An Investigation of an Adaptive Poker Player

Graham Kendall and Mark Willdig

The University of Nottingham, School of Computer Science & IT, Jubilee Campus,
Wollaton Road, Nottingham NG8 1BB, UK
gxk@cs.nott.ac.uk, Mark.Willdig@siemenscomms.co.uk

Abstract : Other work has shown that adaptive learning can be highly successful in developing programs which are able to play games at a level similar to human players and, in some cases, exceed the ability of a vast majority of human players. This study uses poker to investigate how adaptation can be used in games of imperfect information. An internal learning value is manipulated which allows a poker playing agent to develop its playing strategy over time. The results suggest that the agent is able to learn how to play poker, initially losing, before winning as the players strategy becomes more developed. The evolved player performs well against opponents with different playing styles. Some limitations of previous work are overcome, such as deal rotation to remove the bias introduced by one player always being the last to act. This work provides encouragement that this is an area worth exploring more fully in our future work.

1. Introduction

Game playing has a long research history. Chess has received particular interest culminating in Deep Blue beating Kasparov in 1997, albeit with specialized hardware (Hamilton, 1997) and brute force search. However, although arguably, being a 'solved game' chess still receives interest as researchers turn to adaptive learning techniques which allow computers to 'learn' to play chess, rather than being 'told' how it should play (Kendall, 2001). Adaptive learning was being used for checkers as far back as the 1950's with Samuel's seminal work (1959, re-produced in Samuel, 2000). Checkers research would lead to Jonathan Schaeffer developing Chinook, which claimed the world title in 1994 (Schaeffer, 1996). Like Deep Blue, it is arguable if Chinook used AI techniques. Chinook had an opening and ending database. In certain games it was able to play the entire game from these two databases. If this could not be achieved, a form of mini-max search, with alpha-beta pruning was used. Despite Chinook becoming the world champion, the search has continued for an adaptive checkers player. Chellapilla and Fogel's (Chellapilla, 2000) Anaconda was named due to the strangle hold it placed on its opponent. It is also named Blondie24, this being the name it used when competing in internet games (Fogel, 2001). Anaconda uses an artificial neural network (ANN), with 5000 weights, which are evolved by an evolutionary strategy. The inputs to the ANN are the current board position and it outputs a value which is used in a mini-max search. During the training period, using co-evolution, the program is given no information other than whether it won or lost.

Once Anaconda is able to play at a suitable level, it often searches to a depth of 10, but depths of 6 and 8 are also common in play. Anaconda has been available to the delegates at the Congress on Evolutionary Computing (CEC) conference for the past two years (CEC'00, San Diego and CEC'01, Seoul) with Fogel offering a prize of $100 (CEC'00) and $200 (CEC'01) to anybody who could defeat it. The prize remains unclaimed and at the next conference (CEC'02, Hawaii), the prize rises to $300.

Poker also has an equally long research history with von Neumann and Morgensten (von Neumann, 1944) experimenting with a simplified, two-player, version of poker.

Findler (Findler, 1977) studied poker, over a 20 year period. He also worked on a simplified game, based on 5-card draw poker with no ante and no consideration of betting position due to the computer always playing last. He concluded that dynamic and adaptive algorithms are required for successful play and static mathematical models were unsuccessful and easily beaten.

In more recent times three research groups have been researching poker. Jonathan Schaeffer (of Chinook fame) and a number of his students have developed ideas which have led to Loki, which is, arguably, the strongest poker playing program to date. It is still a long way from being able to compete in the World Series of Poker (WSOP), an annual event held in Las Vegas, but initial results are promising. Schaeffer's work concentrates on two main areas (Billings, 1998a and Schaeffer, 1999). The first research theme makes betting decisions using probabilistic knowledge (Billings, 1999) to determine which action to take (fold, call or raise) given the current game state. Billings et. al. also uses real time simulation of the remainder of the game that allows the program to determine a statistically significant result in the program's decision making process. Schaeffer's group also uses opponent modeling (Billings, 1998b). This allows Loki to maintain a model of an opponent and use this information to decide what betting decisions to make.

