Co-Evolution in the Successful Learning of Backgammon Strategy

Jordan B. Pollack & Alan D. Blair[1]
Computer Science Department
Volen Center for Complex Systems
Brandeis University
Waltham, MA 02254
{pollack,blair}@cs.brandeis.edu

Abstract

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 and changing weights if the challenger wins. Surprisingly, this worked rather well. We investigate how the peculiar dynamics of this domain enabled a previously discarded weak method to succeed, by preventing suboptimal equilibria in a "meta-game" of self-learning.

Keywords: coevolution, backgammon, reinforcement, temporal difference learning, self-learning

Running Head: CO-EVOLUTIONARY LEARNING

[1] Current address: Dept. of Computer Science and Electrical Engineering, University of Queensland, 4072, Australia, blair@csee.uq.edu.au

Machine Learning **32**, 1998, 1-16

1. Introduction

It took great chutzpah for Gerald Tesauro to start wasting computer cycles on temporal difference learning in the game of Backgammon (Tesauro, 1992). Letting a machine learning program play itself in the hopes of becoming an expert, 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) and used in Donald Michie’s MENACE tic-tac-toe learner (Michie, 1961); but such self-conditioning systems had later been generally abandoned by the field due to problems of scale and weak or non-existent internal representations. Moreover, self-playing learners usually develop eccentric and brittle strategies which appear clever but fare poorly against expert human and computer players.

Yet Tesauro’s 1992 result showed that this self-play approach could be powerful, and after some refinement and millions of iterations of self-play, his TD-Gammon program has become one of the best backgammon players in the world (Tesauro, 1995). His derived 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). Others have replicated this TD result in backgammon both for research purposes (Boyan, 1992) and commercial purposes.

While reinforcement learning has had limited success in other areas (Zhang and Dietterich, 1996, Crites and Barto, 1996, Walker et al., 1994), 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 alone. How is its success to be understood, explained, and replicated in other domains?

Our hypothesis is that the success of TD-gammon is not principally 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, by which the task dynamically changes as the learning progresses. We test this hypothesis by using a much simpler co-evolutionary learning method for backgammon - namely hill-climbing.

2. Implementation Details

We use a standard feedforward neural network with two layers and the sigmoid function, set up in the same fashion as (Tesauro, 1992) 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 has reached 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 makes 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:

1. add gaussian noise to the weights
2. play the network against the mutant for a number of games
3. 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. 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 to avoid this "Buster Douglas Effect":[2]

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:

$$\text{champion} := 0.95*\text{champion} + 0.05*\text{challenger}$$

This idea, similar to the "inertia" term in back-propagation (Rumelhart et al., 1986) was introduced on the assumption that small changes in weights would lead to small changes in decision-making by the evaluation function. So, by just "biting the ear" off the challenger and adding it to the champion, most of the current decisions are preserved , and 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.

Although we would have liked to rank our players against the same players Tesauro used - Neurogammon and Gammontool - these were not available to us. Figure 1 shows the first 35,000 players rated against PUBEVAL, a moderately good public-domain player trained by Tesauro using human expert preferences. There are three things to note: (1) the percentage of wins against PUBEVAL increases from 0% to about 33% 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-

[2] Buster Douglas was world heavyweight boxing champion for 9 months in 1990.

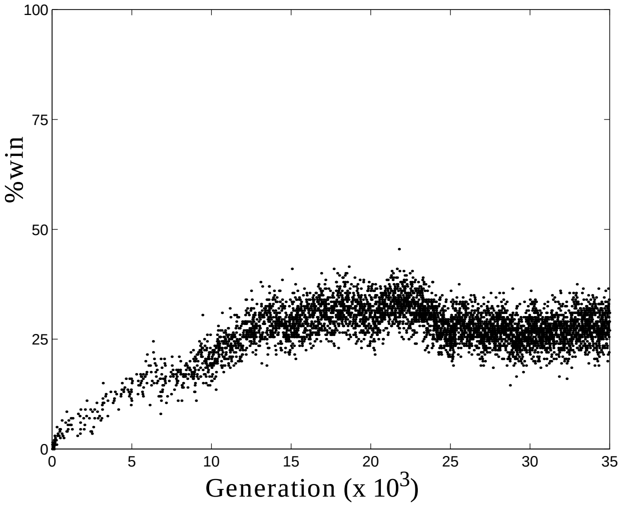


Figure 1: Percentage of wins of our first 35,000 generation players against PUBEVAL. Each match consisted of 200 games.

