Bottom-Up Recognition Learning: A Compilation-Based Model of Limited-Lookahead Learning

Todd R. Johnson[1], Jiajie Zhang[2] and Hongbin Wang[2]

[1]Laboratory for Knowledge-Based Medical Systems
Department of Pathology
The Ohio State University
Columbus Ohio 43210
johnson.25@osu.edu

[2]Department of Psychology
The Ohio State University
Columbus, Ohio 43210
zhang.52@osu.edu, wang@canyon.psy.ohio-state.edu

Abstract

When faced with a novel problem, people can sometimes decide what to do by imagining alternative sequences of actions and then taking the sequence that solves the problem. In many problems, however, various constraints, such as working memory capacity, limit the amount of internal lookahead that people can do. This paper describes Bottom-Up Recognition Learning (BURL), a model of limited-lookahead learning based on final first learning and knowledge compilation. In BURL, knowledge compilation of limited-lookahead search over successive problem-solving trials transfers knowledge from the leaf nodes of a problem space to the top node. Two experiments test BURL's predictions. The first compares the Soar implementation of BURL to human subjects learning to play two Tic-Tac-Toe isomorphs. This experiment shows that BURL can account for learning that occurs when subjects can perform a limited lookahead. The second experiment studies transfer between two strategy acquisition tasks for one isomorph. This experiment shows that BURL must be used in conjunction with other learning methods to fully explain skill acquisition on limited-lookahead tasks.

Introduction

When faced with a novel problem, people can sometimes decide what to do by imagining alternative sequences of actions and then taking the sequence that solves the problem. In many problems, however, various constraints, such as working memory capacity, limits the amount of internal lookahead that people can do. For example, even in a simple game like Tic-Tac-Toe (TTT), people can have trouble imagining complete sequences of moves leading from the beginning of the game to the end. As a result, a person (when playing against a perfect opponent) will often lose several games before learning how to consistently get a draw (the best outcome against a perfect opponent).

What kind of learning process is responsible for this behavior? Newell (1990) has proposed the chunking-learning hypothesis which says that all long term learning occurs through chunking, a form of knowledge compilation. He and his colleagues have tested this hypothesis by demonstrating how chunking, as implemented in Soar (Laird, et al., 1987), supports a wide variety of learning behavior (Steier, et al., 1987). However, limited-lookahead learning is not directly supported by a knowledge compilation learning mechanism. The problem is that knowledge compilation works by compiling the results of search, but a limited lookahead search will not produce a result unless the search reaches the goal state, hence there will be nothing to compile. The lack of result can be circumvented by using a heuristic evaluation of the lookahead state, however this potentially incorrect evaluation will then get compiled leading to erroneous behavior in all later problem-solving trials. As a result, the current weak method for lookahead in Soar will only learn appropriate behavior if it is allowed to exhaustively lookahead. If the chunking-learning hypothesis is true, we must find a different way of using chunking to learn when lookahead is limited. Otherwise, we must reject the hypothesis and consider additional kinds of learning mechanisms.

In this paper, we present Bottom-Up Recognition Learning (BURL), a knowledge-compilation based theory of limited-lookahead learning. According to our theory, people begin with recognition knowledge of the final states of a problem, then, through successive trials, transfer this knowledge up to intermediate states, until eventually, a limited lookahead search from the initial problem state is sufficient to reach an intermediate state whose outcome is recognizable. Recognition knowledge for an intermediate state is acquired through knowledge compilation whenever limited lookahead reaches a recognized state. Thus, state recognition knowledge flows from the bottom of the search space up to the top. In addition to describing BURL, we present two experiments designed to test BURL.