Koller and Pfeffer (Koller, 1997), using their Gala system, allow games of imperfect information to be specified and solved, using a tree based approach. However, due to the size of the trees they state "…we are nowhere close to being able to solve huge games such as full-scale poker, and it is unlikely that we will ever be able to do so."

Luigi Barone and Lyndon While recognise four main types of poker player; Loose, Tight, Passive, and Aggressive. These characteristics are combined to create the four common types of poker players: Loose Passive, Loose Aggressive, Tight Passive and Tight Aggressive players (Barone & While, 1999; 2000). A **Loose Aggressive** player will overestimate their hand, raising frequently, and their aggressive nature will drive the pot higher, increasing their potential winnings. A **Loose Passive** player will overestimate their hand, but due to their passive nature will rarely raise, preferring to call and allow other players to increase the pot. A **Tight Aggressive** player will play to close constraints, participating in only a few hands which they have a high probability of winning. The hands they do play, they will raise frequently to increase the size of the pot. A **Tight Passive** player will participate in few hands, only considering playing those that they have a high probability of winning. The passive nature implies that they allow other players to drive the pot, raising infrequently themselves.

In their first paper Barone and While (Barone, 1998) suggest evolutionary strategies as a way of modelling an adaptive poker player. They use a simple poker variant where each player has two private cards, there are five community cards and one round of betting. This initial work incorporates three main areas of analysis; hand strength, position and risk management. Two types of tables are used, a loose table and a tight table. The work demonstrates how a player that has evolved using evolutionary strategies can adapt its style to the two types of table.

In (Barone, 1999) they develop their work by introducing a *hypercube* which is an *n* dimensional vector, used to store candidate solutions. The hypercube has one dimension for the betting position (early, middle and late) and another dimension for the risk management (selected from the interval 0..3). At each stage of the game the relevant candidate solutions are selected from the hypercube (e.g. middle betting position and risk management 2) and the decision is made whether to fold, call or raise. To make the decision the hypercube entry holds seven real valued numbers which are used as constants to three functions (fold, call and raise). In effect, the functions lead to a probability of carrying out the relevant action. It is the seven real values that are evolved depending on whether the player won the hand or not. Barone reports that this poker player improves on the 1998 version. Their 2000 paper (Barone, 2000) extends the dimensions of the hypercube to include four betting rounds (pre-flop, post-flop, post-turn and post-river) and an opponent dimension so that the evolved player can choose which type of player it is up against. The authors report this player out performs a competent static player.

Poker, being a game of imperfect information, is interesting as a game for the basis of research. Unlike chess and checkers, poker has some information that is unseen. Poker also contains other unknowns such as the playing styles of the other players who may use bluffing (and double bluffing) during the course of the game. These elements add to the research interest. Unlike complete information games where the techniques to solve the games (computational power allowing) have been known and understood for a long time (such as mini-max search and alpha-beta pruning), games of imperfect information have not received the same sort of analysis and, doing so, could prove relevant to many other areas such as economics, on-line auctions and negotiating.

2. The Rules of Poker

The exact rules for poker can be found in many poker books (see, for example, Sklansky, 1994; 1996) and we simply give here the basic rules of one variant (Texas Hold 'Em) so that the reader is able to follow the remainder of this paper. Each player is dealt two cards. These are private cards, only being visible to the player receiving those cards. These cards are normally referred to as hole cards. A round of betting follows this initial deal. Next, three community cards (called the flop) are dealt, face up, in the middle of the table. These cards are used by every player to make the best five card poker hand, using their hole cards. A round of betting follows the flop. Next, another community card (called the turn) is dealt face up in the middle of the table. Another round of betting follows. Finally, another community card (called the river)

is dealt and a final round of betting follows. Once this final round of betting has taken place, assuming there are two or more players who still have an interest in the pot, the cards are shown and the highest poker hand wins. In forming a poker hand, the players can use any combination of their two hole cards and the five community cards to make the best five card poker hand. The various poker hands are as follows, in descending order.

