Learning New Spatially-Oriented Game-Playing Agents through Experience

Susan L. Epstein
Department of Computer Science
Hunter College and The Graduate School of
The City University of New York
New York, NY 10021
epstein@roz.hunter.cuny.edu

Jack Gelfand
Department of Psychology
Princeton University
Princeton, NJ 08544
jjg@princeton.edu

Abstract

As they gain expertise in problem solving, people increasingly rely on patterns and spatially-oriented reasoning. This paper describes the integration of an associative visual pattern classifier and the automated acquisition of new, spatially-oriented reasoning agents that simulate such behavior. They are incorporated into a game-learning program whose architecture robustly combines agents with conflicting perspectives. When tested on three games, the visual pattern classifier learns meaningful patterns, and the pattern-based, spatially-oriented agents generalized from these patterns are generally correct. The trustworthiness and relevance of these agents are confirmed with an algorithm that measures the accuracy of the contribution of each agent to the decision-making process. Much of the knowledge encapsulated by the correct new agents was previously inexpressible in the program's representation and in some cases is not readily deducible from the rules.

Pattern Learning in Game Playing

In this paper we describe the use of an associative visual memory and spatially-oriented reasoning agents in two-person, perfect information, finite-board games. This approach uses two kinds of pattern-oriented learning for game playing: the association of particular patterns with successful or unsuccessful play, and the construction of spatially-oriented heuristics from those patterns. Figure 1(a), where the empty locations are blanks and # denotes "don't care," is an example of the first kind of pattern learning; it links a particular pattern from tic-tac-toe with success for X. In any symmetric orientation and whatever the # squares contain, a human expert associates such a configuration with a win for X.

Along with particular patterns, game-playing experts use more general but equally salient heuristics as spatially-oriented "rules of thumb." Figure 1(b) is an example of the second kind of pattern learning. It is the spatially-oriented heuristic "reflect O's move through the center," proved to be optimal play for X in the game of lose tic-tac-toe (Cohen, 1972). Advice from experts on how to analyze and play games is repeatedly couched in the language of such spatially-oriented patterns. Chess and checkers are discussed in terms of controlling the center of the board, while control of the edges is crucial in Othello (Fine, 1989; Gelfer, 1991; Lee & Mahajan, 1990; Samuel, 1963). Concepts such as shape and thickness are fundamental to the game of Go (Hideo, 1992; Iwamoto, 1976; Yoshio, 1991). As people improve their expertise in game playing, they increasingly employ spatially-oriented heuristics, and treat them as compiled knowledge, integrated but no longer reasoned about.

To learn pattern associations, programs use a feature language and inductive learning algorithms that operate on game states described in that language. De Groot proposed a recognition-association model to explain human chess skill in terms of spatial patterns (de Groot, 1965). Chase and Simon refined this model to include recall from long term memory in terms of spatial *chunks* (Chase & Simon, 1973; Simon & Gilmartin, 1973). There are several chess playing programs that capitalize upon patterns (George & Schaeffer, 1991; Levinson & Snyder, 1991). Applying learned patterns to game playing, however, has proved somewhat problematic. There are usually a great many of them and matching is non-trivial. T2 and Zenith, for example, learned predicate calculus expressions for tic-tac-toe and Othello, respectively (Fawcett & Utgoff, 1991; Yee, Saxena, Utgoff, & Barto, 1990). On one run T2 learned 45 tic-tac-toe concepts with 52 exception clauses after 800 contests, a great many for so simple a game.

In the work described here, learned pattern knowledge is used to construct higher-order, spatially-based reasoning agents. Programs that learn concepts from game-playing experience have in the past been hampered by a predicate calculus representation that lacks incisiveness, and by exhaustive explanation of inconsistencies for positions that may have no consequence in the strategic play of the game

