to 0.68 by 0.035, and BLC: 1 to 1.95 by 0.05, 1 run/value, 8,000 total runs). The plot shows, for ranges of parameter values, how many runs (across the sets of other parameter values) were within the range of subject performance for number of attempts and % correct. The lines on the left are for SYL and ANS. These lines show that very fast speaking rates are too fast, but otherwise there appear to be a relatively wide range of acceptable values. The other line shows that as ANS increases, the percentage of runs that are within the range of human performance increases as well, and then drops off. The line to the far right is for BLC. It shows that BLC=1.6 (and 1.85) led to a local maximum number of runs that were within our subject range. (The percentages of useful model runs in Figure 3 appear to be somewhat low because, for example, the point at 1.5=BLC contains all the values for SYL and ANS, including quite poor combinations.)

Table 3 and Figure 4 thus show the distribution of performance with the peaks of the parameters tested in Figure 3 (SYL=0.15, ANS=0.38, BLC=1.6 and 1.85). These two distributions have more runs that are within the range of performance by the subjects (which is shown in Figure 3), but the resulting distributions of performance shown in Figure 4 are less like the subjects' performance than the default parameters.

Figure 4 suggests that a distribution of parameters is likely to be more representative of the range of subject performance. The settings of the model shown in Figure 4 appear to match individual subjects (or small sets of subjects) much better than they match the whole distribution. We believe this is because the subjects have different speaking rates, different resources (e.g., working memory and knowledge), different appraisals of the task (and thus different noise and anxiety settings), or other differences we have not yet explored.

Table 3. Performance by the model on 4-min. blocks of 7's and 13's, with SD and ranges for SYL=0.15, ANS=0.38, and BLC=1.85).

| (N=100) | 7's | 13's |
|---|---|---|
| Attempts | 58.3 (2.2) [56-68] | 44.3 (1.95) [39-50] |
| %Correct | 65.7 (21.5) [2-84] | 85.3 (13.1) [27-98] |

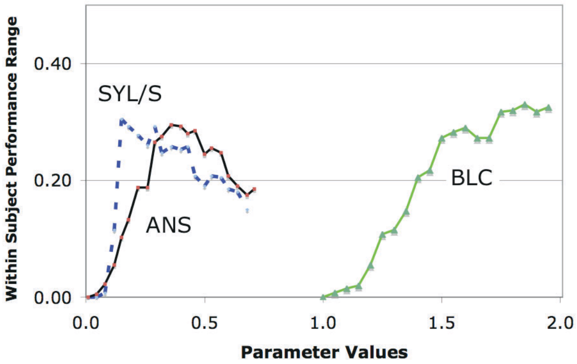


Figure 3. Summary of performance within the range of subject performance (for attempts and % correct) for 7's problems. (Each point is 400 runs.)

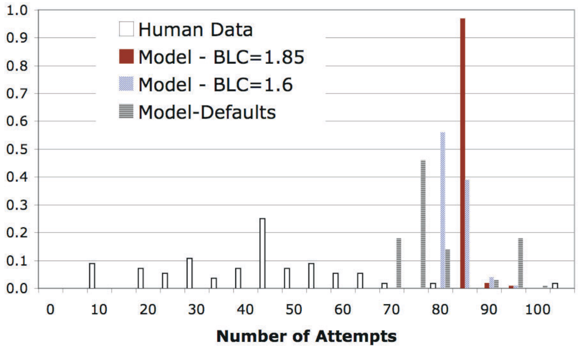


Figure 4. Distribution of attempts (errors not shown) for the 7's problems for the model with better settings and the human data distribution.

## Discussion and Conclusion

The default settings for ACT-R lead the model's performance to match only part of the human data. Examining performance with a wider range of parameter settings suggests that individual differences are what give rise to the distribution that is observed. This is an interesting result, as it suggests ACT-R 6 produces peaked distributions of performance for each setting of parameters. This indicates that we may be able to fit to the average subject, but to fit the subjects' distribution we will have to use a set of parameter settings—the fit is not likely to be a single number, but will be matching of the distribution of individual differences.

### Implications for Serial Subtraction

The analysis here confirm that utterance rate, noise, and base level activation are important in this task. In particular, the development of output mechanisms (speech rate) for architectures is important but somewhat unexplored.

There are a few further measures that would be useful for characterizing behavior on serial subtraction. For example, it would be interesting to know how the pace of subtractions, not just errors, changes over time. Do subjects get faster or slower over time? The error rate could increase because they are performing more subtractions, or it could be that they are performing them more poorly over time. Similarly, it would be interesting to know what errors subjects are making. Are they misretrieving the sub-answers, or are they forgetting to carry or to decrement? How does vocalizing while you are doing subtractions interfere with serial subtraction? We are working on these questions.

## Implications for Architectures

This model and comparison show that distribution of response times and performance variables provide an additional useful, free, inexpensive, and strong constraint—on individuals and on population predictions.

The model was designed to exploit the integrated cognitive systems approach that lies at the core of ACT-R 6. The model performing the whole task including speaking illustrates this theoretical stance that is an important topic in current cognitive architecture research (Gray, 2007). Finally, we are also closer to a position to apply a set of theories of stress implemented as overlays to ACT-R (Ritter et al., 2007) to a sample data set to test the theories of stress on a task with detailed human data.

## Acknowledgements

This project was supported by ONR (N000140310248, FER & LCK), the National Science Foundation (SBR 9905157, LCK), and from the Penn State University College of Health and Human Development (223 15 3605; LCK). The services provided by the GCRC of The Pennsylvania State University are appreciated (NIH Grant M01 RR 10732). We appreciate the dedicated assistance of E. Corwin and M. Stine in completing this project, as well as the research assistants in the Biobehavioral Health Studies Laboratory for subject recruitment, data entry, and data cleaning. Mark Cohen and three anonymous reviewers provided good comments.

## References

Anderson, J. R. (in press). *How can the human mind exist in the physical universe?* New York, NY: OUP.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111*(4), 1036-1060.

Gray, W. D. (Ed.). (2007). *Integrated models of cognitive systems*. New York: OUP.

Kirschbaum, C., Pirke, K.-M., & Hellhammer, D. H. (1993). The Trier Social Stress Test—A tool for investigating psychobiological stress responses in a laboratory setting. *Neuropsychobiology, 28*, 76-81.

Kudlielka, B. M., Buske-Kirschbaum, A., Hellhammer, O. H., & Kirschbaum, C. (2004). HPA axis responses to laboratory psychosocial stress in healthy elderly adults, younger adults, and children: Impact of age and gender. *Psychoneuroendocrinology, 29*, 83-98.

Nater, U. M., La Marca, R., Florin, L., Moses, A., Langhans, W., Koller, M. M., & Ehlert, U. (2006). Stress-induced changes in human salivary alpha-amylase activity—associations with adrenergic activity. *Psychoneuroendocrinology, 31*(1), 49-58.

Ritter, F. E., Bennett, J., & Klein, L. C. (2006). *Serial subtraction performance in the cycling study* (Tech. Report No. 2006-1): ACS Lab, College of IST, Penn State.

Ritter, F. E., Reifers, A., Klein, L. C., Quigley, K., & Schoelles, M. (2004). Using cognitive modeling to study behavior moderators: Pre-task appraisal and anxiety. In *Proceedings of the Human Factors and Ergonomics Society,* 2121-2125. Santa Monica, CA: HFES.

Ritter, F. E., Reifers, A. L., Schoelles, M., & Klein, L. C. (2007). Lessons from defining theories of stress for architectures. In W. Gray (Ed.), *Integrated models of cognitive systems* (pp. 254-262). New York, NY: OUP.

Taylor, S. E., Gonzaga, G. C., Klein, L. C., Hu, P., Greendale, G. A., & Seeman, T. E. (2006). Relation of oxytocin to psychological stress responses and hypothalamic-pituitary-adrenocortical axis activity in older women. *Psychosomatic Medicine, 68*, 238-245.

Tomaka, J., Blascovich, J., Kelsey, R. M., & Leitten, C. L. (1993). Subjective, physiological, and behavioral effects of threat and challenge appraisal. *Journal of Personality and Social Psychology, 65*(2), 248-260.

# Interactions Between Data Labeling and Ratio of Hebbian to Error-Driven Learning in Mixed-Model Networks

**Rebecca J. Robare (RobareOwl@gmail.com)**
Graduate Center, City University of New York
770 Bronx River Road #A64
Bronxville, NY 10708

Cognitive models using mixed-model (that is, combining both task and model learning) learning algorithms, such as Leabra (O'Reilly, 2001), have typically been simulated with supervised-only paradigms. Despite decades of research on learning from partial data (e.g., Dempster, Laird, & Rubin, 1997), only recently have paradigms such as semi-supervised and active learning been applied to more psychologically-oriented models (Love, Medin, & Gureckis, 2004; Robare, 2004). This has been driven, at least in part, by a concern for creating models more reflective of real-world learning environments (Elwin et al., 2007; Li, Farkas, & MacWhinney, 2004; Robare, 2004), which should make mixed-model algorithms and semi-supervised and active learning paradigms more popular in the future.

As paradigms like semi-supervised learning are applied to connectionist networks, often used to model cognitive phenomena in an explicitly brain-based fashion, some unexpected interactions have been observed. Leabra users know that the ratio of task to model learning (the *lmix* variable in the PDP++ software; O'Reilly et al., 1995) interacts with learning rate, *k*-winners-take-all inhibition (kWTA), and the number of epochs needed to train the network (O'Reilly & Munakata, 2000). The ratio of labeled to unlabeled data also interacts with these variables. Although similar interactions have been explored in earlier research on semi-supervised learning and its related paradigms (e.g., Nigam et al., 2000), particularly in the doman of training duration, related as it is to the cost reduction for which semi-supervised learning was initially developed, with the application of semi-supervised learning to connectionist models it is worthwhile to study these interactions anew. There are two primary reasons for this.

First, when the goal of cognitive modeling is not the solving of a computational problem or the development of an artificial intelligence but the explication of human cognition, a different set of considerations apply. Time and cost of human labor, a driving force in other semi-supervised learning research (e.g., Nigam et al., 2000; Yarowsky, 1995), becomes less relevant (though cost may be important in other ways; Antrobus, personal communication, March 2007). Neuroscience provides constraints absent from purely computational research (Sejnowski, 1986), and O'Reilly (1998) has argued persuasively for other constraints on connectionist networks that follow from the requirement for biological plausibility. These constraints mean, or may mean, that the labeled:unlabeled data ratio interacts differently with other variables in a biologically plausible connectionist network than in other kinds of cognitive architectures.

Second, because the drive for using semi-supervised learning in psychologically-oriented models comes not from neuroscience-based constraints but from environment-based constraints, the specific interactions of labeled:unlabeled data ratios with other network variables may have implications for the way we study feedback and reinforcement in real-world environments. Only recently has the notion of selective feedback moved outside the realm of respondent (Rescorla & Wagner, 1972) and operant (Hemmes & Eckerman, 1972) conditioning into cognitive research (Elwin et al., 2007). If we understand, in our models, the relationship between the labeled:unlabeled data ratio and the variables with which it interacts, we will be better able to predict the results of empirical studies on partial and selective feedback and be able to direct observations on the nature of feedback in real-world environments.

The current work focuses on the interaction between the labeled:unlabeled data ratio and the task learning: model learning ratio as applied in Leabra. The Leabra algorithm is a natural fit for semi-supervised paradigms, and the interaction between these two variables is particularly important. The changes wrought in the brain by contact with environmental feedback that is inconstant, incomplete, or absent (Elwin et al., 2007; Li, Farkas, & MacWhinney, 2004; Love, Medin, & Gureckis, 2004; Robare, 2004) will depend on the parameters in the brain that determine changes in synaptic strength.

For example, holding constant kWTA at 0.1 and learning rate at 0.01, for a task learning:model learning ratio of 19:1 (that is, 5% of the weight change comes from Hebbian learning), adding unlabeled data to the training set for a 2:1 ratio of labeled: unlabeled data lowers learning performance, whereas a 3:4 ratio of labeled: unlabeled data improves it. However, when 7.5% of the weight change is from Hebbian learning (a task:model ratio of 12.3:1), the 2:1 labeled:unlabeled data ratio causes less of a decrement than at the 19:1 task:model ratio. These data points, gathered over a series of simulations exploring lexical acquisition and category formation in semi-supervised models, demonstrate the urgent need for more systematic observation of this interaction. Even this casual sequence indicates a complex relationship between the contribution of Hebbian learning to weight-change calculation and the amount of labeled versus unlabeled data presented to the network. Thus we see that at higher amounts of Hebbian learning, less unlabeled data are necessary for performance to improve, whereas more unlabeled data will be required in networks that rely less on Hebbian learning. The precise curve of this relationship has yet to be determined, as has the presence of potential ceiling and floor effects.

## Acknowledgements

I thank Dr. John Antrobus for his constant assistance and advice regarding my work.

## References

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological) 39*, 1-38.

Elwin, E., Juslin, P., Olsson, H., & Enkvist, T. (2007). Constructivist coding: Learning from selecive feedback. *Psychological Science 18*, 105-110.

Hemmes, N. S., & Eckerman, D. A. (1972). Positive interaction (induction) in multiple variable-interval, differential-reinforcement-of-high-rate schedules. *Journal of the Experimental Analysis of Behavior 17*, 51-57.

Li, P., Farkas, I., & MacWhinney, B. (2004). Early lexical development in a self-organizing neural network. *Neural Networks 17*, 1345-1362.

Love, B. C., Medin, D. L., & Gureckis, T. M. (2004). SUSTAIN: A network model of category learning. *Psychological Review 111*, 309-332.

Nigam, K., McCallum, A. K., Thrun, S., & Mitchell, T. M. (2000). Text classification from labeled and unlabeled documents using EM. *Machine Learning, 39*, 103-134.

O'Reilly, R. C. (1998). Six principles for biologically-based computational models of cortical cognition. *Trends in Cognitive Sciences, 2*, 455-462.

O'Reilly, R. C. (2001). Generalization in interactive networks: The benefits of inhibitory competition and Hebbian learning. *Neural Computation, 13*, 1199-1241.

O'Reilly, R. C., Dawson, C. K., McClelland, J. L., & Carnegie Mellon University. (1995). *PDP++*.

O'Reilly, R. C., & Munakata, Y. (2000). *Computational explorations in cognitive neuroscience.* Cambridge, MA: MIT Press.

Rescorla, R. A., & Wagner, A. R. (1972). A theory of Pavlovian conditioning: Variation in the effectiveness of reinforcement and non-reinforcement. In A. H. Black & W. F. Prokasy (Eds.), *Classical conditioning II: Theory and research*. New York: Appleton-Century-Crofts.

Robare, R. J. (2004). Generalization and discrimination in a semantic network trained with semi-supervised learning. In *Proceedings of the Sixth International Conference on Cognitive Modeling* (pp. 412-413). Mahwah, NJ: Lawrence Erlbaum.

Sejnowski, T. J. (1986). Open questions about computation in cerebral cortex. In D. E. Rumelhart, J. L. McClelland, & the PDP Research Group, *Parallel distributed processing, Vol. 2: Psychological and biological models*. Cambridge, MA: MIT Press.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. *Proceedings of the Annual Meeting of the Association for Computational Linguistics* (pp. 189-196).

# Prototypical Relations for Cortex-Inspired Semantic Representations

**Florian Röhrbein (florian.roehrbein@honda-ri.de)**
**Julian Eggert (julian.eggert@honda-ri.de)**
**Edgar Körner (edgar.koerner@honda-ri.de)**

HONDA Research Institute Europe GmbH, Carl-Legien-Str. 30
63071 Offenbach am Main, Germany

## Abstract

Cognitive systems for the representation of declarative knowledge like semantic networks and other graph-based systems are widely unrelated to characteristic neurobiological mechanisms in the brain. In this contribution we report on our efforts in bridging the gap between typical semantic relations like "is part of", "has property" etc. and the laminar wiring pattern of the neocortex. Central to our approach is the identification of the cortical column as a basic building block within the relational network. These columns are typically sectioned into subsystems which comprise different horizontal layers and thereby provide different links for forward, backward and lateral processing. We show how these inter-columnar connections can be related to semantic links, which reflect hierarchical knowledge, temporal ordering and ontological relationship. These dimensions are of outstanding interest for most cognitive tasks. But also arbitrary n-ary relationships can be build by representing the relations as nodes and using only the proposed basic link types. As inference mechanism, a simple locally controlled activation spread was applied. It results directly from the intra-columnar connectivity which is uniform for all nodes. We tested the system with large commonsense databases and obtained promising results including predictions, context influences and feature inheritance.

**Keywords:** cortical column; knowledge representation; relational structures

## Introduction

For the representation of relational knowledge in a graph-based model we have developed a neural-symbolic network which combines ideas from classical semantic networks and recent findings of the neocortical wiring. It consists of columnar-like nodes as uniform entities for the representation of all concepts of the domain, including sensory measurements, motor actions, instances and categories. The nodes are connected by a set of directed links, which can be related to columnar subsystem, as we will argue in the next section.

## Semantic relations and columnar connections

The biological entity, which in our approach corresponds to a network node, is the cortical column. The column is well known as the basic computational unit in the brain and its six-layered architecture has been addressed by several researchers to unravel the functional role (Raizada & Grossberg, 2003; Lücke & von der Malsburg, 2004; Kupper *et al.*, 2006). Here we concentrate on a network build out of columnar-like nodes and do not target at a biologically detailed modeling of the single cortical column. The columns are typically sectioned into subsystems (see Fig. 1) which comprise different horizontal layers and thereby provide different links for bottom-up (BU), top-down (TD) and lateral processing.

We refer to a schema described in (Körner, Tsujino & Masutaki, 1997) which assumes six distinct systems, which we will only briefly sketch here: Subsystem A1 receives input from lower cortical areas, subsystems A2 and B2 project to areas higher in the cortical hierarchy, thus establishing together a bottom-up processing stream. Top-down processing is realized via subsystem C2, which projects to lower areas, targeting in cortical layer I (since there are no neurons in this layer, it is not called a subsystem). The two remaining systems are for lateral processing (B1), which comprises many different cell types and can be subdivided further, and a system which sends primarily motor information to subcortical structures (C1).

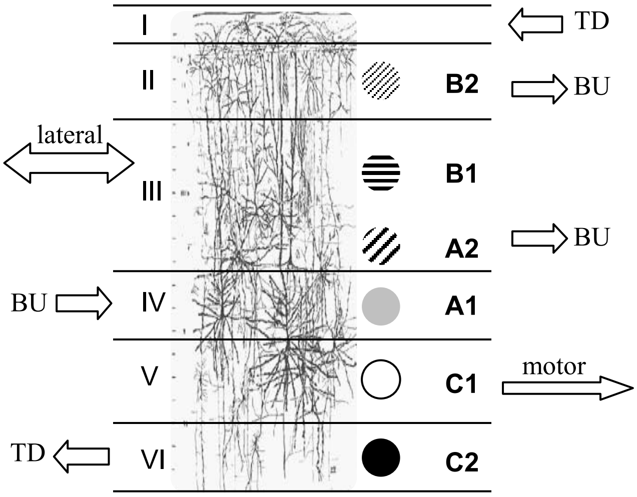


Figure 1: Sketch of the major pathways connecting cortical columns. Shown are cytoarchitecturally defined cortical layers (I-VI, left) and proposed functional subunits (A1 etc., right) with shadings referenced in Fig. 7.

