Evolved Neural Networks Learning Othello Strategies

S. Y. Chong, D. C. Ku, H. S. Lim, M. K. Tan, and J. D. White
Centre for Image Processing and Telemedicine
Multimedia University
75450 Melaka
Malaysia
siangyewch80@yahoo.com

Abstract- Evolutionary computation was used to train neural networks to learn to play the game of Othello. Each neural network represents a strategy based on board evaluations of the game tree generated by a minimax search algorithm. Networks competed against each other in tournament play and selection used to eliminate those that performed poorly relative to other networks. Self-adaptation was used to mutate the weights and biases of surviving neural networks to generate offspring. By monitoring the evolutionary behavior over 1000 generations through game competitions with computer players playing at higher ply-depths using deterministic evaluations, the networks are shown to co-evolve with the style of game play progressing from random to positional and finally to mobility strategy.

1 Introduction

Games have always been an important domain for studies into the behavior of artificial intelligence (AI) systems, especially for a new generation of AI systems that harness evolutionary computation and neural networks for learning and for decision making (Chellapilla & Fogel 1999a, 1999b, Fogel 1993). Games like Othello and checkers possess certain characteristics that make them interesting and viable testbeds to study both the decision making and learning aspects of these evolutionary systems. Firstly, games by nature have specific set of rules that constrains players to certain behaviors (i.e. legal moves), thereby simplifying the problem at hand. Secondly, games have goals for players to achieve (i.e. to win the game), with rewards being given to players that exhibit behaviors (game playing strategies) that allow them to achieve the goals under constraints of finite resources (game pieces). Lastly, games provide enough subtleties for a wide-range of complex behaviors, represented by a diverse environment of players. The challenge posed to researchers is to develop evolutionary system that not only plays the game competently, but also learns to play the game on its own without *a priori* knowledge. So far, such developments have shown promising results not only against computer programs (Chellapilla & Fogel 2000) but also against human players (Fogel 2002).

Among various games, Othello, a deterministic, perfect information, zero-sum game of two players, remains one of the more attractive games to study these systems. Although Othello has a simple set of rules, an average of 60 moves to complete, and a small average branching factor of 7 (Leouski & Utgoff 1996), it is still a challenging game to master, despite its simplicity for players to learn. One reason is that unlike games like chess, every legal move in Othello adds a piece to the board that can only be flipped to another color, not removed. As the game progresses, possible moves are gradually limited. Another reason is that the effect of every legal move in flipping certain opponent's pieces to the player's color makes it difficult to keep track of the piece topology as it can change drastically from one move to another.

In this paper, we discuss the application of a hybrid system of evolutionary computation and neural networks to Othello. Compared to previous methods like (Moriaty & Miikkulainen 1995) where neural networks were evolved to guide a game tree search, our approach evolves neural networks to act as evaluation functions for the minimax search. Co-evolutionary method is used, where neural networks in a population compete with each other for points based on win, lose or draw that decide selection for the next generation. Rather than trying to generate strong Othello play, emphasis is placed on evolved neural networks (ENNs) in learning various game strategies without preprogrammed expert knowledge. It will be shown that ENNs can learn to play the game using concepts from positional and mobility strategies. Section 2 provides background of prior AI studies on Othello. Section 3 provides brief descriptions on game rules. Section 4 details the method that is employed for the study. Section 5 presents results while Section 6 provides discussion. Section 7 concludes the findings of this study.

2 Background of AI Studies on Othello

Othello is one of the board games where AI programs developed can equal, if not surpass, the best human players. However, the impression these AI programs gives to the public can be misleading. Unlike human players that reach the pinnacle of the game by learning on their own through competitions, most of these AI programs are deterministic programs, employing a wide

0-7803-7804-0 /03/$17.00 © 2003 IEEE

range of expert knowledge like Othello strategies and advanced search algorithms that can search for best moves to very deep plies in a relatively short time. A typical example is the work of Rosenbloom who developed Iago, the first master-level Othello-playing program (Rosenbloom 1982). Iago employed an alpha-beta algorithm for tree searching with kill tables and heuristics for board evaluations. No learning mechanism was incorporated. Later, Lee and Mahajan developed Bill. Bayesian learning is used, although optimized state-of-the art searching and timing techniques were still required (Lee & Mahajan 1990). Finally, Buro's Logistello used some brilliant machine learning techniques in its game tree search, convincingly outplayed a world champion (Buro 2002).

