This paper appeared in the *Proceedings of the AAAI Spring Symposium on Multimodal Reasoning*, 1998. Palo Alto, CA: AAAI.

Integrating Pattern Learning in Multimodal Decision Systems

Susan L. Epstein

Department of Computer Science
Hunter College and The Graduate School
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

Warren B. Powell

Department of Civil Engineering
and Operations Research
Princeton University
Princeton, NJ 08544

Abstract

We describe the integration of pattern-based reasoning learned through experience into two decision-making systems. The first is a hierarchical, multimodal game-playing program called Hoyle which integrates various approaches for deciding which move to make. The second is a dynamic programming assignment system which assigns trucking resources to delivery tasks. In this case, we utilize historic patterns of activity in the network generator to limit the candidate tour generation process for the assignment algorithm. In each case, the new knowledge results in increased performance.

1. Introduction

When the initial statement of a task is not given in terms of an effective or efficient representation, the ability to learn new representations of the problem is essential to a high-performance system. In many tasks people and machines optimize representation and sensory processing through their experience (Saitta, Neri, Bajo, Canas, Chaiklin, Esposito, et al. 1995). Knowledge about spatial and temporal patterns of activity may be of particular importance.

This paper describes autonomous decision-making programs in two domains. Each of them gradually acquires pattern knowledge from its experience, and integrates this knowledge into its decision-making structure. Performance improves in both cases as they shift their reliance to these pattern-based representations. Although one could, in principle, reason about a particular task and infer pattern knowledge about its domain, the inherent complexity of most decision-making domains makes it difficult to deduce pattern knowledge from first principles. The research reported here seeks a mechanism that uses experience to focus attention on an application's salient spatial knowledge, and then integrates that pattern knowledge with other factors in the decision-making process.

2. A Multimodal Game Playing Architecture

Hoyle plays games with a mixture of *Advisors*, procedural implementations of particular decision-making rationales. This approach is supported by evidence that people integrate a variety of strategies in order to accomplish problem solving (Biswas, Goldman, Fisher, Bhuva, & Glewwe 1995; Crowley & Siegler 1993; Ratterman & Epstein 1995). Because all Advisors are not equally important or equally trustworthy, Hoyle is organized into the two-tier hierarchy shown in Figure 1.

Advisors in tier 1 are guaranteed to be correct; they perform at most shallow search and do valid inference from perfect knowledge. For example, Victory does a simple, 1-ply lookahead to see if any legal move immediately wins the contest. Its inference procedure is correct, and its opinion is trustworthy. Wiser does a simple, 1-ply lookahead to see if any legal move leads to a state already known to be a win for its side (a *significant win state*) and cached in a lookup table. The significant state cache is correct, so Wiser too is trustworthy.

In contrast, the Advisors in tier 2 are heuristic, because either their reasoning method is not guaranteed valid or the data upon which they rely they is open to question. Tier-2 Advisors are also restricted in their search depth. For example, Vulnerable searches to see if it is threatened by a capture, but, since it only looks ahead 2-ply, Vulnerable's reassurance is not completely trustworthy.

As in Figure 1, the input to Hoyle at any decision point

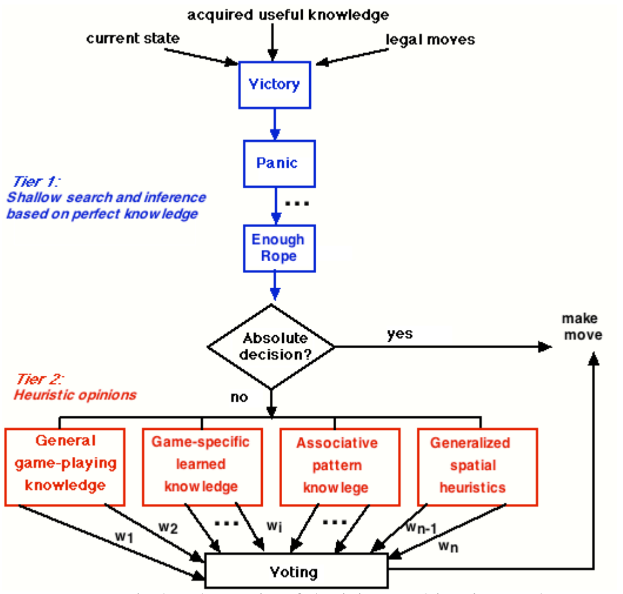
acquired useful knowledge
current state
legal moves
Victory
Panic
Tier 1:
Shallow search and inference based on perfect knowledge
Enough Rope
Absolute decision?
yes
make move
no
Tier 2:
Heuristic opinions
General game-playing knowledge
Game-specific learned knowledge
Associative pattern knowlege
Generalized spatial heuristics
w1
w2
wi
wn-1
wn
Voting

