Coevolution of a Backgammon Player

Jordan B. Pollack & Alan D. Blair
Computer Science Department
Volen Center for Complex Systems
Brandeis University
Waltham, MA 02254
{pollack,blair}@cs.brandeis.edu

Mark Land
Computer Science Department
University of California, San Diego
La Jolla, CA 92093
mland@cs.ucsd.edu

Abstract

One of the persistent themes in Artificial Life research is the use of co-evolutionary arms races in the development of specific and complex behaviors. However, other than Sims's work on artificial robots, most of the work has attacked very simple games of prisoners dilemma or predator and prey. Following Tesauro's work on TD-Gammon, we used a 4000 parameter feed-forward neural network to develop a competitive backgammon evaluation function. Play proceeds by a roll of the dice, application of the network to all legal moves, and choosing the move with the highest evaluation. However, no back-propagation, reinforcement or temporal difference learning methods were employed. Instead we apply simple hill-climbing in a relative fitness environment. We start with an initial champion of all zero weights and proceed simply by playing the current champion network against a slightly mutated challenger, changing weights when the challenger wins. Our results show co-evolution to be a powerful machine learning method, even when coupled with simple hill-climbing, and suggest that the surprising success of Tesauro's program had more to do with the co-evolutionary structure of the learning task and the dynamics of the backgammon game itself, than to sophistication in the learning techniques.

1.0 Introduction

It took great *chutzpah* for Gerald Tesauro to start wasting computer cycles on TD-Gammon (Tesauro, 1992). Letting a machine learning program play itself backgammon in the hopes of bettering itself, indeed! After all, the dream of computers mastering a domain by self-play or "introspection" had been around since the early days of AI, forming part of Samuel's checker player (Samuel, 1959). Donald Michie initiated machine learning work on reinforcement with his MENACE tic-tac-toe learner using matchboxes with the positions drawn on them (Michie, 1961). However such self-organizing systems had generally been fraught with problems of scale and representation and abandoned by the field of AI. Self-playing game learners often learn weird and brittle strategies which allow them to draw each other, yet play poorly against humans and other programs. Yet after millions of iterations of self-play, Tesauro's program has become one of the best backgammon players in the world (Tesauro, 1995) and his weights are viewed by his corporation as significant enough intellectual property to keep as a trade secret except to leverage sales of their minority operating system. (International Business Machines, 1995). However, Tesauro has published the weights for a player using a linear evaluation function called PUBEVAL, which we made use of as a yardstick. Others have replicated this TD result both for research purposes (Boyan, 1992) and reportedly in a commercial product called Jellyfish.

How is this success to be understood, explained, and replicated in other domains? Is TD-Gammon unbridled good news about the reinforcement learning method? For the idea of "conditioning" a machine with rewards and punishments has been rejected by modern cognitive science as part of the associationist paradigm based on its weak or non-existent internal representations. It has been brought back to life in modern machine learning form through work initiated by Klopf, 1982, Barto et al., 1983, and Sutton, 1984. Similarly, there is a lot of work in learning in neural networks following the explosive success of Back-Propagation at overcoming some limitations of the Perceptron(Rumelhart et al., 1986). However, with respect to the goal of a self-organizing learning machine which starts from a minimal specification and rises to great sophistication, TD-Gammon stands quite alone in both the reinforcement and neural network literature.

In general, the problem with learning through self-play is that the player could keep playing the same

Originally in Proceedings of the Fifth Artificial Life Conference, May 1996, Nara, Japan. Revised November 1996.

kinds of games over and over, only exploring some narrow region of the strategy space, missing out on critical areas of the game where it could then be vulnerable to other programs or human experts. Such a learning system might declare success when in reality it has simply converged to a “mediocre stable state” of continual draws or a long term cooperation which merely mimics competition. Such a state can arise in human education systems, where the student gets all the answers right and rewards the teacher with positive feedback for not asking harder questions.