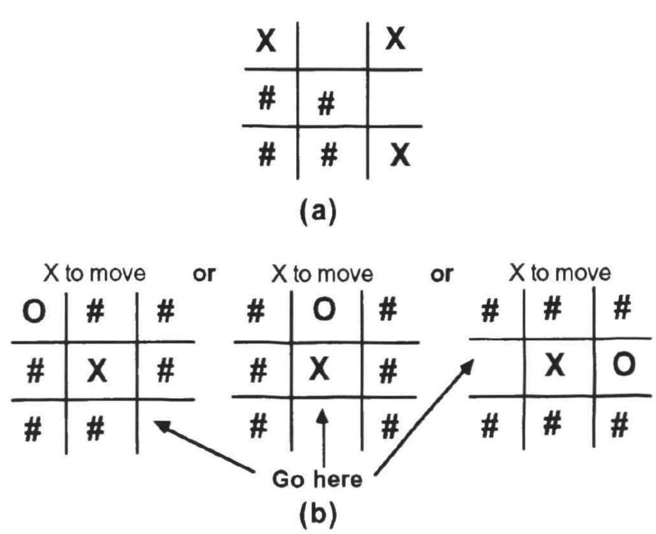

Figure 1: (a) A tic-tac-toe pattern that X associates with winning. # denotes "don't care." (b) "Reflect through the center," a spatially-oriented heuristic for lose tic-tac-toe.

(Fawcett, et al., 1991; Yee, et al., 1990). The process we describe, in contrast, is able to deal with inconsistencies robustly while it focuses attention on those situations containing important visual patterns. Most importantly, a dynamic filtering process continually refines the contents of the pattern memory to assure that, as the game-learning program becomes more expert, concept formation becomes increasingly accurate.

The long-range objective of this work is to create a heuristically-based decision maker that learns rapidly enough to participate in intelligent behavior while it is still acquiring knowledge. With a hierarchical multi-agent system, the presence of other more general problem solving advisors prevents incorrect actions, especially during early experience while learning. This paper reports that pattern-oriented learning functions as anticipated within this environment. We found that the validation process for newly created agents performed properly, and that the system worked smoothly as knowledge was being refined during the learning process. We believe that this process of creating new agents and testing their correctness in a multiple-agent program is unique.

A Game-Learning Program

There is evidence that humans integrate a variety of strategies to accomplish problem solving (Biswas, Goldman, Fisher, Bhuva, & Glewwe, 1995). There is also evidence that multiple, concurrent processing streams exist in the brain, each performing a component of a complex task. Automatic behaviors produce a locus of activity in the brain different from that of similar tasks requiring more cognitive processing (Grafton, Hazeltine, & Ivry, In press; Raichle, et al., 1994; Wallace, Silberstein, Bluff, & Pipingas, In press). In addition, during skill acquisition the locus of activity in the brain shifts from cognitive to associative areas with practice (Grafton, et al., In press; Raichle, et al., 1994; Wallace, et al., In press). The primate visual system has pathways for form, place, motion, and color (DeYoe & Van Essen, 1988; Ungerleider & Mishkin, 1982). Information from these streams is combined to form a perception of the visible world (Kandel, 1991). In addition, it has been found that different parts of the brain are activated when decisions are being made about different strategic aspects of chess (Nichelli, et al., 1994).

The mechanisms we describe below simulate these features. Hoyle is a program that learns to play two-person, perfect information, finite-board games. It is based on a learning and problem-solving architecture for skills called FORR, predicated upon multiple rationales for decision making (Epstein, 1994a). FORR employs multiple concurrent processing streams. Hoyle, as modified here, includes a separate stream for pattern learning. The transitions in the way Hoyle treats patterns model the automaticity shifts detected in humans during skill learning.

Hoyle learns to play in competition against a hand-crafted, external expert program for each specific new game. Whenever it is Hoyle's turn to move, a hierarchy of resource-limited procedures called *Advisors* is provided with the current game state, the legal moves, and any useful knowledge (described below) already acquired about the game.

Hoyle has 23 heuristic Advisors in two tiers. The first tier sequentially attempts to compute a decision based upon correct knowledge, shallow search, and simple inference, such as Victory's "make a move that wins the contest immediately." If no single decision is forthcoming, then the second tier collectively makes many less reliable recommendations based upon narrow viewpoints, such as Material's "maximize the number of your markers and minimize the number of your opponent's." An Advisor outputs its recommendations in the form of *comments.* A comment is of the form

<Advisor, action, strength>

where *strength* is an integer from 0 to 10 that measures the intensity and direction of opinion. Given the Advisors' recommendations, a simple arithmetic vote selects a move that is forwarded to the game-playing algorithm for execution.