Other Theories

Several other theories might also be used to explain limited-lookahead learning. First, people might remember the moves that they made in one trial and then use these again if they get the desired outcome, or avoid these moves if an undesirable outcome occurs. Second, if people can remember their sequence of moves, they might engage in self-explanation (VanLehn, et al., 1991). That is, they could attempt to explain why the sequence of moves led to the resulting state. This process would produce knowledge that would allow them to avoid or select moves in future trials. Finally, people might also treat the task as concept discovery, where the concept being discovered is a strategy

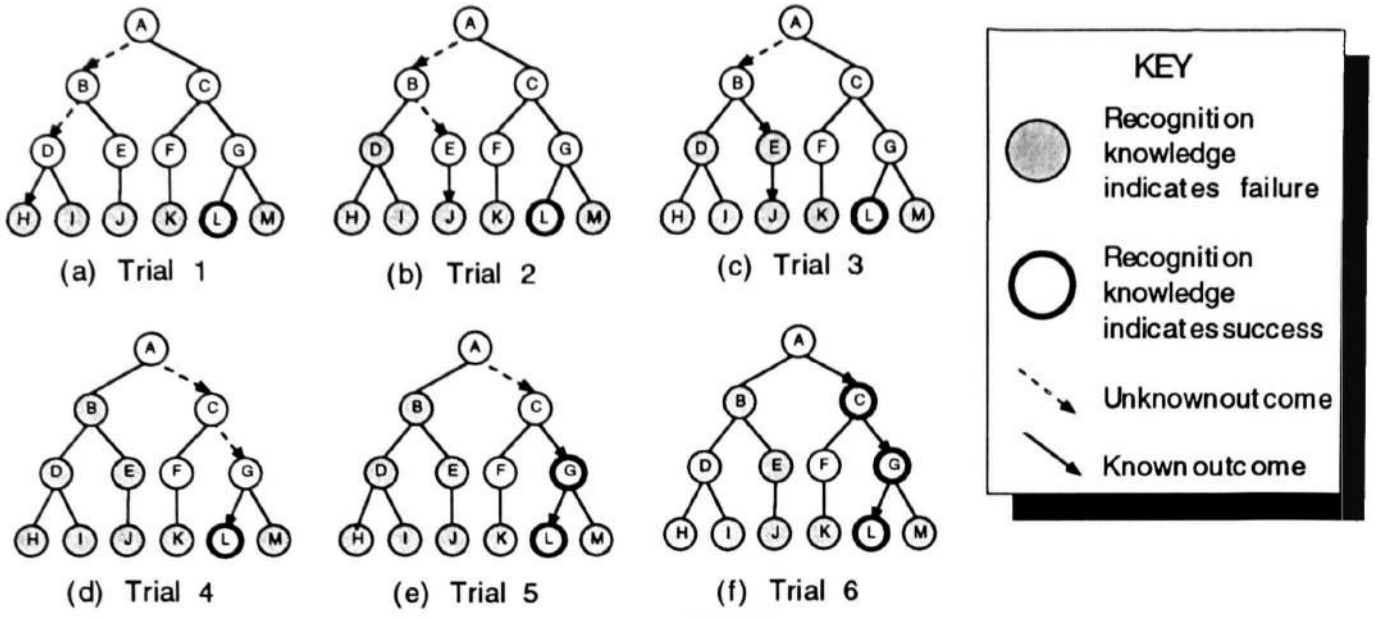

Figure 1: Illustration of basic bottom-up recognition learning, showing how recognition knowledge flows from the bottom of the tree to the top as successive trials are attempted.

and each game is an experiment designed to test a strategy. Although these seem plausible, they are all highly deliberate learning techniques. Our goal, as stated above, was to explore methods that are naturally supported by the chunking-learning theory. This led to the creation of BURL.

BURL is a form of final first learning (e.g., Howes, 1994), a class of learning methods in which state evaluations are backed up from terminal states (either goal or failure states) to higher states in a search space, one trial at a time. This technique has been used in reinforcement learning (Sutton, 1988), to model the acquisition of software interaction skills through exploration (Howes, 1994), and to improve the performance of machine game-playing programs (Samuel, 1967).