The problem is particularly prevalent in self-play for deterministic games such as chess or tic-tac-toe. We have worked on using a population to get around it (Angeline and Pollack, 1994). Schraudolph et al., 1994 added non-determinism to the game of Go by choosing moves according to the Boltzmann distribution of statistical mechanics. Others, such as Fogel, 1993, expanded exploration by forcing initial moves. Susan Epstein, 1994, has studied a mix of training using self-play, random testing, and training against an expert in order to better understand this phenomenon.

Tesauro, 1992 pointed out some of the features of Backgammon that make it suitable for approaches involving self-play and random initial conditions. Unlike chess, a draw is impossible and a game played by an untrained network making random moves will eventually terminate (though it may take much longer than a game between competent players). Moreover the randomness of the dice rolls leads self-play into a much larger part of the search space than it would be likely to explore in a deterministic game.

Our hypothesis is that the success of TD-gammon is not due to the Back-Propagation, Reinforcement, or Temporal-Difference technologies, but to an inherent bias from the dynamics of the game of backgammon, and the co-evolutionary setup of the training. In co-evolutionary learning, the desired task dynamically changes as the learner proceeds. We test this hypothesis by using a much simpler learning method for backgammon - namely hill-climbing - which retains the properties of self-play and initial randomness.

2.0 Setup

We use a standard feedforward neural network with two layers and the sigmoid function, set up in the same fashion as Tesauro with 4 units to represent the number of each player’s pieces on each of the 24 points, plus 2 units each to indicate how many are on the bar and off the board. In addition, we added one more unit which reports whether or not the game is in the endgame or “race” situation, making a total of 197 input units. These are fully connected to 20 hidden units, which are then connected to one output unit that judges the position. Including bias on the hidden units, this is a total of 3980 weights. The game is played by generating all legal moves, converting them into the proper network input, and picking the position judged as best by the network. We started with all weights set to zero. Our initial algorithm was hillclimbing:

i. add gaussian noise to the weights

ii. play the network against the mutant for a number of games

iii. if the mutant wins more than half the games, select it for the next generation.

The noise was set so each step would have a 0.05 RMS distance (which is the euclidean distance divided by $\sqrt{3980}$).

Surprisingly, this worked reasonably well! The networks so evolved improved rapidly at first, but then sank into mediocrity (when tested against Tesauro’s public domain evaluator PUBEVAL). The problem we perceived is that comparing two close backgammon players is like tossing a biased coin repeatedly: it may take dozens or even hundreds of games to find out for sure which of them is better. Replacing a well-tested champion is dangerous without enough information to prove the challenger is really a better player and not just a lucky novice. Rather than burden the system with so much computation, we instead introduced the following modifications to the algorithm:

Firstly, the games are played in pairs, with the order of play reversed and the same random seed used to generate the dice rolls for both games. This washes out some of the unfairness due to the dice rolls when the two networks are very close - in particular, if they were identical, the result would always be one win each. Though, admittedly, if they make different moves early in the game, what is a good dice roll at a particular move of one game may turn out to be a bad roll at the corresponding move of the parallel game. Secondly, when the challenger wins the contest, rather than just replacing the champion by the challenger, we instead make only a small adjustment in that direction:

$$\text{champion} = 0.95*\text{champion} + 0.05*\text{challenger} \quad \text{(EQ 1)}$$

This idea, similar to the “inertia” term in back-propagation, was introduced on the assumption that small changes in weights would lead to small changes in decision-making by the evaluation function. So, by preserving most of the current champion’s decisions, we would be less likely to have a catastrophic replacement of the champion by a lucky novice challenger.

In the initial stages of evolution, two pairs of parallel games were played and the challenger was required to win 3 out of 4 of these games. Running this algorithm on an SGI workstation, we were able to get through about 15,000 generations per day.