Although 23 may appear to be quite a few Advisors, they do a large job with remarkable efficiency. Hoyle learns to play five men's morris with about 9 million states expertly, for example, during exposure to about .012% of the search space, and explicitly retains data on only about .006% of the states in the game graph. Hoyle plays without ever searching more than two ply (one move for each contestant) ahead in the game tree.

Hoyle learns from its experience to make better decisions based on acquired useful knowledge. *Useful knowledge* is expected to be relevant to future play and is probably correct in the full context of the game tree. Examples of useful knowledge include recommended openings and states from which a win is always achievable with perfect play on both sides. Each item of useful knowledge is associated with at least one learning algorithm. The learning methods for useful knowledge vary. The learning algorithms are highly selective about what they retain; they may generalize and they may choose to discard previously acquired knowledge. Further details on Hoyle are available in (Epstein, 1992).

Learning to Use and Apply Patterns

The crux of this paper is the addition to Hoyle of pattern learning and its application in new, game-dependent third-tier Advisors. The implementation of pattern learning and its application were inspired by repeated laboratory experiences with people, in the context of many different games. College students spoke about, reacted to, and relied upon familiar, sometimes symmetrically transposed, patterns while learning (Ratterman & Epstein, 1995). Later, they relied heavily upon these patterns as a kind of compiled expertise.

In this work, visually-perceived regularities are represented as *patterns*, small geometric arrangements of marker types (e.g., black, X) and unoccupied positions (*blanks*) in a particular geographical location. An *associative pattern store* provides a heuristically-organized database that links patterns with contest *outcome* (win, loss, or draw). The associative pattern store includes a set of templates, a waiting list, a pattern cache and generated spatial concepts.

Figure 2 provides an overview of the pattern matcher and the development of pattern-based Advisors from the game-

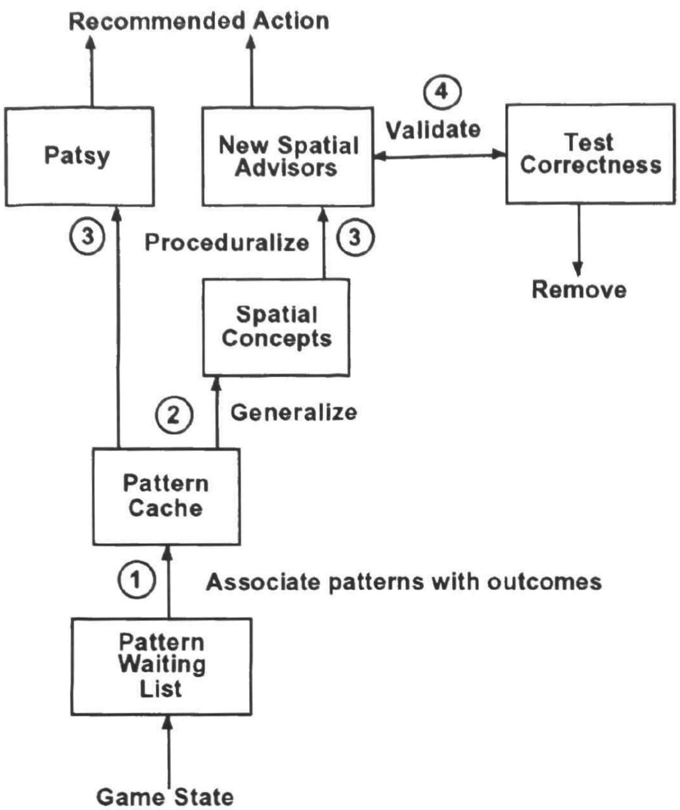

Figure 2: A schematic diagram of the associative pattern learning and spatial concept formation system.

specific associative pattern store. There are four stages detailed here: associate, generalize, proceduralize, and validate. Once patterns are identified, they are associated on the waiting list with winning, losing, or drawing. Patterns that persist over time and are identified with a single consistent outcome move from the waiting list to the pattern cache. Patterns in the cache are proceduralized via an associative pattern classifier, a new, game-independent Advisor called *Patsy*. Periodic sweeps through the pattern cache also attempt to generalize sets of patterns into concepts. Concepts are proceduralized as individual, game-specific Advisors that are then validated during subsequent learning. Finally, the pattern matcher improves as Hoyle learns to constrain pattern generation by excluding uninformative patterns.