Final first learning describes a general class of learning methods, because it does not specify the learning technique used to acquire the knowledge. The learning technique dictates the conditions needed for learning knowledge and the characteristics of the resulting knowledge. For example, BURL must look ahead at least one step to learn any knowledge, but Howes' technique can learn without doing any internal lookahead because it remembers how to construct the previous state and then explicitly constructs evaluation knowledge for that state.

Huffman (1994) has developed a method similar to BURL for use in a Soar-based agent that learns from instruction. To the authors' knowledge, however, the work presented in this paper represents the first effort to apply and evaluate a final first learning technique based on knowledge compilation to human problem solving. General evidence for the use of final first learning by humans can be found in the work of Crowley and Siegler (1993). In a developmental study, they found that children developed rules for playing Tic-Tac-Toe in a final first manner.

Bottom-Up Recognition Learning

Basic BURL is illustrated in Figure 1. Each of the diagrams (a-f) illustrate one complete trial or game. The agent is trying to learn to solve the problem represented by the problem space shown in 1(a), where State L is the goal state. Prior to the first problem solving trial the agent only has recognition knowledge for the leaf states in the problem space. In other words, the agent can only recognize a state in which it has failed or succeeded. Each diagram illustrates what the agent knows by shading states that are recognized as on the path to failure and bolding those that are recognized as on the path to success. Arrows indicate the path that the agent took for that trial. A solid arrow means that the agent knew the outcome of the move. A dashed arrow means that the agent did not know the outcome.

The agent begins the first trial in State A, 1(a). For this example, we will assume that the agent can only look ahead one step. From State A, the agent looks ahead and sees that he can reach either States B or C. Since the agent doesn't recognize either state, he doesn't know which route to take, so a path must be chosen based on a bias. Lets assume that the agent selects B. By *select*, we mean that the agent applies the action leading to B in the external problem situation. From State B, the agent again looks ahead and, this time, sees D and E. Neither state is recognizable, so assume the agent picks D. From State D, however, one step lookahead reveals States H and I, both recognized by the agent as failures. As a result of this lookahead, the agent learns recognition knowledge for State D. Figure 1(b) shows the new state of the agent's knowledge. The new knowledge allows the agent to immediately recognize that State D is on the path to failure. For the first trial (a), however, the agent has already committed to a path leading to failure, so it selects one of the failure paths to H or I to end the trial.

On the second trial, the agent has the recognition knowledge shown in Figure 1(b). Beginning at A, the agent still doesn't recognize B or C as leading to success or failure. Suppose that the agent selects B again. From State B, lookahead reveals D and E. Since D is now recognized as a state that leads to failure, the agent will pick E. Lookahead from E indicates that it is also on the path to failure, so the agent learns recognition knowledge for E (See Figure 1(c)) and then ends the trial with failure at State J.

In the third trial, Figure 1(c), the agent still doesn't recognize either of the states from State A. The agent once again selects B. This time, lookahead from State B reaches D and E, both failure states, which allows the agent to learn to recognize that B is on the path to failure. This leads to the state of knowledge shown in 1(d).

In trial 4, the agent takes the path shown in 1(d), reaching the goal state. Trial 4 results in recognition knowledge to detect that G is on the path to success (1(e)). The agent's recognition knowledge is now sufficient to solve the task on all following trials. In trial 5, lookahead from State C, reveals that G leads to success, so recognition knowledge is acquired for C (shown in 1(f)). After Trial 6, the agent learns that A is on the path to success (this is not shown), completing the learning process.

Note that BURL does more than just cache evaluations. Because it uses knowledge compilation, the recognition knowledge that BURL learns when evaluating one state can potentially apply to multiple states. Knowledge compilation mechanisms build recognition knowledge that tests for only those features that were essential for producing a result. For example, in TTT many states are simply rotations of other states. If recognition knowledge is acquired for one state, it is likely that the same knowledge will apply to the rotated states. This feature distinguishes BURL from simple caching schemes. Generalization allows the agent to recognize additional states, without actually trying them; however, it also can lead to overgeneral knowledge, where states are recognized incorrectly. This would be undesirable for a machine-based game-playing program, but is consistent with human behavior (Larkin, 1981).

