In Proceedings of the Sixth International Conference on Cognitive Modeling, 148-153.
Mahwah, NJ: Lawrence Earlbaum.

A Constraint-based Approach to Understanding the Composition of Skill

Richard L. Lewis (rickl@umich.edu)
Department of Psychology, University of Michigan, Ann Arbor, MI
Alonso H. Vera (avera@arc.nasa.gov)
NASA Ames Research Center, MS 262-3, Moffett Field, CA 94035
Andrew H. Howes (ahowes@cardiff)
School of Psychology, Cardiff University, Cardiff, Wales, UK CF10 3YG

Abstract

A hallmark of human cognition is the ability to compose novel behaviors from an existing repertoire of skills (Newell, 1990). These compositional processes range from search-based problem solving to the rapid, smoothly meshed perceptual-motor coordinations of well-practiced device interaction. In this paper we describe an approach to partially automating the composition of both semi-routine and highly skilled interactive behaviors. This approach, called Cognitive Constraint Modeling (CCM), is characterized by three principles: (a) descriptions of behavior are derived via constraint satisfaction over explicitly declared architectural, task, and strategy constraints; (b) the details of behavioral control (and therefore behavior composition) emerge in part from optimizing behavior with respect to objective functions intended to capture general strategic goals (e.g., go as fast as possible); and (c) the architectural building blocks are based on a simple ontology of resource-constrained cascaded processes. We show that these three principles jointly support modeling two important aspects of an interactive task: the overlapping and anticipatory behavior of highly skilled performance, and the hierarchical control of behavior evident earlier in practice. We contrast this approach with complementary approaches based on modeling the procedural learning processes themselves.

Introduction

A hallmark of human cognition is the ability to compose novel behaviors from an existing repertoire of skills (Newell, 1990). These compositional processes range from search-based problem solving to the rapid, smoothly meshed perceptual-motor coordination of well-practiced device interaction. In this paper we describe an approach to predicting the composition of both semi-routine and highly skilled interactive behaviors.

The a priori prediction of performance time and accuracy is difficult because human performance depends on many complex, interacting constraints that derive both from the task environment and from the constraints on the human cognitive, perceptual, and motor processing architecture. Skilled performance of a routine task usually involves the execution of a number of parallel but interdependent streams of activity. For example, one hand may move to a mouse, while the other finishes typing a word; the eyes track a target while a command label is retrieved from long-term memory. The details of how these cognitive, perceptual and motor processes are scheduled and the nature of their dependencies have significant consequences for the overall time requirement and error rate. Unfortunately, there are currently no modeling approaches that yield a priori prediction of human performance on semi-routine tasks that do not require the modeler to either go through complex simulations of the learning process or to hand-code the intricate sequencing of every type of action that comprises the behavior. In the next section, we briefly sketch the requirements for an effective modeling approach.

Requirements for compositional modeling

Effective tools for modeling semi-routine behavior must support at least the following features: (a) reuse of existing composed tasks; (b) automatic composition; (c) parametric flexibility; and (d) easy manipulation and inspection of theoretical/task assumptions. We discuss each of these in turn, summarizing the technical challenge associated with each.

Compositional reuse and specification One approach to modeling composition is with template modeling. Templates corresponding to common interactive behaviors such as using a mouse-driven pull-down menu can be reused without specifying again all of their internal details. Programmable cognitive architectures such as ACT-R (Anderson & Lebiere, 1998), Soar (Newell, 1990; Laird, Newell, & Rosenbloom, 1987) and Epic (Meyer & Kieras, 1997) provide considerable reuse of architectural-level components, but relatively little support for the flexible recombination of existing strategies at the task level. There are two key reasons for this. First, there are often theoretically irrelevant incompatibilities in existing task models that prevent their straightforward reuse and composition into new tasks. Second, architectural approaches lack an explicit theory of task-level behavior composition to guide the modeler in composing new tasks from existing parts—the composition theory is implicit in the architecture and learning theory.

Encapsulation The principal advantage of composing new behaviors out of existing subcomponents is the ability to work at higher levels of abstraction without worrying about the details of how to achieve the component tasks. Unfortunately, this is difficult to capture in models of skilled human behavior. Consider the following simple example: Suppose we want to compose two behaviors A and B in sequence. Behavior A might be a keyboard button press, and behavior B might be

Table 1: The four technical principles.