Formulating Concepts from the Pattern Cache

Cached patterns are a rich source of information about the marker clusters to be seen during a particular game. Some of them ought to be forgotten; others are worthy of elevation to concepts that drive game-dependent Advisors. The identification of both kinds of patterns is done during a periodic sweep of the pattern cache. Currently, the first sweep of the pattern cache to form concepts is after 15 contests, and then the frequency is recomputed as a function of a confidence parameter after each sweep.

Generalization summarizes a set of detailed experiences into a more useful and efficient representation. Hoyle has two generalization rules to form concepts. Patterns in a cache are said to *agree* when they originate from the same template and pertain to the same stage of the game.

- Given distinct agreeing patterns P1, P2, and P3 with q ?'s that have the same mover and single, non-zero response, and are identical, except that in the *i*th position P1 has a black, P2 a white, and P3 a nil value, construct a new pattern P on the q-1 ?'s other than the *i*th. An example appears in Figure 3(a).
- Given distinct agreeing patterns P1 and P2 such that interchanging the contestants' markers and changing the mover in P1 results in P2 with the opposite single non-zero response, construct a new pattern P with variable place holders α for black and β for white. An example appears in Figure 3(b).

The cache is organized to support fast detection of agreeing patterns.

Proceduralization

Proceduralization is the transformation of expert knowledge into expert behavior. This is a non-trivial task in AI (Mostow, 1983). When there is much data or it conflicts in its potential application, as with pattern knowledge, interesting challenges arise. Each segment of the associative pattern store therefore relates differently to decision making. Patterns on the waiting list have no impact on decision making at all. Patterns in the cache serve as input to the associative pattern classifier, Patsy. Pattern-based concepts become game-specific Advisors.

The new, game-independent, second-tier Advisor Patsy ranks legal next moves based on the way the states they engender match patterns in the cache. Patsy considers the set of possible next states resulting from the current legal moves. Each next state is compared with the patterns in the appropriate, game-specific cache. No new patterns are cached during this process. Each pattern is assigned a value computed from the total number of won, lost and draw contests since the pattern was first seen. The strength of Patsy's comment on each legal next move is a function of the values of the patterns in the state to which it leads. Thus Patsy encourages moves that lead to states introducing patterns associated with a win or a draw, while it discourages moves that lead to states introducing patterns associated with a loss.

Each concept is proceduralized as a new, third-tier, game-specific Advisor. If the perfectly-correct, game-independent first-tier Advisors can select a move with their game-specific useful knowledge, they do so and the second tier is never consulted. If the heuristic but generally correct, game-

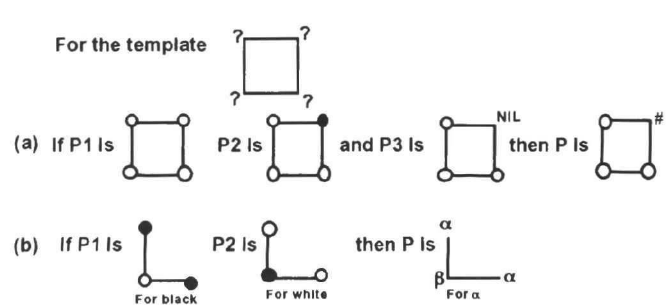

Figure 3: Two generalization rules that are applied to patterns to formulate concepts.

independent second-tier Advisors can agree upon a move with their game-specific useful knowledge, they do so. Otherwise the moves judged equally good by the second tier are forwarded to the newly-created third tier of game-dependent, pattern-based Advisors.