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 rate of change allows the system to drift, which may account for the subsequent degrading of performanceTo 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 (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). The numbers 10,000 and 70,000 were chosen on an ad hoc basis from observing the frequency of successful challenges and the Buster Douglas effect in this particular run, but later experiments showed how to determine the annealing schedule in a more principled manner (see Section 3.2 below).

After 100,000 games using this simple hill-climb, we have developed a surprising 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 wins for 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 their home board, or first 6 points, and then the dice rolls can be used to remove pieces

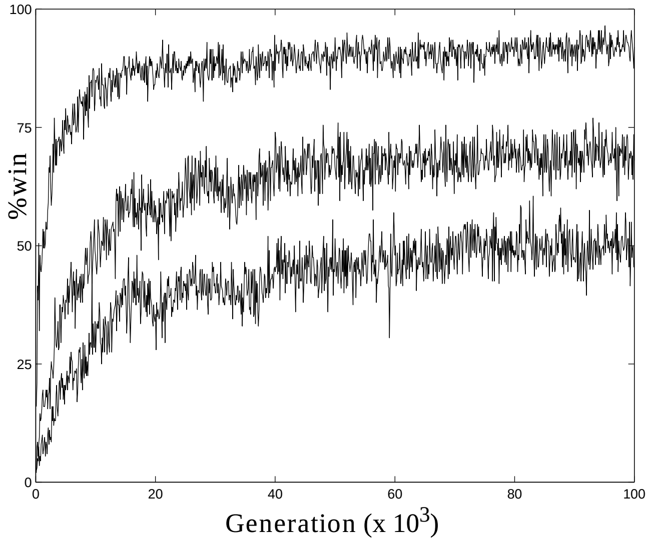
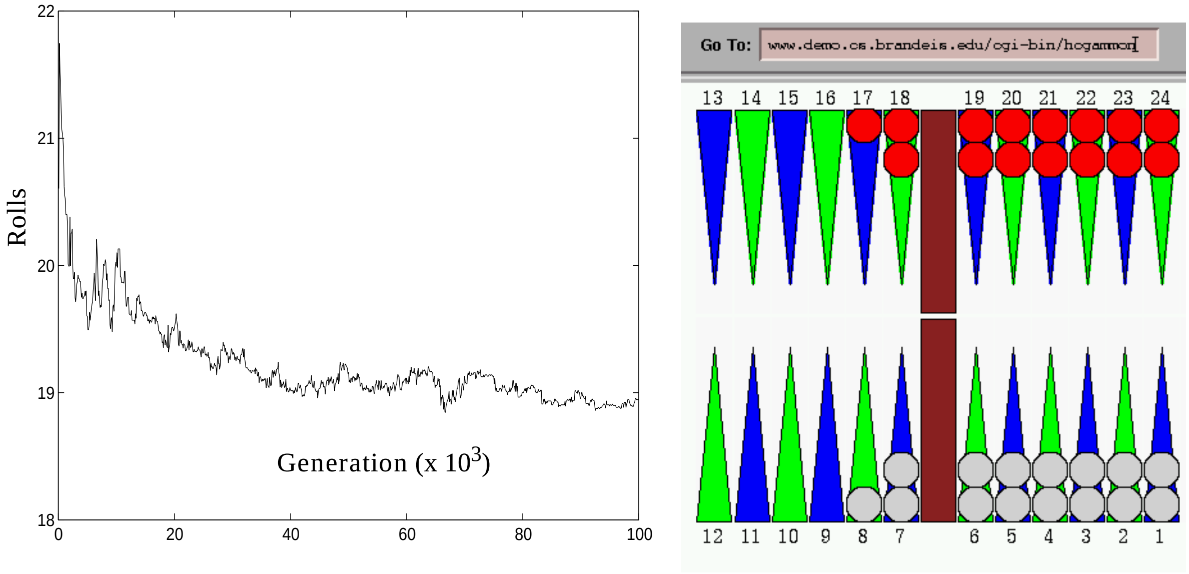


Figure 2: Percentage of wins against benchmark networks 1,000 [upper], 10,000 [middle] and 100,000 [lower]. This shows a noisy but nearly monotonic increase in player skill as evolution proceeds.