P1	Cognitive modeling proceeds by deriving descriptions of behaviors from (possibly underspecified) declarative statements of architectural, strategic, and task constraints. These constraints are an explicit, formal statement of the psychological theory.
P2	Models are based on an ontology of cascading, communicating processes in which the resource consumption of both processes and their communication channels can be reasoned about.
P3	The modeler does not specify all of the control details; rather, details of behaviors are fixed by optimizing algorithms that maximize or minimize objective functions (such as total time of behavior) specified by theorist. This embodies the hypothesis that skilled behavior is the optimal solution to a constraint satisfaction problem.
P4	Explicit goal hierarchies provide (a) the compositional glue for specifying new tasks; (b) the structure over which learning parameters are specified, and (c) the cognitive control structure that organizes the model's behavior.

a mouse-and-click. It is insufficient to specify a serial order on the two behaviors, expand them into their detailed subcomponents, and then impose that ordering constraint on the subcomponents. Human skilled behavior is not so neatly encapsulated—for example, the eye-movements associated with the mouse-movement of behavior B might begin as early as the start of behavior A. What is specified as serial at a high level might be executed with substantial parallelism at the lower levels, and this can have significant implications for overall performance time and accuracy.

In short, the problem is the seamlessness of human behavior. To first approximation, skilled human behavior does not respect task boundaries—certainly not the potentially arbitrary boundaries introduced by a modeling decomposition. In particular, *anticipatory acts*, in which aspects of a later behavior intrude on or modulate an earlier one, are a hallmark of high degrees of skill, and can be seen in behaviors ranging from coarticulation in speech to aggressive eye-movements in computer interaction. The result is that it is not possible to be concerned only with the "inputs" and "outputs" and temporal relationships of component behaviors—the internal details, and how they mesh, do matter.

Parametric flexibility As explained above, most realistic task scenarios require a mix of skilled and novel components. The modeling approach must therefore support flexibility in specifying the degree of learning of subcomponents of the overall task. Current approaches permit either programming only at a fixed skill level (e.g., Epic), or use implemented learning mechanisms, such as production compilation, to simulate the progression through various stages of learning (ACT-R). Although this approach has significant theoretical advantages by providing explanatory accounts, in practice it makes modeling difficult, and it becomes especially cumbersome when the goal is to model mixes of skilled and novel components within a single task. It also puts the specific learning mechanisms, which themselves are evolving hypotheses, on the critical path.

Easy manipulation of theory and task specification Architectural assumptions should be as explicit, inspectable, and modifiable as task specifications. This is important for both the applied HCI practitioner and the cognitive scientist.

Unfortunately, cognitive architectures embody architectural assumptions in underlying program code, and are not easy to change. This would not be a problem if the details of an architectural theory were stable and comprehensive enough to be applied to a wide range of tasks. But in the immediate future many modelers will find it valuable to easily manipulate and add architectural assumptions. Examples of such assumptions may range from relatively low level commitments such as the distributions of times for specific kinds of perceptual processes, to constraints on short term memory retention of specific kinds of task information, to major architectural choice points such as the existence of a central response selection bottleneck.

Addressing the requirements

In the remainder of this paper we describe and illustrate a modeling approach characterized by the three principles of Cognitive Constraint Modeling (CCM) and a fourth principle of composition (Vera, Howes, McCurdy, & Lewis, 2004). Table 1 summarizes these principles. The first principle defines the nature of the cognitive modeling infrastructure; the remaining principles constitute a set of hypotheses about the nature of the human cognitive architecture. In the next sections we explain these principles, and how they address the significant modeling issues above, by demonstrating them in a prototype modeling tool. We also briefly summarize new empirical evidence for the psychological reality of hierarchies in controlling behavior.

Deriving behavior descriptions from constraints Constraints on behavior can be specified in terms of predicate calculus statements relating entities in the environment, tasks, and psychological processes. An entity can be represented as a set of elements where each element is an ordered attribute-value pair. For example, the following asserts that there exists a cognitive process called `initclick` with a start time and a duration:

$$\exists P_i : \{(isa, process), (name, initclick), (resource, cog), (start, S_i), (duration, D_i)\} \subset P_i \quad (1)$$