From this example, we can see that BURL has several interesting properties. First, BURL can learn from failure and success. In fact, it is possible for BURL to learn to correctly solve a problem by either failing many times or succeeding many times. On one hand, if BURL keeps failing, but never succeeds, it will eventually learn not to take failure paths, leaving only successful paths. On the other hand, if BURL consistently succeeds, then it will acquire knowledge of the correct path, preventing it from exploring any failure paths.

Second, the rate of learning depends on two characteristics: 1) the generality of acquired recognition knowledge; and 2) the bias used to select a move when lookahead is insufficient. The generality of the acquired recognition knowledge affects learning rate by increasing the number of recognizable states. In one trial, generalization can enable the agent to learn to recognize states that, without generalization, could take several trials to learn. The generality of the knowledge depends upon many factors of the task and the cognitive representation. We discuss some of these later in the experimental sections.

The bias affects learning because it affects the paths that the agent takes in each trial. What BURL learns in a trial is dependent on the moves made in that trial and previously acquired recognition knowledge. A good bias will tend to push the agent toward a goal state, so the agent will quickly acquire knowledge of states that are on the path to success. In contrast, a bad bias must be overcome through experience. It tends to force the agent down the wrong path, trial after trial, until the agent learns enough recognition knowledge to overcome the bad bias.

Extended BURL

Further Limiting Lookahead. To drastically decrease the amount of search when looking ahead more than 1 step, we modified BURL so that it uses the bias to control lookahead. If the outcome of the current state is not known (i.e., there is no recognition knowledge for whether the current state is or is not on the path to success), then the bias is used to select a move for lookahead. The agent then evaluates the move using an n-step lookahead. If the lookahead search is inconclusive (because the depth limit prevents the agent from reaching recognizable states), then the selected move is taken as the agent's actual move for that turn. However, if lookahead reveals that the move leads to failure, then the move is rejected and the bias is used to select a different move (which is also evaluated using lookahead). The end result is that, in general, search is drastically decreased. For example, on its first move in TTT the modified BURL agent will only need to evaluate 7 states (vs. 56 without this modification): the opponent's 7 responses to the single bias move (assuming n=2). This seems reasonable given the observed response times of about 5 seconds a move for the human subjects.

Implementation in Soar. BURL has been implemented in Soar as a method increment that can be used in place of Soar's default exhaustive lookahead method. The method increment is designed such that any task modeled in Soar that makes use of the default method can instead use BURL by simply adding knowledge of a bias or loading a supplied random bias. No changes to the original task code are needed.

The Experimental Task: Tic-Tac-Toe Isomorphs

We proposed BURL while attempting to model learning results for several isomorphs of TTT that are being used to study the interaction between perception and cognition (Zhang, 1993). Although the original research used 4 isomorphs, we have concentrated our modeling efforts on the two isomorphs (the easiest and hardest of the 4) shown in Figure 2. In the *Line* isomorph (Figure 2a), the first person to get three circles on a straight line wins. In the *Number* isomorph (Figure 2b), the first person to get *three* numbers that add up to 15 wins. The mapping between the two isomorphs is shown in Figure 2c. It should be evident that two or three circles on a straight line are easy to perceive in