Go To: www.demo.cs.brandeis.edu/cgi-bin/hogammon

Figure 3: Average number of rolls to bearoff by each generation, sampled with 200 dice-streams. PUBEVAL averaged 16.6 rolls for the task.

from the board. 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. The graph in Figure 3 shows the average number of rolls to bear-off for each network playing itself using a fixed set of 200 random dice-streams. We note that PUBEVAL is stronger at 16.6 rolls, and will discuss its strengths and those of Tesauro's 1992 results in Section 5.

3. Analysis

3.1. 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 impossible. 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 similar players which make mostly the same moves but which can capitalize on each other's slightly different choices and exposed weaknesses in a tournament. Note that this is a different point than Tesauro originally made - that the feedforward neural network could exploit similarity of positions.

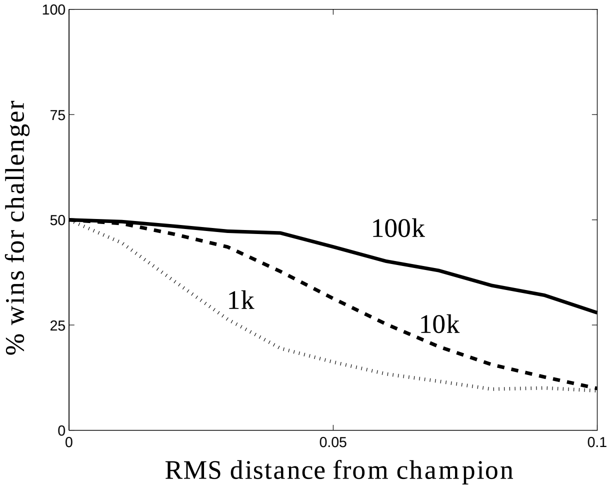


Figure 4: Distance versus probability of random challenger winning against champions at generation 1,000, 10,000 and 100,000.

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 champion networks at generation 1,000, 10,000, and 100,000 from our run, we sampled random players in their neighborhoods at different RMS distances to find out how likely is it to find a winning challenger. A thousand random neighbors at each of 11 different RMS distances played 8 games against the corresponding champion, and Figure 4 plots the fraction of games won by these challengers, as a function of RMS distance. This graph shows that as the players improve over time, the probability of finding good challengers in their neighborhood increases, which accounts for why the frequency of successful challenges goes up.[3] Each successive challenger is only required to

[3] But why does the number of good challengers in a neighborhood go up, and if so, why does our algorithm falter nonetheless? There are several factors which require further study. It may be due to the general growth in weights, to less variability in strategy among mature players, or less ability simply to tell expert players apart with a few games.

take the small step of changing a few moves of the champion in order to beat it. The hope, for co-evolution, is that what was apparently unlearnable becomes learnable as we convert from a single question to a continuous stream of questions, each one dependent on the previous answer.

3.2. Replication Experiments

After our first successful run, we tried to evolve ten more players using the same parameters and the same annealing schedule (10,000 and 70,000), but found that only one of these ten players was even competitive. Closer examination suggested that the other nine runs were failing because they were being annealed too early, before the frequency of successful challenges had reached an appropriate level. This premature annealing then made the task of the challengers even harder, so the challenger success rate fell even lower. We therefore abandoned the fixed annealing schedule and instead annealed whenever the challenger success rate exceeded 15% when averaged over 1000 generations. All ten players evolved under this regime were competitive (though not quite as good as our original player, which apparently benefitted from some extra inductive bias due to having its own tailor-made annealing schedule). Refining other heuristics and schedules could lead to superior players, but was not our goal.