The relevant question in this context now is how semantic links can be ascribed to these pathways which originate from distinct subsystems. A good point to start with will be to look at those semantic relations which seem to be of ubiquitous importance. Indeed, there seem to be very few basic relations which are relevant for concepts on all layers of abstraction, independent of the actual knowledge domain and these might be grouped according the three dimension of hierarchy, sequence and relationship.

Hierarchies are used all over the neocortex as the core organization principle to deal with the nested structure of the surrounding world. Along this dimension of knowledge chunks the notions of BU and TD processing apply. Knowledge about hierarchical relationships is usually expressed in meronymies and holonymies, but also in relations like "is located in" or in the temporal domain ("happens during" etc). In our system three link types are used to build the chunking hierarchy and, following the basic cortical processing streams, columnar subsystems are assigned to each of them (Fig. 1, for details see e.g. Thomson & Bannister, 2003): A "has component" link, which originates in C2 and projects to layer I of nodes on a lower level (top-down). Two "is component of" links stem from different cortical layers (A2, B2) but terminate both in input layer A1. Together they serve for bottom-up information flow, and just differ in the granularity of transmitted information. Note, that an increased level of detail leads to a hierarchy, in which subclasses are represented above superclasses and instances are represented above categories (see e.g. Quian Quiroga *et al.*, 2005), generating a reversed ontological hierarchy.

Sequential information is essential, especially for prediction. We associate corresponding semantic links with the columnar subsystem C1 (compare Lomber & Payne, 2000), but will not make use of it in the work presented here. Instead, we concentrate on ontological knowledge which is expressed in hyponyms and hypernyms. For this dimension (coined "relationship" above) six link types are used: has property / is property of, has subclass / is subclass of and has role / is role of. A suitable columnar subsystem for these connections seems to be B1 because of the existence of distinct functional subsystems within upper layer III (Yoshimura, Dantzker & Callaway, 2005) and the characteristic dense wiring pattern with horizontal connections of different ranges (e.g. Hirsch & Gilbert, 1991). In this line, links denoting subclass relationships connect columns within one level (e.g. within one cortical area), whereas property and role links make inter-area connections, since they connect conceptual representations with more perceptually based ones. Summarizing, we have the following set of link types

- has component / is component of
- has consequence / is consequence of
- has property / is property of
- has subclass / is subclass of
- has role / is role of

All network links proposed here differ in two important aspects to common semantic network links: First, we only use a very restricted set of basic link types, which are biologically justified, since they can be associated with specific neuronal source and target populations each within a specific columnar layer. Second, these links do not vary from node to node, but are common to all nodes. Not all links, of cause, are used by every node, but there are no links which are available only for certain nodes. The motivation for this homogenous layout is that the basic structure of the biological column is widely independent of the cortical site.

In the following the focus will be only on the lateral connections originating and targeting in subpopulations of B1. For details on link types associated with the other subsystems for semantic relations about temporal and spatial ordering, see (Röhrbein, Eggert & Körner, 2007). To motivate the link types associated with B1, we start with quite general considerations concerning the coding of relational structures.

## Representation of arbitrary relations

How can arbitrary relations be expressed within a graph-based framework? The common way to represent facts like "John loves Mary" is to have nodes for the concepts involved ("John", "Mary") and a directed link between them. The link is typically labeled with the relation, which holds between the connecting concepts ("loves"). Unfortunately, this works only for dyadic relations: As soon as a third concept is involved, like in "Mary gives John a cookie", the scheme has to be revised.

One option here is to extend the directed links towards "hyperarcs" (see e.g. Harel, 1988). These are either conceptualized as n-ended arcs and then solve only half of the problem, because still only 2 roles are possible. Cases where directed hyperarcs are sufficient are quite restricted, e.g. for relations like lies-between(X,Y,Z,…) which can be represented by setting head H={X} and tail T={Y,Z,…}. Others, e.g. Boley (1992) in his "directed recursive labelnode hypergraphs" propose links which start at the relation node, cut n-1 argument nodes and end at the nth node of the n-ary relation. He criticizes approaches which introduce additional nodes, because they add "pseudo-entities", but this holds only as long if they cannot be given a reasonable interpretation.

Another solution to handle relations with arity greater than 2 is to have an extra node pointing the all involved concepts (e.g. used in SNePS). The links have to be labeled, otherwise one could not differentiate between the statements "Mary gives John a cookie" and "John gives Mary a cookie". But here the labels do not reflect the type of relation as for binary relations; rather they specify roles like "giver", "recipient" and "object". More as a side effect, in doing so the "artificial" node becomes to represent the whole relation. The provision of labels is a general problem. They are unsatisfactory for several reasons, most importantly for us because biology does not allow for arbitrary link types.

A variant of this approach would be to chunk information into a node which then comes to represent a part of the whole statement like "Mary gives John". This is useful for computational reasons, since reasoning with nodes of arity beyond 3 have proven to be intractable. Conceptually there is no further gain in representing parts of a statement, since it makes no sense to dispense completely with separate concepts for both "Mary" and "John", and the semantic of the link becomes even more obscure.

For a solution without arbitrary link types, two aspects of a "standard link" have to be considered: First, the link has only two ends, and second, there are only two different "values" for the endpoints: "arrow tail" and "arrowhead", usually associated with an "ingoing" and "outgoing" semantic. For a true extension therefore, a graphical notation is needed based on that depicted in Fig. 2 (b) for a ternary relation. Here the link is allowed to have more than two connecting points and at the same time more than two possible values. For a biological interpretation such a graph-based approach still causes a problem, since there are no different connection endings for neurons that can be associated with arbitrary roles. (In fact, there seem to be different link types and associated with them, different roles, but these are not arbitrarily definable, see above.)

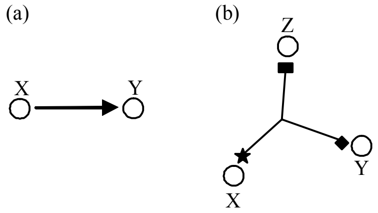


Figure 2: Graphical representations for binary relations (a) have to be extended in two ways to deal with n-ary relations: The number and the type of terminal points. Sketched in (b) is a graphical notation for n=3.

Here we present a very straight-forward solution to this by proposing an additional node for each role. A ternary relation is now represented as sketched in Fig. 3 (b) with new intermediate nodes labeled with digits. This schema can easily be extended for relation with greater arity (c) and is also valid for binary relations (b), thus avoiding any discontinuity. The new nodes have a clear interpretation: For the statement "Mary gives John a cookie" with X=Mary, Y=John and Z=cookie, node 1 represents "Mary acting as giver", node 2 "John acting as receiver" and node 3 "cookie as a gift". These nodes can participate in other relational statements, e.g. the node 2, if it is to be expressed that Mary gives some things to some other people. Of course, the same holds for concepts, since these usually participate in different situations having different roles (see simple example below). Fig. 3 (d) shows how the fact "a can is made of aluminum" is represented and indicates the embedding in our columnar framework with links of type has-role / is-role-of (depicted now as solid bidirectional links).

On the conceptual side, the advantage of the proposed schema lies in the uniform treatment of arbitrary n-ary relations. This is opposed to standard semantic networks which show a tendency to binarize not only relations with more than two roles, but also monadic relations like "has property". E.g., typical KL-ONE representations are restricted to unary and binary predicates. This is not an inherent restriction and n-ary description logics have been proposed (e.g. Schmolze, 1989), but nevertheless n-tuples for n>2 are usually represented indirectly by reification. There are also technical advantages, since the modeling of relations as nodes allow for inheritance, activation spread etc., which we will elaborate on shortly in the next section.

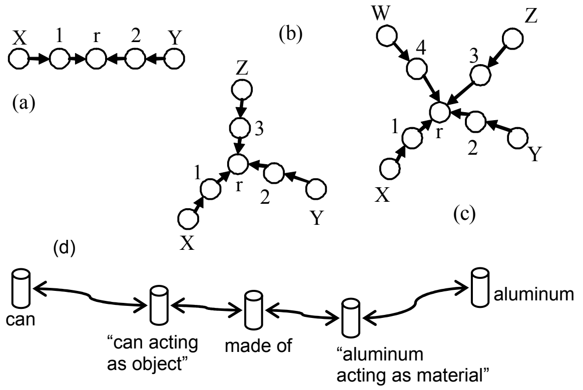


Figure 3: Proposed pattern for representing n-ary relations (a-c, n=2,3,4). The dyadic relation in (d) is represented with 5 nodes connected with has-role / is-role-of links.

## Relational prototypes

For every relation which can be expressed in our framework (like "gives", "made of" etc.) there is a so-called relational prototype, which functions as a template and which is connected with all relations of that type. An example is given in Fig. 4 for the relation "made of". There are two instances of this relation involving three concepts in the same manner as in Fig. 3 (d), but additionally nodes which code the relation are connected (dashed arrows) with corresponding nodes of the relational prototype. These links are of type has-subclass / subclass-of and thus allow for the inheritance of properties (not included in the figure). The nodes constituting the relational prototype (within the gray oval) get their meaning via their connection to all instances of these nodes.

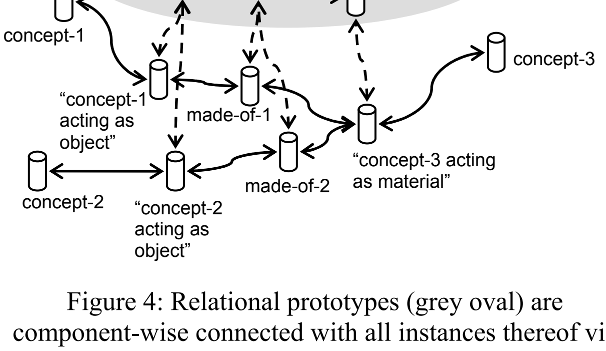

OBJECT
MADE OF
MATERIAL
concept-1
"concept-1 acting as object"
made-of-1
made-of-2
concept-2
"concept-2 acting as object"
concept-3
"concept-3 acting as material"

Figure 4: Relational prototypes (grey oval) are component-wise connected with all instances thereof via has-subclass / is-subclass-of links (dashed arrows).

Note that also the cortex seems to build separate representations for different tasks. For spatial cognition tasks e.g. knowledge about spatial relations has to be provided and all the nodes representing instances of these relations should be arranged in neighbored representations. There is also a psycholinguistic justification of treating relations as abstract concepts, which comes from work of Chaffin and Herrmann (1988). They found basic characteristics, which are known to hold for objects, also for relations. These include decomposability, typicality, codability, multiple inheritance and compositionalibility.

## Activation spread

The activation spread results from intra- and inter-columnar connectivity patterns. Internally each node has an activity vector with one entry for each subsystem. The connections between nodes depend on the represented knowledge and follow the rules outlined above. In the current version of the system, we use the simplest rule for intercolumnar connections without weighting and thresholds: The incoming activation $a_j$ of a specific subsystem $x_{in}$ equals the sum of the activations of the corresponding outgoing subsystems of all those nodes i, which are connected to node j:

$$a_j(x_{in}) = \sum_i w_{ji} a_i(x_{out})$$

The processing within a node depends on the intra-columnar wiring, which is handcrafted, but identical for all nodes in the network. We omit here all definitions except those which are made use of in the example which follows in the next section, i.e. we focus on the subsystems marked in grey in Fig. 5.

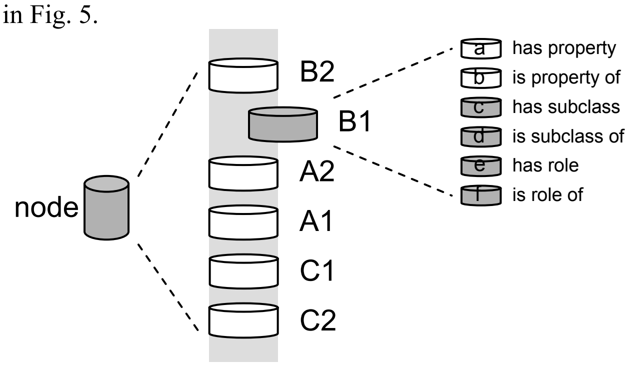


Figure 5: Columnar network nodes are subdivided into functional subsystems and associated semantic link types. In this report the focus is mainly on the shaded parts of B1.

The intra-columnar propagation rules for the activities $a_j(x_{out})$ of a node j are defined in the following for the subsystems $x_{out}$ of B1. We use the abbreviations B1a, …, B1f given in Fig. 5.

has subclass:

$$a_j(B1c_{out}) = a_j(B1b_{in}) + a_j(A1) + a_j(B1c_{in}) + 2 * a_j(B1f_{in})$$

is subclass of:

$$a_j(B1d_{out}) = a_j(B1a_{in}) + a_j(layerI) + a_j(B1d_{in})$$

has role:

$$a_j(B1e_{out}) = 0.5 * a_j(layerI) + a_j(B1e_{in})$$

is role of:

$$a_j(B1f_{out}) = a_j(B1e_{in}) + 3 * a_j(B1c_{in}) * a_j(B1f_{in})$$

Note, that in all cases the activity vector remains unchanged, unless the incoming activity changes, i.e. there is no automatic fading away.

## Toy example

### Network

We tested our network with large knowledge databases, i.e., all relations are extensionally defined. In the following we demonstrate the behavior with a toy example consisting of four pieces of relational knowledge, which are fed into the system:

```
can is-made-of aluminum
can is-used-for drinking
can is-used-for gaming
car is-used-for driving
```

A representation of these statements involves six concepts (for can, car, aluminum etc.) and two relational prototypes (is-made-of and is-used-for). All nodes and relations necessary for representing the knowledge are generated automatically resulting in the network shown in Fig. 6.
This example requires the use of only two different link types: has-role / is-role-of links (depicted as solid bidirectional arrows) and has-subclass / is-subclass-of links (dashed bidirectional arrows). Note that role nodes can be part of more than one relational statement (here node “tool-can”).

### Task

Let’s assume that an object has been presented to the system, which was identified as a can. This successful recognition leads to an activation of the column representing “can”`. In the next time step one might wish to ask the system about the usage of this object. This situation can be directly expressed in our network through the activation of two nodes: Node “can” receives top-down input via layer I which activates neurons in subsystem C2 and parts of subsystem B1. We choose an activity value of 1 and set:

$$a_{can}(layerI) = 1$$

Node “usage” is activated bottom-up leading to a firing of neurons in subsystem A1. For simplicity, the same activity value is assumed here:

$$a_{usage}(A1) = 1$$

Starting at these two nodes, the activation spreads according to the proposed intra-columnar wiring and according to the connections between nodes specified by entries of the knowledge base.

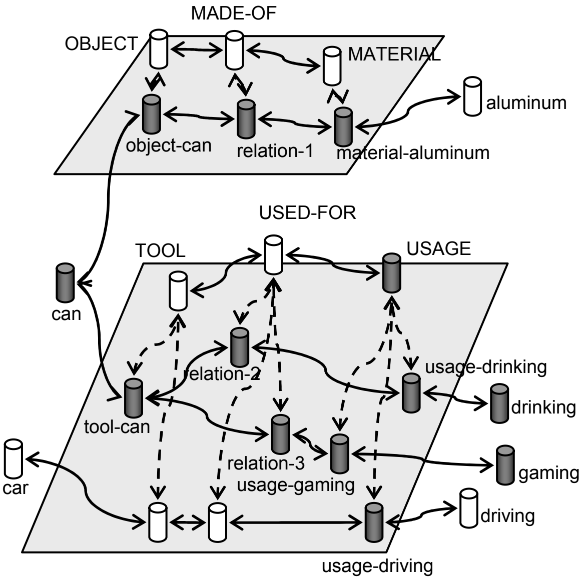


Figure 6: Example network with representations for relations made-of and used-for. See text for details.

## Result

In Fig. 6 all nodes which receive activation are colored: The nodes which received input, several nodes coding the relation and two nodes which represent a concept: “drinking” and “gaming”. This highly selective activation of relevant nodes becomes even more important if we consider realistic knowledge networks like that one we obtained by using freely-available ontological and commonsense databases (see Röhrbein et al., 2007). They typically comprise hundreds of instances for one relational prototype, but the only relational structures which get completely activated are those matching with the concepts contained in the “question”, in this example “can” and “USAGE”. As can be seen from the propagation rules defined above, this is due to the nonlinearity for the is-role-of activity, which leads to the desired gating behavior.

For a quantitative comparison the contribution of the different subsystems to the total node activity can be found in Fig. 7. The depicted activity vectors of all involved nodes show the highest activation for nodes “drinking” and “gaming”. Clearly, the gained sum activity scales with the provided input values, which were here set to 1, but the different subsystems’ contributions depend on the weightings in the definitions above. Moreover, it is not quite clear what should be taken as the “sum activity” of a column (see e.g. measurements by Staiger et al., 2000). On the other hand, the provision of an “answer” as system output, which might consist of a short list of top-ranked nodes, is considered rather as a side effect. The more important result in our view lies in the selective activation of relevant substructures which can be used for subsequent processing steps.

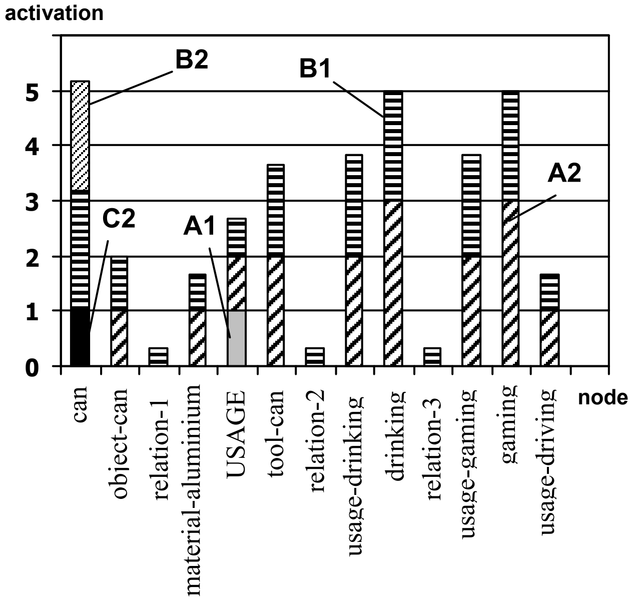


Figure 7: Activity vectors resulting from C2 activation of “can” and A1 activation of “USAGE”. Shading styles refer to columnar subsystems in Fig. 1.

## Discussion

A similar node-based representation is used in LISA (e.g. Hummel & Holyoak, 2003), in which a form of symbolic connectionism is proposed which also avoids labeled links for the representation of relational structures. Hummel and Holyoak argue for a 4-tired hierarchical schema comprising “propositional units”, “role-binding units”, “token units” and “semantic units”. These nodes roughly correspond to neurons, whereas in our approach only one uniform unit is assumed which is related to a larger biological entity, the cortical column. This enables us to differentiate e.g. between superclasses and properties which are treated uniformly in LISA as features at the level of “semantic units”. Related constructs can also be found in connectionist modeling approaches of linguistic aspects. E.g., Hadley and Cardei (1999) introduced p-nodes, which are clusters consisting of a core node connecting sequence nodes with role nodes. Quite similarly to our approach, these role nodes are for the binding of concepts to appropriate roles, but here the possible roles are restricted to a predefined and fixed set of only three role nodes: concepts can be linked either to an “agent role”, an “action role” or a “patient role”. In SHRUTI (Shastri, 1999), "focal-clusters" represent n-ary relations and contain beside n role nodes also a number of special-purpose nodes like enablers and collectors. Shastri does not relate this unit to biological mechanisms like the cortical column, but it might be worthwhile to pursue that direction.