Each pair in the above statement consists of an attribute and a value. Further features may complete the specification of this process (see Howes et al. submitted for a full description). Relationships between the start times and durations of processes can be represented with simple integer-arithmetic inequalities. For example, the following represents the assumption that a motor process is a necessary consequence of an initialization process, that a motor process must start before the end of its initialization process, and that the maximum temporal gap between the two processes is 300ms:

$$\forall P_j : \{(isa, process), (name, initclick), (start, S_j), (duration, Dj)\} \subset P_j \Rightarrow \exists P_i : \{(isa, process), (name, click), (start, S_i)\} \subset P_i \wedge S_j + D_j \leq S_i \wedge S_i - (S_j + D_j) \leq 300 \quad (2)$$

In addition to representing theoretical assumptions about the human cognitive architecture, statements of this form can encode assumptions about the task environment, about instruction taking, and about the strategies that people deploy.

It is important to note that universally quantified constraints specified in a predicate calculus are not production rules. The constraints may to possess a similar surface form to production rules but the semantics are very different. Production rules are procedural representations of knowledge and skill activated in accordance with the control structure of a cognitive architecture. In contrast, constraints such as (2) above are declarative statements of theory. They are not elements of a procedure that generates the behavioral description. The constraint must hold for every circumstance where its antecedent is met. The generation of a model with these constraints is entirely monotonic and the order of expansion of elements of the behavior description can be (and often is) different from the actual predicted order of behavior.

A cascade-based process ontology The primitives of CCM models are *cascaded processes*. Rather than one process following the next, as in discrete-stage theories, processes overlap in time (McClelland, 1979). We assume that two processes must overlap in time to permit information to pass between them. If the processes are temporally separated, there must be some buffering process between them. Figure 1 illustrates this setup: two processes P_i and P_j communicate via a buffer process P_k. The temporal overlaps indicated by m and n ensure that the information pipeline is intact. We have formalized this basic 3-process relationship via a small set of arithmetic constraints, which serve as axioms that define an information processing ontology. This cascade-based ontology has important implications for the ability of CCM to systematically support behavior composition.

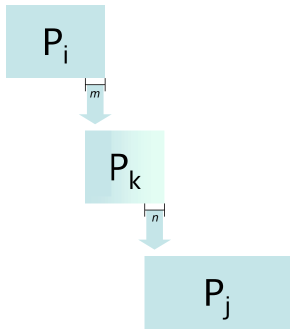

Figure 1: Two processes joined in cascade. See text for a formal specification.

Optimal constraint satisfaction Simon (1992) emphasizes the need for cognitive scientists to uncover optimal solutions given constraints on task environment, strategies, knowledge and human cognitive architecture. Given these constraints, it is possible to generate a prediction of optimal performance (Vera et al., 2004). First, the implications of the theory are derived; these implications take the form of constraints on a set of processes. Subsequently, using a constraint satisfaction engine (we presently use the Constraint Logic Programming Finite Domain engine in Sicstus Prolog), a prediction of the optimal behavior can be calculated by finding a set of variable bindings that are consistent with the defined constraints. A branch-and-bound algorithm is used to generate a schedule with the greatest utility. The particular algorithm is irrelevant to the theory, what is important is the objective function and the fact that its value can be minimized or maximized. In the examples reported in this paper the objective is to minimize time.

Example 1: Composing skilled behavior

We have built a prototype modeling tool, called CORE (Vera et al., 2004), that takes as input a set of mathematically stated constraints on behavior and outputs a prediction. One of the formats for the output, a behavior-graph, similar to a CPM-GOMS-like Pert chart, is illustrated in Figure 2. The prediction in the figure is for a pair of move-and-click mouse tasks (one represented in light gray and the other dark). Each box represents a process. In the figure, time is represented on the horizontal access and each row represents a different resource or processor, perception at the top, through cognition, to motor actions. Note that behaviors associated with the second mouse click begin before the initiation of the first mouse click.

Optimization can lead to the generation of surprising schedules. It can lead to both anticipatory behavior, as in Figure 2, and to strategic deferment. For example, Howes, Vera, Lewis, and McCurdy (2004, submitted) describe how, under certain architectural assumptions, the minimum-time cost schedule for a dual-task behavior required the strategic deferment of response retrieval.