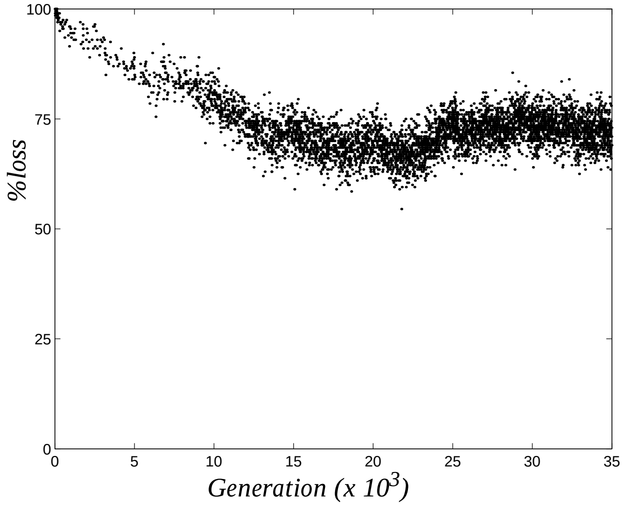

Figure 1: Percentage of losses of our first 35,000 generation players against PUBEVAL. Each match consisted of 200 games.

Figure 1 shows the first 35,000 players rated against PUBEVAL[1]. There are three things to note: (1) the percentage of losses against PUBEVAL falls from 100% to about 67% by 20,000 generations, (2) the frequency of successful challengers increases over time as the player improves, and (3) there are epochs (e.g. starting at 20,000) where the performance against PUBEVAL begins to falter. The first fact shows that our simple self-playing hill-climber is capable of learning. The second fact is quite counter-intuitive - we expected that as the player improved, it would be harder to challenge it! This is true with respect to a uniform sampling of the 4000 dimensional weight space, but not true for a sampling in the neighborhood of a given player: once the player is in a good part of weight space, small changes in weights can lead to mostly similar strategies, ones which make mostly the same moves in the same situations. However, because of the few games we were using to determine relative fitness, this increased frequency of change allows the system to drift, which may account for the subsequent degrading of performance.

To counteract the drift, we decided to change the rules of engagement as the evolution proceeds according to the following "annealing schedule": after 10,000 generations, the number of games that the challenger is required to win was increased from 3 out of 4 to 5 out of 6; after 70,000 generations, it was further increased to 7 out of 8. The numbers 10,000 and 70,000 were chosen on an ad hoc basis from observing the frequency of successful challenges. We are currently investigating how an appropriate annealing schedule may be determined in a more principled manner as the evolution proceed by dynamically adjusting the standard deviation of the gaussian noise, currently fixed at 0.05, as well as the margin of victory required of the challenger. Of course each bout was abandoned as soon as the champion won more than one game, making the average number of games per generation considerably less than 8.

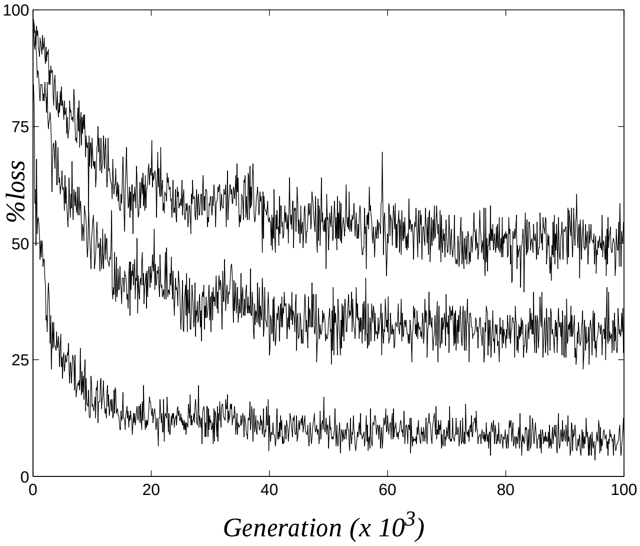

Figure 2: Percentage of losses against benchmark networks 1,000 [lower], 10,000 [middle] and 100,000 [upper]. This shows a noisy but nearly monotonic increase in player skill as evolution proceeds.