Validation of New Advisors

As new, pattern-based Advisors are introduced and Hoyle's skill develops further, some of them may prove irrelevant, self-contradictory, or untrustworthy, despite prior empirical evidence of their validity. Credit/blame assignment in a domain such as this is extremely difficult. At the end of a contest, it is difficult, even for human experts, to pinpoint the move that won or lost. The significant decision may have been early in play, or may have been a set of moves rather than an individual one. Rather than credit or blame a particular move, we have chosen to credit or blame the Advisors that support expert-like behavior.

Consider, for example, a hypothetical game state in which Hoyle has only second-tier comments

<Advisor-1, move-1, strength-1>

and

<Advisor-2, move-2, strength-2>.

Until now, if strength-1 and strength-2 were equal, the vote would be a tie, and one of the moves would have been chosen at random. But if Advisor-2 were more trustworthy in this particular game, its comment should have more influence. This approach holds the rationale behind actions accountable, rather than the actions themselves. Irrelevant and self-contradictory Advisors in a particular game should have weight 0, and more trustworthy Advisors should have higher weights than less trustworthy ones. Empirical experience with Hoyle indicates that these weights are problem-class specific, i.e., a new item of useful knowledge to be learned.

With an external model of expertise as its performance criterion, we use *AWL*, a perceptron-like model, to learn problem-class-specific weights for the [procedure (Epstein, 1994b). AWL runs at the end of every contest Hoyle plays against an external (human or computer) expert. The algorithm considers, one at a time, only those states in which it was the expert's turn to move and Hoyle's first tier would not have made a decision. For each such state, AWL distinguishes among support and opposition for the expert's recorded move and for other moves. Essentially, Hoyle learns to what extent each of its Advisors simulates expertise, as exemplified by the expert's moves. AWL cumulatively adjusts the weights of second-tier and third-tier Advisors at the end of each contest (whether or not the third tier would actually have voted during play), and uses those weights to make decisions throughout the subsequent contest. These weights are determined by a modification of Littlestone's learning algorithm (Littlestone, 1988).

Results

In all the experiments described here, Hoyle alternately moved first in one contest and second in the next. Such a trial continued until Hoyle was said to have learned to play a game because it could draw *n* consecutive contests in this environment. Once it met this behavioral standard, learning was turned off and the program was tested against four challengers that simulated perfect, expert (10% random move selection, 90% perfect), novice (70% random move selection, 30% perfect), and random contestants. During testing, reliability measures the consistency with which the program can continue to win or draw against contestants of varying strengths, and power measures the ability of the program to defeat contestants of various strengths (Epstein, 1994c).

We have used pattern-based learning with Hoyle in tic-tac-toe, lose tic-tac-toe (played exactly like tic-tac-toe except that the first contestant to achieve three of the same playing piece along a row, column, or diagonal *loses*), and five men's morris. This game has two contestants, black and white, each with five markers. A contest at this game is played on a board like that in Figure 5 and has two stages: a *placing stage*, where initially the board is empty, and the contestants alternate placing one of their markers on any empty position, and a *sliding stage*, where a turn consists of sliding one's marker along any line drawn on the game board to an immediately adjacent empty position. A marker may not jump over another marker or be lifted from the board during a slide. Three markers of the same color on immediately adjacent positions on a line form a *mill*. Each time a contestant constructs a mill, she *captures* (removes) one of the other contestant's markers that is not in a mill. Only if the other contestant's markers are all in mills, does she capture one from a mill. The first contestant reduced to two markers, or unable to move, loses.

Since Hoyle had already learned to play all the games studied here expertly after relatively few contests, these experiments were intended to demonstrate that game-dependent visual patterns exist and persist, despite the non-determinism of the learning experience. They also showed that such patterns can be gathered without a combinatoric explosion, and that the transition from waiting list to pattern cache to concept and Advisor is warranted. Furthermore it was shown that new, game-specific Advisors can be learned and managed appropriately, all without reducing the program's ability to play.