In general, the coding of higher-valence relations by introducing additional nodes has already been recommended by Levesque, Brachman (1984) and is proposed also in recent approaches (e.g. Schultheis, Barkowsky & Bertel, 2006), but without considering the need for having role nodes. Another example would be the “relational element theory” put forward by Chaffin and Herrmann (1988). In their investigation on analytical vs. unitary approaches to semantic relations they also propose a decomposition into relational elements (like “agent” and “instrument”), but these are more like properties of relations, e.g. “dimension”, “discrete” etc. than role nodes proper. Furthermore they consider only binary relations.

A final note should be made with respect to the postulation of Firstness, Secondness and Thirdness as the fundamental ontological entities (see Sowa, 2000). This trichotomy has not always been interpreted uniformly, but it seems to fit to our graphical representation, where 1st-ness corresponds to the objects per se, 2nd-ness to the properties of an objects and 3rd-ness to the relation between objects (see Fig. 8).

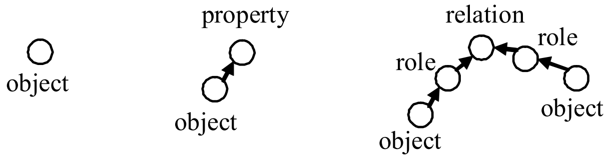


Figure 8: Peircean notions of Firstness, Secondness and Thirdness as major ontological distinctions might be related to network patterns which emerge from our approach.

## References

Boley, H. (1992). Declarative Operations on Nets. In Fritz Lehmann, editor, *Semantic Networks in Artificial Intelligence*, 23, 601-637.

Chaffin, R., & Herrmann, D.J. (1988). The nature of semantic relations: a comparison of two approaches. In: Evens, M.W. (ed.) *Relational models of the lexicon: Representing knowledge in semantic networks.* Cambridge University Press, 289-334.

Hadley, R.F. & Cardei, V.C. (1999). Language acquisition from sparse input without error feedback. *Neural Networks*, 12, 217-235.

Harel, D. (1988). On visual formalisms. *Communications of the ACM*, 31, 514-530.

Hirsch, J.A. & Gilbert, C.D. (1991). Synaptic physiology of horizontal connections in the cat's visual cortex. *The Journal of Neuroscience,* 11, 1800-1809.

Hummel, J.E. & Holyoak, K.J. (2003). A Symbolic-Connectionist Theory of Relational Inference and Generalization. *Psychological Review*, 110, 220-264.

Körner, E., Tsujino, H. & Masutani, T. (1997). A cortical-type modular neural network for hypothetical reasoning. *Neural Networks*, 10, 791-814.

Kupper, R., Knoblauch, A., Gewaltig, M.-C., Körner, U. & Körner, E. (2006). Simulations of signal flow in a functional model of the cortical column. *Neurocomputing*, doi:10.1016/j.neucom.2006.10.085.

Levesque, H.J. & Brachman, R.J. (1984). A fundamental tradeoff in knowledge representation languages. In Brachman, R.J. & Levesque, H.J. (eds.), *Readings in Knowledge Representation*. Morgan Kaufmann, Los Altos, California, 41-70.

Lomber, S.G. & Payne, B.R. (2000) Translaminar differentiation of visually guided behaviors revealed by restricted cerebral cooling deactivation. *Cerebral Cortex*, 10, 1066-1077.

Lücke, J & von der Malsburg, C. (2004). Rapid Processing and Unsupervised Learning in a Model of the Cortical Macrocolumn. *Neural Computation*, 16, 501-533.

Quian Quiroga, R., Reddy, L., Kreiman, G., Koch, C. & Fried, I (2005). Invariant visual representation by single-neurons in the human brain. *Nature*, 435. 1102-1107.

Raizada, R. & Grossberg, S. (2003). Towards a theory of the laminar architecture of cerebral cortex: Computational clues from the visual system. *Cerebral Cortex*, 13, 100-113.

Röhrbein, F., Eggert, J. & Körner, E. (2007). A Cortex-Inspired Neural-Symbolic Network for Knowledge Representation. In: *Proceedings of the IJCAI-07 Workshop on Neural-Symbolic Learning and Reasoning* (NeSy-07). CEUR Workshop Proceedings.

Schmolze, J.G. (1989). Terminological Knowledge Representation Systems Supporting N-ary Terms. In: Brachman, R.J., Levesque, H.J., Reiter, R. (Eds.): *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann: 432-443.

Schultheis, H., Barkowsky, T. & Bertel, S. (2006). LTM-C: An Improved Long-Term Memory for Cognitive Architectures. *Proceedings of the 7th International Conference on Cognitive Modeling* ICCM 2006: 274-279.

Shastri, L. (1999). Advances in SHRUTI - A neurally motivated model of relational knowledge representation and rapid inference using temporal synchrony. *Applied Intelligence.*, 11, 79-108.

Sowa, J.F. (2000). *Knowledge Representation: Logical, Philosophical, and Computational Foundations*, Brooks Cole Publishing Co., Pacific Grove, CA.

Staiger, J.F., Kötter, R., Zilles, K., Luhmann, H.J. (2000). Laminar characteristics of functional connectivity in rat barrel cortex revealed by stimulation with caged-glutamate. *Neuroscience Research*, 37, 49–58.

Thomson, A.M. & Bannister, A.P. (2003). Interlaminar connections in the neocortex. *Cerebral Cortex*, 13, 5-14.

Yoshimura, Y., Dantzker, J.L. & Callaway, E.M. (2005). Excitatory cortical neurons form fine-scale functional networks. *Nature*, 433, 868-873.

# Cognitive Redeployment in ACT-R: Salience, Vision, and Memory

**Terrence C. Stewart (terry@ccmlab.ca)**
**Robert L. West (robert_west@carleton.ca)**
Carleton Cognitive Modelling Lab
Institute of Cognitive Science, Carleton University
1125 Colonel By Drive, Ottawa, Ontario, K1S 5B6 Canada

## Abstract

Cognitive redeployment is the idea that an important part of the evolution of cognition is the adaptation and re-use of existing cognitive modules for new purposes. In this paper, we apply this idea to the ACT-R declarative memory system and the ACT-R visual system. We have developed a common underlying implementation of these systems as part of the Python ACT-R project. As a result, we can apply aspects of vision to memory and vice-versa. In particular, we apply the concepts of base-level learning, spreading activation, and partial matching from declarative memory to vision. We also apply the recent visual saliency model from vision to declarative memory. This results in a variety of new models of existing phenomena which rest on a simpler and more general theoretical framework.

## Introduction

In previous work (Stewart & West, 2006; Stewart & West, in press), we have argued that it is useful to deconstruct the ACT-R cognitive architecture (Anderson & Lebiere, 1998) into its more basic components. By combining these components in various ways, we can explore the effects of *architectural* changes, as well as the more commonly explored effects of numerical parameter changes. In other words, instead of just adjusting values such as latency or noise, it is possible to investigate the effects of adding multiple memory systems, multiple production systems, and new buffers for communication. This system is called Python ACT-R and is integrated with a complete set of tools (CCMSuite) for creating experimental environments, running parallel simulations, and performing data analysis. All source code and experimental data can be found at <http://ccmlab.ca/ccmsuite.html> including the models described here.

In this paper, we explore the implications of treating the ACT-R declarative memory system and the ACT-R visual system as being built out of the same underlying components. These systems are traditionally treated as quite separate in ACT-R; indeed, until recently the visual system was part of an add-on component called ACT-R/PM. However, these systems have a similar interface, in that they both can be given requests to find items that fit a particular pattern. The precise algorithms for determining which items are found by the requests do differ between vision and declarative memory, but we show in this paper that there are underling similarities that allow us to combine these algorithms, and apply the declarative memory ones to vision and the vice versa.

The goal of this work is to determine whether there is an advantage to treating vision and memory as systems built from the same type of generic system. By consolidating the models of these two phenomena and exploring their similarities (and differences), we hope to arrive at a more parsimonious cognitive model.

### Cognitive Redeployment

If a generic memory system can be shown to accommodate both the vision and the declarative memory systems within the ACT-R cognitive architecture, then it can be seen as an example of *cognitive redeployment*. Michael Anderson (2007) argues that "the brain evolved by preserving, extending, and combining existing network components, rather than by generating complex structures de novo." His *massive redeployment* hypothesis (Anderson, forthcoming) states that "cognitive evolution proceeds in a way analogous to component reuse in software engineering, whereby existing components ... are used for new purposes and combined to support new capacities, without disrupting their participation in existing programs" (Anderson, 2007, p.13). This is the theoretical motivation behind our generic model. Based on this we believe that it is useful to examine how cognitive modules can be modified and re-purposed to perform different cognitive functions.

The Python ACT-R system supports this exploration by allowing the modeller to customize the ACT-R architecture. In the current Lisp ACT-R system, the underlying modules can certainly be modified, but this requires delving into the underlying implementation of the architecture. In Python ACT-R, creating memory systems and customizing them in various ways is a basic part of the modelling process (much like defining productions and chunks). No understanding of the underlying implementation is needed. This makes architectural exploration available to any cognitive modeller, not just those willing to implement models from scratch.

## Salience

We began this project by updating Python ACT-R's capabilities by implementing ACT-R's new visual salience system (Byrne, 2006). This was originally developed to model the "pop-out" effect familiar to vision researchers. This is implemented in Python ACT-R as a separate module that can connect to the visual system. However, because the visual system in Python ACT-R is built using the generic system as the declarative memory system, it was also possible to connect the salience module to declarative memory. However, it should be noted that we are not claiming that the same *physical neurons* in the brain may be responsible for both visual and memory effects. Rather, we believe that a similar cognitive structures may be involved, and that a common computational model may be used for both.

To develop this idea, we first describe the ACT-R declarative memory system and the vision system. Next, a generic system is described that is capable of being customized via particular sub-components to perform either task. We then demonstrate its capabilities, showing particular differences between this vision model and the standard ACT-R vision model. Finally, we investigate what capabilities are gained by the declarative memory system now that the visual salience system can be integrated with it. The result is both a novel model of *distinctiveness* in memory and a demonstration of the usefulness of cognitive redeployment.

## Declarative Memory

The ACT-R declarative memory system consists of a module for storing *chunks* (small meaningful collections of data, such as "Fido is a dog") and a retrieval buffer for storing the currently recalled chunk. Chunks are placed into memory whenever they are used by an ACT-R production rule. A production rule can also make a request from memory, asking for a chunk that fits a particular pattern (such as "Fido is a *what?*"). If a chunk is found, then its contents are placed in the retrieval buffer.

In many cases, there will be several chunks that match the requested pattern. In these situations, the chunk with the highest *activation* is retrieved. The activation of a chunk is based on a variety of factors, the most important of which is the *base level learning* equation. Here, a chunk's activation is increased when it is used, and this activation decays over time. There is also a formula for adjusting the activation of chunks that almost match the memory request pattern (partial matching), a formula for increasing the activation of chunks that are similar to chunks already in buffers (spreading activation), and a random amount of noise that leads to variability in behaviour.

However, there is no way of *modifying* chunks once they are placed in memory. Therefore, to avoid the problem of repeatedly retrieving the most active chunk in situations where this is not appropriate, ACT-R has a system for maintaining "Fingers of Instantiation" (or FINSTs). This allows a memory request to indicate that it should not recall a recently retrieved item. Details and formulas for the declarative memory system can be found in (Anderson & Lebiere, 1998).

## Vision

The vision system in ACT-R (formerly a part of ACT-R/PM) has two sub-systems and two buffers associated with these sub-systems. These two sub-systems are the "where" and "what" systems, where the former is in charge of directing visual attention, and the later in charge of determining information about the object at the attended location.

To perceive an object, an ACT-R production is first used to instruct the vision system to find a location in space. This can be something as simple as "find any object" or as complex as "find a red object on the right side of the screen that's above the object currently being attended to". When an object is found that matches these criteria, the location of the object is placed in the visual location buffer. Next, the "what" system can be used to attend to that location and get information about the object. In ACT-R, this does not involve a theory of visual processing. Instead, the features that have been attached to the object by the modeller are placed in the visual buffer. For example, a picture of a house might be set to create a chunk in the visual buffer that simply says "house".

The vision system also supports a number of other behaviours. If an object moves while being attended to, then attention will follow that object (object tracking). If nothing is being attended to, but a new item appears which matches the previous search request, then it will automatically be attended to (buffer stuffing). Requests can also be made to only attend to "new" or "old" objects (i.e. objects that have recently appeared or been visible for a longer period of time). This functionality is similar to FINST system in declarative memory.

One of the most recent advances in the ACT-R vision system is the addition of a theory of *visual salience* (Byrne, 2006). Here, when an object is being searched for, the system no longer chooses randomly among matching items. Instead, the visual features of the object are examined, and objects that have more *rare* features are more likely to be attended to. This causes the well-known pop-out effect to occur, e.g., where a single red object in a set of blue objects is much more likely to be attended to.

## Generic Chunk Storage and Retrieval

Both the declarative memory system and the vision system operate according to the same basic premise: requests to find chunks conforming to a particular pattern are made, and the result is placed into an associated buffer (note, we are treating objects in the visual field as chunks here). This is the basic definition of a *content-addressable* memory system, and has a simple implementation: chunks are placed in the memory, these are examined individually to see which ones match, and one of these is chosen as the final result.

The key question is what to do when multiple chunks match. In declarative memory, the base level learning, partial matching, and spreading activation systems (as well as random noise) all combine to form an *activation* value, and the chunk with the highest activation value is returned. The visual system can be thought of as operating in a similar way, with the activation of the chunks in the visual field boosted according to their level of salience.

Our approach is to treat the different process involved in retrieval as *sub-modules*. Without any sub-modules, the activation value of all chunks in the generic memory system is considered to be zero. Since all the chunks have the same activation value, if more than one chunk matches, the system chooses randomly between them.

As part of the model creation process, Python ACT-R allows the modeller to attach particular sub-modules to adjust the activation values. This is similar to, but more flexible than, the standard ACT-R approach of turning on or off the various aspects such as partial matching or spreading activation. The following sub-modules have been written:

**Base Level:** the standard ACT-R base level learning equation. Activation increases whenever a chunk is added into memory, and decays over time. This also includes the new optimized version (Petrov, 2006).

**Spreading Activation:** increases activation for slots that have chunk values matching those in particular buffers. Allows for configuring which buffers are used and the strength of the spreading.

**Partial Matching:** allows chunks that do not exactly match the request to be returned, but at a decreased level of activation. Note that this sub-module requires a slightly more complex integration with the memory module, as it not only adjusts the activation levels, but also adjusts the set of chunks that could be returned.

**Noise:** a configurable amount of logistic noise added to the activation of each chunk. Can also add a random fixed amount to a chunk when it is first created. Equivalent to the Lisp ACT-R parameters ANS and PAS.

**Salience:** increases the activation of chunks based on the rarity of the slot values. Requires specifying a context of which chunks to consider when determining how rare particular values are.

To complete the implementation, the chunk storage system also defines a *threshold* (a value that activations must be above in order to be retrieved) and a *latency* (a value determining how lower activations require more time to recall). Furthermore, a FINST (fingers of instantiation) system is also available, which keeps track of what chunks have been recently recalled, and allows those chunks to be temporarily ignored.

The first four of these sub-modules should be familiar to ACT-R users. They form the terms in the standard ACT-R activation equation. The fifth sub-module implements the new salience estimation process (Byrne, 2006). With this approach, the declarative memory system and the vision system can constructed from the same basic system.

## Vision as Memory

The core part of a vision system can now be implemented via the new generic system. For the most part, this generic system will be familiar to those who have worked with the ACT-R declarative memory system. Using this for vision requires a few special considerations. Chunks representing the currently visible objects are stored in a generic memory system that can be thought of as representing the visual trace. The ACT-R model can then make requests of that system to find particular objects, just as is done in the standard ACT-R vision system. However, there are two major complications to this process.

First, there needs to be a way to fill the memory with the currently available chunks. For this type of function, Python ACT-R also allows for the creation of *separate production systems*. These can be specified in exactly the same way as the core production system, allowing this sort of sub-module to be easily implemented. This separate production system for vision can also be used to implement buffer-stuffing and tracking (see West, Stewart, Pyke, & Emond, 2006 for an example of using this approach for buffer stuffing).

We have found that the production system for filling the visual memory requires a very short action time (the time required for a production to fire), around 5 to 10 msec in order to be reasonably responsive to changing stimuli. Of course, since this is a separate production system it, does not affect how quickly the productions in the ACT-R procedural memory fire (normally 50 msec). Furthermore, if this system uses the Base Level sub-module and a high threshold (~5.5) and decay (~0.95), then it is not necessary to also clear the memory: the decay of activation for old chunks will automatically ensure objects that no longer are visible are not returned. This can be seen in Figure 1.

One difference between this system and the standard ACT-R vision system is that there is no bias towards returning *new* objects. Instead, Figure 1 shows that older objects can have a higher activation if they have been in the visual field for a longer amount of time. However, this effect disappears if chunks that were in the visual field on the previous cycle are included in the context set for determining salience. In this case, new objects will be returned if they a sufficiently salient.

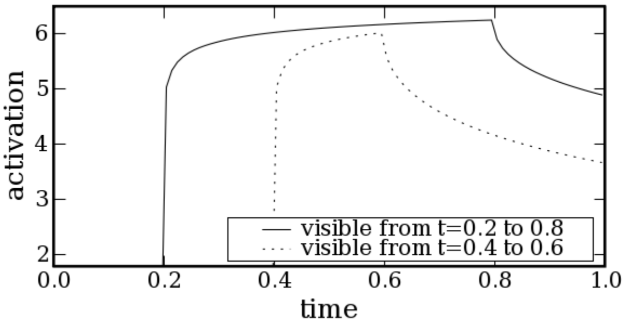


Figure 1: Activation levels of two chunks in the new vision system, visible for different periods of time. Activation uses the Base Level learning equation with d=0.95.

The second complication involves the separate "what" and "where" systems in ACT-R vision. One approach is to ignore this distinction, and not model the vision system to this fine a degree. This is the approach taken by the SOS vision system for ACT-R (West & Emond, 2002), and has been shown to be useful in situations where the details of vision are not an important part of the cognitive process being modelled.

However, an alternative approach is to implement vision using two separate memory systems: one that is filled with the *locations* of all available objects. This can be used to implement the "where" system, and results in filling a visual location buffer with a location that can be attended to. The "what" system can then be a separate memory, which is only filled with items near the currently attended visual location. This maps well onto the behaviour of the current ACT-R vision system.

In either case, the result is a memory-based system that constantly maintains a collection of chunks representing the visible objects available to the ACT-R model. As objects appear and disappear, the activation levels of chunks in this visual system change. When a retrieval request is made (i.e. when ACT-R chooses to attend to a visual object), a chunk representing that object is placed into a visual buffer. The activation levels of the chunks control this process just as they do in ACT-R declarative memory. The result is a variation of the generic chunk storage system that functions as a visual system.