After 100,000 games, we have developed a surprisingly strong player, capable of winning 40% of the games against PUBEVAL. The networks were sampled every 100 generations in order to test their performance. Networks at generation 1,000, 10,000 and 100,000 were extracted and used as benchmarks. Figure 2 shows the percentage of losses of the sampled players against the three benchmark networks. Note that the three curves cross the 50% line at 1, 10, and 100, respectively and show a general improvement over time.

The end-game of backgammon, called the "bear-off," can be used as another yardstick of the progress of learning. The bear-off occurs when all of a player's pieces are in the player's home, or first 6 points, and then the dice rolls can be used to remove pieces. To test our network's ability at the end-game, we set up a racing board with two pieces on each player's 1 through 7 point and one piece on the 8 point as shown in Figure 3. In playing from this position, the skill involves knowing to first move the three pieces from the 7 and 8 points, and then aggressively removing pieces rather than moving them around (which a

[1] PUBEVAL is quite a strong machine player, trained on a database of expert preferences using comparison training. (Tesauro, personal communication). Figures 1 and 4 have been corrected since an earlier release of this paper.

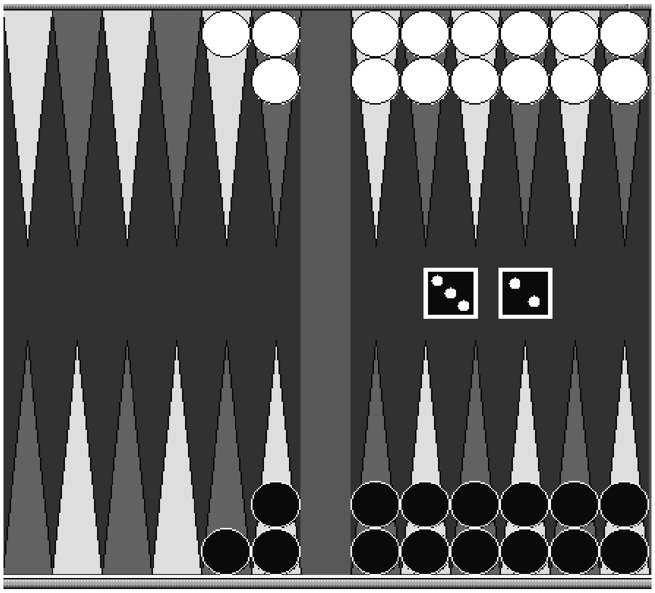

Figure 3: Starting position for bear-off trials

random player might do). The graph in Figure 4 shows the average number of rolls to bear-off of each generation network playing itself using a set of 200 random dice-streams.

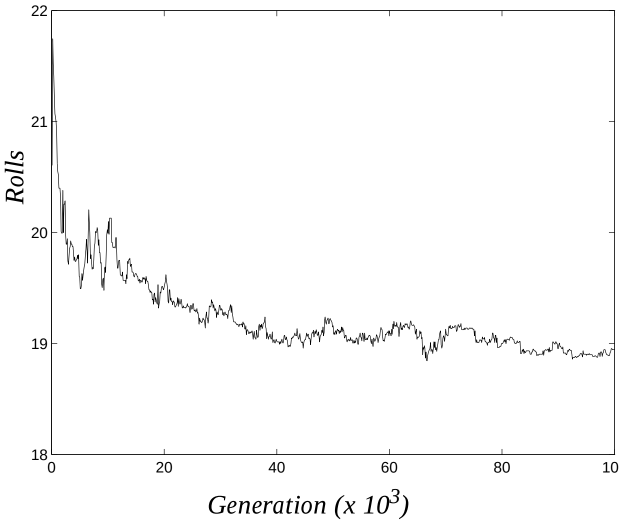

Figure 4: Average number of rolls to bearoff by each generation, sampled with 200 dicestreams. PUBEVAL averaged 16.6 rolls for the task.