The potential computational overhead for concept formation is avoided. Very few of the possible patterns ever appear on the waiting list or in the cache. Even fewer are emphasized as the conceptual grounds for a heuristic Advisor, and some are learned to be uninformative. In tic-tac-toe, despite the potentially large number of patterns, after learning there were 58 patterns in the waiting list, 22.2 patterns in the cache, 4.2 uninformative patterns, and 6.4 concepts, all for draws. In lose tic-tac-toe, with just as many potential patterns, after learning there were 58.8 patterns in the waiting list, 57.2 patterns in the cache, 1.4 uninformative patterns, and 19 concepts, some for draws and others for losses.

Furthermore, the Advisor Patsy is highly weighted by the AWL validation algorithm. After learning tic-tac-toe, Patsy's average rank by weight among the Advisors in the second tier was 3 out of 17; after learning lose tic-tac-toe Patsy's average rank was 6.5 out of 17. AWL assesses Patsy to be a valuable Advisor. The growth in the weight of Patsy and in the weights of the pattern-based Advisors simulates the transition from high-level reasoning to skill learning.

With sufficient experience, Hoyle learns only correct as-

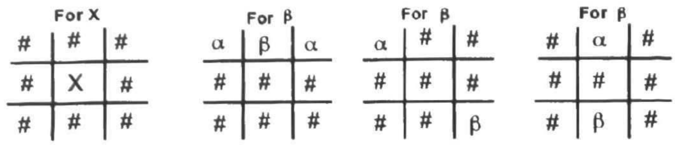

Figure 4: Some learned concepts for tic-tac-toe and lose tic-tac-toe. Note that the mover for a concept is in the current state, but the pattern is matched for the subsequent state.

sociations, ones considered relevant and significant by human experts. The first concept in Figure 4, for example, describes control of the center. Although it appears to be a simple pattern, it is actually a generalization over a set of persistent patterns. The second concept in Figure 4 blocks a potential row of three in its center.

In addition, concepts are learned which were previously inexpressible in Hoyle's representation. An example of this appears in lose tic-tac-toe where, to play the role of X perfectly, one must move in the location that is the reflection, through the center, of O's last move. Such reflection was not previously expressible in Hoyle's useful knowledge, but is now learned as the last pair of draw concepts in Figure 4. (Note that, with symmetry, vertical reflection through the center encompasses horizontal reflection and one diagonal reflection encompasses the other.)

The program experiences the rules of a game only as a set of "black boxes" that return the current state, the legal moves from it, and whether or not a state results in a win, a loss, or a draw. Consider, for example, what we term here *confinement,* the concept of restricting a five men's morris marker to a corner so that it can no longer slide. (Recall that a morris contestant unable to slide loses.) Confinement, the rightmost concept in Figure 5, is learned by Hoyle on every run. The concept of a mill (three markers of the same color on immediately adjacent positions on a line) was also previously outside the program's knowledge. (Hoyle only knows that certain moves permit it to capture, but not why.) Now on every run of five men's morris, Hoyle learns the first two concepts in Figure 5 as a pair of Advisors that subgoal on mills.

We found that value of pattern-based heuristics is confirmed in continued play. The reflection Advisors for lose tic-tac-toe and the mill Advisors for five men's morris have weights that remain among the top few in the third tier during learning with AWL. Although the reflection Advisors tend to emerge only after 80 or so contests, they typically achieve weights higher than 10 of the 17 second-

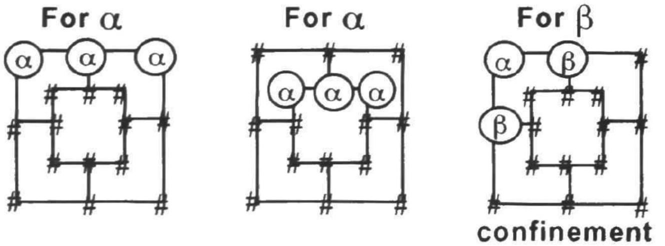

Figure 5: Some learned concepts for five men's morris. Note that the mover for a concept is in the current state, but the pattern is matched for the subsequent state.

tier Advisors, i.e., learned, game-specific knowledge proves more powerful than much of the more general game-independent knowledge supplied by the other advisors.