3.3. Relative versus Absolute Expertise

Does Backgammon allow relative expertise or is there some absolutely optimal strategy? Theoretically there exists a perfect “policy” for backgammon which would deliver the minimax optimal move for any position, and this perfect policy could exactly rate every other player on a linear scale, in practice, and especially without running 10,000 games to verify, it seems there are many relative cycles and these help prevent early convergence.

In cellular studies of iterated prisoner’s 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 seem 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 can arise and hold a temporal oligopoly preventing further advance. On the other hand, it may be that such a basic form of instability prevents the formation of sub-optimal oligopolies and allows learning to progress. These problems are specific to non-zero-sum games; in zero sum games, appropriate use of self-play can be shown to converge to optimal play for both parties (Littman96)

4. Discussion

We believe that our evidence of success in learning backgammon using simple hill-climbing in a relative fitness environment indicates that the reinforcement and temporal difference methodology used by Tesauro in his 1992 paper which led to TD-Gammon, while providing some advantage, was not essential for its success. Rather, a major contribution came from the co-evolutionary learning environment and the dynamics of backgammon. Our result is thus similar to the bias found by Mitchell et al in Packard’s evolution of cellular automata to the “edge of chaos” (Packard, 1988, Mitchell et al., 1993).

Obviously, 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 an opponent’s behavior, co-evolution usually requires a population. Therefore, there must be something about the domain itself which is helpful because it permitted both TD learning and hill-climbing to succeed through self-play, where they would clearly fail on other problem-solving tasks of this scale. In this section we discuss some issues about co-evolutionary learning and the dynamics of backgammon which may be critical to learning success.

4.1. Evolution versus Co-evolution

TD-Gammon is a major milestone for a kind of evolutionary machine learning in which the initial specification of the 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 an environment which responds to its own improvements - hopefully in a never-ending spiral, though this is an elusive goal to achieve in practice. While this co-evolutionary effect has been seen in population models, it is completely unexpected for a “1+1” hill-climbing evolution. Co-evolution has been explored on the sorting network problem (Hillis, 1992), on tic-tac-toe and other strategy games (Angeline and Pollack, 1994, Rosin and Belew, 1995, Schraudolph et al., 1994), on predator/prey games (Cliff and Miller, 1995, Reynolds, 1994) and on classification problems such as the intertwined spirals problem (Juille and Pollack, 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 had substantial success.

Since a weak player can sometimes defeat a strong one, it should in theory be possible for a network to learn backgammon in a static evolutionary environment (playing against a fixed opponent) rather than a **co-**evolutionary one (playing against itself). Of course this is not as interesting an acheivement as learning without an expert on hand, and if TD-gammon had simply learned from Neurogammon, it wouldn’t have been as startling a result. In order to further isolate the contribution of co-evolutionary learning, we had to modify our training setup because our original algorithm was only appropriate to self-play. In this new setup the current champion and mutant both play a number of games against the *same opponent* (called the *foil)* with the same dice-streams, and the weights are adjusted only if the champion loses all of these games while the mutant wins all of them. The number of pairs of games was initially set to 1 and incremented whenever the challenger success rate exceeded 15% when averaged over 1000 generations. The lower three plots in Figure 5, which track the performance of this algorithm with each of the three benchmark networks from our original experiments acting as foil, seem to show a relationship between learning rate and probability of winning.

Against a weak foil (1k) learning is fast initially, when the probability of winning is around 50%, then tapers off as this probability increases. Against a strong foil (100k) learning is very slow initially, when the probability of winning is small, but speeds up as it increases towards 50%. All of these evolutionary runs were outperformed by a co-evolutionary version of the foil algorithm (co-ev) in which the champion network itself plays the role of the foil. Co-evolution seems to maintain a high learning rate throughout the run by automatically providing, for each new generation player, an opponent of the appropriate skill level to keep the probability of winning near 50%. Moreover, weaknesses in the foil are less likely to bias the learning process because they can be automatically corrected as the co-evolution proceeds (see also Section 4.3).