Royal Flush: Ten, Jack, Queen, King and Ace, all in the same suit.
Straight Flush: any sequence of five cards, all of the same suit.
Four of a Kind: four cards having the same value, one from each suit.
Full House: three cards of the same value combined with two cards of the same value. For example, Three 2's and a pair of Queens.
Flush: all five cards have the same suit.
Straight: all five card values are in sequence, made up from at least two suits.
Three of a Kind: three cards all having the same value.
Two Pairs: two cards of the same value, combined with another two card of the same value. For example, two 9's and two 3's.
A Pair: two cards having the same value.
Single Card: the highest value card is used to value the hand.

When betting, the players have three choices to make. They can either fold (throw in their cards and relinquish all claims to the money in the pot), they can call (match the amount of money bet so far) or they can raise (increase the current bet, thus forcing all the other players to match this amount or fold). To start the betting it is usual to put in some form of ante. This is a mechanism to start the betting by giving the players an interest in the pot.

In this paper we have not implemented a full version of Texas Hold 'Em, preferring a version of poker, where the players are dealt five cards and, after a round of betting, are allowed to trade two cards before a final round of betting. This version is known as draw poker and was considered as a suitable test bed for this initial investigation.

3. Experiments

We have implemented the four playing styles (loose passive, loose aggressive, tight passive and tight aggressive) described above so that we can sit each of them at our tables and find out if our approach can adapt to each of these styles. Each playing style will play to a specific set of rules using the value of their current hand and the current value of the pot to decide whether to fold, call, or raise.

Table 1: The Loose Aggressive Players Strategy

1st Round Strategy		
Hand From	**Hand To**	**Action**
0	Pair 8's	Fold
Pair 9's	Pair K's	Call
Pair A's	2 Pairs Ace High	Raise 5 if Pot <= 100 otherwise Call
Three 2's	Three 4's	Raise 10 if Pot <= 150 otherwise Call
Three 5's	Three J's	Raise 15 if Pot <= 200 otherwise Call
Three Q's	Three A's	Raise 20 if Pot <= 250 otherwise Call
Straight	Royal Flush	Raise 25 if Pot <= 300 otherwise Call
2nd Round Strategy		
Hand From	**Hand To**	**Action**
0	Pair 8's	Fold
Pair 9's	Three 6's	Call
Three 7's	Three A's	Raise 5 if Pot <= 150 otherwise Call
Straight 6 High	Straight A High	Raise 10 if Pot <= 200 otherwise Call
Flush 6 High	Full House A High	Raise 15 if Pot <= 250 otherwise Call
Four 2's	Four A's	Raise 20 if Pot <= 300 otherwise Call
Straight Flush 6 High	Royal Flush	Raise 25 if Pot <= 400 otherwise Call

Table 2: The Loose Passive Players Strategy

1st Round Strategy		
Hand From	**Hand To**	**Action**
0	Pair 8's	Fold
Pair 9's	Three J's	Call
Three Q's	Flush A High	Raise 5 if Pot <= 100 otherwise Call
Full House 2 High	Royal Flush	Raise 10 if Pot <= 150 otherwise Call
2nd Round Strategy		
Hand From	**Hand To**	**Action**
0	Pair 8's	Fold
Pair 9's	Three A's	Call
Straight 6 High	Straight A High	Raise 5 if Pot <= 100 otherwise Call

Flush 6 High	Four 5's	Raise 10 if Pot <= 150 otherwise Call
Four 6's	Royal Flush	Raise 15 if Pot <= 200 otherwise Call

Table 3: The Tight Aggressive Players Strategy

1st Round Strategy		
Hand From	**Hand To**	**Action**
0	Pair A's	Fold
2 Pairs 3 High	Three 4's	Call
Three 5's	Three J's	Raise 5 if Pot <= 150 otherwise Call
Three Q's	Three A's	Raise 15 if Pot <= 200 otherwise Call
Straight 6 High	Royal Flush	Raise 25 if Pot <= 300 otherwise Call
2nd Round Strategy		
Hand From	**Hand To**	**Action**
0	Pair A's	Fold
2 Pairs 3 High	Three 10's	Call
Three J's	Three A's	Raise 5 if Pot <= 150 otherwise Call
Straight 6 High	Straight A High	Raise 10 if Pot <= 200 otherwise Call
Flush 6 High	Full House A High	Raise 15 if Pot <= 250 otherwise Call
Four 2's	Four A's	Raise 20 if Pot <= 300 otherwise Call
Straight Flush 6 High	Royal Flush	Raise 25 if Pot <= 400 otherwise Call