3.0 Discussion

3.1 Machine Learning and Evolution

We believe that our evidence of success in learning backgammon using simple hillclimbing indicates that the reinforcement and temporal difference methodology used by Tesauro in TD-gammon was non-essential for its success. The success came from the setup of co-evolutionary self-play biased by the dynamics of backgammon. Our result is thus similar to the bias found by Mitchell, Crutchfield & Graber in Packard's evolution of Cellular Automata to the "edge of chaos"(Packard, 1988, Mitchell et al., 1993).

TD-Gammon is a major milestone for a kind of evolutionary machine learning in which the initial specification of model is far simpler than expected because the learning environment is specified implicitly, and emerges as a result of the *co-evolution* between a learning system and its training environment: The learner is embedded in a learning environment which responds to its own improvements in a never-ending spiral. While this effect has been seen in population models, it is completely unexpected for a "1+1" hillclimbing evolution.

The process of co-evolution, as seen from evolutionary ecology, is of two species forming part of each other's environment and thus certain characteristics are co-adapted, either to an equilibrium or in a continuing arms-race. In Artificial Life, there have been many recent results on formal and computational ecology models which appear to have arms race dynamics (Holland, 1994, Kauffman, 1993, Ray, 1992, Lindgren, 1992).

The idea of machine learning based on evolution is most often thought of in terms of the genetic algorithm field pioneered by Holland (Holland, 1975). Much of this work however has become focused on optimization to a fixed goal expressed as an absolute fitness function. Using the idea of co-evolution in learning recognizes the difference between an optimization based on absolute fitness and one based on relative fitness (with respect to the rest of the population). This was explored by Hillis (Hillis, 1992) on the sorting problem, by Angeline & Pollack (Angeline and Pollack, 1994) on genetically programmed Tic-Tac-Toe players, on predator/prey games, e.g. (Cliff and Miller, 1995, Reynolds, 1994), and by Juille & Pollack on the intertwined spirals problem (Juille and Pollack, 1995). Rosin & Belew applied competitive fitness to several games (Rosin and Belew, 1995). However, besides Tesauro's TD-Gammon, which has not to date been viewed as an instance of co-evolutionary learning, Sims' artificial robot game (Sims, 1994) is the only other domain as complex as Backgammon to have substantial learning success.

3.2 Learnability and Unlearnability

Learnability can be formally defined as a time constraint over a search space. How hard is it to randomly pick 4000 floating-point weights to make a good backgammon evaluator? It is simply unlearnable.

How hard is it to find weights better than the current set? Initially, when all weights are random, it is quite easy. As the playing improves, we would expect it to get harder and harder, perhaps similar to the probability of a tornado constructing a 747 out of a junkyard. However, if we search in the neighborhood of the current weights, we will find many players which make mostly the same moves but which can capitalize on each other's slightly different choices and exposed weaknesses in a tournament.

Although the setting of parameters in our initial runs involved some guesswork, now that we have a large set of "players" to examine, we can try to understand the phenomenon. Taking the 1000th, 10,000, and 100,000th champions from our run again, we sampled random players in their neighborhoods at different RMS distances to find out how likely is it to find a winning challenger. We took 1000 random neighbors at each of 11 different RMS distances, and played them 8 games against the corresponding champion. Figure 6 plots the average number of games won against the three champions in the range of neighborhoods. This graph demonstrates that as the players improve over time, the probability of finding good challengers in their neighborhood increases. This accounts for why the frequency of successful challenges goes up. Each successive challenger is only required to take the small step of changing a few moves of the champion in order to beat it. Therefore, under co-evolution *the unlearnable becomes learnable* as we convert from a single question to a continuous stream of questions, each one dependent on the previous answer.