This small example illustrates how the automatic composition of highly skilled behavior can be achieved

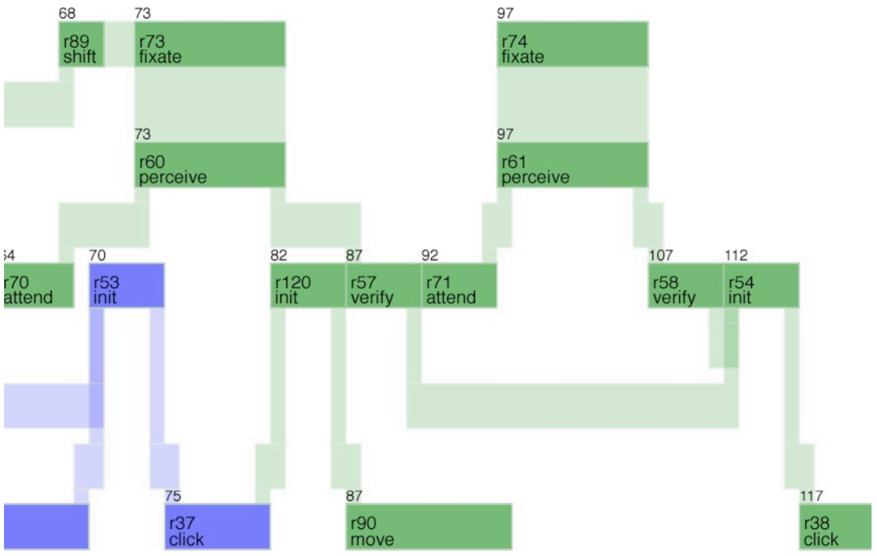

Figure 2: Interleaved mouse move-click tasks.

with a combination of optimization and cascaded information-processes. Cascades prevent cognitively implausible process orderings that are possible when the relationships between processes are described in terms of temporal dependencies, without specifying the processing resources for communicating between the processes. (For example, the process ordering `init(x), init(y), click(y), click(x)` is legal if the relationship between an `init` and a `click` is specified as a temporal dependency—but this may not be cognitively plausible because it assumes no cost to buffering information between the cognitive intention and the motor action.)

Example 2: Hierarchical composition at multiple skill levels

But how is behavior organized earlier in practice? We hypothesized that the hierarchical organization of behavior early in practice results in longer response times at subtask boundaries. Fixed task hierarchies must be retrieved, one goal at a time, from memory. The retrievals that occur at these task boundaries may have implications not only for response times, but also for the interleaving of subtasks that is a feature of skilled behavior. As described in the introduction, we are aiming toward a modeling framework that flexibly and easily spans mixtures of novice and expert behavior.

It is a commonplace of modern cognitive psychology that much complex human behavior is hierarchically organized in some way, and there is substantial evidence for this in domains ranging from language to problem solving. However, there is no existing compelling evidence for goal-subgoal structures in routine, non-problem-solving tasks with relatively fixed goal structures. Lewis, Vera, and Remington (2004, in preparation) set out to obtain such evidence, which we briefly describe next.

Empirical evidence for hierarchical control structures Using a simulation of an Automated Teller Machine (ATM), Lewis et al. (2004, in preparation) designed a set of studies to analyze the effects of task hierarchy on the acquisition of skill. They hypothesized that, for example, participants who were told that they must enter a 4-digit PIN in order to access all of their accounts and that they must then enter the specific account number for the desired transaction (e.g., withdrawal from checking versus savings) would form a different task representation than those told to enter their PIN and account number together in order to access all of the functions. The sequence of numbers in each case could be exactly the same, yet because of a potentially different hierarchical representation, there should be a longer pause between the two numbers in one case than the other.

Fifteen University of Michigan undergraduate students participated in the study. The ATM simulation interface, had five elements: a numeric keypad, function buttons to each sided of the numeric keypad (e.g. “OK”, “Cancel”), a display screen, buttons at the sides of the screen and two slots (one for inserting the card and the other for retrieving money/receipt). The entire task was mouse-driven, including clicking on the card and money/receipt slots. Each subject performed the same task 100 times. Subjects were instructed on the interface and given a set of of actions to perform: enter card, enter PIN, enter account, enter amount, etc.