Table 4: The Tight Passive Players Strategy

1st Round Strategy		
Hand From	**Hand To**	**Action**
0	Pair A's	Fold
2 Pairs 3 High	Straight A High	Call
Flush 6 High	Four 5's	Raise 5 if Pot <= 100 otherwise Call
Four 6's	Straight Flush A High	Raise 10 if Pot <= 150 otherwise Call
Royal Flush	Royal Flush	Raise 15 if Pot <= 200 otherwise Call
2nd Round Strategy		
Hand From	**Hand To**	**Action**
0	2 Pairs A High	Fold
Three 2's	Three A's	Call
Straight 6 High	Flush A High	Raise 5 if Pot <= 100 otherwise Call

Full House 2 High	Four A's	Raise 10 if Pot <= 150 otherwise Call
Straight Flush 6 High	Royal Flush	Raise 15 if Pot <= 250 otherwise Call

In order to test our adaptive poker player, we adopt the following rules. At the start of each hand each player places an ante of one unit into the pot. There will be two rounds of betting. Each round will pass around the table a maximum of three times, unless every player except one decides to fold, or all players call.

Non-evolving players will play to the strategies described above (tables 1 thru 4). The evolving player considers three factors when deciding whether to fold, call or raise, these being hand strength, the number of players left at the table and the money in the pot. As well as these factors, a learning value will be evolved and will also dictate the actions of the evolving player. The learning value is manipulated throughout the training period of the evolving player, assisting in its decision whether to fold, call or raise. There is a learning value (ranging over the interval 1..10) associated with each possible hand. The algorithm, in deciding whether to fold, call or raise is as follows.

```
If lv < 6 then FOLD
    elseif lv >= 6 AND lv < 8 then CALL
    elseif lv >= 8 then
        ac = (lv/LOG(pv)/(np/lv)
        if ac < 10 then CALL
        else RAISE by SQRT(ac) * w
```

where

lv = learning value for the hand being played
pv = the current value of the pot
np = number of players left in the current game
ac = players action, returning a value greater than 0
w = a weighting factor dependant on *lv*

```
        if lv < 8 then w = 1
        if 8 < lv > 8.99 then w = 3
        if 9 < lv > 9.99 then w = 4
        if lv > 9.99 then w = 5
```

Example of the use of this algorithm is shown in figures 1 thru 3.

lv = 8, *np* = 5, *pv* = 50 *ac* = 7.53, player will call	*lv* = 8, *np* = 3, *pv* = 50 *ac* = 12.57, player will raise 11 units

Figure 1: Player Calls **Figure 2:** Player Raises 11 units

lv = 9, *np* = 3, *pv* = 50 *ac* = 15.89, player will raise 16 units

Figure 3: Player Raises 16 units

Figure 1 has more players participating in the current game resulting in the evolving player calculating it is less likely to win the pot so it decides to call. A

reduced number of players contesting the pot increases the evolving players chances of winning, influencing its action to raise, as shown in figure 2.

Figures 3 highlights the difference in the raise value when the learning value is adjusted between the values of 8 (figure 2) and 9 (figure 3). The learning value of 9 is associated with better hand rankings, thus there is a better chance of winning. As the possibility of winning is increased with the higher learning value, more emphasis is placed on driving the pot harder, raising it, in the hope of increased winnings.

The learning values, *lv*, are associated with hand strength. Each hand is given a value, *lv*, which is used in the formulae outlined above. Initially, all values are set to 10, so that the evolving player will raise every time. Using this method every hand is assumed to be good until we find out otherwise. This is seen as preferable to assuming every hand is bad until we know otherwise as this was one of the criticisms that Barone made of his own work. He experienced a royal flush so infrequently that he folded it when one did appear, on the basis that the program had not learnt that this was good hand.

Our adaptation technique is simple. If the evolving player wins a game, with other players either calling or raising, then the learning value is incremented by 0.1, but will never exceed 10. If the player folds after raising or calling the learning value will be decreased by 0.1 unless it is already zero.