We note that there is a refinement of the contents of both the waiting list and pattern memory due to the threshold for a pattern to get into the waiting list, aging in both the waiting list and pattern memory, and the management of both consistent and inconsistent entries. Although we did not perform a quantitative study of this memory refinement process, we did find that without it performance was degraded. This process is ongoing and constantly refines the storage of important patterns with experience.

Discussion

Our work not only integrates pattern learning with high-level reasoning, it also suggests how the former gradually comes to support and enhance the latter. We do not advocate reliance on pattern-learning alone. That would ignore the other higher-level processes quite evident in humans. Indeed, Hoyle learns many other kinds of useful knowledge detailed elsewhere (Epstein 1992). Pattern learning is, however, an important component in skill development, one that those interested in the simulation of human intelligence or the design of adaptive game-playing programs cannot afford to ignore.

Each of the patterns Hoyle now learns is a generalization over a class of states that occurs with some frequency and contains a simple configuration of spatially-related markers. These patterns occur in the context of a particular stage of the game and are consistently associated with a single outcome. An associative pattern classifier provides learning whose possibly premature guidance is tempered by the higher-level reasoning of the other Advisors. When we force patterns to prove their reliability and importance before they can enter the cache, we reduce the combinatorics that would otherwise confront the generalizer. More experienced, concept-based Advisors gradually emerge to emphasize broader generalities, and are expected to advocate expert play to retain their status. Finally, the identification and exclusion of uninformative patterns constrains the pattern generator and thereby focuses the entire process more intelligently.

For this initial test we used simple games and made a number of simplifications in the individual components of the program. The Advisor, Patsy, based on individual patterns was placed in the second tier of Hoyle. The correct tier assignment for the new Advisors created from pattern-based concepts is another subject of current research. They were all placed in a third tier for the experiments described here, to avoid interference with a preexisting second tier that already worked quite well. To improve computational efficiency, however, and to model the transition to automaticity, the pattern-based Advisors should reside in the second tier. If they competed in parallel with the other second-tier Advisors, the pattern-based Advisors should comment faster and with greater weight in situations where they are applicable, and thereby supplant the others.

Future work includes more difficult games and other kinds of visual biases for spatial relations (such as center, edge, perimeter, bounded regions, length, and area), and causally-based pattern generation where one or more patterns that give rise to concepts are combined to create new, larger, somewhat less regular patterns. We intend to experiment with other learning algorithms to determine which is best for our application, and to develop and test a suite of generalization rules and meta-rules to construct concepts from patterns.

Acknowledgments

We acknowledge helpful discussions with Ron Kinchla, Philip Johnson-Laird, and Nick Littlestone. This work was supported in part by a grant from the James. S. McDonnell Foundation to the Human Information Processing Group at Princeton University, grant #9423085 from the National Science Foundation, and ONR grant #00014-93-1-0510.

References

Biswas, G., Goldman, S., Fisher, D., Bhuva, B., & Glewwe, G. (1995). Assessing Design Activity in Complex CMOS Circuit Design. In P. Nichols, S. Chipman, & R. Brennan (Ed.), *Cognitively Diagnostic Assessment.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Chase, W. G., & Simon, H. A. (1973). The Mind's Eye in Chess. In W. G. Chase (Ed.), *Visual Information Processing*. New York: Academic Press.

Cohen, D. I. A. (1972). The Solution of a Simple Game. *Mathematics Magazine*, *45*(4), 213-216.

de Groot, A. (1965). *Thought and Choice in Chess.* The Hague: Mouton.

DeYoe, E., & Van Essen, D. (1988). Concurrent Processing Streams in the Monkey Visual Cortex. *Trends in Neuroscience*, *11*, 219-226.

Epstein, S. L. (1992). Prior Knowledge Strengthens Learning to Control Search in Weak Theory Domains. *International Journal of Intelligent Systems*, *7*, 547-586.

Epstein, S. L. (1994a). For the Right Reasons: The FORR Architecture for Learning in a Skill Domain. *Cognitive Science*, *18*(3), 479-511.

Epstein, S. L. (1994b). Identifying the Right Reasons: Learning to Filter Decision Makers. *Proceedings of the AAAI 1994 Fall Symposium on Relevance.* (pp. 68-71). Palo Alto, CA: AAAI.