However, despite these AI systems' excellent achievements, they still cannot learn to play the game on their own. Instead of deterministic approaches and the need for human expert interjection, a feasible alternative approach is to use a hybrid system comprising of neural networks and evolutionary computation techniques. This system will need to learn to play Othello starting from a non-playing state without relying on expert knowledge. Such a self-learning system is possible, since it is known that neural networks can represent complex behaviors and that evolutionary algorithms (EAs) can optimize complex behaviors. There are many examples to this approach, in perfect information games such as checkers (Chellapilla & Fogel 1999a), tic-tac-toe (Fogel 1993), Othello (Moriaty & Miikkulainen 1995), and imperfect information games such as Backgammon (Pollack & Blair 1998).

In this paper, an EA is used to co-evolve neural networks. Fundamentally, EAs do not need fitness functions to work. They only require a method to rank competing solutions to determine which solution is better than the other. Chellapilla and Fogel showed that with just "win, lose or draw" information, ENNs not only can learn to play the game of checkers, but to defeat strong players rated as experts from a commercial program (Chellapilla & Fogel 2000) or compared to human players (Fogel 2002, pp. 283). For our experiment, as the neural networks are co-evolving, computer players using deterministic Othello game playing algorithms are used to monitor the ENNs' game strategies through Othello game competitions. In addition, it should be noted that competition results are not given to neural networks to provide an independent observation. We will demonstrate that the resulting ENNs learn to play Othello with game playing strategies using concepts of positional and mobility.

3 Othello Game Rules

Othello is played by two players, Black and White, alternatively placing their pieces on an 8 x 8 board. Starting from an initial board position, Black moves first. A legal move is one where the new piece is placed adjacent horizontally, vertically, or diagonally to an opponent's existing piece. The piece must be placed in such a way that at least one of the opponent's pieces lies between one of the player's existing pieces and the new piece. The move is completed when the surrounded pieces are flipped to player's own color. The game is completed when neither player can make a legal move, which generally occurs when there are no empty squares remaining on the board. The winner is the player with the most pieces on the board at the end of the game. If both players possess an equal number of pieces, both players are awarded a draw.

4 Method for Evolving Neural Networks

The simulation program consists of a search algorithm, a neural-network-based evaluation function, and an EA to generate offsprings from parent neural networks and that selects the best neural networks for the next generation. A separate module is used to monitor the evolution process without participating. The program was written in ANSI-C, and ran on a 1.8 GHz Intel processor-based PC, covering over 1000 generations in 16 days for certain evolutionary behaviors to be observable.

The search algorithm is based on the basic minimax algorithm. The basic principle of the algorithm that the best possible move for every turn is the one that minimizes the maximum damage the opponent can inflict, is valid if heuristics used to determine the value of the moves when searching through the game tree are accurate. In the simulation, the evaluations during the minimax search is provided by neural networks. During evolution, ply-depth is restricted to 2 for neural networks. As such, the reliability of game heuristics provided by neural networks will govern the level of play of the ENN.

The neural network used to provide board evaluations through its input-output response to the minimax is based on the Multilayer Perceptron (MLP) model used in Chellapilla and Fogel's experiments on evolving MLP's to learn to play checkers (Chellapilla & Fogel1999b). Referring to Fig. 1, the MLP consists of an input layer, three hidden layers, and an output. The input layer is a vector consisting of 64 components representing the 64 Othello board positions. The components are represented in a computable manner as +1 (Black piece), -1 (White piece), or 0 (empty square). The first hidden layer is designed to function as a spatial preprocessing layer. It consists of 91 nodes, divided into 7 different node types. Each node type is responsible for different board subsection sampling as shown in Fig. 1. For example, there are 36 nodes to sample all possible 3 x 3 board subsections of the 8 x 8 game board. The arbitrary subdivisioning is to provide the neural network a chance to capture possible spatial characteristics of the board