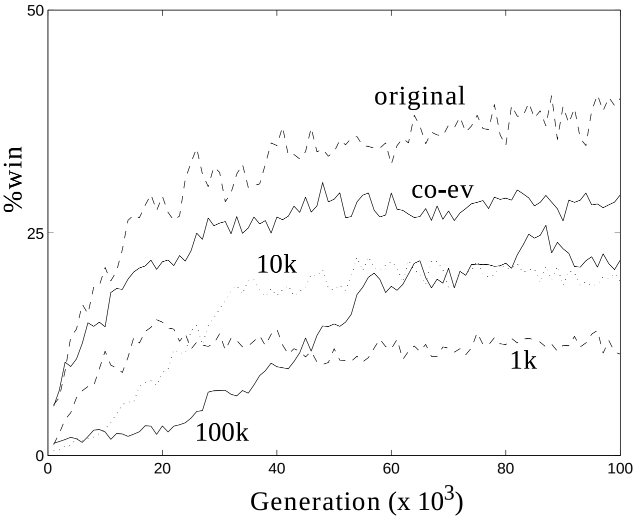


Figure 5: Performance against PUBEVAL of players evolved by playing benchmark networks from our original run at generation 1k, 10k and 100k, compared with a co-evolutionary variant of the same algorithm. Each of these plots is an average over four runs. The performance of our original algorithm is included for comparison.

4.2. The Dynamics of Backgammon

In general, the problem with learning through self-play discovered repeatedly in early AI and ML is that the learner could keep playing the same 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 would then be vulnerable to other programs or human experts. This problem is particularly prevalent in deterministic games such as chess or tic-tac-toe. 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. We have worked on using a population to get around the limitations of self-play (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. Epstein, 1994, has studied a mix of training using self-play, random testing, and playing against an expert in order to better understand these aspects of game learning.

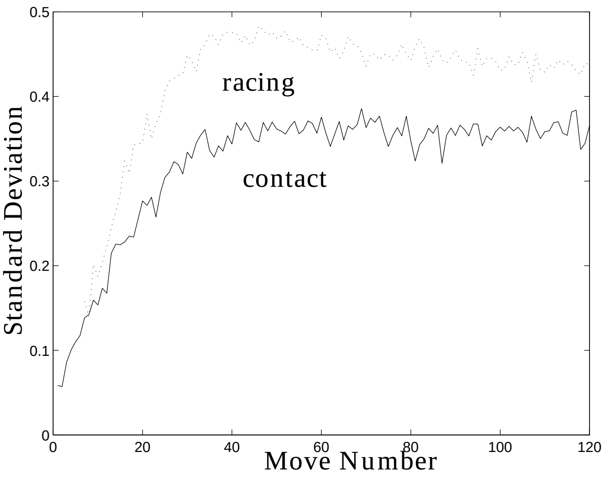


Figure 6: (a) Standard deviation in the probability of winning for contact positions and racing positions.

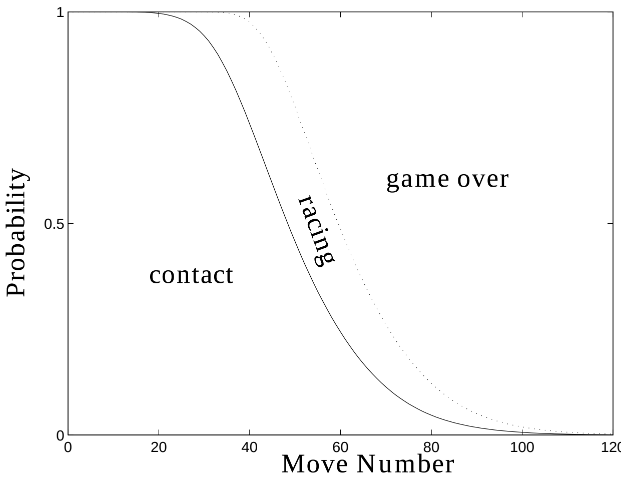


Figure 6: (b) Probability of a game still being in the contact or racing stage at move *n*.