Epstein, S. L. (1994c). Toward an Ideal Trainer. *Machine Learning*, *15*(3), 251-277.

Fawcett, T. E., & Utgoff, P. E. (1991). A Hybrid Method for Feature Generation. *Proceedings of the Eighth International Workshop on Machine Learning* (pp. 137-141). San Mateo, CA: Morgan Kaufmann.

Fine, R. (1989). *The Ideas behind the Chess Openings.* New York: Random House.

Gelfer, I. (1991). *Positional Chess Handbook.* New York: Macmillan.

George, & Schaeffer. (1991). Chunking for Experience. In D. F. Beal (Ed.), *Advances in Computer Chess VI.* London: Ellis Horwood.

Grafton, S., Hazeltine, E., & Ivry, R. (In press). Localization of Independent Cortical Systems in Human Motor Learning. *Science*.

Hideo, O. (1992). Good Shape. In *Opening Theory Made Easy.* San Jose, CA: Ishi Press.

Iwamoto, K. (1976). *Go for Beginners.* New York: Random House.

Kandel, E. (1991). Chapter 30: Perception of Motion, Depth, and Form. In E. Kandel, J. Schwartz, & T. Jessel (Ed.), *Principles of Neural Science*. Amsterdam: Elsevier.

Lee, K. F., & Mahajan, S. (1990). The Development of a World Class Othello Program. *Artificial Intelligence*, *43*(1), 21-36.

Levinson, R., & Snyder, R. (1991). Adaptive Pattern-Oriented Chess. *Proceedings of the Eighth International Machine Learning Workshop* (pp. 85-89). San Mateo, CA: Morgan Kaufmann.

Littlestone, N. (1988). Learning Quickly when Irrelevant Attributes Abound: A New Linear-threshold Algorithm. *Machine Learning*, *2*, 285-318.

Mostow, D. J. (1983). Machine Transformation of Advice into a Heuristic Search Procedure. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Ed.), *Machine Learning: An Artificial Intelligence Approach*. Palo Alto: Tioga Publishing.

Nichelli, P., Grafman, J., Pietrini, P., Alway, D., Carton, J., & Miletich, R. (1994). Brain Activity in Chess Playing. *Nature*, *369*, 191.

Raichle, M., Fiez, J., Videen, T., MacLeod, A., Pardo, J., Fox, P., & Petersen, S. (1994). Practice-Related Changes in Human Brain Functional Anatomy during Nonmotor Learning. *Cerebral Cortex 4*, 8-26.

Ratterman, M. J., & Epstein, S. L. (1995). Skilled like a Person: A Comparison of Human and Computer Game Playing. *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society*, in press. Hillsdale, NJ: Lawrence Erlbaum Associates.

Samuel, A. L. (1963). Some Studies in Machine Learning Using the Game of Checkers. In E. A. Feigenbaum, & J. Feldman (Ed.), *Computers and Thought.* New York: McGraw-Hill.

Simon, H. A., & Gilmartin, K. (1973). A Simulation of Memory for Chess Positions. *Cognitive Psychology*, *5*, 29-46.

Ungerleider, L., & Mishkin, M. (1982). Two Cortical Visual Systems. In D. Ingle, M. Goodale, & R. Mansfield (Ed.), *Analysis of Visual Behavior.* Cambridge: MIT Press.

Utgoff, P. E. (1986). Shift of Bias for Inductive Concept Learning. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Ed.), *Machine Learning: An Artificial Intelligence Approach Volume II.* Palo Alto: Tioga Publishing.

Wallace, J., Silberstein, R., Bluff, K., & Pipingas, A. (In press). Semantic Transparency, Brain Monitoring and Evaluation of Hybrid Cognitive Architectures. *Connection Science*.

Yee, R. C., Saxena, S., Utgoff, P. E., & Barto, A. G. (1990). Explaining Temporal Differences to Create Useful Concepts for Evaluating States. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 882-888). Palo Alto, CA: AAAI Press.

Yoshio. (1991). *All about Thickness.* Mountain View, CA: Ishi Press.