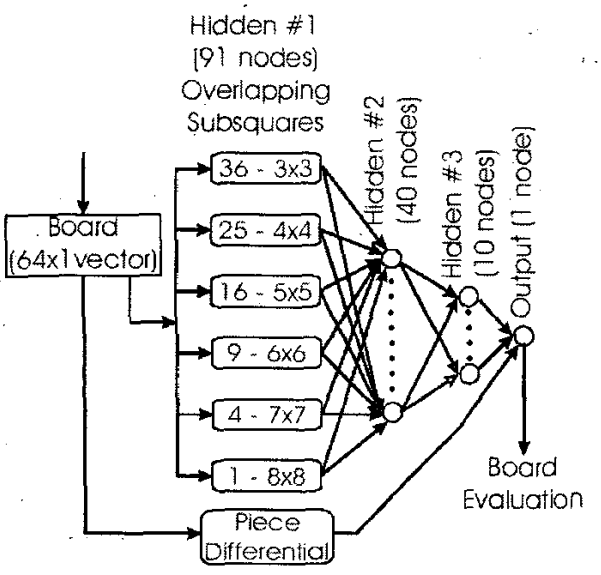

Fig. 1 Model of the neural-network-based evaluation function. The model is based on a feed-forward MLP, consisting of a 64 x 1 vector input (Othello board representation), 3 hidden layers and a single node output. The first hidden layer, with 91 nodes, acts as a spatial preprocessing layer that acts to sample the input to provide spatial information to the network. The second hidden layer (40 nodes), third hidden layer (10 nodes) and the output node are fully connected. The output represents the evaluation of the board position.

for use during board position evaluations, without explicit spatial features of the game programmed into the neural network. The second layer and third layer consist of 40 nodes and 10 nodes respectively. All first layer nodes are connected to all second layer nodes, which are then connected to all third layer nodes. A single node is used for output, connected to all nodes in the third layer. Each of these nodes will sum its inputs and a bias before passing the sum to a hyperbolic tangent function bounded by ±1 that provides nonlinearity. However, for the output node, the additional of an input consisting of the board's piece differences is also supplied. The neural network's effectiveness to provide Othello strategies will depend on how it uses piece differential information and spatial information provided by the spatial preprocessing layer for decision making, as well as how it generalizes the input-output mappings to effective Othello strategies.

Co-evolution is used to evolve neural networks. Beginning with neural networks playing randomly, co-evolution was implemented to improve neural networks' overall Othello playing strategies and also used to refine potentially higher-level strategies without losing generality or narrowing strategies to certain playing styles. The procedure for simulating co-evolution starts with the weights and biases $[w_i(j)]$ of 10 neural networks (strategies) as parents being randomly generated by sampling from a uniform distribution over [-0.2, 0.2]. The components of the self-adaptive parameter vector $[\sigma_i(j)]$ for the weights and biases of each neural network were initially set to 0.05. Components of $[\sigma_i(j)]$ control the step size of the mutation. Each of the 10 offspring neural networks was generated from the parent neural networks (PNNs) through self-adaptation as follows:

$$\sigma'_i(j) = \sigma_i(j) * \exp(\tau * N_j(0,1)),\ i=1\ldots10;\ j = 1,\ldots,Nw$$
$$w'_i(j) = w_i(j) + \sigma'_i(j) * N_j(0,1),\ i=1\ldots10;\ j = 1,\ldots,Nw$$

where N_w is the total number of weights and biases (5900) for each neural network, $\tau = (2(N_w)^{0.5})^{-0.5} = 0.08068$, and $N_j(0,1)$ is a standard Gaussian random variable re-sampled for every j. After generating offspring, each neural network from the population competes (Black) against 5 randomly chosen neural networks (White), where on average, it would be randomly picked to play White in equal number of games when other neural networks are evaluated for fitness. Point distribution is set to 5, 1 and 0 for win, draw and loss respectively to encourage neural networks to play for wins rather than for draws. A neural network is restricted from competing against the same opponent more than once unless they switched color in the two games. The game ends after more than 10 move skips (because neural networks cannot make legal moves). Current board pieces are counted to award wins, loses or draws to respective neural networks. At the end, 10 neural networks with most points are selected to be parents for the next generation. The process of generating offspring, tournament competition and selection is then repeated.

why does we choose it randomly?