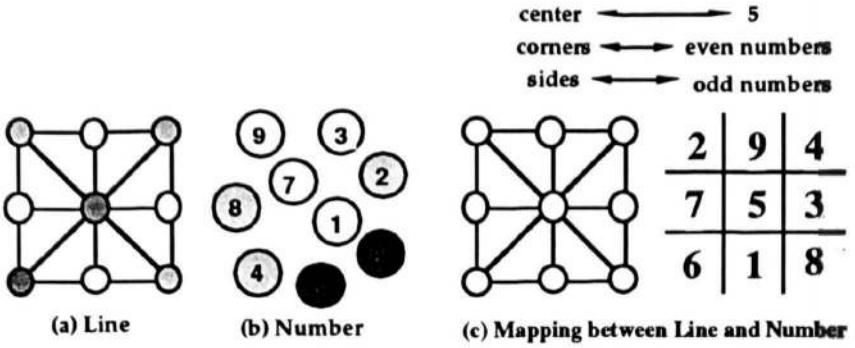

Figure 2: (a) The Line isomorph. (b) The Number isomorph. (c) The mapping between the two isomorphs. Modified from (Zhang, 1993).

Line, but that the same information is more difficult to see in *Number*, because it must be computed by adding pairs or triples. In addition, the symmetry information in *Line* (center, corners, sides) is easier to perceive than in *Number* (5, even and odd numbers).

Previous Empirical Results

In the two experiments reported in Zhang (1993), subjects played either the *Line* or the *Number* isomorph (see Figure 2) against a perfect computer opponent until they got 10 draws in a row. They were told that they could not win. In Exp. 1, the computer opponent always selected an even number[1] as the first move (see Figure 2c). To draw, subjects had to select 5, followed by any odd number (the five-odd strategy). In Exp. 2, the computer always selected 5 as the first move. To draw, subjects had to select any even number, followed by any remaining even number (the even-even strategy). We will refer to the 4 conditions using Isomorph-Experiment, such as *Number-2*, meaning the *Number* isomorph under the learning conditions for Exp. 2.

The empirical results for Experiments 1 and 2 are shown in Figure 3 (the bars marked as *Subjects*). It reports the average number of games needed to get the first draw and the average number of games played before 10 draws in a row were achieved (not counting the 10 draws). If a subject did not get 10 draws in a row after playing 50 games, 50 was used as the number to 10 Draws. The experiments reveal three important behavioral regularities: 1) *Line* is easier than *Number* regardless of the computer's first move; 2) the even-even strategy is easier to learn than the five-odd strategy; and 3) one draw is not sufficient to acquire the appropriate strategy.

The BURL Model for TTT

In applying BURL to TTT we made the following assumptions based on the theory of perception and cognition proposed by Zhang and on the empirical results. (1) Subjects can detect final draw and lose states (leaf states). (2) For a given state, subjects first consider moves that block the opponent. If no block is needed, then all possible moves are considered. (3) Subjects assume the opponent will move to win, if possible, otherwise block the subject from winning, otherwise, take one of the available moves. (4) In *Line*, symmetry information is available to the subject (i.e., they can detect and use information about the relationships between corners, sides and the center). Recognition knowledge learned for one trial of *Line* should therefore apply to symmetric situations. (5) In *Number*, symmetry information is not available. Recognition knowledge is therefore sensitive to the specific numbers tested during a single trial. (6) The biases were set by analyzing the subject's moves on the first few trials. This analysis revealed that only the subjects' first move of each trial was biased. All following moves were equivalent to random selection (prior to learning). The bias for the first move is probabilistic and was set to match the distribution of moves selected by the subjects. (7) Based on the speed of subjects' responses, we assume only a two step lookahead (n=2).

Given the biases and the assumptions about the use of symmetry, *Line* should be easier than *Number*, because symmetry information is available and the bias tends to select the correct first move. In addition, we would expect that the tasks in Exp. 2 should be easier than those in Exp.1, because for the first move in Exp. 2, subjects have a 50% chance of selecting the correct first move (an even number or a corner), whereas in Exp. 1 subjects only have a 12.5% chance of selecting the correct first move (5 or the center). In addition, the correct first move for *Line* in Exp. 2 corresponds to the subjects' bias. The critical point for the model is whether or not it can produce the specific changes in difficulty across the 4 conditions.