### Applying Memory Sub-Modules to Vision

Implementing vision via the standard ACT-R memory system can, by itself, be seen as a useful modelling advance, as it provides a parsimonious explanation of both systems. Having both vision and memory built using the same cognitive components shows how it is possible to redeploy cognitive facilities from one task to another. Of course, significant work is still needed to rigorously compare the performance of this new vision system, the ACT-R vision system, and real human vision. This work is on-going.

However, it is worth pointing out two significant advantages that are available to this new vision system due to its integration with existing memory models. Just as the Base Level sub-module was used to implement item decay from visual memory, it is also possible to use both the Partial Matching sub-module and the Spreading Activation sub-module.

Applying Partial Matching to vision provides a natural implementation of many of the special-case features that are used in the current ACT-R vision system. Instead of the VISUAL-MOVEMENT-TOLERANCE parameter, which indicates how far away from a point an object can be and still be noticed, the partial matching system can gradually reduce the activation of chunks farther away from the point of interest. A similar approach can be taken for features such as colour, where a search for a *red* should also find objects that are *pink* (although perhaps at a slightly lower activation level).

Another new possibility arises with Spreading Activation. Here, the activation of visual chunks can be increased based on the current contents of the goal buffer (or any other buffer). This can be seen as a type of top-down processing influencing visual attention. This aspect of top-down and bottom-up salience is discussed in more detail at the end of this paper.

A detailed analysis of these possibilities is still in progress. However, the mere fact that these options appear naturally from this method of modelling vision is promising in and of itself.

## Memory as Vision

It is also possible for us to apply the salience module used in the vision system to the declarative memory system. When used in the context of vision, this module leads to pop-out effects. If this same system is applied to declarative memory, then it results in a bias towards recalling unique chunks. This can be seen as an implementation of *distinctiveness*. Chunks that have similar slot values will tend to have lower activations than chunks that have distinctive values. This makes it easier to recall rare or special items in memory (a well known memory effect).

We now turn to a specific example of the modelling capabilities gained by using the salience module to create a distinctiveness effect in declarative memory.

### Release from Proactive Interference

The Salience sub-module increases the activation of chunks with unique slot values. This can be seen as functionally equivalent to decreasing the activation of chunks with similar slot values. That is, if a chunk is similar to (i.e. is semantically related to) another chunk in memory, then the activation for those chunks will be less than it would be otherwise.

This precisely corresponds to the classic phenomenon of release from proactive interference. Here, a list of words is presented to a subject, and they are then asked to recall as many as possible (usually with a distractor task to eliminate rehearsal). Four groups of words are usually presented. The first three groups all contain words of a similar category (e.g., household items). The fourth group, however, presents words from a different category (e.g., animals). The observed effect is that recall accuracy decreases over the first three groups, and then increases for the fourth, although it does not usually increase up to the accuracy of the first group.

To implement this in ACT-R, the chunks representing words are assumed to have a slot indicating what category they are in. The model uses the Salience, Noise, and Base Level sub-modules, as well as the FINST system (to stop the model from recalling the same word repeatedly). Results from the model are shown in Figure 2.

These results show the general pattern seen in human data. The exact shape of this curve is highly dependent on a large number of factors (Hasher et al., 2002), so we have not attempted to fit this data to a particular set of results. The model currently has enough free parameters to adjust to fit any similar shape, so further data is required to constrain the model.

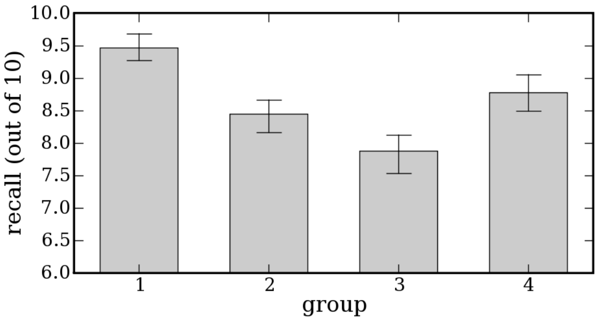


Figure 2: Recall accuracy for the Release from Proactive Interference task. The words presented in groups 1-3 are all from the same category, while group 4 is different.

In particular, one issue is the *context* for determining how rare particular slot values are. Clearly, not *all* chunks in declarative memory should be considered. Instead, we are examining the possibility of limiting the chunks to only those above a certain activation, or weighting them by their activation.

## Salience and Spreading Activation

In examining the effects of these modules in various configurations, there is a certain similarity between the Salience sub-module and the Spreading Activation sub-module. Both of them are used to increase activations of particular chunks (i.e. to make them more or less likely to be recalled) based on contextual information. For the Salience system, this increase is based on the uniqueness of the chunk among some set of chunks. For Spreading Activation, this increase is based on the similarity between the chunk and some specific buffer content.

One way of interpreting this difference is to consider Salience to be a model of *bottom-up* attention, while Spreading Activation is *top-down* attention. That is, Salience is a low-level, highly automatic process that is only somewhat controllable by high-level reasoning. It may be possible to adjust the context used to determine Salience (for example, by only considering objects in the left half of the visual field, (c.f. Byrne, 2006), or only considering chunks of a particular type), but it is not organized for fine-grained control.

In contrast, the Spreading Activation system can be used to focus attention on chunks based on their similarity to one specific focused chunk. Here, chunks that are related are connected in a semantic web, with shared chunk values allowing attention to one chunk to increase the activation of chunks that are connected to it. Applying this to vision allows for a similar focusing of attention on objects related to the current topics of thought.

It is also interesting to note that both sub-modules share a similar equation, based on the logarithm of one over the number of chunks which share that slot value. Indeed, it is possible to see Salience as a version of Spreading Activation that spreads from *every possible* chunk (or every chunk in the context set if this set is defined in some way), rather than from one particular chunk. This suggests that these two sub-modules may share a common underlying implementation as well.

## Conclusion

Inspired by the idea of cognitive redeployment, we have been exploring how the components of ACT-R can be adapted to perform different cognitive tasks. This architectural flexibility has been an important part of our Python ACT-R project, allowing for the cognitive architecture itself to be adjusted. This capability was exploited here to show that two disparate systems (vision and declarative memory) may share common underlying components.

In addition, the commonalities between vision and declarative memory allow us to take particular features from vision (or memory) and apply them to memory (or vision). If the visual salience system is connected to declarative memory, we find a natural implementation of the distinctiveness phenomenon, leading to a novel model of release from proactive interference. Furthermore, salience and spreading activation seem to form a complementary pair; one implementing bottom-up attention, and the other implementing top-down attention.

## References

Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought.* Mahwah, NJ: Erlbaum.

Anderson, M. (2007). Evolution of cognitive function via redeployment of brain Areas. *Neuroscientist*, 13(1):13–21.

Anderson, M. (forthcoming). The massive redeployment hypothesis and the functional topography of the brain. *Philosophical Psychology.*

Byrne, M. (2006). *A theory of visual salience computation in ACT-R.* 13th Annual ACT-R Workshop, Pittsburgh, PA.

Hasher, L., Chung, C., May, C., & Foong, N. (2002). Age, time of testing, and proactive interference. *Canadian Journal of Experimental Psychology*, 56(3), 200-207.

Petrov, A. A. (2006). *Computationally efficient approximation of the base-level learning equation in ACT-R.* 7th International Conference on Cognitive Modeling. Trieste, Italy.

Stewart, T.C. & West, R. L. (2006) *Deconstructing ACT-R.* 7th International Conference on Cognitive Modelling. Trieste, Italy.

Stewart, T.C. & West, R.L. (in press) Deconstructing and Reconstructing ACT-R: Exploring the Architectural Space. *Cognitive Systems Research.*

West, R.L, & Emond B. (2002). *SOS: A simple operating system for modeling HCI with ACT-R.* 7th Annual ACT-R Workshop. Pittsburgh, PA.

West, R. L., Stewart, T. C., Emond, B., & Pyke A. (2006). *Modelling emotion in ACT-R.* 13th Annual ACT-R Workshop, Pittsburgh, PA.

# Fast Learning in a Simple Probabilistic Visual Environment: A Comparison of ACT-R's Old PG-C and New Reinforcement Learning Algorithms

**Franklin P. Tamborello, II (tambo@rice.edu)**
**Michael D. Byrne (byrne@rice.edu)**
Department of Psychology, 6100 S. Main, MS-25
Houston, TX 77005, USA

### Abstract

A visual search task used red highlighting to cue the location of the target with varying degrees of probability. The probability that the cue was a valid indicator of target location on any given trial changed during the course of the experiment, and human subjects adapted to this change very rapidly. ACT-R models using the old PG-C and the new reinforcement learning algorithm matched human data from a previous experiment in this paradigm quite well, but only the model that learned by reinforcement mimicked human performance in a new experiment with dynamic highlighting validity.

## Introduction

Life is full of environments and tasks people must interact with, and usually they are not perfectly predictable. How do people learn and behave in simple probabilistic environments? Previous research using a simple visual search task included a cue, red highlighting, that had some probability of indicating the location of a target or a distractor, termed "validity" (Fisher & Tan, 1989; Tamborello & Byrne, in press). In brief, the Fisher and Tan task consisted of finding one of four possible targets in a small array of distractors, where the highlighting validity was manipulated as a between-subjects factor. This task is interesting because trials in this task typically take less than one second to complete: will humans be sensitive enough to the probabilistic nature of this rapid environment to adapt their behavior toward efficiency, or will the time-scales involved be too minute for humans to detect? People do appear to optimize a at this level in deterministic environments (Gray & Boehm-Davis, 2000), but it is unclear whether they do so in probabilistic ones.

Tamborello and Byrne found that a cognitive model implemented in the ACT-R cognitive architecture (Anderson et al., 2004) must learn the utility of each and every move of visual attention in the task in order to simulate the differential use of highlighting (termed "sensitivity") to aid visual search. Sensitivity is the response time for trials with invalid highlighting minus the response time for trials with valid highlighting. This quantity is useful as a measure of relative use of highlighting. Tamborello and Byrne's  study implemented an ACT-R model that used a learning mechanism, "PG-C" (Anderson et al., 2004), that has since been replaced with a reinforcement learning algorithm (Anderson, 2007; see also ACT-R Research Group, 2007). In brief, ACT-R fires a series of production rules, which are IF-THEN rules stating under what conditions they match, and when they match, what the model does. When multiple production rules match a set of circumstances, they compete. ACT-R resolves the competition by selecting the rule with the highest estimated utility. PG-C estimated a production rule's utility by multiplying the estimated probability (P) of a achieving a goal if that production fires by the value of the goal (G, in seconds), and then subtracting the cost (C, in seconds) of firing that production.

The reinforcement utility learning mechanism now used in ACT-R instead calculates the current production rule's utility as a function of the amount of reward propagated to that production rule. Over many applications the production rule's utility converges on the average amount of reward it receives. Others (e.g., Gray et al., 2006) have claimed that reinforcement learning algorithms work much better than PG-C in certain probabilistic environments, particularly those with costs and rewards at small time scales, such as the Fisher and Tan task. Indeed, Tamborello and Byrne (in press) speculated that this may be why their PG-C model failed to fit their human data very well at low validities. Part of the motivation for this study was to determine whether the reinforcement learning algorithm could do better with low validity highlighting than the PG-C algorithm. Additionally, a new experiment examined human ability to cope with changing environmental probabilities. Could a model built for Tamborello and Byrne's experiment generalize to the new one? Any model that hopes to explain how people behave in probabilistic environments of small time-scales will need to capture major effects from both studies.

## The ACT-R Models: Static Validity

Two ACT-R models simulated runs on the current dynamic validity experiment as well as the static validity experiment from Tamborello and Byrne (in press). The previous experiment was identical to the current study's except that highlighting validity remained static throughout the experiment and a wider range of validity conditions were run. In the static validity experiment, highlighting validity was set as a between-subjects factor at increments of 12.5% all the way from 0% to 100%. The same models were run on both the static and dynamic validity experiments.

The models were identical except for which utility learning algorithm they used, PG-C or reinforcement. On any given highlighted trial, the red item was set as the default visual location. For all trials, the default hand location was set to left hand with index finger on "4." With every move of attention, two productions competed:

"attend-red" and "avoid-red." Attend-red requested a move of visual attention to the red item, or else avoid-red requested the location of an unattended black item. If the attended item was red and the target (a "valid" trial), the model could press the appropriate key after a single shift of visual attention. If the attended item was red and a distractor (an "invalid" trial), the reinforcement model propagated a reward of -0.2 (the PG-C model marked a failure) and attended the nearest unattended black item until the target was found. If the model had initially avoided the red item, it could still choose to attend it at any time. This is also a crucial production conflict point because in the case of a standard trial, the models simply moved attention to the nearest unattended item until the target was found. The reinforcement model began a simulation run with a prior utility of 0.01 for the production that would find the red item after the black distractor had been attended. The PG-C model had 75 successes and 25 failures for this same production's priors.

## Results and Discussion

Both models fit data from the static highlighting study (the original Tamborello and Byrne experiment) fairly well. The reinforcement model correlated 0.91 (mean deviation 115 ms) with the human data, while the PG-C model correlated 0.89 (mean deviation 110 ms). Figure 1 depicts mean response times (RTs) on valid and invalid trials for the human data, the reinforcement model, and the PG-C model.

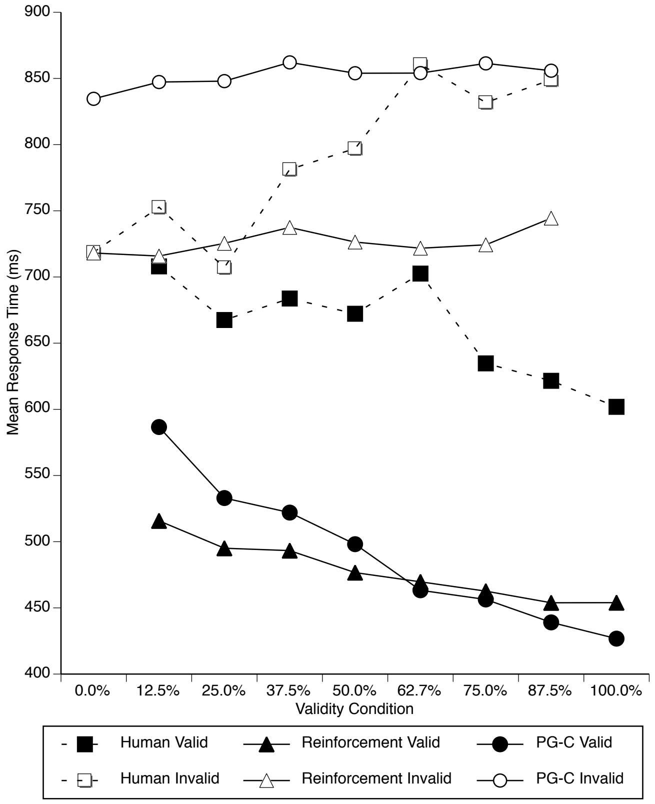


Figure 1. Mean response times of the human data, reinforcement model, and PG-C model for valid and invalid trial types for Tamborello and Byrne (in press) data.

There is a particular difficulty for the models in the previous experiment. Assuming subjects really did keep their fingers on the 1–4 number keys, and that the location of the red item was immediately available at trial onset, ACT-R predicts they should take about 300 ms to complete a validly highlighted trial when they initially attend to the highlighted item: 50 ms to decide to attend to the red item, 85 ms to move visual attention, 50 ms to decide to press the appropriate key, then 120 ms to complete the motor movement. Yet in the study reported by Tamborello and Byrne (in press), humans averaged 602 ms to complete highlighted trials when highlighting was 100% valid. That is, given that half of all trials have no highlighting at all, the other half have valid highlighting, and no trials ever have invalid highlighting, people take twice as long on average to complete valid trials as ACT-R's action latencies predict.

To average 600 ms on a trial that should take 300 ms in a best-case scenario, in the wort case subjects must be taking approximately 900 ms to complete the trial, which is about as long as it would take to search the entire five-item display. Were people really avoiding highlighting as often as using it even when it is always valid? Can either the reinforcement or PG-C model capture that effect, or are people perhaps engaging in some action not captured by the models? Humans averaged 602 ms to complete highlighted trials at 100% validity, whereas the reinforcement model averaged 454 ms and the PG-C model averaged 409 ms.

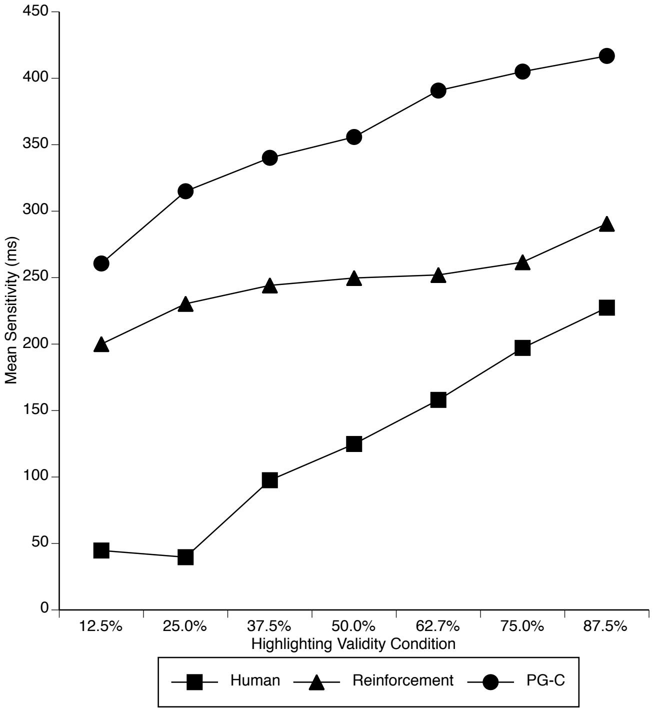


Figure 2. Mean sensitivity per validity condition for humans, the reinforcement model, and the PG-C model for Tamborello and Byrne (in press) data

Figure 2 shows the sensitivity exhibited both by humans and the two models as a function of validity condition. Clearly the reinforcement model does a better job in terms of absolute fit, though the slopes generated by the two models were about equally off (reinforcement model correlation was 0.56; PG-C was 0.53). Interestingly, the PG-C model generated a function which was too steep, and the reinforcement learning model a somewhat too shallow slope.

## The Dynamic Validity Experiment

### Method

The dynamic validity experiment replicated Tamborello and Byrne's (in press) static validity experiment, except that the validity levels changed during the experiment.

**Participants.** One hundred nine Rice University undergraduates participated to fulfill experiment participation requirements for their psychology classes.

**Design.** The experiment incorporated a mixed design utilizing two within-subjects variables, block and trial type (standard, meaning no highlighting; valid, the target was highlighted; and invalid, a distractor was highlighted), and three between-subjects variables: magnitude of validity change, direction of validity change, and change onset timing. Magnitude of validity change refers to by how many percentage points the highlighting validity proportion changed, either 34% or 68%. Direction of validity change refers to whether the highlighting became more valid or less valid. Finally, the experiment was divided into six blocks, affording short rest periods for the subjects between each block. The change in validity occurred at the beginning of either block three (termed "early") or block five ("late"). Thus the total number of between-subjects conditions was eight.