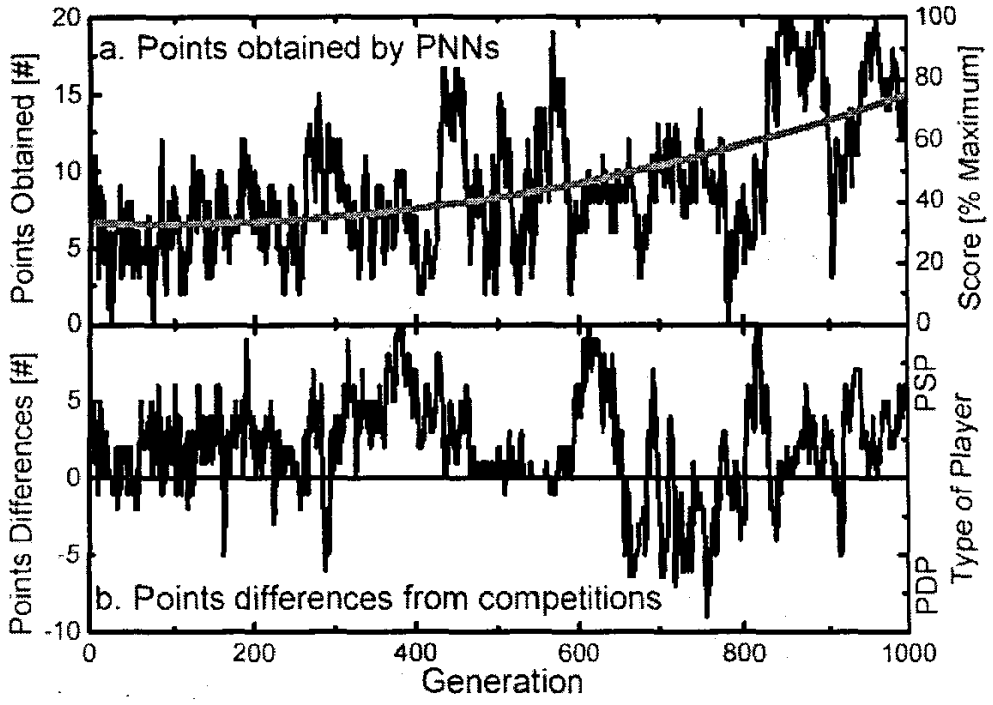

Fig. 2 Analysis of the evolution of the neural networks. Each PNN competed against the PDP and PSP in two separate games. (a) The black line indicates points obtained by the PNNs when competing against PDP and PSP playing at ply-depth of 4. The gray line is the result of a least-squares fit of a second-order polynomial, indicating improvements as the neural network evolves. (b) The black line shows the difference of points obtained by PNNs that can defeat PSP to points obtained by the same PNNs that can defeat PDP. A positive points difference will indicate that the major contributors to the points in (a) are successful competitions of PNN against PSP, while negative points will indicate that the major points contributors are from successful competitions against PDP.

To provide independent monitoring of population fitness and to observe its changes as neural networks evolve, the parents for the next generation (after selection) are compared with computer players through game competitions. For comparisons, computer players will use two different algorithms, one for each game, for a total of two games with each neural network. One algorithm is based solely on a simple piece differential evaluation function with a minimax search. The other uses minimax search also, but employs a more sophisticated evaluation function, using simplified heuristics for positional play based on Iago's evaluation function (Rosenbloom 1982). Comparison starts with neural network player competing with the piece differential computer player playing at ply-depth of 2 (the same ply-depth is used by neural networks). For every neural network win, the relative level of play of the computer player is increased by increasing its ply-depth by 2, up to a maximum of 6. Similar comparison is done with the positional computer player. Just as the Othello game provides an interesting case in studying strategy and strategy changes in (Billman & Shaman 1990), the two computer players will also provide some insight into the evolutionary behavior of the neural networks in terms of strategy changes as well.

5 Results

The simulation was conducted for 1000 generations, which took about 2 weeks of continuous running on a 1.8 GHz machine. To monitor the evolution of the neural networks ensemble (restricted to a ply-depth of 2), comparisons of 10 PNNs with the two computer players were conducted at every generation. For a rough comparison, a positional computer player (PSP) outperforms a piece differential computer player (PDP) in the game when both are playing at same ply-depth. Furthermore, comparisons with the two computer players that use different strategies will allow the observation of how ENNs respond to opponents playing different styles.