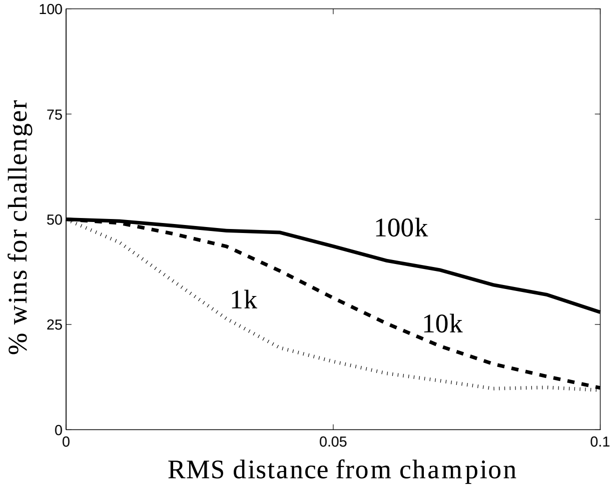

Figure 5: Distance versus probability of random challenger winning against champions at generation 1,000, 10,000 and 100,000.

3.3 Relative versus Absolute Expertise

Does Backgammon allow relative expertise or is there some absolutely optimal strategy? While theoretically there exists a perfect "policy" for backgammon which would deliver the best move for any position, and this perfect policy could exactly rate every other player on a linear scale, in practice it seems there are many relative cycles. Figure 6 shows a graph of the "food chain" over every 5000th player in our sequence of 100,000. By playing them 1000 games against each other and showing the dominance relations with arrows, we can see many relative expertise cycles, albeit with small margins of victory, such as [45,000 beats 70,000 beats 85,000 beats 45,000].

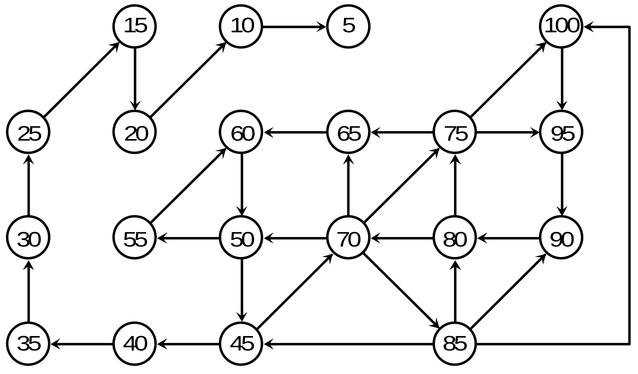

Figure 6: A partial graph of "who eats who", showing for each 5000th player, the immediate dominance relationships.

In spatial studies of iterated prisoners dilemma following (Axelrod, 1984), a stable population of "tit for tat" can be invaded by "all cooperate" which then allows exploitation by "all defect". This kind of relative expertise dynamics, which can be seen clearly in the simple game of rock / paper / scissors (Littman, 1994) might initially seen as very bad for self-play learning, because what looks like an advance might actually lead to a cycle of mediocrity. A small group of champions in a dominance circle might arise and hold a temporal monopoly preventing further advance. On the other hand, it may be that such a basic form of instability prevents the formation of sub-optimal monopolies and allows learning to progress.

3.4 Avoiding Mediocre Stable States

We are not suggesting that 1+1 hillclimbing is an advanced machine learning technique which others should bring to many tasks. Without internal cognition about opponents behavior, co-evolution usually requires a population. Therefore, there must be something about the dynamics of backgammon itself which is helpful because it permitted both TD learning, and hill-climbing, to succeed where they would clearly fail on other tasks and in other games of this scale. If we can understand why the backgammon domain led to successful acquisition of expert strategies from random initial conditions, we might be able to re-cast other domains in its image.

We believe it is not simply the dice rolls which overcome the problems of self-learning. Others have tried to add randomness to deterministic games and have not generally met with success. There is something special about backgammon which we suspect to be more critical; namely, the instability of the game with respect to predictions of winning. What is seen as exciting about backgammon to observers is that the outcome of the game continues to be uncertain until all contact is broken and one side has a clear advantage. There are many situations in backgammon where one dice roll, or an improbable sequence, can dramatically reverse which player is expected to win.