**Procedure.** Subjects were instructed to place the fingers of their dominant hand on the 1, 2, 3, and 4 keys of the number row at the beginning of the trials and to keep them there throughout the experiment. At the start of a trial, subjects viewed crosshairs for 500 ms at the intended fixation point, in the center of the computer screen. This was then replaced by a horizontal array of five different numerals. The numerals were printed in black 14-point Times New Roman font on a 17-inch CRT computer monitor at a resolution of 1024 by 768 pixels. The highlighting simply used red text. One numeral from the potential target set, {1 2 3 4} was chosen at random, while four distractors from the distractor set {5 6 7 8 9} were also chosen at random. The target and distractors were sorted randomly. The subjects' task was to find the number in the display that was less than five and immediately press the corresponding key on the number row at the top of the keyboard. Subjects were instructed to respond as quickly as possible without making any mistakes. The array disappeared upon the subject's key press, and one second later the next trial began. In the event of an incorrect response, the computer beeped and paused the experiment for two seconds. This time penalty discouraged simple guessing.

Depending upon which condition a subject was assigned to, the initial validity they encountered was 16%, 34%, 68%, or 84%. Blocks consisted of 60 trials, and with the onset of the third or fifth block the validity level changed. Subjects assigned to the 16% initial validity condition then received 84% validity, and vice versa. Similarly, subjects assigned to

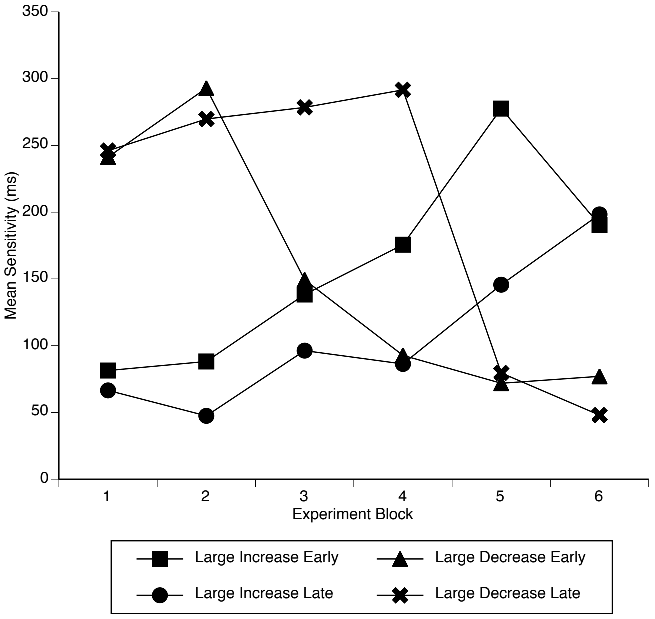


Figure 3. Mean human sensitivities for large change conditions.

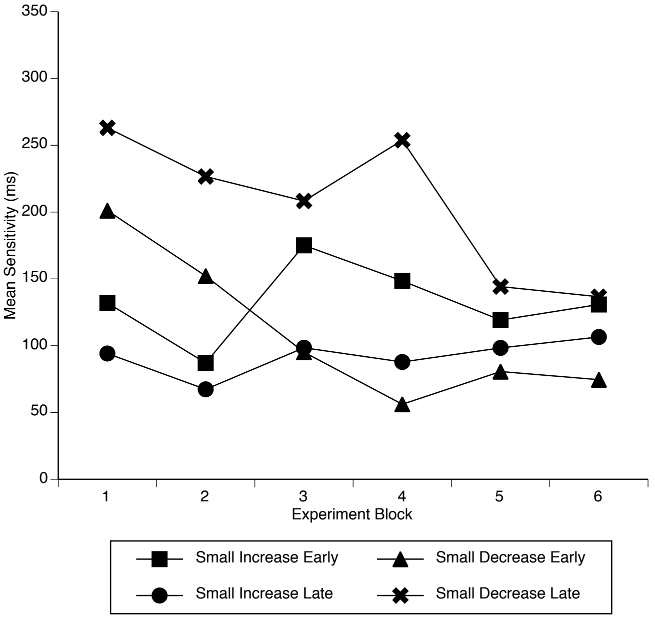


Figure 4. Mean human sensitivities for small change conditions.

the 34% initial validity condition then received 68% validity, and vice versa. The experiment required a little less than 30 minutes for subjects to complete. Instructions did not indicate that the highlighting validity rate would change during the course of the experiment. In many real-world tasks of searching some visual display, such as a web page, the user is not informed a priori about how useful visual cues will be.

Since subjects' sensitivity to changes in validity was assessed by changes in response times, we attempted to avoid confounding with practice effects. Therefore, subjects had a full block of 60 practice trials before beginning the actual experiment. The important task components to practice were searching the field to find the target and pressing the appropriate button in response. Allowing subjects to acclimatize to whatever level of highlighting validity they were assigned to before response times are actually recorded might prove detrimental to the attempt to assess their sensitivity to the highlighting validity. Therefore, practice trials were all of the standard type (no highlighting at all).

## Results and Discussion

Outliers were removed prior to statistical analysis. This was done both for single trials and entire subjects. An outlier trial was defined as a trial in which the response time was more than three standard deviations from the subject's overall mean. Those response times were replaced with the subject's mean response time. Each subject's mean response time per condition was similarly screened against the mean response time for all subjects, per condition. Any subject whose mean response time was more than three standard deviations from the mean response time for all subjects in more than one condition was considered an outlier subject. Two such subjects were found, and their data were removed from further analysis. Figure 3 depicts the mean sensitivity for each of the four conditions with large validity proportion changes while Figure 4 depicts those means for the four conditions with small validity proportion changes.

The slopes of the change in sensitivity for the block preceding change onset, the block of the change onset, and the block after the change onset were examined. These three blocks represent prior sensitivity, sensitivity under adjustment, and posterior sensitivity, respectively, and were therefore of most interest for analyzing changing sensitivity in subjects as they adjusted to new highlighting validity proportions. Among the factors size of change, direction of change, and onset of change, only direction had any significant effect on slope, $F(1,98) = 56.13$, $p < 0.01$. There was also a reliable interaction of size with direction, $F(1, 98) = 13.62$, $p < 0.01$. All other F's < 2, p's ≥ 0.17. The direction by size interaction coupled with the main effect of direction means that change direction matters, but it matters more when the change is large than small.

Did subjects in the decreasing validity conditions change their sensitivity faster in response to the changing validity than did subjects in the increasing validity conditions? Presumably subjects would notice a drop in highlighting validity faster than they would notice an increase in validity because of their greater attention paid to a higher prior level of validity. In fact the mean absolute slope for the increase conditions was 56.8 ms per block, and 82.6 for the decrease conditions. A *t*-test of the absolute slope for the two groups failed to yield a reliable difference in degree of sensitivity response to the changing validity in the two groups, $t(104) = 1.92$, $p = 0.58$. The observed effect size in this analysis was

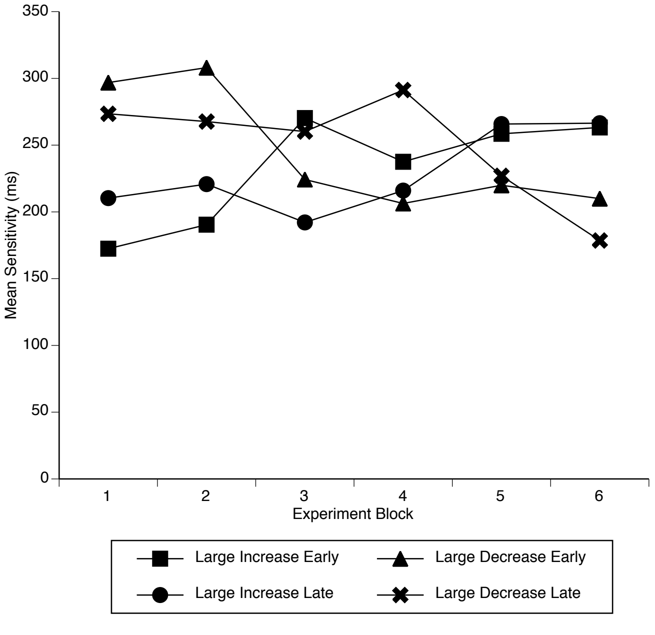


Figure 5. Mean reinforcement model sensitivities for large change conditions.

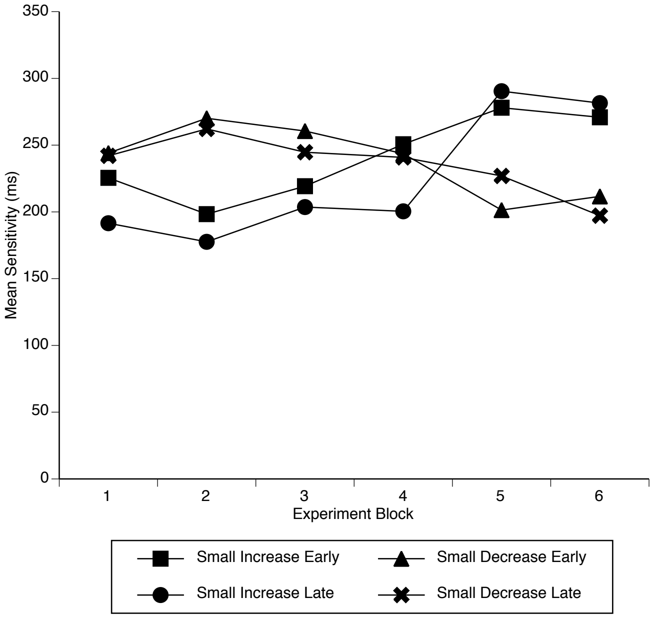


Figure 6. Mean reinforcement model sensitivities for small change conditions.

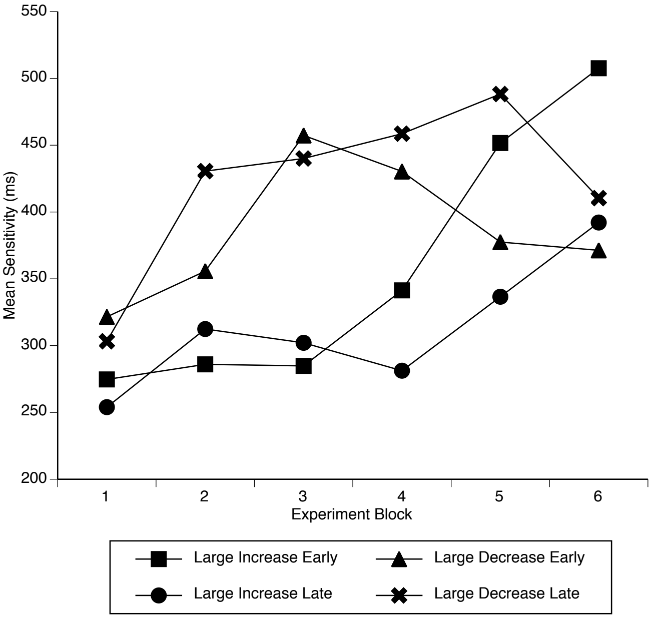


Figure 7. Mean PG-C model sensitivities for large change conditions.

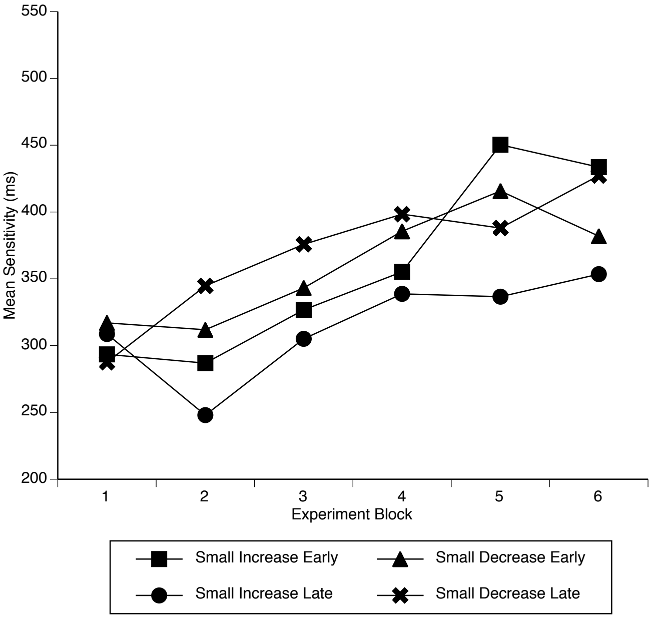


Figure 8. Mean PG-C model sensitivities for small change conditions.

medium-small, Cohen's d = 0.38, and the power to detect an effect of that size was 0.48. The current evidence is therefore inconclusive as to whether the absolute rate of change in sensitivity was different depending upon whether subjects experienced increasing or decreasing validity.

We were also surprised by the lack of reliable effect of late vs. early change; That is, it did not seem matter how much prior experience subjects had with a particular level of validity. Subjects adapted equally well with two additional blocks of experience in a particular validity condition.

## The ACT-R Models: Dynamic Validity

As for the dynamic highlighting task, the reinforcement model correlated 0.64 with the human data (mean deviation = 115 ms), while the PG-C model correlated 0.75 (mean deviation = 110 ms). Figures 5 through 8 plot mean sensitivity for the reinforcement and PG-C models. Compare these with Figures 1 and 2. Note how the reinforcement model generally shows the same qualitative trends in its sensitivity functions as do humans. The PG-C model has some hint of those trends, but while the effect size for the large decrease conditions is on the order of 200 ms for the humans and 100 ms for the reinforcement model, it only approximately 50 ms for the PG-C model. Note also how the overall size of the sensitivities keeps increasing throughout the duration of the experiment for the PG-C model (linear $F$ (1, 7) = 29.11, $p$ = 0.001), but not the human data (linear $F$ (1, 7) = 0.88, $p$ = 0.38) nor the reinforcement model (linear $F$ (1, 7) = 0.04, $p$ = 0.84) (Figure 9). The sensitivity function of the reinforcement model generated a slope more closely resembling that of humans ($r$ = 0.86, $p$ = 0.007) than did the PG-C model ($r$ = 0.25, $p$ = 0.55) (Figure 10).

The generally better qualitative fit of the reinforcement model over the PG-C model suggests that learning in a domain like the present study's experiment, a dynamic, probabilistic one of small scale, probably requires a flexible strategy that is more strongly influenced by recent experience than more distantly past experience. One major difference between the standard PG-C algorithm and the reinforcement algorithm is that the reinforcement algorithm uses the last reward propagated to the currently rewarded production to compute the current reward. That last reward, of course, included its previous reward, and so on. However, the PG-C algorithm weighted all past events the same. Lovett (1998) implemented a model of a probabilistic task using a variant of the PG-C utility learning algorithm that incorporated a decay mechanism, though unfortunately this algorithm is computationally expensive. It may be that the need for a decay mechanism to model a probabilistic task indicated the necessity for a fundamental change to ACT-R's utility learning algorithm that would discount distally past experience.

On a side note, a crucial factor in generating reasonable fits to the human data was the degree of prior utility advantage for the production that would seek the red item after a black distractor had been fixated. Models which set this prior too low actually exhibited strong, negative initial sensitivity. We were surprised by how unstable the model's performance was as a function of this single prior utility, which again suggests the critical importance of small time advantages.

Finally, the astute reader will have noticed that the models either attend to the highlighting or avoid it, with no strategy

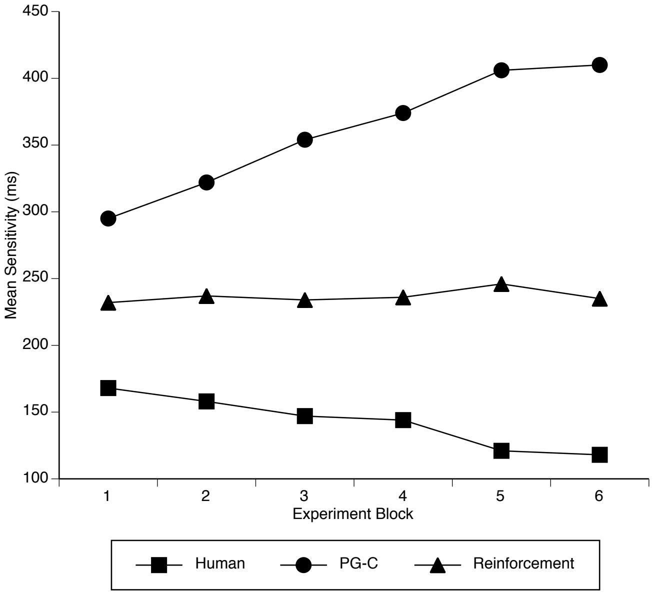


Figure 9. Sensitivity across experiment blocks.

that simply ignores the highlighting. We did this because the highlighting provides information about probable target location as long as validity is not equal to random chance of any one item being the target. When validity is low, one can use highlighting to rule out one search location, and thus by avoiding that location stand to gain approximately 200 ms over a strategy that simply ignores the information provided by highlighting. Gray et al. (2006) demonstrated that people do tend to be efficient in tasks that take place at small time scales. If Gray et al.'s findings generalize to the Fisher & Tan task, then it stands to reason that people will take advantage of information at their disposal for the sake of speed. However, it would still be desirable to actually test this assumption in the future with models that can ignore highlighting rather than or in addition to avoiding it so that such a possibility could be ruled out by data that speak directly to the matter rather than by assumptions based on prior evidence.

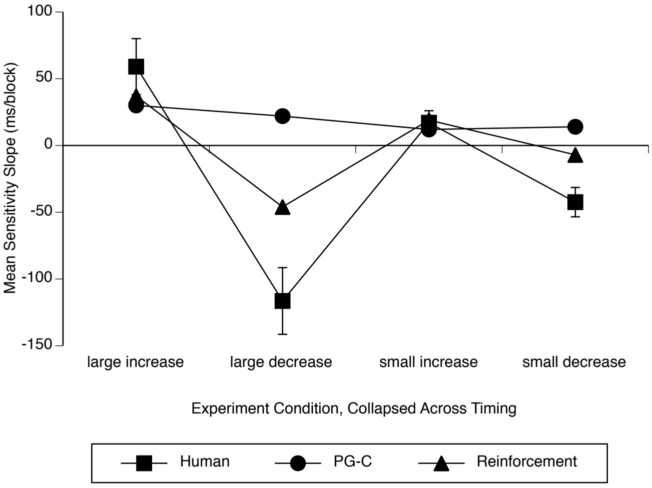


Figure 10. Slope of sensitivity functions, collapsed across timing conditions. Error bars indicate standard error of the mean. Note the interaction of change size and direction in the human and reinforcement model data.

## References

ACT-R Research Group. (2007). Unit 6: Selecting Productions on the Basis of Their Utilities and Learning these Utilities. Accessed on January 31, 2007 from http://act-r/psy.cmu.edu/

Anderson, J. R. (2007). *How can the human mind occur in the physical universe?* New York: Oxford.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Quin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111*, 1036-1060.

Fisher, D.L., & Tan, K.C. (1989). Visual displays: The highlighting paradox. *Human Factors, 31(1)*, 17 – 30.

