In Proceedings of the Sixth International Conference on Cognitive Modeling, 112-117.
Mahwah, NJ: Lawrence Earlbaum.

Learning to Choose the Most Effective Strategy: Explorations in Expected Value

Wayne D. Gray, Michael J. Schoelles, & Chris R. Sims
Cognitive Science Department
Rensselaer Polytechnic Institute
[grayw, schoem, simsc] @rpi.edu

Abstract

Small variations in how a task is designed can lead humans to tradeoff interaction-intensive for memory-intensive strategies. In this paper we introduce one such task, Blocks World, and present empirical data that shows such tradeoffs. Our attempts to model the acquisition of these tradeoffs using the default ACT-R conflict resolution mechanisms have met with failure. We have now run our model using four different methods for tallying a strategy's success and failure. For each run, we discuss the formulation of expected value used for conflict resolution and the reasons why the model does or does not match the human data.

Introduction

Few, if any, tasks are so new as to require the invention of strategies that have never been used by the task performer. Hence, in most situations, settling on a strategy or set of strategies for performing a task is not so much a matter of learning new strategies as it is learning which strategy, out of a set of already acquired strategies, works best in the current task environment.

This paper is motivated by our attempts to model strategy selection in Blocks World using ACT-R. First we introduce Blocks World and the empirical phenomena we seek to model. Second, in ACT-R, the *expected value* equation (Anderson, Bothell, Byrne, & Lebiere, in press; Anderson & Lebiere, 1998) determines which of two or more alternative strategies will be selected. We present data from a model that uses the default expected value equation and discuss why we found this mechanism inadequate for modeling Blocks World. Third, we layout and discuss three alternative bases for calculating expected value and present data from the original model run with each of these alternatives. We discuss how each alternative influenced model behavior as well as its fit or misfit to the empirical data. Fourth and finally, we summarize our work and draw conclusions regarding both the Blocks World task specifically, as well as our three alternative bases for calculating expected value.

Blocks World

Blocks World is a simple task that has been used to study the trade-off between interaction-intensive and memory-intensive strategies (Ballard, Hayhoe, & Pelz, 1995; Fu & Gray, 2000; Gray & Fu, 2000). The task is to copy a pattern of colored blocks shown in the *Target window* to the W*orkspace* window, using the colored blocks in the R*esource* window (for our version see Figure 1).

The Blocks World Studies

Each trial begins with a random placement of 8 colored blocks into empty spaces (defined by an invisible 4 x 4 grid) in the Target window. Unlike Figure 1, during the study all three windows are covered by gray windows. In our studies the gray windows that cover the Resource window and the Workspace window vanish as soon as the cursor enters those windows. The between-Ss manipulation varies how effortful it is to uncover the Target window. Across three studies (a brief description of the first is available as Fu & Gray, 2000, the other two are not published) we have varied difficulty "intuitively", by varying the Fitts' Law Index of Difficulty, and by lockout time.

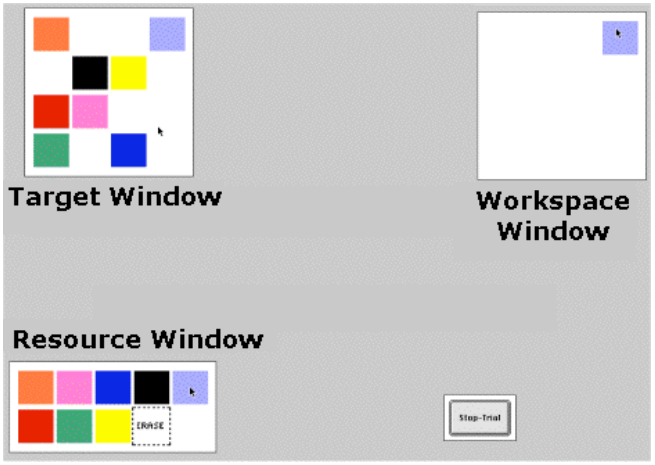

Figure 1. The blocks world task at the start of a new trial. In the actual task all windows are covered by gray boxes and at any time only one window can be uncovered. (The labels do not appear in the actual task. The Start/Stop button is shown at the lower right.)

Subjects were asked to do 40 (E1) or 48 (E2-3) trials. Each trial continued until they had correctly duplicated in the Workspace window the pattern (color and location) of blocks shown in the Target window.