Simulation Results

The model was tested by simulating the same number of subjects as that used in the empirical studies (20 subjects for each of the 2 conditions in Exp. 1 and 15 subjects for each of the conditions in Exp. 2). Each simulated subject (the model) played against the same computer opponent used in the empirical studies until getting 10 draws in a row or having played 50 games. The results are shown in Figure 3, alongside the empirical data. At first glance, BURL appears to account for the three regularities revealed in the experiments described above: *Line* is always easier than *Number*, the even-even strategy (*Line-2* and *Number-2*) is

[1] We use 5, even and odd numbers in *Number* to refer to the positions in *Line* as well, since they are equivalent to the center, corners, and sides in *Line*. See Figure 2c.

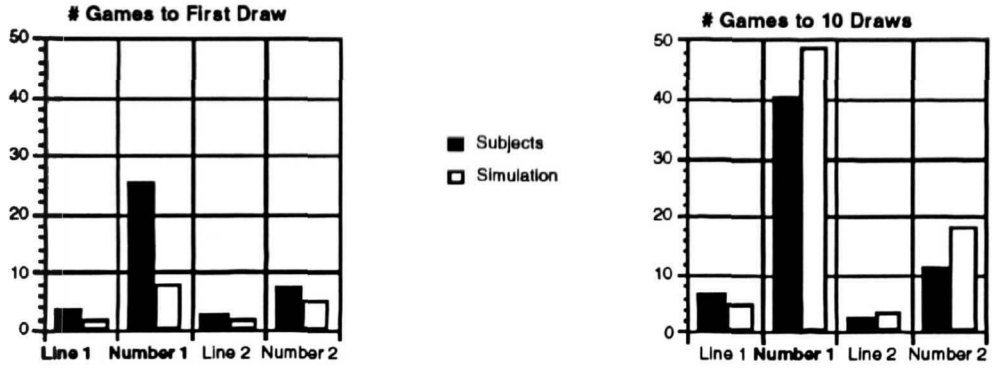

Figure 3: Comparison of simulation and empirical data from Experiments 1 and 2. The isomorph names in Bold-face indicate a significant difference between the simulation and subjects. From left to right: $t(38) = .316$, $p < .03$; $t(35) = 4.8$, $p < .0001$; $t(28) = 1.12$, $p = .27$; $t(28) = 1.84$, $p = .08$; $t(38) = .316$, $p = .75$; $t(38) = -1.64$, $p<.0001$; $t(28) = -1.22$, $p=.23$; $t(28) = -2.5$, $p=.02$.

easier to acquire than the five-even strategy, and one draw is not sufficient to acquire the appropriate strategy. However, when we look closely, BURL appears to explain behavior on *Line*, but not on *Number*. For *Number*, the simulation and subjects differ significantly on both the number of games to the first draw and the number needed to get 10 draws. To determine why, we collected verbal protocols from several subjects for each isomorph. The protocols for *Line* provided very little information, since subjects were able to readily perceive the information needed to make a move; however, the *Number* protocols revealed that successful subjects treated the task as a concept acquisition task. These subjects stated a possible strategy for selecting moves, then proceeded to test the strategy by trying it for several games. In addition, although subjects appear to do some lookahead on *Number*, most only attempt to determine if they need to block the computer—they do not check to see if they can make a move that would force the computer to block them. Thus, subjects only attempt a partial one-step lookahead. Clearly, this is not the type of task for which BURL is applicable. For that reason, in the next experiment, we will look only at *Line*.

Transfer Across Strategies

Although BURL appears to simulate the subjects' behavior on *Line*, more data is needed to determine precisely where the model might be right or wrong. Since BURL makes specific predictions about the recognition knowledge that will be acquired during a task, a good test is to see if subjects are learning the predicted knowledge. One way to test these predictions is with a transfer study. When the simulation learns to correctly solve *Line-1*, it has acquired a large body of recognition knowledge. We might expect some of this knowledge to transfer to *Line-2*. Likewise, knowledge for *Line-2* should also transfer to *Line-1*. Thus, we have two transfer conditions: *Line-1* to *Line-2* and *Line-2* to *Line-1*. Because *Line-2* is relatively quick to learn, comparatively little recognition knowledge is acquired. Thus, we would expect to see a lot of transfer from *Line-1* to *Line-2*, but little transfer from *Line-2* to *Line-1*.