Gray, W. D., & Boehm-Davis, D. A. (2000). Milliseconds matter: An introduction to microstrategies and to their use in describing and predicting interactive behavior. *Journal of Experimental Psychology: Applied, 6,* 322-335.

Gray, W. D., Sims, C. R., Fu, W. T., & Schoelles, M. J. (2006). The soft constraints hypothesis: A rational analysis approach to resource allocation for interactive behavior. *Psychological Review, 113(3)*, 461 – 482.

Lovett, M. (1998). Choice. In J. R. Anderson & C. Lebiere, *The atomic components of thought*. Mahwah, NJ: Erlbaum.

Tamborello, F. P., II, & Byrne, M. D. (in press). Adaptive but non-optimal visual search behavior in highlighted displays. *Journal of Cognitive Systems Research*.

# Towards a Complete, Multi-level Cognitive Architecture

**Robert Wray[1] (wray@soartech.com), Christian Lebiere[2] (cl@cmu.edu), Peter Weinstein[3] (peter.weinstein@altarum.org), Krishna Jha[4] (kjha@atl.lmco.com), Jonathan Springer[5] (springer@reservoir.com), Ted Belding[6] (ted.belding@newvectors.net), Bradley Best[7] (bradjbest@gmail.com) & Van Parunak[6] (van.parunak@newvectors.net )**

[1]Soar Technology, Inc., 3600 Green Court Suite 600, Ann Arbor, MI 48105
[2]Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh PA 15213
[3]Altarum Institute, 3520 Green Court Suite 300, Ann Arbor, MI 48105
[4]Lockheed Martin Advanced Technology Labs, 3 Executive Campus, Cherry Hill, NJ 08002
[5]Reservoir Labs, 632 Broadway Suite 803, New York, NY 10012
[6]NewVectors, 3520 Green Court Suite 250, Ann Arbor, MI 48105
[7]Adaptive Cognitive Systems LLC, 1709 Alpine Avenue, Boulder, CO 80304

## Abstract

The paper describes a novel approach to cognitive architecture exploration in which multiple cognitive architectures are integrated in their entirety. The goal is to increase significantly the application breadth and utility of cognitive architectures generally. The resulting architecture favors a breadth-first rather than depth-first approach to cognitive modeling by focusing on matching the broad power of human cognition rather than any specific data set. It uses human cognition as a functional blueprint for meeting the requirements for general intelligence. For example, a chief design principle is inspired by the power of human perception and memory to reduce the effective complexity of problem solving. Such complexity reduction is reflected in an emphasis on integrating subsymbolic and statistical mechanisms with symbolic ones. The architecture realizes a "cognitive pyramid" in which the scale and complexity of a problem is successively reduced via three computational layers: Proto-cognition (information filtering and clustering), Micro-cognition (memory retrieval modulated by expertise) and Macro-cognition (knowledge-based reasoning). The consequence of this design is that knowledge-based reasoning is used primarily for non-routine, novel situations; more familiar situations are handled by experience-based memory retrieval. Filtering and clustering improve overall scalability by reducing the elements to be considered by higher levels. The paper describes the design of the architecture, two prototype explorations, and evaluation and limitations.

## Introduction

Cognitive architecture research seeks to define a collection of integrated processes and knowledge representations that provide a general foundation for intelligent behavior across an increasingly broad spectrum of problems. While most cognitive architectures seek to cover the requirements for general intelligence, all today fall far short of that goal. They represent "works in progress" that tend to have niche strengths in a few problem domains.

Investigations of human psychology suggest that human behavior derives from the integration of three mechanisms: 1) powerful perceptual and "pre-cognitive" processing; 2) memory, along with processes that contextualize and generalize experience to make it readily applicable in future situations; and 3) satisficing ("good enough") reasoning. As an example, Best (2005) found that humans generate reasonably good solutions to the NP-complete traveling salesman problem (TSP) in nearly linear time for problems up to about 40 cities. Humans accomplish this via parallel perceptual processes that group cities by proximity, evaluation of goodness of path (dependent on experience and memory), and a localized, serial search process. We introduce C3I1 ("three cognitions, one intelligence.") a novel cognitive architecture that integrates three cognitive-architecture-level approaches, each targeted to one of these distinguishing human capacities.

Our approach is similar to the common approach to cognitive architecture development, which adapts and incorporates existing algorithms into an architecture. Cassimatis (2006) and Chong and Wray (2005) offer recent examples of the typical approach. Our approach is different primarily in terms of scale: rather than integrating individual algorithms, we map three cognitive-architecture-level systems to each of the three functionalities outlined above. The methodological hypothesis motivating this approach is that by integrating these complete, relatively mature approaches, we can enable faster exploration of alternatives in design space than approaches that integrate sub-components of existing cognitive architectures. If true, this will enable more rapid and thorough evaluation of the three-level architecture than constructing *de novo* a new architecture designed around the functional capabilities.

An obvious potential drawback is the potential redundancy in functional capability at each level, with consequences for software engineering complexity and run-time performance. The architecture reported in this paper is not the end-goal, but rather a description of the results of our experiences along this methodological path. The results suggest that even the superficial integrations we report here have significant functional value. However, long-term, our goal is to refine and deepen the initial integrations described here, based on lessons learned from the explorations.

This paper reports the initial progress and lessons learned in terms of evaluating the methodological hypothesis. We report progress toward three goals: 1) defining the functional requirements for each level of the architecture, 2) evaluating the utility of the general design pattern via prototype implementations, and 3) identifying functional synergies that could result long-term in tighter integration. In order to explore design consequences empirically, we have chosen specific candidate systems for each level of the system. Pre-cognitive processing, or Proto-cognition, is accomplished via configurations of swarming algorithms. ACT-R is used for memory formation and expertise-based retrieval (Micro-cognition), serving as a bridge between the sub-symbolic Proto-cognitive component and a symbolic Macro-cognitive layer. Macro-cognition, the locus of symbolic reasoning and explicit knowledge, is realized via Soar. Each instantiation was chosen because of its maturity and capability with respect to the practical realization of the target functionality, rather than any regard to cognitive plausibility or a definite commitment to a particular choice of method for any level. Other approaches to the same functionality could be substituted, allowing for an evaluation of the properties of constituent mechanisms.

## Design Principles

The design of C3I1 is inspired by two principles particularly important in the human cognitive architecture: progressive filtering and structuring of perception ("cognitive pyramid") and tight integration of functionalities and modalities.

### The Cognitive Pyramid

There is a computational trade-off between the quantity of data that can be processed at any one time and the sophistication of reasoning applied to that data. The human cognitive architecture solves this problem by collapsing and integrating large volumes of perceptual information into more abstract aggregates that can then be stored, retrieved, manipulated and composed more tractably. We refer to this volume-complexity tradeoff as a "cognitive pyramid". When the volume of incoming data is high, the architecture applies local processing to the data, the function of which is to aggregate and filter. At progressively higher levels in the architecture, because total data volume is reduced, more computationally expensive methods (such as symbolic reasoning) can be brought to bear tractably.

While this principle has been laid out in purely computational terms, the architecture of the human brain follows similar principles for information processing. When information reaches the brain, it is first processed by the sensory cortex. Subsequent rapid processing greatly reduces the complexity of the data into increasingly abstract representations. Quick reaction can be obtained by comparing that abstracted information to stored patterns in massively parallel procedural and declarative information stores hypothesized to be located in the basal ganglia and neocortex, respectively. If more sophisticated processing is required such as reasoning or planning, relatively slow, sequential processing occurs, controlled by contextual structures in the prefrontal cortex. While we do not equate the three cognitive levels with specific brain structures, nor argue for the neural plausibility of our algorithms, the organizing principles bear substantial similarities.

### Tight Integration

A second architectural principle for C3I1 is tight integration: fine-grained, full-spectrum interaction between the cognitive levels. Tight integration may seem like an unintuitive requirement, given the components to be integrated. However, both Soar and ACT-R suggest that tight integration results in better leveraging of the inherent power of individual components (Anderson et al. 2004; Jones and Wray 2006). With loose integration, information about the situation becomes "trapped" inside subsystems and cannot be efficiently communicated. In a tightly integrated approach, all cognitive components will share a common memory, integrated control process, and a shared language for communication with individual components.

The neurological architecture of human cognition is characterized by both forward and reverse projections from interconnected areas. This observation inspired a design choice that each information flow between cognitive layers would have a reverse learning flow. For example, even in the prototype we describe, Proto-cognition's filtering and clustering are modulated by signals from Micro-cognition indicating goodness-of-fit and applicability of high salience clusters to on-going problem solving. This allows the architecture to adapt and optimize itself to the nature of its processing and the structure of its environment. In this paper, we focus on describing the implemented and tested version of the architecture. This prototype currently falls far short of a tightly integrated architecture, but suggests specific opportunities for beneficial, synergistic integrations as discussed in the conclusions.

## C3I1 Implementation

This section outlines the role of the three primary components of the architecture. Importantly, the individual architectures proposed for each level are not unique choices; other architectures or methods likely could have been applied at each level. The focus instead concerns how these more general systems have been applied to the specific role proposed for each level in the architecture.

### Proto-cognition

The primary role of Proto-cognition ("Proto") in the architecture is to reduce the scale of raw input data. This general goal may be accomplished in a variety of ways; the current design envisions two functions: 1) organizing data into structured, hierarchical representations and 2) providing skeletal subsolutions for the higher layers. These processes reduce the effective search space for other layers. Currently, we have explored two specific swarming-based mechanisms for Proto: SODAS (decentralized hierarchical clustering) and marker-based stigmergy (topological sorting).

SODAS (Parunak et al. 2006) implements dynamic decentralized hierarchical clustering via swarming. In SODAS, nodes represent either data nodes or summaries of subtrees of nodes; links represent parent-child relationships, so that the network forms a hierarchical tree of data clusters. SODAS' merge operator is identical to classical agglomerative hierarchical clustering, providing a fully general hierarchical clustering method. SODAS' promote operator provides SODAS with the additional ability to reorganize and improve existing cluster trees, making it arguably more general than most other clustering methods.

Marker-based stigmergy depends on special-purpose information ("pheromones") that agents deposit in the environment to support coordination. An example in nature is ant foraging, where ants that have found food leave pheromones as they return to the nest, which attract other ants to the food source. In the context of C3I1, it is used to provide skeletal solution frameworks to the higher layers for such problems as topological sorting. Here, the nodes are spatial locations, place agents, and each node has links to each of its spatial neighbors (for instance in a 2D lattice). Swarming agents roam over this spatial topology, reacting to pheromone concentrations and other agents, and depositing their own pheromones.

## Micro-cognition

The role of Micro-cognition ("Micro") is to store and deploy problem-solving expertise. A key part of the ability to solve complex problems efficiently is in learning from experience and then deploying that knowledge as a substitute for the time-intensive reasoning processes by which it was accumulated. One can view the process as a space-time tradeoff, where the system trades increased storage space for expertise in exchange for reduced processing time in finding an expert-level solution. This approach relies on the engineering assumption that storage is increasingly less expensive and that special-purpose hardware can provide storage with the right properties (robust associative content-based retrieval, constant time access, etc.); response time is usually a less negotiable constraint. The C3I1 prototype uses ACT-R (Anderson and Lebiere 1998) for the Micro-cognitive layer. ACT-R includes functionality for generalizing patterns that take the form of either declarative knowledge, capturing a solution to various sub-problems, or procedural skills, resulting in more efficient execution.

Micro is defined by sequential cognition that relies on a hybrid mix of symbolic and statistical knowledge, such as applying a set of patterns to the current situation to make the best guess under tight computational constraints. The computation is a mix of sequential and parallel, similarity- and history-based processes and bottom-up and top-down constraints. Key operations include retrieving declarative knowledge from long-term memory and selecting the most promising rule from the procedural skill set. These operations are implemented by a combination of symbolic pattern-matching and subsymbolic processes:

**Partial matching** compares stored memories to the current problem to determine the closest-matching pattern. A current problem will seldom exactly match previous experience. Partial matching uses similarity-based generalization similar to that of neural networks to find the closest match to an item in memory. While the pattern is described symbolically, the internal mechanism generalizing that pattern is statistical.

**Bayesian learning** operates at the subsymbolic level of declarative memory. Its role is to guide the retrieval process to the most likely pieces of relevant information. It combines parameters derived from computations at the Proto level that reflect factors such as the compatibility between problem and solution, together .with quantities learned at the Micro level that capture environmental heuristics such as recency and frequency of use.

**Reinforcement Learning** controls and tunes procedural skill acquisition, using successes and failures as input. Subsymbolic utilities are associated with skills to control selection and application and learned using online reinforcement learning algorithms. RL alone typically scales poorly in combinatorial state spaces, but using RL with general production rules instead of discrete, combinatorial states significantly improves efficiency.

## Macro-cognition

The primary task of Macro-cognition ("Macro") in C3I1 is to perform a deliberate "problem search" to solve a specific (sub)problem initiated by Micro. In essence, Macro brings knowledge to bear in deliberating over non-routine, novel tasks. Soar (Newell, 1990) is used for Macro-cognition.

Problem search is potentially expensive. In order to reduce the cost of problem search, Macro should ensure responsiveness to the current problem (not waste cycles working on an out-dated problem), support multiple methods of reasoning (apply the most appropriate reasoning to the problem), support efficient knowledge search (minimize the cost of bringing knowledge to bear) and improve performance with experience. Soar implements and integrates a number of influential ideas and algorithms that support these requirements, including:

**Efficient, pattern-directed control**: The flow of control in Macro is determined by context and the associations it triggers. Macro requires least commitment representation; there should be no fixed, design-time mapping between the justifications for an activity and its implementation. This distinction supports multimethod reasoning because the approach to some specific activity (e.g., resolving a request from Micro) can be pursued in multiple different ways.

**Reason maintenance**: Reason maintenance ensures that Macro is responsive to the environment. It embeds dynamics of belief change directly in the architecture. This facilitates multimethod reasoning because reason maintenance is implemented consistently across any reasoning method.

**Meta-reasoning**: Macro can recognize when it has no available options or has conflicting information about its options and, in response, creates a goal to resolve this "impasse." This process facilitates multimethod reasoning, because different approaches (e.g., reasoning by analogy vs. lookahead planning) can be identified and brought to bear to attempt to resolve individual subgoals.

**Knowledge compilation**: Macro should improve with experience, converting the results of reasoning to new knowledge representations that summarize reasoning. This allows the system to skip reasoning steps when a similar situation is later encountered, improving responsiveness.

## Memory and Control

In addition to the individual components, the architecture requires memory store(s), a control process, and a message passing language and infrastructure.

**Unified Memory** A significant design decision was whether to create a single memory, shared by each of the three layers, or to use the individual memory systems available within each of the existing components. A unified memory would be more efficient than passing partial results, especially given high volumes of data (e.g., Proto passing up clustering results). A shared memory also ensures consistency by making sure that all cognitions work from the same problem state. While, in theory, separate memories for each cognition could be synchronized, in practice, resolving synchronization issues (such as race conditions) is oftentimes difficult and labor-intensive. On the other hand, the three cognitions currently have significantly different memory models. A common memory model might be inefficient for the operations of individual cognitions. For instance, memory organized for Soar and ACT-R might not be a practical organization for use by swarming. As well, reimplementation costs are significant. While the various cognitions are somewhat modular, changes to their memory systems would have broad repercussions throughout their implementations. We used the distinct memories of the individual cognitions in the prototype, but are also pursuing unified memory integration.

**Integrated Control** Explicit control is needed to coordinate the three cognitions. Integrated control would mean a single process that would invoke the primitives provided by the three cognitions. Integrated control would also enable it to be more adaptive and generally more sensitive to the operations of the cognitions than control structures located within each cognition or outside of the cognitions entirely. The prototype assumes a simple up-and-down data flow: Proto computes a first pass, Micro attempts to apply its knowledge to some subproblem, and Macro is called upon if Micro's expertise is insufficient for the problem. Results then flow down to Micro and Proto and the cycle repeats. In parallel, there are reverse flows for each information flow, providing modulation and feedback. We also developed explicit control strategies for each prototype application. This approach is not an appropriate long-term solution, but it allowed an exploration of the requirements and implications of integrated control.

**Inter-cognition Communication & Control:** In order to communicate with other components, and with external systems that present "problems" to the architecture, we developed a XML-schema-based language. Message types are organized into three categories:

**Knowledge management**. Messages (e.g., `AddChunk`, `RemoveChunk`) allow management of problem data and general domain knowledge. For example, any of the cognitions could send an `AddChunk` message when a new inference or cluster was created, which would then be communicated to the other cognitions.

**Cognition invocations & responses**. Messages such as `MicroRequestInference` encode requests between cognitive components and corresponding responses (e.g., `MacroRespondInference`).

**Operational protocols**. Protocols specify how to declare problems to C3I1 and how to deliver solutions.

The set of messages is fixed and general across problem domains. We developed functions within Swarming, ACT-R, and Soar to interpret the messages in a domain-general manner. The types of data contained in the messages are problem-specific. For example, the AddChunk message is always used to add data, but the structure of the data added will be different for different applications. Thus the language is modular; there is a generic part into which domain-specific parts can be inserted. XMLBlaster is used to transmit messages between cognitions, facilitating a distributed, flexible experimentation environment.

# Prototype Explorations

In order to elicit requirements and empirically explore implications of C3I1, we built prototypes for two very different application domains: evidence marshalling and unmanned-aerial vehicle (UAV) route planning.

## Evidence Marshalling

Evidence marshalling is the process of collecting and organizing information (evidence) to support a hypothesis. We used a simple but realistic data set used to train intelligence analysts. In this scenario, many distinct reports provide hints about a possible coordinated terrorist attack in three different cities. We applied C3I1 to uncovering specific connections in this dataset (after hand-encoding facts from the reports into C3I1 assertions). Proto-cognition uses SODAS to cluster similar facts together. Micro applies its expertise to "link" facts and clusters and infer new conclusions. When it lacks knowledge to make a link, Micro calls Macro. Macro, using ontologies and domain knowledge, searches for a link between facts of interest.

The primary lesson learned from this prototype was that the basic functional roles we had defined in the design demonstrated utility in application. It also allowed us to define and implement the initial version of the interaction language and give the cognitions the ability to pass messages. However, we also quickly recognized that evidence marshalling was a poor prototyping domain: there would be significant expense to encode a very large database of facts, but a large store of facts was needed to demonstrate the power of Proto's clustering. The example problem was small enough (< 1000 assertions) that the problem could be readily addressed within ACT-R or Soar.

## Route Planning for Unmanned Aerial Vehicles

The application, a simplified representation of the routing of an unmanned aerial vehicle (UAV), includes these elements:

- **Terrain**: The scenario was defined to take place on a 2D terrain overlaid with a 100x100 coordinate grid. Positions of other components were specified in terms of this grid.
- **Targets**: A set of target locations on the map. The UAV had to visit all targets, in an order of its choosing.
- **Threats**: A set of threat locations was defined on the map, each with a threat radius. The UAV had to complete its tour without entering any threat radius.