The results of the simulation are summarized in Fig. 2. Fig. 2(a) plots the points obtained by the PNNs at each generation. The points given depend on the game competitions between the PNNs and the computer players

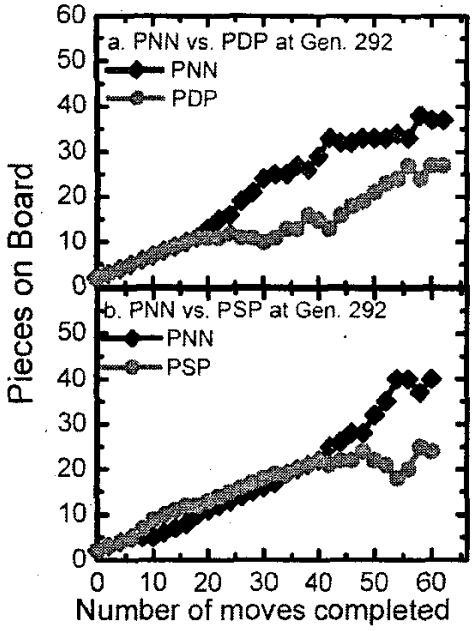

Fig. 3 Analysis of game competitions between a PNN at generation 292 with the PDP and the PSP showing evidence of positional strategy being employed by the PNN. (a) Total number of pieces for the PNN and the PDP as a function of moves made. (b) Total number of pieces for the PNN and the PSP as a function of moves made.

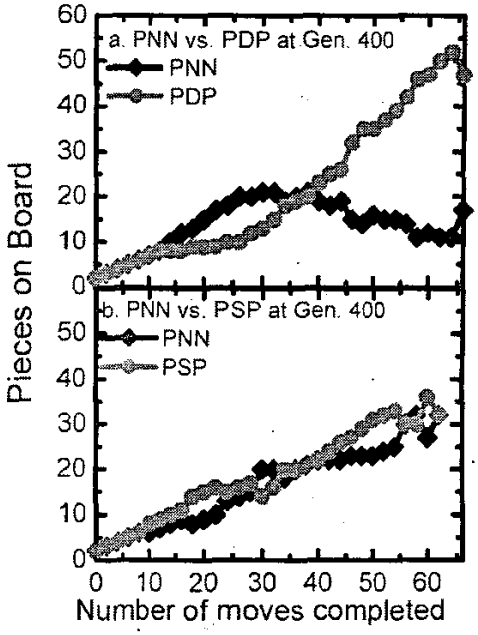

Fig. 4 Analysis of game competitions between a PNN at generation 400 with the PDP (a) and the PSP (b). Poor performance of the PNN indicates that the neural network population searching for other game playing strategies.

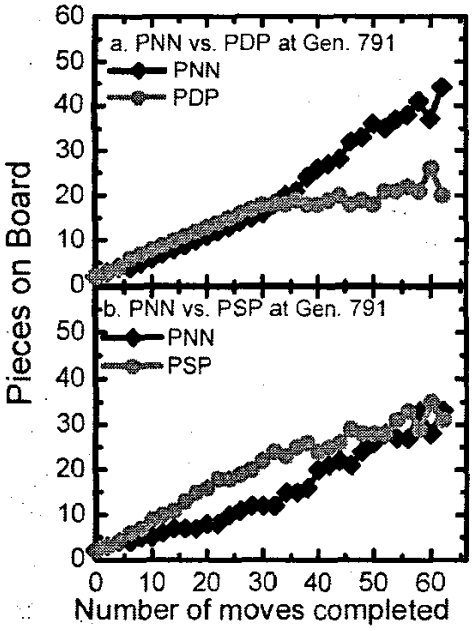

Fig. 5 Analysis of game competitions between a PNN at generation 791 with the PDP (a) and the PSP (b) showing evidence of play using concepts of mobility being employed by the PNN.

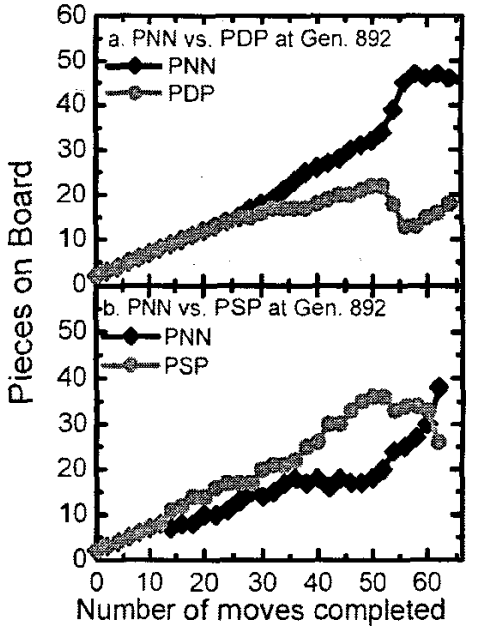

Fig. 6 Analysis of game competitions between a PNN at generation 892 with the PDP (a) and the PSP (b) showing evidence of perfected mobility play being employed by the PNN.