The human and model data reported here are based on the version of Blocks World used in our third study. In that study the costs of opening the Target window were varied by increasing the lockout time (i.e., the delay in uncovering the Target window after the cursor had been moved into it). The three conditions reported have a 0 (0-Lock), 400 (400-Lock), and 3,200 (3200-Lock) millisecond lockout time.

Strategies

To access the information in the Target window subjects could adopt either an interaction-intensive or a memory-intensive strategy. An extreme interaction-intensive strategy would entail uncovering the Target window to obtain color information for a single block, obtaining that block from the

Resource window, another uncovering of the Target window to obtain the block's position information, followed by placing the block in the Workspace window. In contrast, an extreme memory-intensive strategy would entail one look at the Target window to encode both color and position for all 8 blocks.

We did not expect to find either extreme strategy to be popular with our subjects. However, as the cost of accessing information in the Target window increased, we expected to find that subjects shifted from more interaction-intensive strategies to more memory-intensive ones.

The Blocks World Results

For the current report, our measure of performance is the number of blocks correctly placed after the first, but before the second, uncovering of the Target window. At the time of the first uncovering, each of the lockout conditions has 8 blocks that have to be placed. Our empirical data shows that the number of blocks placed on the first uncovering varies significantly between conditions. Hence, on subsequent uncoverings, the number of remaining to-be-placed blocks differs between conditions.

Likewise, as it takes some time for the models and humans to settle on stable strategies, we only report data for trials 25-48. The process of "settling in" is interesting but beyond the scope of this short report.

There were 18 subjects in each of the three conditions. For these subjects, Figure 2 shows that as lockout time increases, the number of blocks placed in the Workspace window increases. Human subjects are clearly trading off interaction-intensive for more memory-intensive strategies.

Failure to Pick a Good Strategy: Issues in Credit Assignment and Expected Value

In data reported in Fu and Gray (2000), we showed that as the costs of opening the Target window increased subjects spent more time with the window open before going off to place the blocks. As the number of blocks placed also increased, the obvious inference is that the increased time spent with the Target window open, reflects increased time spent encoding a larger number of blocks.

To capture human adaptation to the cost of opening the Target window, we implemented a set of 8 *DO-strategies*. These strategies, DO-1 through DO-8, varied in the number of blocks they encoded per opening of the Target window. After each round of encoding, the model would go to the Resource window and attempt to retrieve the memory (declarative memory element or DME) of an encoded, but not-yet-placed block. If a DME was retrieved, a block of that color was picked up from the Resource window and placed in the Workspace window. After placing a block in the Workspace window the model tried to retrieve another DME of another encoded, but not-yet-placed block. When no more DMEs of not-yet-placed blocks could be retrieved, the model picked a new DO-strategy according to its expected value and another round began. A trial ended with all 8 blocks correctly placed in the Workspace window.

Model Details

The above description generally characterizes our modeling approach. This section provides further details on the construction and operation of our model.

Limits on DO-strategies. On reflection, it will be clear that all DO-strategies could encode their full range of blocks on the first uncovering of every trial, but not thereafter. For example, if on the first round of encoding, DO-4 fired, encoded 4 blocks, and placed 3, on the next round only 5 to-be-placed blocks would remain. Hence, on round 2, DO-5, DO-6, DO-7, and DO-8 would all encode 5 blocks. At best this would blur the distinction between DO-strategies. At worst, it seems cognitively implausible that, for example, people would fire a strategy to encode 8 blocks when only 1 block remained to be placed. To avoid this problem we wrote our model so that a DO-strategy would compete only if the number of to-be-placed blocks was greater than or equal to the strategy's *DO-number* (i.e., if 4 blocks remained, only DO-4, DO-3, DO-2, and DO-1 would be in the conflict set).

Calculating Expect Value. The DO-strategies compete with each other based on their expected value. ACT-R's expected value equation is:

$$EV = PG - C + /- noise \qquad \text{Equation 1}$$

P reflects the probability that a production has been successful in the past. P is simply calculated as the ratio:

$$P = \frac{successes}{(successes + failures)} \qquad \text{Equation 2}$$

G is a constant expressed in units of time. G is loosely thought of as the number of seconds that a person would be willing to pursue a given goal. The default value of G is 20.