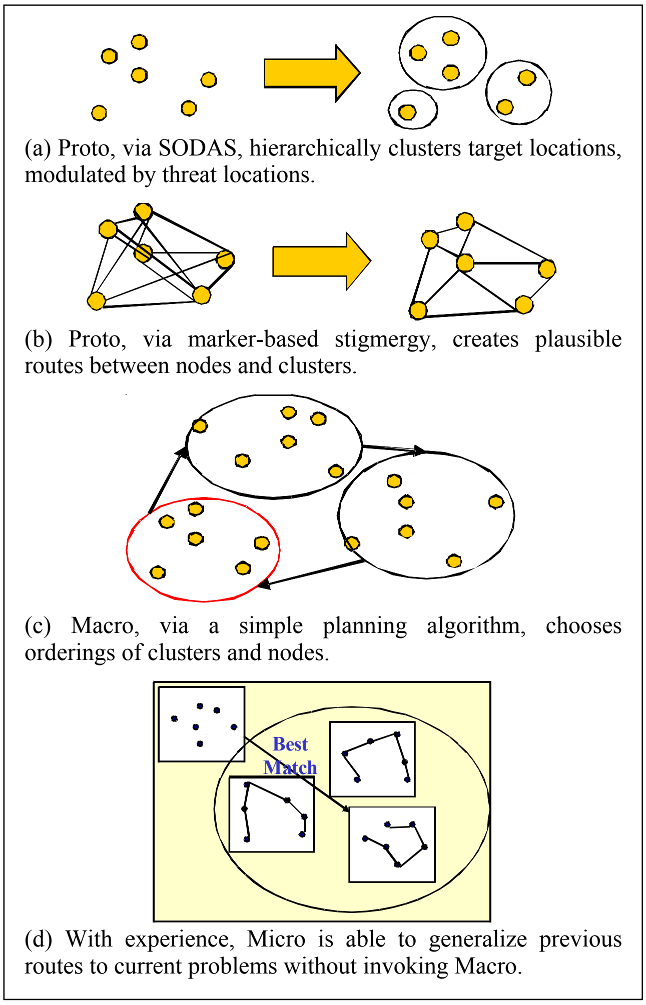
We defined this scenario to be very similar to the Traveling Salesman Problem (TSP). However, threats and dynamic ("pop-up") locations were included in the scenarios, which give the problem more realistic requirements. Figure 1 illustrates the basic solution developed, drawing heavily from the aforementioned model of Best (2005). Proto both clusters target points (in 1a) via SODAS and produces, via marker-based stigmergy, a skeletal solution (in 1b) representing possible routes between targets in a particular cluster. Although not shown in the figure, Proto performs these functions in a space warped by threats, so that simple Euclidean distance is not sufficient for producing "good" potential routes.

While Proto continues to cluster targets (and clusters of targets), Micro attempts to match groups of clusters against prior routes stored in memory. Initially, it will have no matching routes; Macro is invoked to find a routing between clusters. Because the number of clusters at the highest level is small, the routing was determined via a straightforward implementation of a TSP solution (in 1c), using distance measures between targets as computed by Proto (in 1b). We also explored the use of heuristic planning within Macro, using categorizations of rough route geometries (arcs, lines, and loops, e.g.,) to simplify route planning. However, in practice, Proto reduced the number of targets sufficiently that Macro could easily compute an analytic solution.

Once Macro finds a routing, it provides the route to Micro, which stores the route in memory. As Micro gains experience, it will increasingly be able to retrieve route segments from its memory (in 1d), without recourse to Macro. A hardware support strategy has been developed for this memory, based on an understanding of the functionality of the human visual system and strategies that are used for visual search (Pederson et al. 2006).

Once a route is computed between clusters, the system recursively develops routes within the targets (and/or subclusters) within each of the clusters in the evolving route. This approach also has the desired property that pop-up targets and pop-up threats typically introduce local changes, requiring only new routing within a cluster, rather than a high-level reconsideration of the overall route. We did explore tunings of SODAS clustering so that it would prefer making minimal changes to existing clusters when dynamic locations or threats were presented.

The approach we have taken does not guarantee a solution in linear time but is, on average, much less expensive than an analytic solution. The routes resulting from this decomposition are obviously also not always optimal but are generally close to it. We have observed some cases in which poor routes are generated and are investigating using Macro as a "critic" to recognize these situations and repair them. This additional functionality may represent a meta-cognitive role for Macro-cognition in the future.



(a) Proto, via SODAS, hierarchically clusters target locations, modulated by threat locations.

(b) Proto, via marker-based stigmergy, creates plausible routes between nodes and clusters.

(c) Macro, via a simple planning algorithm, chooses orderings of clusters and nodes.

(d) With experience, Micro is able to generalize previous routes to current problems without invoking Macro.

Figure 1: Operation of C3I1 on route planning example

## Evaluation & Conclusions

We proposed C3I1 to address a greater range and scale of problems with an architecture. We explored two different applications domains to evaluate the extent to which these goals are being achieved. We consider three general dimensions of evaluation: generality, taskability, and efficiency. While these criteria appear purely functional, they are similar to proposed tests for a theory of cognition (Anderson & Lebiere, 2003) such as behaving arbitrarily as a function of the environment and operating in real time.

**Generality**: C3I1 seeks to provide a very general computational cognitive substrate, enabling both high degrees of applicability and robustness. As a general architecture, C3I1 shows promise. The same core mechanisms and design demonstrated value in the highly symbolic, knowledge-intensive evidence marshalling domain, as well as in the more algorithmic, more dynamic routing problem. However, special-purpose mechanisms were developed within each layer for each application (e.g., in routing, threat-warped clustering, topologically-oriented memory retrieval, and the TSP sorting heuristics). Our investigations, however, suggest a fixed, domain-independent set of mechanisms for Proto is feasible.

**Taskability**: C3I1 should not only apply to a broad range of problems, but it should do so without requiring extensive developer modification and have inherent ability to adapt to the requirements and constraints of particular problems. Taskability includes the ability to handle novel variations of problems and improve with experience. It also includes extensibility: the architecture should require small (and increasingly smaller) human effort to achieve good results in different applications. The taskability of C3I1 is both less clearly encouraging and more tentative. In the near-term, significant developer involvement will be needed in each layer; essentially, the approach requires a manual decomposition and mapping of the problem to the functionality of the three layers. Over time, we believe that tighter integration of the mechanisms will improve taskability, as a more constrained architecture will guide solutions more explicitly.

**Efficiency**: General, architectural solutions are always likely to be less efficient than special-purpose ones. To compensate, overall efficiency of C3I1, especially in the core loops of processing, is critical. Efficiency includes responsiveness to specific situations, overall resource management, and scalability across problems of increasing complexity. We have evaluated the computational bounds at each level. While presentation of this analysis is not possible in this brief paper, in terms of worst-case performance, the architecture should perform acceptably well, assuming specialized, parallel hardware for Proto and Micro. A key assumption is that Proto will reduce the overall scale of the problem by several orders of magnitude, in order to make Macro problem search generally tractable. Our initial explorations suggest Proto can accomplish a level filtering and aggregation that greatly reduces the raw scale of reasoning problems, but it is an open, empirical question how generally this power can be realized across domains and applications.

In terms of the methodological hypothesis, the explorations did identify potential opportunities for much tighter, more synergistic integrations of the components. Integration between swarming and ACT-R will focus on unifying the ACT-R activation calculus and the swarming output, including devising methods for learning within ACT-R to influence Swarming. Integration between ACT-R and Soar based on the pattern of usage in the prototypes would center on impasses and chunking. ACT-R currently calls upon Soar when no relevant knowledge exists to apply directly, which is a concept very similar to the Soar impasse mechanism, but not a process directly supported in ACT-R. The information returned by Soar is a summary of the result of its reasoning. Thus, this process is functionally identical to Soar's chunking and makes it a good candidate for the focus of integration for Soar and ACT-R.

As mentioned, other alternatives are possible for the cognitions of C3I1. For example, neurologically-inspired clustering might be appropriate for perceptually-dominated domains. These observations point to the question of whether C3I1 is an architecture, making specific commitments to particular mechanisms, or a framework, representing a design pattern for building general, intelligent systems but not making specific commitments to representations and algorithms. While our intention is to follow up the promise observed in the prototypes by investigating more fine-grained integration of the current technologies, it would also be equally valid to explore and evaluate alternative technologies. Similarly, the C3I1 decomposition also serves as a suggestive guide for developing less brittle and more scalable solutions to specific classes of domain problems (such as route planning), where the goal is a point solution. All of these options point to the value of considering existing cognitive architectures as computational primitives for integration into larger and increasingly capable intelligent systems.

## Acknowledgments

The research reported in this paper was performed via the support of the DARPA Architectures for Cognition Information Processing (ACIP) program, contract FA8750-04-C-0266, administered by the US Air Force Research Laboratory. The authors would like to thank program managers Robert Graybill and William Harrod for the support of this research. DARPA Distribution Statement A (Approved for Public Release, Distribution Unlimited), DISTAR Case #8726, 3/28/2007.

## References

Anderson, J. R., and Lebiere, C. (1998). *The Atomic Components of Thought*. Mahwah, NJ: Erlbaum.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., and Qin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111(4)*, 1036-1060.

Anderson, J. R. & Lebiere, C. (2003). The Newell test for a theory of cognition. Beh. & Brain Sciences 26, 587-637.

Best, B. J. (2005) A Model of Fast Human Performance on a Computationally Hard Problem. *Proceedings of the 27th Annual Conference of the Cognitive Science Society*.

Brueckner, S. A., and Parunak, H. V. D. (2002) Swarming Agents for Distributed Pattern Detection and Classification. *AAMAS Workshop on Ubiquitous Computing*. Bologna, Italy.

Cassimatis, N. L. (2006). A Cognitive Substrate for Human-Level Intelligence. *AI Magazine*, 27(2).

Chong, R. S., and Wray, R. E. (2005). Constraints on Architectural Models: Elements of ACT-R, Soar and EPIC in Human Learning and Performance. In *Modeling Human Behavior with Integrated Cognitive Architectures*, Gluck, K. and Pew R., eds., Erlbaum, 237-304.

Jones, R. M., and Wray, R. E. (2006). Comparative Analysis of Frameworks for Knowledge-Intensive Intelligent Agents. *AI Magazine*, 27, 57-70.

Newell, A. (1990). *Unified Theories of Cognition.* Cambridge, MA: Harvard University Press.

Parunak, H. V. D., Rohwer, R., Belding, T. C., and Brueckner, S. A. (2006) Dynamic Decentralized Any-Time Hierarchical Clustering. *Fourth International Workshop on Engineering Self-Organizing Systems.*

Pederson, K., Lethin, R., Springer, J., Manohar, R., and Melhem, R. (2006) Enabling Cognitive Architectures for UAV Mission Planning. *10th Annual Workshop High Performance Embedded Computing* (HPEC 2006).

# Scheduling of Eye Movements and Manual Responses in Performing a Sequence of Choice Responses: Empirical Data and Model

**Shu-Chieh Wu (scwu@mail.arc.nasa.gov)**
San Jose State University and NASA Ames Research Center, Mail Stop 262-4, Moffett Field, CA 94035 USA

**Roger W. Remington (R.Remington@psy.uq.edu.au)**
School of Psychology, University of Queensland, St Lucia, QLD 4072 AUSTRALIA

**Harold Pashler (hpashler@ucsd.edu)**
Department of Psychology, University of California, San Diego, La Jolla, CA 92093 USA

## Abstract

The seamless and effortless integration between eye movements and cognitive functions signifies tight coordination between eye movement control and the underlying perceptual, cognitive, and motor processes. In this paper we aim to deconstruct the coordination between eye movements and manual responses exhibited in performing a sequence of choice responses through a combination of experimental and modeling methods.

## Introduction

Eye movements are integral to human cognition. The seamless and effortless integration between eye movements and cognitive functions signifies tight coordination between eye movement control and the underlying perceptual, cognitive, and motor processes. As the front end of cognitive functions, how the eyes move must bear constraints imposed by the mechanics of the oculomotor system as well as the demand of information processing. But do the eyes fixate and move simply to fulfill the need for information acquisition, or are there other factors involved in determining the scheduling of eye movements? In this paper we aim to shed light on this question by examining the timing of eye movements in performing a sequence of choice responses using a combination of experimental and modeling methods.

A great majority of research on the coordination between eye movements and cognitive functions focuses on eye movements that occur during reading. In reading, when and where the eyes move are shown to be determined by oculomotor control as well as to a larger extent by visual inputs (i.e., words), in terms of both their physical and linguistic properties (for a review, see Reichle, Rayner, & Pollatsek, 2003). However, most cognitive functions performed in daily life involve not only eye movements but also manual responses or limb movements. Does the need to produce overt manual responses in natural behavior impose further constraints on the scheduling of eye movements?

Efforts to characterize eye movements that occur in natural behavior have focused on activities with well-defined scripts, such as golf putting, driving, tea making, and block-copying (for a review, see Hayhoe & Ballard, 20005). It has been found that in these activities the eyes often move in anticipation of upcoming actions. More fascinating is the tactical timing of anticipatory eye movements. The eyes appear to move to acquire information just prior to when the information is needed in the action (Johansson et al., 2001). This just-in-time characteristic of eye movement control (cf. Ballard et al., 1995) exemplifies the type of additional constraints imposed by the process of producing manual responses.

As a foray into modeling eye-hand coordination observed in natural behavior, we devised a task complex enough to capture many of the same elements found in natural behavior but simple enough so that its underlying processes can be readily identified (Wu & Remington, 2004). The task was a typing-like task modeled after Pashler (1994). Participants viewed a row of five letters sequentially and responded to each individually. The letters were small and distributed widely so that moving the eyes to fixate each was necessary to performing the task. We evaluated the coordination between eye movements and manual responses through the timing of eye movements, the timing of manual responses, and three derived eye-hand measures: 1) eye-hand span (EHS), which represents the elapsed time between the initial fixation on a particular stimulus to the moment when the corresponding manual response is generated; 2) dwell time, which represents the duration for which fixation is maintained on a particular stimulus; and 3) release-hand span (RHS), which represents the elapsed time between the end of fixation on a particular stimulus to the moment when the manual response is generated. Dwell times and release-hand spans make up eye-hand spans.

Using this simple task we found patterns of anticipatory eye movements commonly seen in natural behavior (Wu & Remington, 2004). Figure 1 shows the pattern of observable events in one experiment that manipulated letter luminance, along with the key dependent measures (Wu & Remington, 2004). The stimuli are listed in the order in which they were responded from top (leftmost) to bottom (rightmost). Horizontal bars reflect the time from fixation to response for each stimulus (S1-S5). RT1 refers to the response time to the first letter (S1). IRI (Inter-Response Interval) is the time between the overt manual responses for each pair of successive stimuli. Fixations are represented by the shaded portion of the bars. Anticipatory eye movements were evident by the fact that the response to a given item was made during fixation on the next item.

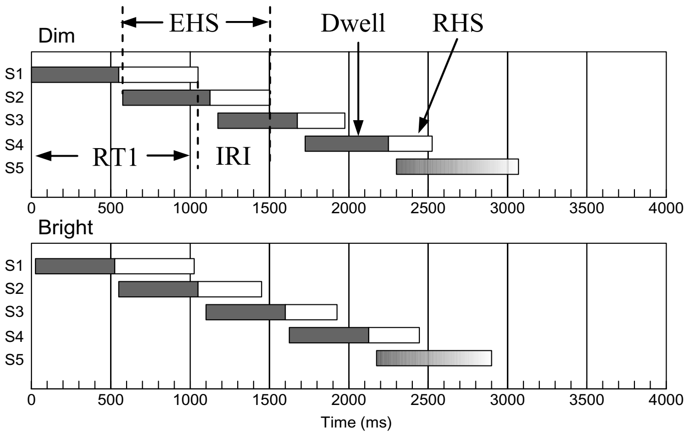


Figure 1. The time lines of fixations and manual responses in Dim and Bright luminance conditions (based on the data from Wu & Remington, 2004)

Results from this task also revealed emergent properties difficult to account for by concatenating processes underlying the constituent discrete responses. Figure 2 plots the results of RT1, IRIs, EHS, and Dwell times measured on each of the five stimuli from the same experiment (Wu & Remington, 2004). As each of the choice response was thought to include identical processes, EHS should remain constant across the series. The empirically observed EHS however decreased across the series. That is, there was a decoupling between eye movement and manual response timing. While the eyes moved across the series at a constant pace, indicated by mostly constant dwell times between S1-S4, manual responses were not produced at a constant rate (with IRI averaged around 450 ms) until after an exceedingly long delay on RT1.

The pattern of regularity in the timing of manual responses (i.e., constant IRIs) was first reported by Pashler (1994), who interpreted it in terms of the central bottleneck stage theory. Each choice response is thought to comprise three sequential stages: Stimulus Encoding (SE), Response Selection (RS), and Response Execution (RE). He posited that RS operations on the current item can proceed concurrently with SE operations on subsequent items, and that RE on the current item can proceed in parallel with RS on the subsequent items. RS is the rate limiting operation, and the duration of IRI is a direct measure of the duration of the central RS stage.

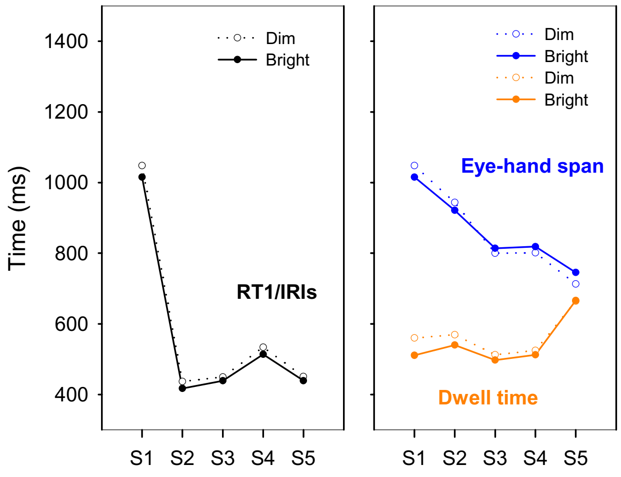


Figure 2. Patterns of RT, EHS, and Dwell results from Wu & Remington (2004)

The bottleneck theory however could not account for the substantial elevation of RT1. One possible explanation is that RT1 included a cost for performing the sequence that only affects the initial responses and dissipates over time. As the regularity of eye movements appeared to be established from the very beginning, this suggests that eye movements and manual responses could become coupled once the initial preparation cost has completely dissipated. We tested this hypothesis in the present research by examining the effects of sequence length.

## Experiment

Experiment investigates the timing of eye movements and manual responses in sequences of differing length. This would allow us to determine if they eventually appear coupled in a longer sequence of responses. It also will allow us to see if the preparation time, reflected in RT1 elevation, is a function of sequence length.

### Method

**Participants** Sixteen undergraduate students recruited from local colleges near NASA-Ames participated in the experiment for course credits.

**Apparatus** The experiment was carried out on a Pentium 4 PC with a 21-inch monitor. Participants were seated about 28 inches from the monitor. Responses were made using a PC keyboard with fingers of their right hand. Eye movements were monitored using a head-mounted high-speed eye tracker (Applied Sciences Laboratory, Model 501) with eye-head integration function, sampling at 120Hz.

**Stimuli and Display** The primary stimulus display consisted of a row of nine letters (0.13° x 0.26°) approximately 3.20° apart and centered around the middle of the display. The stimulus letters on each trial were aligned with the leftmost position, with the rest of the positions occupied by small filled squares.