Figure 1: A schematic of decision making in Hoyle.

is the current state of the world, a list of the legal actions possible, and a collection of useful knowledge. Useful knowledge is heuristic, learned from experience, and expected to enhance performance. Good openings are an example of useful knowledge for game playing. The other kinds of useful knowledge for game playing include significant states, ones certain to result in a win for a particular contestant assuming perfect play on both sides. Each kind of useful knowledge has its own learning algorithm, triggered at the end of a contest or after a move.

A full list of Hoyle's Advisors, grouped by tier, appears in Table 1. Observe that individual Advisors employ a variety of reasoning methods. In particular, there are a number of model-based heuristics in tier 2. Coverage, for example, represents the board as a set of prespecified straight lines and reasons about their control. There is also an entire set of reasoners, including Pitchfork, for a bipartite graph representation (Epstein 1990).

Although every Hoyle Advisor is theoretically relevant to the games in its domain, the significance of any particular tier-2 Advisor will vary from one game to the next. Material, for example, is important only in games where pieces can be captured, and thus is irrelevant in tic-tac-toe. FORR, the architecture on which Hoyle is based, provides a facility to learn appropriate a weight for each tier-2 Advisor that reflects its significance in a particular game (Epstein 1994a).

Decision making in Hoyle considers each tier in turn. The first tier sequentially attempts to compute a decision based upon perfect knowledge, no more than 2-ply search, and valid inference. If the game-independent tier-1 Advisors can select a move, they do so, and the second tier is never consulted. Otherwise the Advisors in tier 2 collectively make less reliable comments from individual, narrow heuristic viewpoints, such as Material's "maximize the number of your playing pieces and minimize the number of your opponent's." When a decision must be made in tier 2, Hoyle selects the move with maximal support, summing the product of the strength from each comment about the move with the weight of the commenting Advisor. The weights of each Advisor's vote are game specific and are learned by playing against a perfect player.

One of our current research interests is learning new tier-2 pattern-oriented Advisors, as described below. The challenge comes in incorporating these new reasoners so that the system improves, without degrading its performance as the learned Advisors join the ranks of the other tier-2 Advisors.

Table 1: Hoyle's Advisors for game playing. Tier 1 is in its prespecified order. Advisors with a * apply useful knowledge.

Name	Description	Reasoning Method
Tier 1		
Victory	Makes winning move from current state if there is one.	Search
Wiser*	Makes correct move if current state is remembered as certain win.	Lookup
Sadder*	Resigns if current state is remembered as certain loss.	Lookup
Panic*	Blocks winning move non-mover would have if it were its turn now.	Search and lookup
Don't Lose*	Eliminates any move that results in immediate loss.	Search and lookup
Shortsight*	Advises for or against moves based on two-ply lookahead.	Search and lookup
Enough Rope*	Avoids blocking losing move non-mover would have if it were its turn.	Search and lookup
Tier 2		
Anthropomorph*	Moves as winning or drawing non-Hoyle expert did.	Lookup
Candide	Formulates and advances naive offensive plans.	Model-based
Challenge	Moves to maximize its number of winning lines or minimize non-mover's.	Model-based
Coverage	Maximizes mover's influence on predrawn board lines or minimizes non-mover's.	Model-based
Cyber*	Moves as winning or drawing Hoyle did.	Lookup
Freedom	Moves to maximize number of its immediate next moves or minimize non-mover's.	Model-based
Greedy	Moves to advance more than one winning line.	Model-based
Leery*	Avoids moves to state from which loss occurred, but where limited search proved no certain failure.	Lookup
Material	Moves to increase number of its pieces or decrease those of non-mover.	Model-based
Not Again*	Avoids moving as losing Hoyle did.	Lookup
Open*	Recommends previously-observed expert openings.	Lookup
Patsy*	Supports or opposes moves based on their patterns' associated outcomes	Lookup
Pitchfork *	Advances offensive forks or destroys defensive ones.	Model-based
Vulnerable	Reduces non-mover's capture moves on two-ply lookahead.	Search
Worried	Observes and destroys naive offensive plans of non-mover.	Model-based
Learned spatial Advisors	Supports or opposes moves based on their creation or destruction of a single pattern.	Lookup