C is a ratio of the sum of all past efforts attributed to the production divided by all past uses:

$$C = \frac{efforts}{(successes + failures)} \qquad \text{Equation 3}$$

Finally, noise adds variability to the expected value, but rather than constituting unexplained variability it seems to be an essential element. Too little noise leads the system to prematurely settle on strategies that gain an early advantage in P and C. Too much noise prevents the model from settling on any strategy, regardless of the values of P and C.

Credit Assignment. The credit assignment issue is "when" – when are the parameters in the expected value equation updated? These quantities could be updated for all productions once per trial; that is, after all 8 blocks are placed. However, as our model interacts with the same software as our humans interact with, many hundreds of productions fire on each trial. Indeed, we counted 762 productions firings on a randomly sampled trial that took 128 seconds of ACT-R time to complete. (This count includes many refirings by some productions.) As all trials ended successfully, each production fired on a trial would have the value of its successes updated by one. (If it fired multiple times, it would receive multiple updates.) Each production fired on a trial would have its efforts

incremented by the difference in ACT-R time between when it was selected and the end of the trial. (If it fired multiple times, its efforts would be updated for each firing by the difference between firing time and trial end time.)

Perhaps more to the point, placing 8 blocks entails a number of different DO-strategies firing a number of different times. This is the problem of structural credit assignment. Given a number of competing strategies, to what extent should each be credited with contributing to the final success of the trial? Updating all productions at the end of each trial would make it extremely difficult for credit assignment to properly credit the success, failure, and cost of any given DO-strategy. (Note that due to forgetting, the higher DO-strategies seldom, if ever, placed their complete allotment of encoded blocks.)

Rather than updating credit assignment once per trial, we updated it once per firing of a DO-strategy. Credit assignment time began ticking when a DO-strategy was selected. Time ended when the model could no longer retrieve the DME of a not-yet-placed block.

Model Runs. One model was run with four different schemes for updating successes and failures. For convenience, we refer to the model when it is running a particular updating scheme as, e.g., "the Success-Weighted model." However, each of these "models" used the same production rules, the same DMEs, and the same settings for all ACT-R parameters. The only change between models is in the updating of successes and failures that are discussed in the next section.

With two exceptions, all ACT-R parameters were left at their defaults. Specific parameters[1] important to our model include *enable subsymbolic computations* (:esc t); enable randomness (:er t); optimize learning (:ol t); parameter learning (:pl t); and base level learning (:bll 0.5). Although we make special mention of these parameters, this set is required by any model in which expected value and declarative memory activation is learned. They are all set to their default values. Our two exceptions do not have definite default values. We set activation noise (ans) to 0.23 and expected gain noise (egs) to 0.3. Activation noise is the noise added and subtracted to the activation of a DME on each retrieval attempt. The value we picked is within the normal range of this parameter and is one that we have used in other studies. Expected gain noise is the noise added and subtracted to the expected value of a production each time it appears in a conflict set (see Equation 1). The value we picked is within the range that we typically use in models (0.25 to 0.50). The setting of both egs and ans were done *a priori* – neither were tuned to the particular results of our models.

Problems in Updating Successes and Costs

The default scheme for calculating expected values based on successes, failures, and costs does not reproduce the data. The number of blocks placed by the model that used ACT-R's default scheme for updating successes and failures (the *Vanilla model*) is plotted at the bottom of Figure 2. It can be seen that this model differs quantitatively (RMSE = 2.26) as well as qualitatively from the human data. By the 25th trial, only one block is being placed in each lockout condition. As this is preliminary work, we are more concerned with the qualitative mismatch in trends across lockout conditions than with the quantitative mismatch in absolute number of blocks placed. Why does the number of initial blocks placed not increase for the higher lockout conditions?

Unfortunately, the answer to our question is as obvious as it is basic to the ACT-R calculation of expected value. As each round ends with a success, the value of P stays at 1.0. The expected value is driven entirely by the costs. As we go from DO-1 to DO-8 the model spends more time encoding blocks, more time getting blocks from the Resource window, and more time placing blocks in the Workspace window. For the higher DO-strategies the costs soar and the expected value plummets (see Table 1).

Table 1: Expected values after 48 trials for DO-strategies 1-8 by lockout condition in the Vanilla ACT-R model

Lockout	1	2	3	4	5	6	7	8
0	8.4	4.8	3.8	1.9	0.4	-1.0	-6.0	-4.7
400	8.1	5.1	4.0	3.7	-1.0	-1.5	-6.3	-6.7
3200	5.3	2.5	0.6	1.4	-1.0	-2.4	-9.8	-9.5

We believe that the failure to capture the qualitative trends reflects a fundamental flaw with the default ACT-R mechanism for credit assignment in tasks involving a sequence of steps over a time period of secs to 10s of secs. We elaborate this argument in the next section.