**Design and Procedure** There were three sequence length conditions (3, 5, and 9). Trials of different sequence length conditions were intermixed. There were a total of 180 trials, 60 in each condition. The trials were administered in 3 blocks of 60. Prior to the experiment participants received 24 practice trials of all sequence length conditions.

Each trial began with the presentation of a fixation cross in the center of the display for 1 second. Then the fixation was erased and a small filled square appeared at the leftmost stimulus position. Participants were instructed to move their eyes to fixate the small square when it appeared and maintain fixation at that location. The small square remained for 500 ms, followed by a blank interval of 500 ms. Then 3, 5, or 9 letters appeared simultaneously aligned to the leftmost position, with the rest of the positions occupied by small filled squares. Participants were asked to look at the letters one at a time, decide what they were, and make responses accordingly. They were advised to respond as fast as possible but avoid making errors, and to not group responses. The letters were erased after the participant had responded to all of the letters on a trial. The next trial followed immediately.

## Results and Discussion

Figure 3 shows the manual response results in the three sequence length conditions. One striking feature was the perfectly aligned results from the three conditions, which all showed the typical pattern of RT1 elevation followed by short and relatively constant IRIs. In addition, in previous experiments IRIs have been found to show moderate increase over the series, with a peak on S4 when the sequence length was 5. It appeared that with extended sequence length IRIs appeared to increase slowly to a new level after every 3 items. It is not clear what caused the pattern of increase.

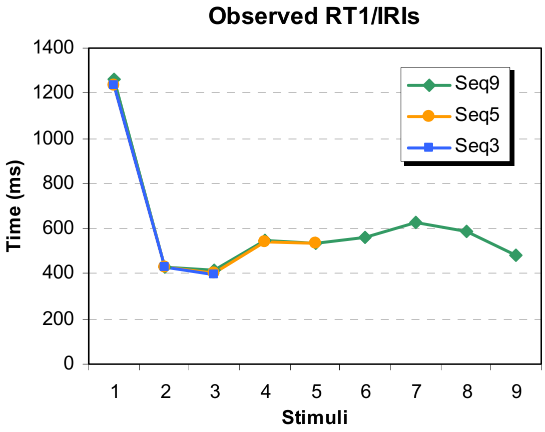


Figure 3. RT1 and IRI results from the present experiment

Figure 4 shows the results of EHS, RHS, and Dwell times in the three conditions. Because results from Sequence 3 and 5 conditions were perfectly aligned with those from Sequence 9 condition, for simplicity only the results from Sequence 9 are plotted. With an extended sequence length, EHS again decreased from the beginning but leveled out after S3, while dwell times stayed relatively constant across the series. RHS also decreased during the first few stimuli and level out after S5. Collectively, the results showed that after three responses, the timing of eye movements and manual responses appeared to be coupled, producing a constant EHS at around 800 ms.

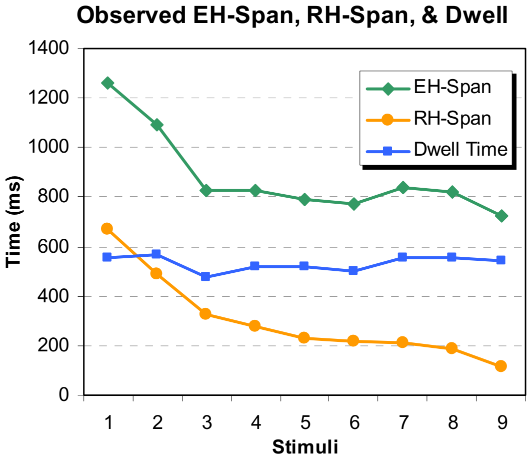


Figure 4. EHS, RHS and Dwell results from the Sequence 9 condition of the present experiment

# Model

Previously we reported a model that produced good fits to the data shown in Figure 2 (Remington, Lewis, & Wu, 2006). Here we extend that same model (basic components illustrated in Figure 5) to see how it handles the present data. Here we highlight some basics of the models. A more detailed description can be found in Remington et al., 2006.

**Assumptions** The model made three key assumptions. First, RS is the rate limiting stage, following the central bottleneck theory. Second, the eyes remain fixated on the current stimulus until SE is complete. Third, the timing of the eye movement is strategically chosen so that SE of the next stimulus is completed at the same time as RS on the current stimulus is completed. We referred to this as the “just-in-time” assumption, since it attempts to minimize wait states in central processing by assuring that perceptual processing is complete as close as possible to when the central processor becomes free.

**Model construction** The timing of eye movements and manual responses was constructed separately based on their respective hypothesized underlying components. Figure 5 presents the task model of producing a manual choice response. The timing of eye movements was governed by a separate control process that included stages necessary to initiate a saccade (I, denoting Init operator) and to maintain fixation for stimulus encoding (SE) (Figure 6). The

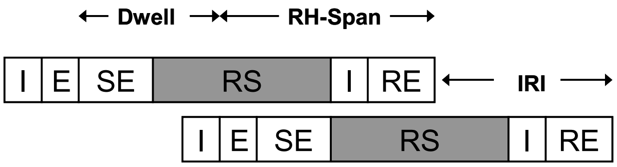


Figure 5. Task model. Processing of each stimulus consists of stimulus encoding (SE, 100 ms), response selection (RS), response execution (RE, 150 ms), and preceded by an eye movement (E, 30 ms). E and RE are preceded by Init operators (I, 50 ms). Dwell, RHS, and IRI are indicated (not to scale)

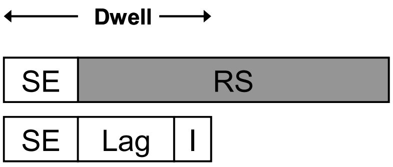


Figure 6. Task model for processes underlying fixations. SE (150 ms) represented the period of time during which the eyes remained fixated for stimulus processing. Each fixation ended with an Init operator (50 ms) that programs the next saccade.

correspondence between eye movements and manual choice response processing was borne by a Lag parameter that extends fixation beyond what is necessary for stimulus processing to realize just-in-time scheduling. Modeling eye movement timing using a separate control process fulfills the contention that these movements were generated by a lower-level open-loop process not entirely dependent on choice response processes.

**Parameters** Numerical parameter estimates for several necessary parameters were assigned values consistent with existing literature (e.g., Vera et al., 2005), described in the caption of Figures 5 and 6 . The durations for some internal, unobservable states were estimated from data based on theorized processes of the task. For example, RS duration was estimated using averaged IRIs (524 ms). Lag was estimated by first estimating dwell time on a single stimulus, which was estimated by first estimating the total time involved in completing processes on the critical path of the first 8 items, based on the assumption that the eyes remain fixated until the central processor is about to be free. An initial preparation cost was added to RT1, which was derived based the observed RT1 and estimated durations of stages involved in making the choice response.

**Simulation Results** The model was implemented and Monte Carlo simulations run in the statistical package of R. Model predictions, shown in Figures 7 and 8, were averaged results from 1000 runs. To a large extent, the model again captured the signature pattern of the data. However, the model did not produce the coupling between the timing of eye movements and manual responses found in the observed data between S3-S7, indicated by constant EH-Spans.

## General Discussion

The present experiments examined the effects of sequence length on the timing of eye movements and manual responses. Three key findings have direct implications for models of the underlying cognitive mechanisms. First was the striking lack of any sequence length effects on RT1 elevation. This clearly indicates that whatever preparation or start-up costs are reflected in RT1 elevation, they did not accrue from an item-by-item evaluation. That is, if this cost reflects motor planning then the plan is established without consideration of all the items.

Second, sequence length had no effect on asymptotic levels of IRI or dwell time. Again, this suggests that for these regular sequences a simple move-and-respond plan was implemented without examining all items in advance and iterated over items without incurring an item by item cost. It is important to note the simplicity and regularity of our sequences, which may have made these two outcomes possible. It remains to be seen whether the same patterns would be in evidence in visual search with heterogeneous and irregularly dispersed items.

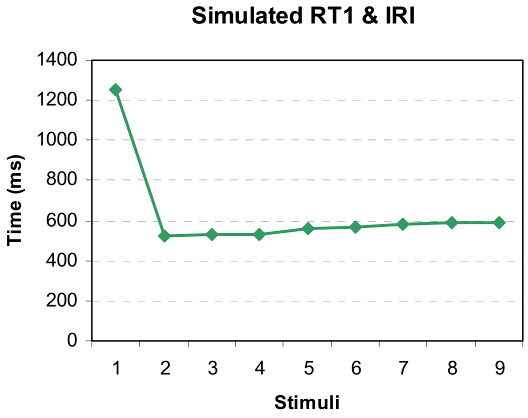


Figure 7. Model predictions for RT1 and IRIs

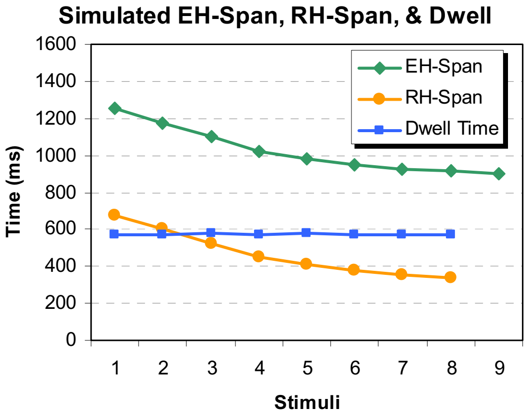


Figure 8. Model predictions for EH-Span, RH-Span, and Dwell time

The third outcome of particular importance is the apparent convergence of eye movements and manual responses to a regular tightly coupled phase. This is shown in the EHS curve of Figure 4. The flattening of the EHS after S3 indicates convergence on a more or less constant rhythmic execution of saccades and manual responses. The decoupling of eye and hand responses seen in previous experiments appears to be confined to early portions of the sequence. Indeed, this raises the possibility that the intended strategy in planning to perform the sequence is to achieve this regular rhythmic execution of eye movements and manual responses. The strategy is not reflected in the first few items because of the RT1 elevation, which requires about three items to dissipate. It is this added RT1 processing component that produces the large initial EHS and subsequent decrease that suggested that the eyes and hands were decoupled. The new empirical evidence instead suggests the possibility that the scheduling of eye movements is constrained not only by visual stimulus processing but also response production. By having participants perform choice responses of various sequence length, we showed that the degree of coordination evolved with the sequence. Coupled eye-hand responses were only found in extended sequences after the initial cost diminishes.

The model that we previously developed to fit the results from 5-item sequences had only moderate success in accounting for the results from 9-item sequences. Specifically, the model continued to predict a decreasing EHS even after 9 items consistent with the decoupled timing of eye movements and manual responses. In addition, the observed data also presented a challenge to the parameter estimates used in the model. Between S3-S7, EH-Spans were about 800 ms when IRIs were about 600 ms. If the duration of IRIs indeed represents the duration of RS, that leaves about 200 ms stimulus encoding, response initiation and execution, which according to model parameters should take 300 ms altogether (100 ms for SE, 50 ms for Init, 150 ms for RE). As it appears that observed IRIs increased with sequence length, it is possible that processes other than RS were involved in determining the timing of manual responses. In future revisions of the model we will explore the possibility that subjects plan the sequence with the goal of maintaining a constant EHS.

The way in which the model accomplishes eye movement scheduling must also be reexamined. Currently, the model estimates the saccade lag parameter by considering the total fixation time on an entire trial. Saccade lag is the parameter that delays the onset of the saccade to attempt to align the end of stimulus encoding on N+1 with the end of response selection on N. Since the total time includes preparatory operations that give rise to RT1 elevation, the estimates of dwell time assume that people consider this when programming sequences and retain these estimates long after the preparatory effects dissipate. In future versions of the model we will explore the consequences of estimating eye movement timing using the average parameter values rather than the total estimated time.

We have described the model as implementing a just-in-time assumption. It is true that the model adjusts saccade lag with the explicit just-in-time goal of having response selection free right when stimulus encoding completes. Because of the RT1 elevation and stochastic stages, just-in-time performance is not achieved in model simulation results. Estimation of the saccade lag directly from average durations of response selection, stimulus encoding, and eye movement latencies may insure better just-in-time performance in practice. Then again, explorations of alternative strategies, such as the eye-hand coupling discussed above, may produce good performance without such an assumption.

To guarantee a just-in-time schedule in model simulation results would require a very different modeling approach. Our model uses global estimates to set up eye movement timing routines. This contrasts closed-loop control models in which some explicit model construct monitors the momentary progress of cognitive processing and bases the decision to move on the completion of underlying operations (see Reichle, Rayner, & Pollatsek, 2003 for a recent review of modeling approaches to reading). In principle such closed-loop control could achieve more precise timing as eye movement initiation could be adjusted with variations in completion. However, it is unclear whether this would obtain in actual behavior, as the demands of monitoring and deciding on the adjustment could conceivably place increased demands on cognitive processing, interfering with task operations. It may well be that optimal performance is achieved not by precise timing, but by good enough timing done without drawing on central limited capacity resources for either evaluation or execution.

## Acknowledgments

This research was supported by grant #FA9550-06-1-0219 form the Air Force Office of Scientific Research, and by grant #DP0666772 from the Australian Research Council.

## References

Ballard, D. H., Hayhoe, M. M., & Pelz, J. B. (1995). Memory representation in natural tasks. *Journal of Cognitive Neuroscience, 7,* 66-80.

Hayhoe, M. M., & Ballard, D. (2005). Eye movements in natural behavior. Trends in Cognitive Sciences, 9, 189-194).

Johansson, R. S., Westling, G., Bäckström, A., & Flanagan, J. R. (2001). Eye-hand coordination in object manipulation. *Journal of Neuroscience, 21,* 6917-6932.

Pashler, H. (1994). Overlapping mental operations in serial performance with preview. *Quarterly Journal of Experimental Psychology Section A-Human Experimental Psychology, 47,* 161-191.

Reichle, E. D., Rayner, K., & Pollatsek, A. (2003). The E-Z Reader model of eye-movement control in reading: Comparisons to other models. *Behavioral and Brain Sciences, 26,* 445-526.

Remington, R. W., Lewis, R., & Wu, S.-C. (2006). Scheduling mental operations in a multiple-response sequence: Modeling the effects of a strategy to minimize variance in the timing of saccades. In D. Fum, F. del Missier, & A. Stocco (Eds.), *Proceedings of the 7$^{th}$ International Conference on Cognitive Modeling*.

Vera, A., John, B. E., Remington, R. W., Matessa, M., Freed, M. A. (2005). Automating human-performance modeling at the millisecond level. *Human-Computer Interaction, 20,* 225-265. Vickers, J. N. 1992. Gaze control in putting. *Perception, 21,* 117-132.

Wu, S.-C., & Remington, R. W. (2004). Coordination of component mental operations in a multiple-response task. In S.N. Spencer (Ed.), *Proceedings of the Eye Tracking Research and Applications Symposium 2004.* New York: ACM SIGGRAPH.

# Author index

Archer, R. 55–60

Ball, J.T 163–168, 233–234
Belding, T. 325–330
Bellet, T. 43–48
Bello, P. 169–174
Berman, M.G. 175–180
Best, B. 325–330
Bignoli, P. 169–174
Borst, J. 133–138
Bosse, T. 181–186, 187–192, 193–198, 241–246
Both, F. 199–204
Brudzinski, M.E. 139–144
Brumby, D.P. 121–126
Byrne, M.D. 319–324

Cassimatis, N. 169–174
Chandrasekaran, B. 205–210, 267–272
Chartier, S. 97–102
Chen, P.-C. 145–150
Choi, D. 115–120
Chu, A. 31–36
Cohen, M.A. 157–162
Crossman, J.A. 49–54
Czechowski, K. 211–216

Dickison, D. 79–84
Dye, H.A. 217–218

Eggert, J. 307–312

Fan, X. 145–150
Ferguson, R.W. 211–216
Freudenthal, D. 109–114, 219–224

Georgeon, O. 43–48
Gerritsen, C. 181–186
Giguère, G. 97–102
Gobet, F. 109–114, 219–224
Gonzalez, C. 61–66, 67–72
Gorski, N.A. 225–230
Gray, W.D. 127–132, 231–232, 279–283
Grintsvayg, A. 231–232, 279–283

Haddawy, P. 247–252
Harris, J. 233–234
Haynes, S.R. 157–162
Heiberg, A. 163–168, 233–234

Henning, M.J. 43–48
Herd, S. 91–96
Heuvelink, A. 199–204, 235–240
Hoogendoorn, M. 241–246
Howes, A. 31–36, 121–126

Jha, K. 325–330
Jilk, D. 91–96
Jones, R.M. 49–54
Jonker, C.M. 241–246
Juvina, I. 73–79, 91–96

Kase, S.E. 299–304
Kazi, H. 247–252
Kennedy, W.G. 253–254
Kim, J.W. 255–260
Klien, L.C. 299–304
Koedinger, K. 37–42
Kong, X. 261–266
König, T. 115–120
Körner, E. 307–312
Koubek, R.J. 255–260
Kurup, U. 267–272

Laine, T. 273–278
Laird, J.E. 25–30, 85–90, 225–230
Langley, P. 115–120
Lathrop, S.D. 25–30
Lebiere, C. 49–54, 67–72, 325–330
LeFevre, J. 7–12
Lewis, R.L. 31–36, 225–230
Lina, J.-M. 97–102
Lindsey, R. 231–232, 279–283
Liu, Y. 175–180, 285–290

Mappus, R.L., IV. 211–216
Martens, S. 91–96
Martin, M. 67–72
Mata, R. 291
Matessa, M. 55–60
Memon, Z.A. 193–198
Mille, A. 43–48
Mui, R. 55–60

Nejati, N. 115–120
Newman, L.I. 103–108

Park, C. 115–120
Parunak, V. 325–330
Pashler, H. 331–335

Pavlik, P.I., Jr. 37–42
Pearson, D. 225–230
Petkov, G. 293–298
Pine, J.M. 109–114, 219–224
Polk, T. 103–108
Pontier, M. 187–192
Presson, N. 37–42
Proulx, R. 97–102
Pyke, A. 7–12

Ratwani, R.M. 139–144
Remington, R.W. 331–335
Ritter, F.E. 157–162, 255–260, 299–304
Robare, R.J. 305–306
Röhrbein, F. 307–312

Salvucci, D.D. 121–126
Sama, R. 151–156
Schoelles, M.J. 127–132, 299–304
Schultheis, H. 19–24
Schunn, C.D. 261–266
Shahbazyan, L. 293–298
Silber, R. 163–168
Springer, J. 325–330
Stewart, T.C. 313–317
Suebnukarn, S. 247–252

Taatgen, N.A. 13–18, 73–79, 79–84, 91–96, 133–138
Tamborello, F.P., II. 319–324
Trafton, J.G. 139–144, 253–254
Treur, J. 181–186, 187–192, 193–198, 241–246

Upal, M.A. 151–156

Van Maanen, L. 1–6
Van Rijn, H. 1–6, 13–18
Veksler, B.Z. 127–132, 231–232, 279–283

Wang, Y. 85–90
Weinstein, P. 325–330
West, R.L. 7–12, 313–317
Wray, R. 325–330
Wu, C. 175–180
Wu, S. 331–335

Yen, J. 145–150
Yovtchev, T. 205–210