2.1 Incorporating new reasoners

Spatially-oriented representations are useful for two-dimensional board games. Initial directions and rules given to novices, however, are typically statements of relationsamong a few playing pieces. As game players become more expert, they rely on pattern knowledge and spatial heuristics to direct play. Indeed, much advice from experts on how to analyze and play board games is conveyed through spatially-oriented concepts. Chess and checkers are discussed in terms of controlling the center of the board (Gelfer 1991) and concepts such as shape and thickness are fundamental to the game of Go (Ishida 1991; Otake 1992). Our approach to learning spatially-oriented heuristics for game playing is based upon learning this information through experience while playing the game. For a system to learn through experience, it must be able to perform at some low level of competence that supports the kind of experience required to achieve a higher level of performance through practice. The approach described here employs multiple expert decision modules some of which have general game playing knowledge and others which acquire new game-specific knowledge while playing.

For Hoyle, a *pattern* is a visually-perceived regularity, represented as a small geometric arrangement of playing pieces (e.g., black or X) and *blanks* (unoccupied positions) in a particular geographic location. A move can create a pattern by providing some missing piece or blank, or, in games where pieces are not permanently placed, destroy a pattern by removing one.

When it first learns a new game, Hoyle constructs a set of board-dependent *templates* as a filter for its perceived patterns: straight lines, L's, triangles, squares, and diagonals of a limited size composed of legal piece positions. When a template is instantiated with some combination of pieces, it becomes a pattern. The *associative pattern store* links patterns with *contest outcome* (win, loss, or draw). Patterns in the cache are proceduralized such that if an individual pattern is created by a legal move, the pattern Advisor, *Patsy,* votes accordingly.

Periodic sweeps through the pattern cache also attempt to generalize sets of patterns into spatial concepts. Each concept is proceduralized as an individual, game-specific, learned spatial Advisor. Each new Advisor is placed in tier 2, and a weight is learned for it. Despite the care with which patterns are filtered, some of these new Advisors are useless (silent) or simply wrong. Weight learning eventually detects that; their weights sink to zero, effectively removing them from decision making. Their introduction, however, could cause an intermediate reduction in Hoyle's prowess, until their weights drop low enough. Therefore Hoyle phases in these new, learned spatial Advisors, artificially suppressing their weights until they have shown themselves repeatedly reliable.

2.2 Results

Hoyle now learns pattern associations and game-specific spatial Advisors while it plays. The games we used were tic-tac-toe and *lose tic-tac-toe* (played like tic-tac-toe but whoever gets three in a row, column, or diagonal first loses). Both are *draw games,* that is, play between two perfect contestants must, by the nature of the game graph, end in a draw. Because tic-tac-toe and lose tic-tac-toe have the same board, they begin with the same templates. Tic-tac-toe is extremely easy for Hoyle to learn well, and we expected no improvement; it was present only to demonstrate that weights, patterns, and learned spatial Advisors were game-board independent. Lose tic-tac-toe is a far more difficult game to learn to play well, both for people and for machines. It has been solved mathematically and the correct strategies for the two contestants are different (Cohen 1972). Thus it forces the program to distinguish between patterns and concepts good for one contestant and those good for both.