A learning system itself can be viewed as a meta-game, between teacher and student, which are identical in a self-play situation. The teacher's goal is to correct the student's mistakes, while the student's goal is to placate the teacher and avoid correction. A mediocre stable state for a self-learning system can be seen as an equilibrium situation in this meta-game. If the game includes draws, a player which learns to draw itself will have solved its meta-game equilibrium and stop learning. If draws are not allowed, it may be possible for a self-playing learner to collude with itself - to simulate competition while actually cooperating.[2] For example, if slightly suboptimal moves would allow a player to "throw" a game, a player under self-play could find a meta-game equilibrium by alternately throwing games to itself!

We cannot prove it yet, but our hypothesis is that the dynamics of backgammon discussed above actively prevent this sort of collusion from forming in the meta-game of self-learning.

4.0 Conclusions

We have noticed several weaknesses in our player that stem from the training which does not yet reward or punish the double and triple costs associated with severe losses ("gammoning" and "backgammoning") nor take into account the gambling process of "doubling." We are continuing to develop the player to be sensitive to these issues in the game. As noted in (Sutton, 1988), the goals of good game playing and accurate position evaluation are not quite the same. In backgammon, the opponent's stable configuration is immutable in the course of a single move. This effectively partitions the space of positions into a large number of `regions' in such a way that two moves from different regions are never directly compared. Indeed, preliminary studies on the evaluation of end-game situations suggest that our player does not operate by global position analysis. Rather, the opponent's configuration serves to multiplex an evaluation function that is then tuned to discriminate only those positions in the current region.[3]

TD-Gammon remains a tremendous success in Machine Learning, but the causes for its success have not been well understood. We do not claim that our 100,000 generation player is as good as TD-Gammon, ready to challenge the best humans, but it is surprisingly good considering its humble origins from hill-climbing with a relative fitness measure. Interested players can challenge our evolved network using a web browser through our home page at:

`http://www.demo.cs.brandeis.edu`

Replicating some of TD-Gammon's success under a much simpler learning paradigm, we find that the Reinforcement and Temporal Difference methods are not the primary cause for success; rather it is the dynamics of backgammon combined with the power of co-evolutionary learning. If we can isolate the features of the backgammon domain which enable evolutionary learning to work so well, it may lead to a better understanding of the conditions necessary, in general, for complex self-organization.

Acknowledgments

This work is supported by ONR grant N00014-96-1-0418 and a Krasnow Foundation Postdoctoral fellowship. Thanks to Gerry Tesauro for providing PUBEVAL and subsequent means to calibrate it, Jack Laurence and Pablo Funes for development of the WWW front end to our evolved player, and comments from the Brandeis DEMO group, Brendan Kitts, and Chris Langton. The backgammon graphic was captured from xgammon, by Klasen and Steuer, distributed under the GNU general public license.

References

Angeline, P. J. (1994). An alternate interpretation of the iterated prisoner's dilemma and the evolution of non-mutual cooperation.

[2] For example, in a prisoner's dilemma, if the payoff for temptation 7 instead of 5, the long term bonus for alternating defections (3.5) would be more rational than cooperating(3.0). See Angeline, 1994 for related discussion

[3] Further work, incorporating look-ahead, might employ two networks: net A (for pruning) would be tuned to discriminate siblings in the game tree, while net B (for leaf evaluation) is tuned for global position comparison.

Angeline, P. J. and Pollack, J. B. (1994). Competitive environments evolve better solutions for complex tasks. In Forrest, S., editor, *Genetic Algorithms: Proceedings of the Fifth International Conference.*.

Axelrod, R. (1984). *The evolution of cooperation*. Basic Books, New York.

Barto, A., Sutton, R., and Anderson, C. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, 13.