All our tests have five players seated at the table. Player 1, except for initial testing to confirm the system is operating fairly, will always be the evolving player. Players 2 thru 5 will be the non-evolving players as defined in tables 1 thru 4. Each player will have 10,000 units allocated to them making a total of 50,000 units at the table. The evolving player will have its learning values initialised to ten at the start of each training session. The deal and betting will move clockwise around the table. The player to the dealers left will always play first. Initial testing, using tables of similar players with no evolving player, showed that the game was fair, in that no single player or position dominated.

The evolving player must initially be trained by allowing manipulation of the learning values. It is interesting to monitor the evolving player during this learning period.

Initially, the evolving player plays every hand. This can be seen in epochs (hands played) 1 to 100 in table 5. After this, learning values are being reduced and the number of hands played gradually decreases.

Table 5 : Number of Games Played and Won During the Training Period

	25	**50**	**75**	**100**	**200**	**300**	**400**	**500**	**1000**	**1500**	**2000**	**2500**	**3000**
Hands Played	25	50	75	100	195	257	324	393	647	925	1187	1421	1664
Hands Won	5	12	19	28	60	81	102	123	216	312	409	484	560
% Played	100	100	100	100	97.5	85.6	81.0	78.6	64.7	61.6	59.3	56.8	55.5
% Won	20.0	24.0	25.3	28.0	30.7	31.5	31.5	31.3	33.3	33.7	34.5	34.0	33.6

Table 5 also shows another interesting result. The percentage of hands played gradually falls, yet the number of hands won increases, demonstrating that the player is learning. At the end of the training period less than half of the hands are being played with over 33% of them winning.

Next we consider how the training process affects the number of units the evolving player wins or loses over a specific time period. As highlighted above, the player will soon realise that playing in every pot (and raising it, due to the high learning values) is not the best method of playing poker. As the player begins to adapt, the losing streak eventually levels off and changes into a winning streak, creating a better player, maximising its winnings against a variety of players. Figures 4 and 5 show how the program learns to play poker against two different tables, where a table consists of players of the same playing style.

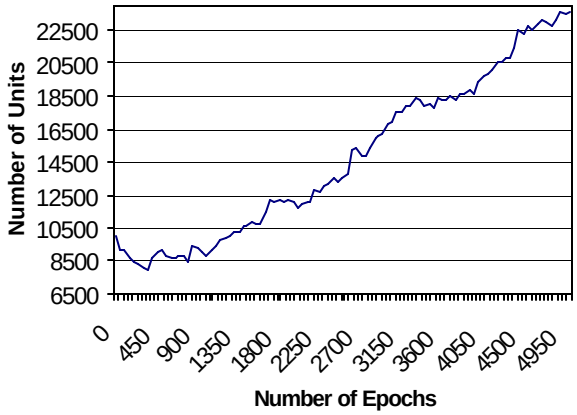

Figure 4: Learning Curve, against Loose Aggressive Players

From figure 4 and 5 an obvious losing trend can be seen, particularly in figure 5. The evolving player initially loses, before the graph levels off and then rises. Figure 4 has an initial losing period until epoch 350, when the losing streak begins to level off as the learning starts to have an impact. By epoch 1300 the learning process is almost complete, and the program begins to win and eventually wins more units than it initially started with. Figure 5 takes slightly longer to learn, the initial losing streak continues until epoch 650. This losing streak levels off until epoch 2650, when the learning process allows the player to regain its earlier losses and by epoch 3600 the evolving player is back in the black. Figure 4 and 5 emphasises that learning against a table of Loose Aggressive players is quicker than that of a table of Tight Aggressive players due to the fact that tight players play less hands themselves. In addition, the evolving player wins more money against a loose player than a tight player.