Hoyle learns to value *pattern-oriented* play (i.e., Patsy and the learned spatial Advisors) highly. After learning in 200 tic-tac-toe contests, 32.6% of all weight is assigned to Advisors that are pattern-oriented, and Patsy and the best learned spatial Advisor always have the top two weights. In lose tic-tac-toe 29.3% of all weight is assigned to Advisors that are pattern-oriented, and Patsy ranks second on all but one run, where it ranks third. On 80% of the runs, Hoyle learned at least one spatial Advisor for lose tic-tac-toe with weight at least one, and that Advisor ranked fifth on average. Learning new spatially-oriented heuristics also reduces the number of contests Hoyle requires to develop consistent expert performance. With patterns and the learned spatial Advisors, the program never lost at lose tic-tac-toe during learning after contest 29.0, versus contest 56.0 without patterns and the learned spatial Advisors.

Two manifestations of Hoyle's learned spatial orientation are its performance against different opponents and its response in novel situations. Recall that Hoyle trains against a perfect player, which should result in stereotyped movement in the game tree (Epstein 1994b). During testing, however, the challengers with varying degrees of randomness in their responses introduce states in the game tree provably never experienced by Hoyle during learning. Hoyle with patterns is better able to win and draw in most of the categories, despite the differences between its trainer and its challengers.

3. Historical Patterns of Activity in Motor Carrier Assignment Systems

Motor carrier companies carry freight in trucks from point to point as either a full truckload dedicated to a single destination or fractional truckloads that make stops and discharge freight at many destinations. In either case an optimization problem exists for large truckload carriers who operate thousands of trucks and can dispatch hundreds of drivers per hour. Dispatchers must assign individual drivers to loads while minimizing *deadhead* miles (empty miles from a driver's location to a pickup point) and maximizing truck usage within schedule limitations. At present, this task is handled by human dispatchers. Dynamic programming assignment systems have been installed in many trucking firms to augment the decision-making strategies of the human dispatchers (Bertsekas & Tsitsiklis 1996; Powell & Shapiro 1997) . Though these automated systems are provably optimal at an instant in time, dispatchers agree with their assignments from only 40% to as high as 80% of the time.

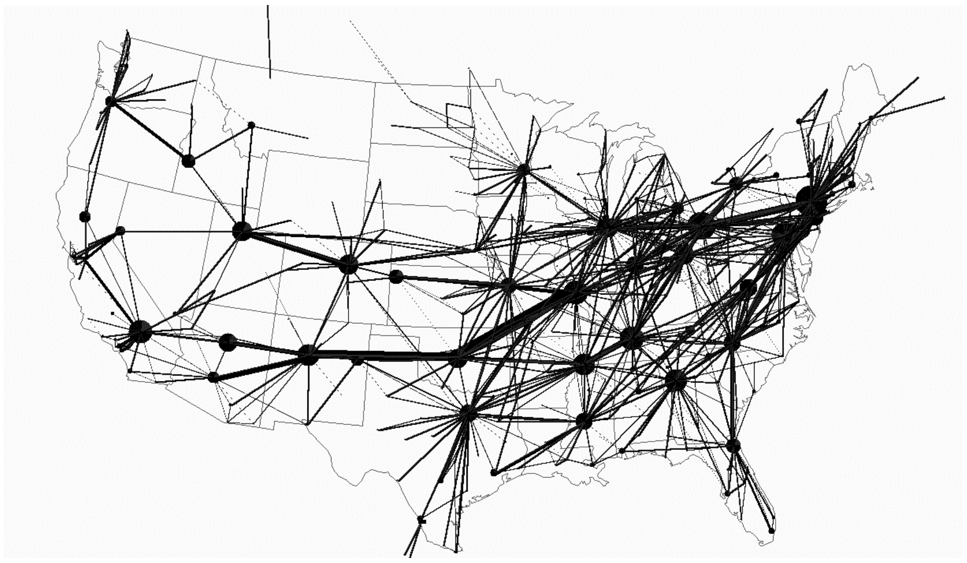

Figure 3: A map of one day's motor carrier activity at Yellow Freight.