Boyan, J. A. (1992). Modular neural networks for learning context-dependent game strategies. Master's thesis, Computer Speech and Language Processing, Cambridge University.

Cliff, D. and Miller, G. (1995). Tracking the red queen: Measurements of adaptive progress in co-evolutionary simulations. In *Third European Conference on Artificial Life*, pages 200–218.

Epstein, S. L. (1994). Toward an ideal trainer. *Machine Learning*, 15(3).

Fogel, D. B. (1993). Using evolutionary programming to create neural networks that are capable of playing tic-tac-toe. In *International conference on Neural Networks*, pages 875–880. IEEE Press.

Hillis, D. (1992). Co-evolving parasites improves simulated evolution as an optimization procedure. In C. Langton, C. Taylor, J. F. and Rasmussen, S., editors, *Artificial Life II*. Addison-Wesley, Reading, MA.

Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. The University of Michigan Press.

Holland, J. H. (1994). Echoing emergence. In Cowan, G., Pines, D., and Meltzer, D., editors, *Complexity: Metaphors, Models, and Reality*, pages 309–342. Addison-Wesley.

Juille, H. and Pollack, J. (1995). Massively parallel genetic programming. In Kinnear, P. A. . K., editor, *Advances in Genetic Programming II*. MIT Press, cambridge.

Kauffman, S. A. (1993). *The origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press.

Klopf, A. H. (1982). *The Hedonistic Neuron*. Hemisphere Publishing Corporation, Washington, D.C.

Lindgren, K. (1992). Evolutionary phenomena in simple dynamics. In C. Langton, C. Taylor, J. F. and Rasmussen, S., editors, *Artificial Life II*. Addison-Wesley, Reading, MA.

Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning: Proceedings of the Eleventh International Conference*, pages 157–163. Morgan Kaufmann.

International Business Machines, Press Release (Sept. 12, 1995). "IBM's family funpak for OS/2 Warp hits retail shelves".

Michie, D. (1961). Trial and error. In *Science Survey, part 2*, pages 129–145. Penguin.

Mitchell, M., Hraber, P. T., and Crutchfield, J. P. (1993). Revisiting the edge of chaos: Evolving cellular automata to perform computations. *Complex Systems*, 7.

Packard, N. (1988). Adaptation towards the edge of chaos. In Kelso, J. A. S., Mandell, A. J., and Shlesinger, M. F., editors, *Dynamic patterns in complex systems*, pages 293–301. World Scientific.

Ray, T. (1992). An approach to the synthesis of life. In C. Langton, C. Taylor, J. F. and Rasmussen, S., editors, *Artificial Life II*. Addison-Wesley, Reading, MA.

Reynolds, C. (1994). Competition, coevolution, and the game of tag. In *Proceedings 4th Artificial Life Conference*. MIT Press.

Rosin, C. D. and Belew, R. K. (1995). Methods for competitive co-evolution: finding opponents worth beating. In *Proceedings of the 6th international conference on Genetic Algorithms*, pages 373–380. Morgan Kaufman.

Rumelhart, D., Hinton, G., and Williams, R. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

Samuel, A. L. (1959). some studies of machine learning using the game of checkers. *IBM Joural of Research and Development*.

Schraudolph, N. N., Dayan, P., and Sejnowski, T. J. (1994). Temporal difference learning of position evaluation in the game of go. In *Advances in Neural Information Processing Systems*, volume 6, pages 817–824. Morgan Kauffman.

Sims, K. (1994). Evolving 3d morphology and behavior by competition. In *Proceedings 4th Artificial Life Conference*. MIT Press.

Sutton, R. (1984). *Temporal Credit Assignment in Reinforcement Learning*. PhD thesis, University of Massachusetts, Amherst.

Sutton, R. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44.

Tesauro, G. (1992). Practical issues in temporal difference learning. *Machine Learning*, 8:257–277.

Tesauro, G. (1995). Temporal difference learning and TD-Gammon. *Communications of the ACM*, 38(3):58–68.