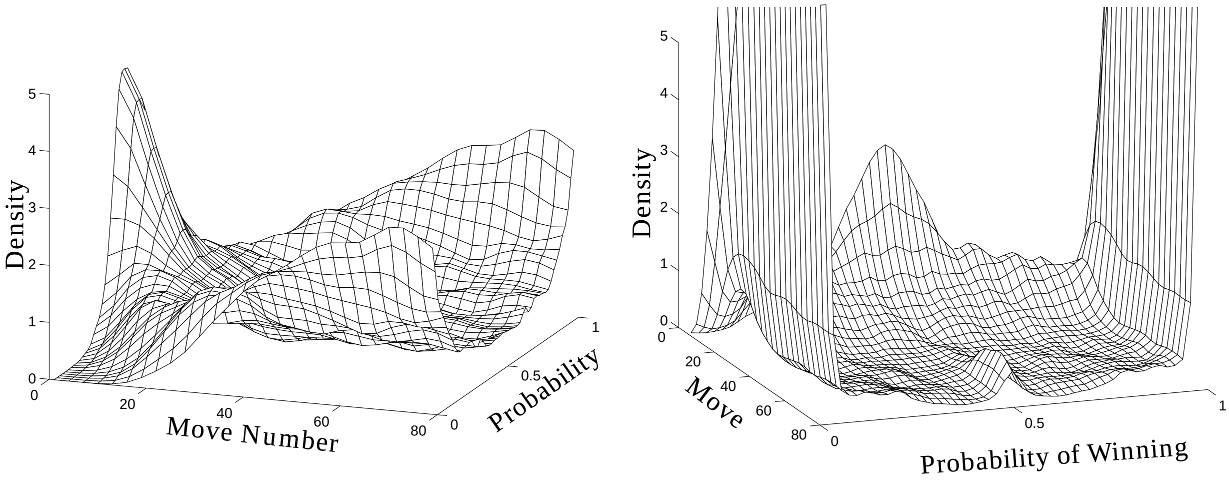


Figure 7: Smoothed distributions of the probability of winning as a function of move number, for contact positions (left) and racing positions (right).

We believe it is not enough to add randomness to a game or to force exploration through alternative training paradigms. There is something critical about the dynamics of backgammon which sets its apart from other games with random elements like Monopoly - namely, that the outcome of the game continues to be uncertain until all contact is broken *and* one side has a clear advantage. In Monopoly, an early advantage in purchasing properties leads to accumulating returns. What many observers find exciting about backgammon, and what helps a novice sometimes overcome an expert, is the number of situations where one dice roll, or an improbable sequence, can dramatically reverse which player is expected to win.

In order to quantify this “reversibility” effect we collected some statistics from games played by our 100,000th generation network against itself. For each *n* between 0 and 120 we collected 100 different games in which there was still contact at move *n*, and, for *n*>6, 100 other games which had reached the racing stage by move *n* (but were still in progress). We then estimated the probability of winning from each of these 100 positions by playing out 200 different dice-streams. Figure 6 shows the standard deviation of this probability (assuming a mean of 0.5) as a function of n, as well as the probability of a game still being in the contact or racing stage at move n. Figure 7 shows the distribution in the probability of winning, as a function of move number, symmetrized and smoothed out by convolution with a gaussian function.

These data indicate that the probability of winning tends to hover near 50% in the early stages of the game, gradually moving out as play proceeds, but typically remaining within the range of about 15% to 85% as long as there is still contact, thus allowing a reasonable chance for a reversal. These numbers could be different for other players, less reversability for stronger players perhaps and more for weaker ones, but we believe the effect remains an integral part of the game dynamics regardless of expertise. Our conjecture is that these dynamics facilitate the learning process by providing, in almost every situation, a nontrivial chance of winning and a nontrivial chance of losing, therefore *potential to learn from the consequences of the current move*. This is in deep contrast to many other domains in which early blunders could lead to a hopeless situation from which learning is virtually impossible because the reward has already become effectively unattainable. It seems this feature of backgammon may also be shared by other tasks for which TD-learning has been successful (Zhang and Dietterich, 1996, Crites and Barto, 1996, Walker et al., 1994).