playing at ply-depth of 4. For a successful defeat of the PSP by the PNN, 1 point is awarded. Likewise, 1 point is awarded to the same PNN if it also defeats the PDP. Thus, a single PNN can obtain a maximum of 2 points and the 10 PNNs at a generation can obtain a maximum of 20 points (a percentage is also calculated at each generation, with 100 % meaning that the PNN population scored maximum points at its generation). As such, Fig. 2(a) shows the progress of the neural network evolution. For example, for the first 400 generations, the PNN population scored about an average of 25 % of the maximum score while from generation 850 onwards, the average increases to about 75 %. Fluctuations are observed throughout the evolution process, while the least-squares fit of a second-order polynomial (3rd, and 4th-order coefficients are essentially zero) indicates that improvement of the play of the ensemble of PNNs increases as evolution progresses. In addition, another analysis of the evolution process can be observed with Fig. 2(b). This graph is a result of taking the difference between the points obtained by PNNs that can outperform the PSP playing at ply-depth of 4 with the points obtained by PNNs that can outperform the PDP playing at the same ply-depth also. A positive number will indicate that at that particular generation, more PNNs are competitive against PSP compared to the PDP. For example, at around generation 400 and 600, Fig. 2(b) shows that more PNNs can defeat the PSP, while between the period (around generation 500), Fig. 2(b) shows that the PNNs are comparable to both computer players (since the difference value is small and near zero).

Figures 3, 4, 5 and 6 are results from analysis of game-plays of the first PNN (selected by the selection mechanism employed in the simulation) against the two computer players. In all the figures, there are a total of two graphs. The first graph, (a), plots the number of pieces acquired by the PNN and the PDP playing at ply-depth of 4 throughout the game. Graph (b) plots the number of pieces acquired by the PNN and the PSP playing at ply-depth of 4 throughout the game. Figures 3 to 6 are results from analysis of the game between the PNN and the computer players at generation 292, 400, 791 and 892.

In Fig. 3, it can be observed that the PNN at generation 292 outperformed both the PDP and PSP playing at ply-depth of 4. With reference to data in Fig. 2(a), the PNN population scored 13/20 (65 %) while in Fig. 2(b) the difference is negative (4 PNNs outperformed the PSP compared to 9 PNN can outperform the PDP), indicating that the majority of the score obtained in Fig 2(a) at this generation is from competitions between PNNs and PDP. As for Fig. 4, the PNN at generation 400 lost to the PDP but drew with the PSP. At this generation, the PNN population scored 6/20 (30 %), with a positive difference (6 PNNs outperformed the PSP but all PNNs lost to the PDP). For Fig. 5, at generation 791, the PNN outperformed the PDP and only narrowly defeated the PSP. The PNN population at this generation scored 4/20 (20%), with a zero points difference. Lastly, for Fig. 6, at generation 892, the PNN outperformed both PDP and PSP. The PNN population performed admirably, scoring 20/20 (100 %), with a zero points difference.

6 Discussion

The use of computer players such as PDP and PSP in this experiment is crucial, in that the comparisons between both computer players and with the neural networks provide an independent and external observation to the evolutionary behavior of the ENNs, just like the use of game transcripts in (Billman & Shaman 1990) to study strategy changes in game playing of Othello players. As such, there are several important considerations to be noted regarding such an observation method. First of all, the fact that the computer players use fixed evaluation functions is important because they provide fixed standards that can be used to evaluate the ENNs' playing strategies. This is due to the fact that the PDP and PSP are deterministic players and will always respond in a manner they are programmed regardless of how the ENNs play. For example, the PDP will always try to capture as many pieces as possible while the PSP will always try to capture strategic locations in the board. Secondly, the computer players were designed such that they are neither substantially stronger nor substantially weaker than the ENNs. This is crucial because computer players that either defeat all the neural networks or loses to all the neural networks will not provide a good measure in observing the evolutionary behavior as well as the dynamical fitness of the ENNs as they co-evolve. In addition, the fact that the computer players' level of play is tunable by changing the ply-depth of the minimax search algorithm provides some measure of flexibility to assess the ENNs level of play. Lastly, the general use of the minimax algorithm by all the players (PDP, PSP, and ENN) is crucial in that analysis can be made directly on the learning behavior of the ENN, focusing on the input-output mappings of the neural networks and how evolution plays a hand in perfecting the mappings to represent Othello game strategies.