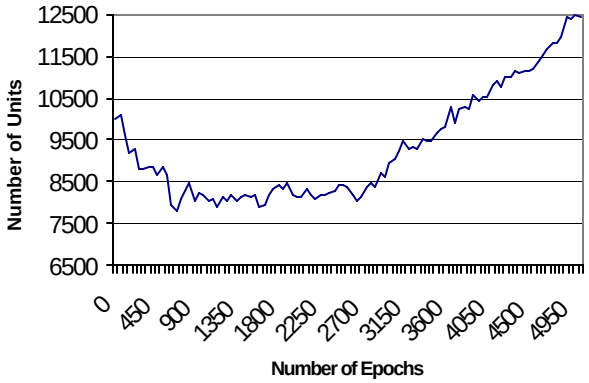

Figure 5: Learning Curve, against Tight Aggressive Players

Table 6 shows the results when the evolved player (i.e. after training) is played against players of a single style, after a training period of 5000 epochs. A value of 5000 was chosen as it appears that a player can be trained in about 4000 epochs (figures 4 and 5). The figure of 5000 was chosen as an insurance against slow learning due to an unfavourable distribution of cards. The results in table 6 are played over 1000 hands, averaged over five runs

Table 6: Units Won by each player (Player 1 is the evolving player)

Player	Loose Aggressive (^{played}/won)	Tight Passive (^{played}/won)	Loose Passive (^{played}/won)	Tight Aggressive (^{played}/won)
1	13342 $^{507}/_{153}$	11123 $^{355}/_{196}$	15632 $^{500}/_{157}$	10816 $^{446}/_{167}$
2	9696 $^{362}/_{196}$	9536 $^{122}/_{83}$	8520 $^{376}/_{200}$	9527 $^{212}/_{147}$
3	8602 $^{386}/_{198}$	9459 $^{119}/_{80}$	8724 $^{373}/_{194}$	9730 $^{216}/_{115}$
4	8879 $^{343}/_{160}$	9420 $^{98}/_{64}$	8189 $^{342}/_{161}$	9837 $^{184}/_{122}$
5	9496 $^{341}/_{169}$	9706 $^{108}/_{68}$	8935 $^{345}/_{135}$	10079 $^{197}/_{129}$

The evolved player beats all the other players, whilst the non-evolving players perform evenly across all of the tables. It is also interesting to note that the evolving player participates in more games, due to a more aggressive nature. However, this does not mean that the player is guaranteed to win. In fact, the opposite is true; the more games played the more likely it is the player will be open to defeat, playing with lower rank cards. Therefore, it suggests that when the evolving player holds a strong hand it takes a very positive approach, by raising frequently.

It is an interesting observation that the non-evolving loose players play more hands than tight players due to an overestimation of their hand value. In general, the loose

players lose more than tight players and the evolving player does better against the loose players. It is well known that tight poker players will do better than loose players but there is a balance to be struck otherwise a tight player would only ever bet with the best hand. It would appear that the evolving player has found such a balance.

So far, the players at a given table have all been of the same type. Figure 6 shows how the evolving player competes when there are four different types of player at the same table (we also tested a variety of different players at the same table and the results are similar).

The results confirm our intuition that the loose players do worse than the tight players. It is also pleasing to see that the evolved player beats all the other players.

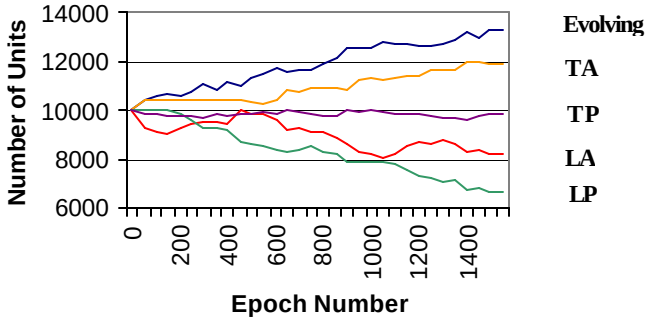

Figure 6: A Table Consisting of each Type of Player

4. Conclusions and Discussion

This paper has carried out an initial investigation as to how a computer program can learn how to play poker. We realize that this is only an initial investigation but we feel that we have shown the method we propose, although simple in its implementation, does show that an adaptive poker player is a promising research direction. Not only would several competing research groups be able to promote the research domain but, a sustained research strategy could derive benefits in other areas such as bluffing, negotiation and dealing with imperfect information. These insights would be valuable in domains such as on-line auctions, game playing theory, negotiating and real world economics.