4.3. Avoiding Suboptimal Equilibria in the Meta-Game of Learning

A learning system can be viewed as an interaction between teacher and student in which the teacher's goal is to expose the student's weaknesses and correct them, while the student's goal is to placate the teacher and avoid further correction.

We can build a model of this teacher / student interaction as a formal game, which we will call the *Meta-Game of Learning* (MGL) to avoid confusion with the *game* being learned. In this meta-game, the teacher T presents the student S with a sequence of questions Q_i prompting responses R_i from the student. (In the backgammon domain, all the questions and responses would be legal positions, rolls and moves). S and T each receive payoffs in the process, which they attempt to maximize through their choices of questions and answers, and their limited abilities at self-modification.

We generally assume the goal of learning is to prepare the student for interaction with a complex environment E that will provide an objective measure of its performance.[4] E and T thus play similar roles but are not assumed to be identical. The question then is: Can we find a payoff matrix for S and T which will enable the performance of S to continually improve (as measured by E)? If the rewards for T are too closely correlated with those for S, T may be tempted to ask questions that are too easy. If they are anti-correlated (for example if T=E), the questions might be too difficult. In either case it will be hard for S to learn (see Section 4.1).

[4] For a general theory of evolution or self-organization, E is not necessary.

An attractive solution to this problem is to have two or more students play the role of teacher for each other, or indeed a single student act as its own teacher, thus providing itself with questions that are always at the appropriate level of difficulty. The dynamics of the MGL, under such a self-teaching or co-evolutionary situation, would hopefully lead to a continuing spiral of improvement but may instead get bogged down by antagonistic or collusive dynamics, depending on the payoff structure.

In our hillclimbing setup we may think of the mutant (teacher) trying to gain advantage (adjustment in the weights) by exploiting weaknesses in the champion, while the champion (student) is trying to avoid such an adjustment by not allowing its weaknesses to be exploited. Since the student and teacher are of approximately equal ability, it is to the advantage of the student to narrow the scope of the search, thus limiting the domain within which the teacher is able to look for a weakness. In most games, such as chess or tic-tac-toe, the student could achieve this by aiming for a draw instead of a win, or by always playing a particular style of game. If draws are not allowed, the teacher and student may figure out some other way to *collude* with each other - for example, each “throwing” alternate games (Angeline, 1994) by making a suboptimal sequence of early moves. These effects in self-learning systems, which may appear as early convergence in evolutionary algorithms, narrowing of scope, drawing or other collusion between teacher and student, are in fact *Nash equilibria in the MGL*, which we call *Mediocre Stable States.*[5]

Our hypothesis is that certain features of backgammon operate against the formation of mediocre stable states in the MGL: backgammon is *ergodic* in the sense that any position can be reached from any other position[6] by some sequence of moves, and the dice rolls apparently create enough randomness to prevent either player from following a strategy that narrows the scope of the game appreciably. Moreover, early suboptimal moves are unlikely to provide the opponent with an easy win (see Section 4.2), so collusion by the throwing of alternate games is prevented.

Mediocre stable states can also arise in human education systems, for example when the student gets all the answers right and rewards the teacher with positive teaching evaluations for not asking harder questions. In further work, we hope to apply the same kind of MGL equilibrium analysis to issues in human education.

5. Conclusions

TD-Gammon remains a tremendous success in Machine Learning, but the causes for its success have not been well understood. The fundamental research in Tesauro’s 1992 paper which was the basis for TD-Gammon, reportedly beat Sun’s Gammontool 60-65% of the time (depending on number of hidden units) and achieved parity against Neurogammon 1.0.

Following this seminal 1992 paper, Tesauro incorporated a number of hand-crafted expert-knowledge features, eventually engineering a network which achieved world

[5] MSS follows Maynard Smith’s ESS (Maynard Smith, 1982)

[6] with the exception of racing situations and positions with some pieces out of play.

master level play (Tesauro, 1995). These features included concepts like existence of a prime, probability of blots being hit, and probability of escape from behind the opponent's barrier. The evaluation function was also improved using multiple ply search.