Weighting Successes and Failures

As success in Blocks World is defined as correctly placing 8 blocks in the Workspace window, the Blocks World paradigm allows us to define partial success in terms of the number of blocks placed. Hence, if a DO-strategy places one block into the Workspace window it is less successful than a DO-strategy that places four blocks.

Table 2: Three weighting schemes for changing ACT-R's system for calculating expected value.

	P		C	
Weighting	**Success**	**Failure**	**Success**	**Failure**
Success-Weighted	Yes	No	Yes	No
All-Weighted	Yes	Yes	Yes	Yes
Mixed-Weighted	Yes	Yes	Yes	No

Prior models of ACT-R have apparently not had to deal with such nuances. In thinking about how to overcome this limit to ACT-R we generated a number of schemes that weight the updating of the successes and failures parameters by the number of blocks placed. Three of these schemes are discussed below and shown in Table 2.

Implementing these schemes required adding a hook to ACT-R's *parameters learning* function to bypass the normal updating of the successes and failures parameters with the updates required by each scheme. The hook

[1] The only detailed discussion of ACT-R parameters that we know of is in Anderson and Lebiere (1998). Updated documentation, ACT-R5parameters.doc, can be retrieved from http://act-r.psy.cmu.edu/tutorials/.

function is called by the parameters learning function for each production in the sequence.

Success-Weighted

The most basic change is to vary the count of successes to reflect the number of blocks correctly placed. This update is shown in the first row of Table 2.

As the current model almost always successfully places at least one block, the Success-Weighted update is equivalent to dropping "failures" from the calculation of P (compare Equation 2 and 4) and C (compare Equation 3 and 5). This change has the effect of setting P to one.

$$P = \left(\frac{successes}{successes}\right) \qquad \text{Equation 4}$$

$$C = \frac{efforts}{successes} \qquad \text{Equation 5}$$

The effect of our change is to increase the denominator of C. Rather than adding one to successes each time a DO-strategy has fired, our change adds in the number of blocks that have been successfully placed. (For example, if DO-3 places 3 blocks successes will be incremented by 3. Hence, the cost in terms of the additional time required to encode and place multiple blocks is amortized over the number of blocks actually placed.

All-Weighted

An alternative update would be to weight both the number of successes and the number of failures that an update returns. This alternative is shown in the 2nd row of Table 2.

For *All-Weighted*, the equations for P and C are the same as the default equations shown in Equations 2 and 3. However, All-Weighted differs from the default in two ways. First, both successes and failures can be updated on a given round. Second, the number of successes and failures is weighted by the amount of the goal accomplished or attempted. As per the Success-Weighted scheme, All-Weighted increases the denominator of C and P by the number of blocks correctly placed (columns 2 and 4 of Table 2). However, unlike Success-Weighted, failures are also credited (columns 3 and 5). Failures are defined as the difference between the number of blocks encoded versus the number of blocks placed (DO-number minus number-placed). If, for example, DO-8 fires and encodes 8 blocks but places only 3, then DO-8 will be credited with 3 successes and 5 failures.

Unlike Success-Weighted, All-Weighted affects the value of P by differentially changing both the numerator and denominator (as per Equation 2). A DO-strategy that encodes more blocks than it can retrieve from memory will be severely punished by a decrease in P (the denominator increases faster than the numerator).

On the other hand, regardless of the number of successes and failures, for a given DO-strategy, All-Weighted equally increments the denominator of C (see Equation 3). For example, if DO-6 fires, encodes 6 and places 6, 6 successes will be added to the denominator for C. If the next time DO-6 fires it encodes 6 but places 3, the denominator will again be incremented by 6 (3 successes + 3 failures).

It would be one thing if All-Weighted were neutral with respect to the effect of success and failure on C; however, it seems to reward failure. If 6 blocks are encoded and only one is placed, then the time (and therefore effort) between initiating the strategy and finishing the strategy is less than if 6 blocks were placed, but the effect on the denominator is the same. Counterintuitively, for the same DO-strategy, costs are reduced more by an early failure than by an eventual complete success.