As for the results of the simulation, Fig. 2(a) shows that the level of play among the PNN population improves with time (generation). As the neural network co-evolves, the scores that the PNNs obtained increase. Starting with an average of 25 % of maximum score, the neural networks have evolved to a state where the PNN population, from generation 800 onwards, can score 75 to 80 % of the maximum score. The results of the experiment in Fig 2(a) indicates that starting from a non-playing state (the ENNs were randomly initialized), co-evolution had managed to evolve the ENNs to a state they can play Othello competently, with about 8 PNNs out of 10 can outperform the PDP and PSP playing at ply-depth of 4. Clearly, through co-evolution, the ENNs have learned to play the game of Othello on their own.

However, what is more interesting is that throughout the evolution of the neural networks, certain playing

strategies as well as changes in playing strategies are observed. Before further discussion, it should be noted that generally, a positional strategy is very effective against a strategy based solely on piece differentials. In addition, mobility strategy in turn is effective against a positional strategy in an Othello game. Looking at the evolution of the neural networks, Fig. 2(b) shows that at the start of the evolution to about generation 200, the majority of the PNNs were more competitive against PSP than against PDP (the curve resides in positive values). Although the figure might suggest that the neural networks are adopting a strategy that is effective only against positional strategy used by PSP, more in-depth analysis from the games will suggest otherwise. During this period, the neural networks seemed to engage in random moves with hints of positional strategy being employed, and that the style of play seemed to be effective against the PSP but ineffective against PDP. This is particularly interesting since for human players learning to play Othello, novices also seemed to uniformly begin with positional strategy (Billman & Shaman 1990). However, upon further evolution, the PNNs seemed to have grasped the underlying concepts regarding positional strategy. This is observed at generation 292. As suggested by Fig. 3, the PNN used a viable form of positional strategy that was more superior compared to the one used by the PSP (Fig. 3(b)), and whose overall game play was very effective against the PDP (Fig. 3(a)).

Further analysis will reveal that as the neural networks evolve, changes from positional to mobility style of plays have occurred, although the transition is not always smooth or stable. For example, with a population of neural networks playing strong positional strategy, it would have been difficult for the neural networks to make further improvements as to the strength of the positional strategy. Other strategies are being searched and adopted by PNNs that might overcome positional style of play, which at times might be narrow in nature. For example, at generation 400, the majority of the PNNs were competitive against the PSP but none of the PNNs can defeat the PDP (Fig. 2(b)). Game analysis will show that a PNN at this generation performed badly against the PDP (Fig. 4(a)) but managed to draw with the PSP (Fig. 4(b)). Further analysis will show that the PNN was still trying to play a particular variation of positional strategy.

Upon further evolution, it seems that the evolving neural networks have managed to discover a particular strategy that is effective against positional style of play, that is, a strategy that uses concepts of restricting the opponents' number of moves (mobility strategy). For example, at generation 791, Fig. 5(b) shows that the neural network is attempting a mobility strategy, where the intermediate goal of minimizing the opponent's number of moves usually results in a particular graph seen in (Moriaty & Miikkulainen 1995). The same strategy employed by the PNN also worked well against a PDP (Fig. 5(a)).

Further co-evolution of neural networks seemed to perfect the mobility style of play, culminating at generation 892, where all the PNNs can defeat both PDP and PSP (Fig. 2(a)). In addition, game analysis from Fig. 6 indicates that the neural network used a viable form of mobility strategy. In playing against the PSP, the neural network demonstrated that mobility play is a more sophisticated form of Othello playing strategy compared to positional play (Fig. 6(b)). Furthermore, the same strategy used by the particular neural network was also effective against the PDP (Fig. 6(a)), where at the start to middle part of the game, the neural network kept a low number of pieces but a high number of move possibilities before quickly taking advantages of its much more superior positions and move options to defeat the PDP.