Our current research plans will consider using Texas Hold 'Em as a more suitable poker variant. We are also experimenting with evolutionary strategies in place of the simple learning technique we currently employ. We also plan to experiment with co-evolution so that different strategies have to fight to survive to a future generation. Finally, we will also incorporate bluffing and negotiation so as we feel these elements are needed in order to compete with the best human players.

References

Barone L. and While L. 1998. Evolving Computer Opponents to Play a Game of Simplified Poker. In proceedings of the 1998 International Conference on Evolutionary Computation (ICEC'98), pp 108-113

Barone L. and While L. 1999. An Adaptive Learning Model for Simplified Poker Using Evolutionary Algorithms. In proceedings of Congress of Evolutionary Computation 1999 (CEC'99), July 6-9, Washington DC, pp 153-160.

Barone L. and While L. 2000. Adaptive Learning for Poker. In proceedings of the Genetic and Evolutionary Computation Conference 2000 (GECCO'2000), July 10-12, Las Vegas, Nevada, pp 560-573

Billings, D., Papp, D., Schaeffer, J. and Szafron, D. 1998a. Poker as a Testbed for AI Research. In Proceedings of AI'98, The Twelfth Canadian Conference on Artificial Intelligence, Mercer, R.E., and Neufeld, E. (eds), Advances in Artificial Intelligence, Springer-Verlag, pp 228-238

Billings, D., Papp, D., Schaeffer, J. and Szafron, D. 1998b. Opponent Modelling in Poker. In Proceedings of the 15th National AAAI Conference (AAAI-98), pp 493-499

Billings, D., Peña, L, P., Schaeffer, J. and Szafron, D. 1999. Using Probabilistic Knowledge and Simulation to Play Poker. In Proceedings of AAAI-99 (Sixteenth National Conference of the Association for Artificial Intelligence).

Chellapilla, K. and Fogel, D. B. 2000. Anaconda Defeats Hoyle 6-0: A Case Study Competing an Evolved Checkers Program against Commercially Available Software. In Proceedings of Congress on Evolutionary Computation, July 16-19 2000, La Jolla Marriot Hotel, La Jolla, California, USA, pp 857-863

Findler N. 1977. Studies in machine cognition using the game of poker. Communications of the ACM, 20(4):230-245.

Fogel D. 2001. Blondie24: Playing at the Edge of AI, Morgan Kaufmann, ISBN 1-55860-783-8

Hamilton, S., Garber, L. 1997. Deep Blue's Hardware-Software Synergy. IEEE, pp 29-35

Kendall, G and Whitwell, G. 2001. An Evolutionary Approach for the Tuning of a Chess Evaluation Function using Population Dynamics. In proceedings of Congress of Evolutionary Computation 2001 (CEC 2001), May 27-30 2001, COEX, Seoul, Korea, pp 995-1002.

Koller, D. and Pfeffer, A. 1997. Representations and Solutions for Game-Theoritic Problems. Artificial Intelligence 94(1-2), pp 167-215

Neumann, J von and Morgenstern, O. 1944. Theory of Gamesand Economic Behavior. Princeton, N.J.: Princeton University Press

Samuel A. L. 1959. Some Studies in Machine Learning using the Game of Checkers. IBM Journal of Research and Development, 3(3), pp 210-229

Samuel A. L. 2000. Some Studies in Machine Learning using the Game of Checkers. IBM Journal of Research and Development, Vol. 44 No. 1/2 January/March, pp 207-226

Schaeffer, J. 1996. One Jump Ahead: Challenging Human Supremacy in Checkers, Springer, Berlin

Schaeffer, J., Billings, D., Peña, L, P. and Szafron, D. 1999. Learning to Play Strong Poker. In Proceedings of the Sixteenth International Conference on Machine Learning (ICML-99) (invited paper)

Sklansky, D. 1994. Theory of Poker. Two Plus Two Publishing, ISBN 1-880685-00-0

Sklansky, D. 1996. Hold'em Poker. Two Plus Two Publishing, ISBN 1-880685-08-6