Mixed-Weighted

Mixed-Weighted is an alternative to All-Weighted that simply drops the count of failures from the denominator of costs. The expected value equation for Mixed-Weighted borrows its calculation of P from Equation 2 and its calculation of C from Equation 5. As per All-Weighted, if a DO-strategy promises more than it can deliver, then it is punished by a reduction in P. As per Success-Weighted, costs are reduced in proportion to the amount of the goal accomplished. Credit is not given for promises, only for results.

Model Data: Comparing Weighting Schemes[2]

The four models differ only in the scheme they use for counting successes and failures. In all other respects, in terms of productions, DMEs, and all other parameters, the models are identical.

Success-Weighted Model. As Figure 2 shows, unlike Vanilla ACT-R, the Success-Weighted model is not absolutely flat and consistently overshoots human performance. However, it is fair to say that Success-Weighted is a poor fit both qualitatively and quantitatively to the human data (RMSE = 1.42).

Table 3: Expected values after 48 trials for DO-strategies 1-8 by lockout condition in the Success-Weighted model

Lockout	1	2	3	4	5	6	7	8
0	7.1	8.8	10.0	10.2	10.3	10.4	10.2	9.9
400	6.3	8.1	9.4	10.2	10.4	10.4	10.1	9.8
3200	3.6	6.1	7.7	9.0	9.4	9.5	9.4	9.1

Across all three lockout conditions (see Table 3) the expected values of the smallest DO-strategies, DO-1 and DO-2, is much below that of the other DO-strategies. Post-hoc comparisons show that the comparison of DO-1 and DO-2 versus DOs3-8 was significant [$F(1, 105) = 1816$, $p < .0001$] and accounted for 83% of the variance due to DO-Strategy. DO-3 is close to the higher DOs for 0-Lock, it begins diverging slightly for 400-Lock, and by 3200-Lock it is still close, but 1.3 units of expected value away from the next highest expected value. Hence, the three lockout conditions are relying on essentially the same pool of DO-strategies with the slight increase in number placed for 400-Lock and 3200-Lock due to the less frequent use of DO-3 in favor of a slightly increased use of the higher DO-strategies.

All-Weighted Model. Compared to the Vanilla and Success-Weighted models, the All-Weighted model is a much better fit. As shown by Figure 2, this is the first model

[2] For each of the three models reported here, the model was run six times for each of the three lockout conditions.

that comes close to capturing the qualitative and quantitative (RMSE = 0.60) trends in the human data.

Table 4: Expected values after 48 trials for DO-strategies 1-8 by lockout condition in the All-Weighted model

Lockout	1	2	3	4	5	6	7	8
0	6.4	8.1	7.8	9.0	7.9	7.5	7.0	6.8
400	5.5	7.8	8.9	8.5	7.4	7.1	7.3	6.3
3200	2.6	4.9	6.8	8.1	8.0	7.6	6.6	6.7

Across the DO-strategies the difference between the maximum and minimum expected value varied from 2.6 for 0-Lock, to 3.4 for 400-Lock, and 5.5 for 3200-Lock (see Table 4). Post-hoc comparisons showed that DO-1 and DO-2 had a much lower expected value for 3200-Lock than did the other DO-strategies [$F(1, 35) = 67.7$, $p < .0001$] with this comparison accounting for 79% of the variance due to DO-strategy. This same comparison accounted for 15% of the variance for 400-Lock and 4% of the variance for 0-Lock. Hence, in contrast to the Success-Weighted model, it is clear that for the All-Weighted model a different mix of DO-strategies was favored across the three lockout conditions.

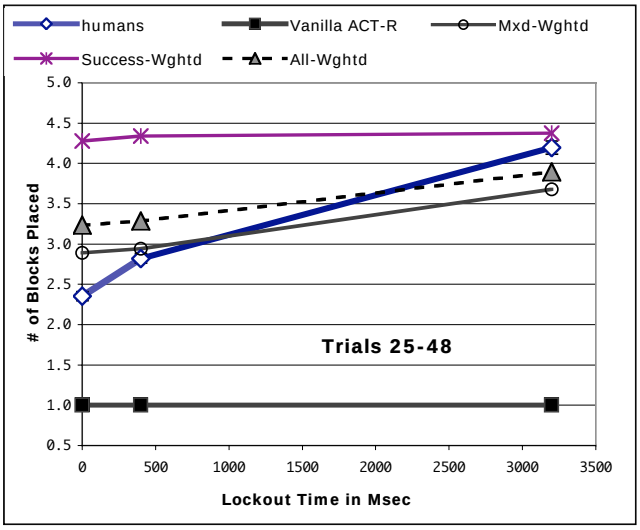

Figure 2: Blocks placed following the first uncovering of the Target window for Humans versus four ACT-R models. Except for the use of different systems for weighting successes and failures, all models use the same parameters and same productions.