The best players we've been able to evolve can win about 45% of the time against PUBEVAL, which we believed to be about the same level at Tesauro's 1992 networks. Because Tesauro had never compared his 1992 networks to PUBEVAL, and because he used Gammontool's heuristic endgame in the ratings, the level of play achieved by these players has been somewhat murky:

> "the testing procedure is to play out the game with the network until it becomes a race, and then use Gammontool's algorithm to move for both sides until the end. This also does not penalize the TD net for having learned rather poorly the racing phase of the game."(p 272)

When we compare our network's performance to PUBEVAL, it must be noted that we use our network's own (weak) endgame, rather than substituting in a much stronger expert system like Gammontool. Gerald Tesauro, in a commentary in this issue, has graciously cleared up the matter of comparing PUBEVAL to his 1992 results, and differs somewhat from our conclusions below.

There are two other phenomena fom the 1992 paper, which are relevant to our work:

> "Performance on the 248-position racing test set reached about 65%. (This is substantially worse than the racing specialists described in the previous section.)" (p. 271)

> "The training times ...were on the order of 50,000 training games for the networks with 0 and 10 hidden units, 100,000 games for the 20-hidden unit net, and 200,000 games for the 40-hidden unit net." (p. 273)

Because we achieve similar levels of skills, and observe these same phenomena in training, endgame weakness, and convergence, we believe we have achieved results substantially similar to Tesauro's 1992 result, without any advanced learning algorithms. We could make stronger players by tuning the learning parameters, and adding more input features, but that is not our point.

We do not claim that our 100,000th generation player is anywhere near as good as the current enhanced versions of 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. Tuning our parameters or adding more input features would make more powerful players, but that was not the point of this study.

We also do not claim there is anything "wrong" with TD learning, or that hillclimbing is just as good as reinforcement learning in general! Of course it isn't! Our point is that once an environment and representation have been refined to work well with a machine learning method, it should be benchmarked against the weakest possible algorithm so that credit for learning power can be properly distributed.

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. Interested players can challenge our 100,000th network using a web browser through our home page at:

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

In conclusion, replicating some of Tesauro’s 1992 TD-Gammon 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 co-evolutionary and reinforcement 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 was 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, the anonymous referees, Justin Boyan, Tom Dietterich, Leslie Kaelbling, Brendan Kitts, Michael Littman, Andrew Moore, Rich Sutton and Wei Zhang.

References

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 Inter national Conference.*

Angeline, P. J. (1994). An alternate interpretation of the iterated prisoner’s dilemma and the evolution of non-mutual cooperation. In Brooks, R. and Maes, P., editors, *Proceedings 4th Artificial Life Conference*, pages 353–358. MIT Press.

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

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.

Crites, R. and Barto, A. (1996). Improving elevator performance using reinforcement learning. In Touretzky, D., Mozer, M., and Hasselmo, M., editors, *Advances in Neural Information Processing Systems*, volume 8, pages 1024–1030.

International Business Machines (Sept. 12, 1995). IBM’s family funpak for OS/2 warp hits retail shelves.

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

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.

Littman, M. L. (1996) Algorithms for Sequential Decision Making. Ph.D. Dissertation, Providence: Brown University Computer Science Department..

Maynard Smith, John (1982). *Evolution and the Theory of Games,* Cambridge: Cambridge University Press

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.

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 Journal 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 Brooks, R. and Maes, P., editors, *Proceedings 4th Artificial Life Conference*. MIT Press.

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

Tesauro, G. (1989). Connectionist learning of expert preferences by comparison training. In Touretzky, D., editor, *Advances in Neural Information Processing Systems*, volume 1, pages 99–106, Denver 1988. Morgan Kaufmann, San Mateo.

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.

Walker, S., Lister, R., and Downs, T. (1994). Temporal difference, non-determinism, and noise: a case study on the ‘othello’ board game. In *International Conference on Artificial Neural Networks 1994*, pages 1428–1431, Sorrento, Italy.

Zhang, W. and Dietterich, T. (1996). High-performance job-shop scheduling with a time-delay td(lambda) network. In Touretzky, D., Mozer, M., and Hasselmo, M., editors, *Advances in Neural Information Processing Systems*, volume 8.