As shown in Figure 3, there is evidence for a multi-tiered hierarchical representation of the task through the first 100 trials of performance. The data were analysed according to transitions corresponding to a three-level hierarchy traversal (e.g., from enter PIN to enter account), a two-level traversal, and a one-level traversal. All of these transitions were experimentally controlled for Fitt’s law variance. Figure 3 shows a clear differentiation for each type of transition. Actions requir-

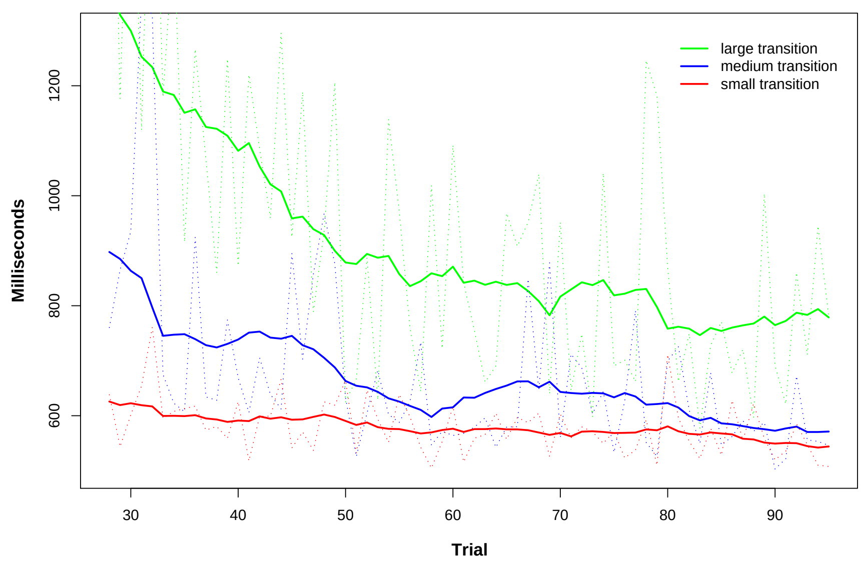

Figure 3: Reaction times for button presses in a simulated ATM banking task. Buttons are grouped according to the size of the transition required in the task hierarchy (Lewis, Vera, & Remington, in preparation).

ing three-level transitions took 200–400ms longer over the 100 trials than mouse clicks requiring only two-level transitions. Similarly, the difference between two-level an one-level traversals was on the order of 100–200ms. Two additional interesting effects are that the curves begin to converge but do not fully overlap late in practice (the three-level traversal curve remains 50-100ms above the other two) and that the one-level traversal curve remains relatively flat throughout (i.e., participants start out fast on those transitions and remain fast).

A model of hierarchical control and learning We specified a CCM model of Lewis et al.'s data focusing on the effect of hierarchy on inter-mouse-click latencies. Two constraints were of particular importance:

1. There is a time cost to retrieving subtasks from memory. The average time cost of retrieving a single subtask is fixed but the traversal of multiple layers of the hierarchy requires multiple retrievals. More retrievals therefore require more time.

2. Learning consists of incrementally removing intermediate layers in the task hierarchy from bottom to top. This assumption reflects the idea that subtasks are chunked together into more specialized and efficient (in terms of memory and time costs) procedures.

Figure 4 shows the model's predicted behavior for novice performance. This behavior graph is an optimal solution that was derived using CORE from a set of mathematically specified constraints. Time is represented from left to right. For now, we are not modeling the perceptual processing required to achieve the task. Therefore the cognitive processing consists of the retrieval of task information and the initiation (transmit) of motor responses.

Above the cognitive processor in the figure are the buffers that represent the goal information. At the top is the top level task (`do banking`), it is predicted to occupy the buffer for at least as long as is required to cue the retrieval of its two subtasks `type_pin` and `type_account`. Each of `type_pin` and `type_account` cue the retrieval of further subtasks. Eventually the correct sequence of motor responses is retrieval and initiated. Most importantly for our current purposes, the contiguous retrieval of task `r61` and its first subtask `r77` results in a longer latency between `r111` and `r112` than between `r110` and `r111`. I.e., consistent with the human data, the model predicts longer latencies at subtask boundaries.

Figure 5 illustrates the model's predicted behavior for expert performance. The behavior graph has been generated from task knowledge that was derived from the specified novice task knowledge using a set of five universally quantified predicate calculus statements. The statements encode a theory of how procedures are composed with practice. They specify how to infer the structure of an expert task hierarchy from a given novice task hierarchy and how to ensure that the task ordering is maintained once the control information provided by the hierarchy has been removed. In Figure 5 the hierarchy has been replaced by a buffer (represented below cognition) that connects the transmit processes. This buffer represents a working memory for control state information.