We used the model to simulate 20 subjects for each condition. After the simulation, we collected data from 20 real subjects for each condition. The results are shown in Figure 4. When the number of games needed to get 10 draws in a row was used as the measure of transfer, as expected, the model shows transfer from *Line-1* to *Line-2* ($t(38) = 2.65$, $p<.01$), but none from *Line-2* to *Line-1* ($t(38) = -0.57$, $p=.57$). However, the subjects show transfer in the opposite direction, from *Line-2* to *Line-1* ($t(36) = 2.18$, $p<0.04$), but not from *Line-1* to *Line-2* ($t(36)=-0.043$, $p=0.97$). When the number of games needed to get the first draw was used as a measure, the simulation and experimental results were consistent: there was no significant transfer from *Line-1* to *Line-2* (Simulation: $t(38)=1.45$, $p=.16$; Subjects: $t(36) = -1.64$, $p=0.11$) or from *Line-2* to *Line-1* (Simulation: $t(38)=-0.13$, $p=.90$; Subjects: $t(36) = 0.66$, $p=0.52$) (see Figure 6). Transfer was measured as the difference between the subjects who played the isomorph first and those who played it second.

It seems that subjects acquired meta-knowledge from the first task: whenever getting the first draw, simply make the same moves that lead to the first draw to get the rest of the 10 draws. This can be easily seen from Figure 5. When *Line-1* was played as the first task, subjects needed 4.28 games to get the first draw and 10.11 games before getting 10 in a row. Thus, on average subjects got 10 draws in a row within 5.8 games of getting the first draw. In contrast, when *Line-1* was played as the second task, subjects needed 3.35 games to get the first draw and only 3.9 games before getting 10 in a row. Thus, on average subjects got 10 draws in a row within 1.5 games of getting the first draw. This meta-knowledge should have a much larger effect on *Line-1* because it is much harder than *Line-2*. Since *Line-2* is relatively easy (subject's who attempt it first need only 3.9 games on average to get 10 draws), the effect of the meta-knowledge might not be readily apparent. In addition, the standard deviation for *Line-2* is quite high: 1.092 for the first draw and 3.161 for 10 draws in a row.

Conclusion

BURL represents a natural solution to the problem of explaining how knowledge compilation can be used to directly learn when lookahead is limited. The implementation of BURL as a method increment in Soar provides the Soar user with a more cognitively plausible lookahead learning method than the default, exhaustive

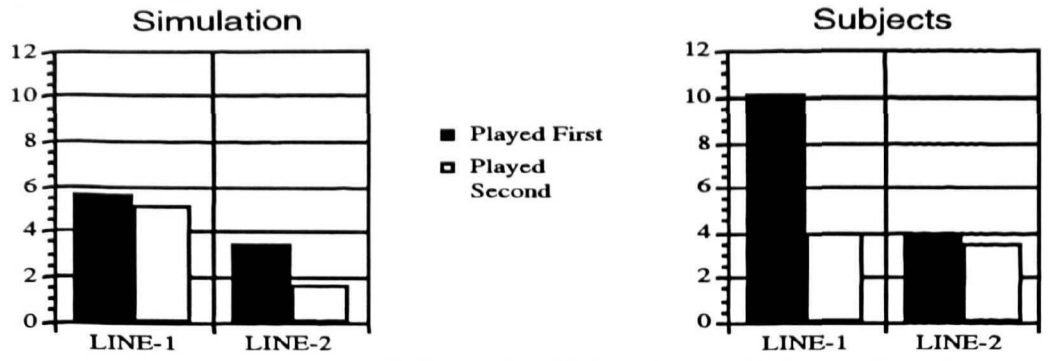

Figure 4: The number of games before getting 10 draws in a row in the transfer study.