Mixed-Weighted Model. The Mixed-Weighted model is the best fitting of the three both qualitatively and quantitatively. Quantitatively it has the smallest RMSE (0.44). Qualitatively, this model shows the greatest increase in blocks placed across lockout conditions. The difference between number of blocks placed at 0-Lock versus 3200-Lock is 1.84 for humans (see Figure 2), 0.79 for Mixed-Weighted, 0.66 for All-Weighted, and 0.09 for Success-Weighted.

In terms of expected value, performance in the 3200-Lock condition is dominated by DOs3-6 (see Table 5). The expected value of these DO-strategies were quite similar. The next closest DO-strategy was 0.50 expected value units below this range. In contrast, for 400-Lock the both DO-2 and DO-8 were within the same range of expected value as DOs3-6. For 0-Lock, DO-2 fell within the range of values shown across DOs3-6. Hence, compared to Success-Weighted and All-Weighted, for the Mixed-Weighted model as lockout time increases the extreme DO-strategies (high as well as low) are less likely to be selected.

Table 5: Expected values after 48 trials for DO-strategies 1-8 by lockout condition in the Mixed-Weighted model

Lockout	1	2	3	4	5	6	7	8
0	6.7	8.5	9.1	8.4	8.3	8.0	7.0	6.5
400	5.5	7.7	8.9	7.9	8.3	7.2	7.0	7.2
3200	2.7	6.0	6.9	7.1	6.9	6.9	6.4	6.3

Comparing Models. Across the four models the number placed gets closer and closer qualitatively and quantitatively to the human data, and the pattern of expected values for the DO-strategies begins to seem like a reasonable reflection of what humans must be doing.

The default, or vanilla, ACT-R model simply cannot handle these data. A strategy is either a success or a failure. If it places at least one block it is successful. The vanilla model was entirely driven by costs to an exclusive use of DO-1 over all runs of the model for trials 25-48. (Note that this model was so consistent that we only did three runs for each condition. In contrast, each of the other models was run six times per condition.)

The Success-Weighted model reduced the costs of the higher DO-strategies by the number of blocks they successfully placed. This cost reduction sufficed to boost the expected value of all higher DO-strategies. The expected values for DOs4-8 fell within 0.46 expected value units of each other for 0-Lock, within 0.56 for 400-lock, and within 0.52 for 3200-lock. Hence, the number placed was much higher than for the Vanilla model, but the number placed did not vary between lockout conditions.

The All-Weighted model punished strategies that encoded more than they placed by lowering their P value, but worked against itself by reducing costs based solely on the number encoded. This bias in reducing costs actually worked to favor strategies that encoded a lot but placed little. (More detail on this aspect has to await a fuller report in which we examine and report changes in P and C across models.) This all worked to favor DO-strategies in the range of DO-3 on up.

Like the All-Weighted model, the Mixed-Weighted model punished strategies that encoded more than they placed by lowering their P value. Unlike that model, it only reduced costs for the number of blocks actually placed. This combination worked to favor DO-strategies in the range of DO-4 to DO-6 over both the lower and higher strategies. The Mixed-Weighted model provided the best qualitative and quantitative fit to the empirical data.

Discussion and Conclusions

We divide this section into a brief discussion of alternative changes to expected value computations, conclusions about our work on the expected value equation, and conclusions about our model of Blocks World.

Other Changes to the Expected Value Equation

Rather than changing how successes and failures are calculated, our initial instinct was to change G – the value of the goal. It made much sense to us that if the value of completing a trial was worth, say 24, then the value of placing each block would be worth 3. (A strategy that placed 3 blocks would have its G incremented by 9 units.)

Whatever the merits of this scheme, unlike P and C, G does not accumulate separately for each production. Rather, G is a global value that is applied equally to calculate the expected value for each item in the conflict set. To experiment with G would require learning G. This would require more extensive changes to the current version of ACT-R (5.0) than the changes reported in this paper.