Discussion

We have described the application of a tool for making inferences about the implications of formally specified theories of behavior to the problem of understanding the composition of skill. The tool uses a constraint logic-programming environment to support inference given a specification of the constraints on the task environment,

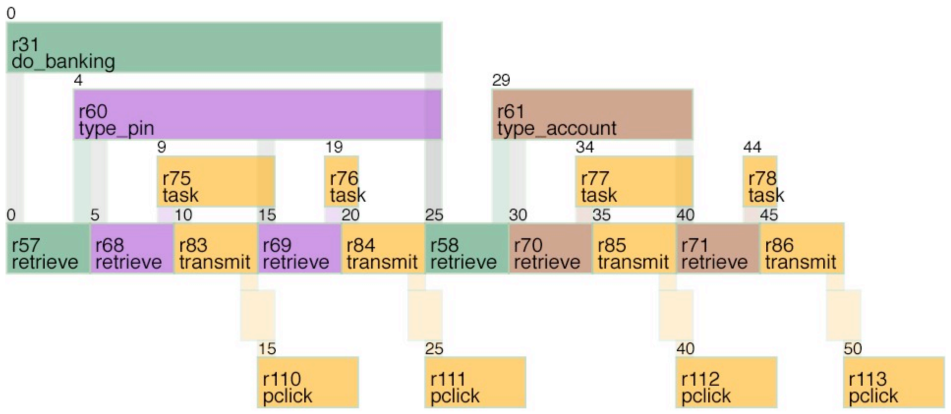

Figure 4: Prediction of novice performance on the ATM task. There is a greater duration between click r111 and r112 than between r110 and r111 because of the time required to retrieve the type-account subtask.

on perception, on cognition, and on action.

A particular model constructed with the tool was used to illustrate the way in which composition can be understood as incremental replacement of layers in the task hierarchy with process chaining. The behavior predicted by the model was consistent with human performance on a simulated ATM task. Most importantly, predicted inter-move latency corresponded to the task boundaries.

More generally, the framework and model begin to address the requirements for compositional modeling that we articulated in the introduction: (a) existing composed tasks were reused while avoiding the restricted-interleaving associated with encapsulation (Example 1); (b) automatic composition was achieved by separating constraints from scheduling and then using an optimal scheduling algorithm to predict the optimal behavior given constraints on task environment and strategy and architecture (see Simon, 1992); (c) parametric flexibility was achieved by specifying the declarative relationship between task knowledge structures at different levels of skill, rather than by simulating the incremental transitions of the human cognitive learning algorithm; (d) specifying constraints on cascading information processes in terms of mathematical relationships between their start times and durations supported rapid specification and manipulation of theory (only five rules were required to articulate a simple theory of skill composition).

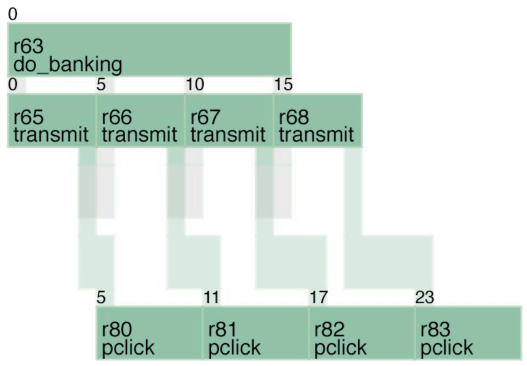

Figure 5: Prediction of skilled performance on the ATM task. The duration between the clicks is shorter and more homogenous than in the novice schedule.

Acknowledgments

This work was supported by ONR Grants N000140310087 and N0001404IP2002.

References

Anderson, J. R., & Lebiere, C. (1998). *Atomic Components of Thought.* Lawrence Erlbaum.

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

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

Lewis, R. L., Vera, A. H., & Remington, R. (2004, in preparation). *The hierarchical control of routine behavior: New theory and data.*

McClelland, J. L. (1979). On the time relations of mental processes: An examination of systems of processes in cascade. *Psychological Review*, *86*, 287–330.

Meyer, D. E., & 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, A. (1990). *Unified Theories of Cognition.* Cambridge, Massachusetts: Harvard University Press.

Simon, H. A. (1992). What is an explanation of behavior? *Psychological Science*, *3*, 150–161.

Vera, A. H., Howes, A., McCurdy, M., & Lewis, R. L. (2004). A constraint satisfaction approach to predicting skilled interactive cognition. In *Proceedings of CHI2004.* Vienna, Austria.