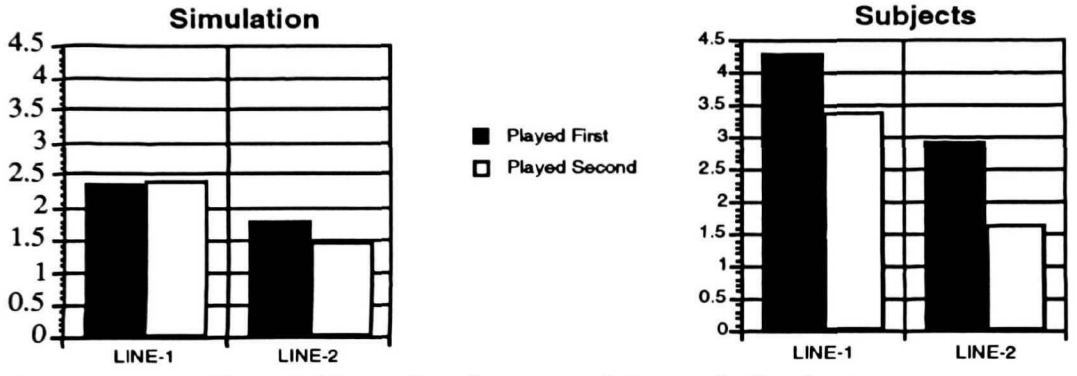

Figure 5: The number of games needed to get the first draw.

method. In addition, this work provides additional support for the chunking-learning hypothesis by showing how chunking can be applied to limited lookahead learning. The experiments described above show that BURL can explain some of the behavioral regularities of human learning; however, they also reveal possible discrepancies in the type of knowledge acquired during problem solving. To refine BURL we are planning to conduct a detailed model-tracing analysis of the model using individual subject moves. This will allow us to use the precise bias used by each subject and will provide additional details concerning where the model fits and does not fit the data.

Acknowledgements

We thank the members of the OSU Center for Cognitive Science and the Soar community for their discussions and comments on this work.

References

Crowley, K. & Siegler, R. S. (1993). Flexible strategy use in young children's tic-tac-toe. *Cognitive Science, 17*(4), 531-561.

Howes, A. (1994). A model of the acquisition of menu knowledge by exploration. In *Proceedings of the CHI '94 Conference on Human Factors in Computing Systems*

Huffman, S. B. (1994) *Instructable Autonomous Agents.* PhD thesis, University of Michigan.

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

Larkin, J. H. (1981). Enriching formal knowledge: A model for learning to solve textbook physics problems. In J. R. Anderson (Ed.), *Cognitive Skills and Their Acquisition* (pp. 311-334). Hillsdale, New Jersey: Lawrence Erlbaum Assoc.

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

Samuel, A. L. (1967). Some Studies in Machine Learning Using the Game of Checkers. II-Recent Progress. *IBM Journal,* 601-617.

Steier, D. M., Laird, J. E., Newell, A., Rosenbloom, P. S., Flynn, R. A., Golding, A., Polk, T. A., Shivers, O. G., Unruh, A. & Yost, G. R. (1987). Varieties of learning in Soar: 1987. In P. S. Rosenbloom, J. E. Laird, & A. Newell (Ed.), *The Soar Papers: Research on Integrated Intelligence* (pp. 537-548). Cambridge, Mass: MIT Press.

Sutton, R. S. (1988). Learning to predict by the methods of temporal difference. *Machine Learning, 3,*9-44.

VanLehn, K., Jones, R. M. & Chi, M. T. H. (1991). Modeling the self-explanation effect with Cascade 3. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society* (pp. 137-142). New Jersey: Lawrence Erlbaum Associates.

Zhang, J. (1993). The interaction between perceptual and cognitive processes in a distributed problem solving task. In *Working Notes of the 1993 AAAI Fall Symposium on Games: Planning and Learning*