In a significant percentage of the cases where the human dispatchers disagree with the automated system they do so with good reason. This is because, though the algorithms are provably optimal, they sometimes lack complete input knowledge of the state of the system and accurate forecasts of future trends. There are also important constraints that are not always explicitly expressed and entered as input. Because of the spatial and temporal nature of the task, this lack of knowledge and the errors in suggested assignments tend to be expressible in a pattern-based representation. Also, because the location of the shippers, destinations and trucking company terminals tend to remain somewhat constant, regular patterns of spatial or temporal activity tend to occur over time. Most importantly, many patterns of activity do not tend to occur for reasons that, as stated above, are not entered as input. These historical patterns can be used as a way to narrow down the choices of possible routes to be considered for truck assignments by the algorithm.

3.1 Integrating Historical Patterns of Activity into the Decision Process

The dynamic programming assignment algorithm uses a network generator which generates candidate tours for truck deliveries. These multi-leg tours are constructed incrementally by linking together single arcs. This process must be restricted in some way because it will result in a combinatorial explosion if all possible combinations of arcs are allowed. In the integrated system, a memory of previous patterns of activity is used to augment the tour generation process. This process results in some momentum because of the use of historical patterns. If there is any change in the factors that enter into an assignment decision, such as changes in union rules or other constraints, this will result in changes of the optimal patterns of activity. However the rest of the dynamic programming algorithm and network generator remain intact and reasonable assignments will be made As time goes on, the pattern assignments which result from the new conditions will change and the pattern data base will again use these new patterns as candidate tours.

3.2 Results

The pattern-based system has been installed at the operations headquarters of the Yellow Freight System in Kansas City, KS. This freight system handles approximately 6000 dispatches each day to 400 delivery points. The system described here is in daily use at Yellow Freight and supplies suggestions of optimal truck assignments to the human dispatchers. The operation of the system was checked by using the dynamic programming algorithm with and without the historical pattern memory in the network generator module. A useful measure of the performance of the system is the number of deadhead miles expressed as a percentage of total miles traveled. When the patterns are not used, the algorithm suggests assignments in which the deadhead miles are 8% - 9% of the total. If the historical pattern database is used this figure drops to 5.5% - 6.0 %.

References

Bertsekas, D. and Tsitsiklis, J. 1996. *Neuoro Dynamic Programming* . Belmont, MA: Athena Scientific.

Biswas, G., Goldman, S., Fisher, D., Bhuva, B. and 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.

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

Crowley, K. and Siegler, R. S. 1993. Flexible Strategy Use in Young Children's Tic-Tac-Toe. *Cognitive Science*, 17 (4): 531-561.

Epstein, S. L. 1990. Learning Plans for Competitive Domains. In *Proceedings of the Seventh International Conference on Machine Learning*, 190-197. Austin: Morgan Kaufmann.

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. Toward an Ideal Trainer. *Machine Learning*, 15 (3): 251-277.

Ishida, Y. 1991. *All about Thickness* . San Jose, CA: Ishi Press.

Otake, H. (1992). Good Shape. In *Opening Theory Made Easy* (pp. 62-111). San Jose, CA: Ishi Press.

Powell, W. and Shapiro, J. 1997. A dynamic programming approximation for ultra large-scale dynamic resource allocation problems. submitted :

Ratterman, M. J. and Epstein, S. L. 1995. Skilled like a Person: A Comparison of Human and Computer Game Playing. In *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society*, 709-714. Pittsburgh: Lawrence Erlbaum Associates.

Saitta, L., Neri, F., Bajo, M., Canas, J., Chaiklin, S., Esposito, F., Kayser, D., Nedellec, C., Sabah, G., Tiberghien, A., et al. (1995). Knowledge Representation Changes in Humans and Machines. In P. Reimenn, & H. Spada (Ed.), *Learning in Humans and Machines* Oxford: Elsevier
Science.