Other than our work, the only work we know that explores changes in ACT-R's calculation of expected value is that of Belavkin and colleagues. Whereas our work focuses on discriminating among strategies, Belavkin's work is focused more on changes in the range of expected values considered as the model gains expertise within a domain (Belavkin & Ritter, 2003) or on when to give up on a strategy (Belavkin, 2003).

Conclusions for calculations of expected value

This research into the parameters of the ACT-R expected value equation arose out of failure in trying to fit a model to the Blocks World data. As soon as we ran the model with ACT-R's default expected value settings (the Vanilla model) we realized that unless we could amortize costs and punish strategies that encoded more than they could place that we could not hope to fit the human data. The problem stemmed from the sparse reinforcement in the Blocks World environment, as well as the binary nature of success or failure in ACT-R. This dilemma prompted us to explore the space of expected values equations generated by changes in how successes and failures were counted and accumulated.

Our Mixed-Weighted model provides the best fit to our data and we believe it makes the most intuitive sense. For problems such as Blocks World where eventual success can be easily quantized into smaller units, it makes sense to us to reward and punish strategies based on how much of the problem they succeed in solving.

Conclusions for Blocks World

Although we believe the Mixed-Weighted model is a general solution to similar problems, we do not believe that we have adequately modeled the Blocks World data.

Under the Mixed-Weighted scheme, a key to a strategy's success or failure is its ability to retrieve from memory the items it has encoded. Currently we have run our models with optimized learning on (:ol t). This is the default for ACT-R models. However, in related research (Sims & Gray, 2004) we have come to believe that optimized learning overestimates the amount that can be retrieved in situations like the Blocks World where there is a long interval between an item's early encoding and rehearsal versus its later retrieval.

Indeed, for the larger DO-strategies there is a sizable period of time between the encoding and rehearsal of the first encoded block and the encoding and rehearsal of the last block on that round; this period, of course, precedes the long placement period. We believe that more realistic forgetting might work to drive the 0-Lock condition to rely more on lower DO-strategies so that the number placed for 0-Lock declines to somewhere closer to the human data (see Figure 2). Contrarily, more forgetting may drive the higher lockout conditions (such as 3200-Lock) to use fewer of the larger DO-strategies and more of the middle DO-strategies. We expect that this shift would have the counterintuitive effect of boosting number placed for 3200-Lock from its current 3.7 blocks to someplace closer to the 4.2 blocks of the humans in this condition.

Acknowledgments

The work reported was supported by a grant from the Office of Naval Research ONR #N000140310046. Additional support was provided by a subcontract to Rensselaer Polytechnic Institute from contract #MDA-904-03-C-0408 to Booz Allen Hamilton from the Advanced Research and Development Activity (ARDA). Thanks to Wai-Tat Fu for running human subjects as well as many other contributions to this project.

References

Anderson, J. R., Bothell, D., Byrne, M. D., & Lebiere, C. (in press). An integrated theory of the mind. *Psychological Review*.

Anderson, J. R., & Lebiere, C. (Eds.). (1998). *Atomic components of thought*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Ballard, D. H., Hayhoe, M. M., & Pelz, J. B. (1995). Memory representations in natural tasks. *Journal of Cognitive Neuroscience, 7*(1), 66–80.

Belavkin, R. V. (2003). Conflict resolution by random estimated costs. In D. Al-Dabass (Ed.), *Proceedings of the 17th European Simulation Multiconference* (pp. 105-110). Nottingham, UK.

Belavkin, R. V., & Ritter, F. E. (2003). The use of entropy for analysis and control of cognitive models. In F. Detje, D. Dorner & H. Schaub (Eds.), *Fifth International Conference on Cognitive Modeling*.

Fu, W.-T., & Gray, W. D. (2000). Memory versus perceptual-motor tradeoffs in a Blocks World task. In L. R. Gleitman & A. K. Joshi (Eds.), *Twenty-second Annual Conference of the Cognitive Science Society* (pp. 154–159). Hillsdale, NJ: Lawrence Erlbaum Associates.

Gray, W. D., & Fu, W.-T. (2000, November). *Memory versus perceptual-motor tradeoffs in a Blocks World task.* Conference of the Psychonomics Society, New Orleans.

Sims, C. R., & Gray, W. D. (2004). Episodic versus semantic memory: An exploration of models of memory decay in the serial attention paradigm. In M. C. Lovett, C. D. Schunn, C. Lebiere & P. Munro (Eds.), *6th International Conference on Cognitive Modeling–ICCM2004*. Pittsburgh, PA.