Finally, to look at the performance of the PNN at generation 892, a computer player that incorporates both positional and mobility strategies of Iago for its evaluation function is used. At ply-depth of 2, the PNN played quite well compared to this computer player that was also playing at the same ply-depth. The PNN narrowly lost to the computer player at 30 to 34. It was only when the ply-depth was adjusted to 4 that the PNN can convincingly outperform the computer player at 43 to 21. Although the PNN's ply-depth needed to be increased to defeat this simplified version of Iago that used some viable mobility strategies for its play, the PNN's play that resulted in restricting the computer player's movement to an average of 1 possible legal move from move number 48 onwards of the game should be well noted.

In addition to ENNs success in discovering and adoption to mobility strategy, it is perhaps much more interesting to take into consideration parallels between how ENN and human learn to play the game. Just like a human player, ENNs will start from a simple variation of positional strategy before learning to perfect the game play through continued evolution. In addition, like a human player also, upon reaching a particular superior level of positional play, there will be attempts to search for strategies that can defeat positional play. During this period, ENNs also find the same difficulty as a human player, and often resort back to positional play. It is only after further evolution that the ENNs managed to discover and to perfect the mobility strategy that is very effective against positional play, but also in overall effectiveness against other styles of play. In addition, observations of Fig. 2 also prove to be interesting as it provides indications as to how the changes of game playing strategies occur as the neural networks are learning (evolving) to play the game. A high points score in Fig. 2(a) and a small points difference in Fig. 2(b) will indicate a strong overall playing strategy. A high points score but a large points difference will indicate a narrow playing strategy. Thus, analysis of Fig. 2 will show that as the neural networks evolve, it will learn to play from a narrow playing strategy to an overall playing strategy.

7 Conclusion

The results of the experiment have shown that not only co-evolving neural networks learn to play the zero-sum game Othello without relying on preprogrammed expert knowledge, but also learn to use concepts of positional strategy and mobility strategy in their game play. More interesting is the fact that the ENNs learn the various strategies much like a human player does. Starting with random play with hints of positional play, the neural networks co-evolve to perfect their current play to a competent level of positional play. After that, further evolution will reveal successful attempts of the ENNs to discover mobility strategy and to adopt the game play.

Acknowledgments

The authors gratefully acknowledge and thank Dr. David B. Fogel for encouragement and guidance in the early stages of this work, as well as anonymous reviewers in revising earlier work.

Reference

Billman, D. and Shaman, D. (1990) "Strategy knowledge and strategy change in skilled performance: A study of the game Othello," American Journal of Psychology, Vol. 103 (2), pp. 145-166.

Buro, M. (2002) "Improving heuristic mini-max search by supervised learning," Artificial Intelligence, Vol. 134, pp. 85-99.

Chellapilla, K. and Fogel, D. B. (1999a) "Evolving Neural Networks to Play Checkers without Expert Knowledge," IEEE Trans. Neural Networks, Vol. 10:6, pp. 1382-1391.

Chellapilla, K. and Fogel, D. B. (1999b) "Evolution, Neural Networks, Games, and Intelligence," Proc. IEEE, Vol. 87:9, pp. 1471-1496.

Chellapilla K. and Fogel D. B. (2000) "A Case Study Competing an Evolved Checkers Program against Commercially Available Software," Proceedings of the 2000 Congress on Evolutionary Computation, IEEE Press, Piscataway, NJ, pp. 857-863.

Fogel, D. B. (1993) "Using Evolutionary Programming to Construct Neural Networks that are Capable of Playing Tic-Tac-Toe," Proc. of 1995 IEEE Intern. Conf. on Neural Networks, San Francisco, CA, pp. 875-880.

Fogel, D. B. (2002) *Blondie24: Playing at the Edge of AI*, San Francisco, CA: Morgan Kaufmann.

Lee, K. F. and Mahajan, S. (1990) "The Development of a World Class Othello Program," Artificial Intelligence, Vol. 43, pp. 21-36.

Leouski, A. V. and Utgoff, P. E. (1996) "What a Neural Network Can Learn about Othello," Technical Report 96-10, University of Massachusetts.

Moriarty, D. E. and Miikkulainen, R. (1995) "Discovering Complex Othello Strategies through Evolutionary Neural Networks," Connection Science, Vol. 7, No. 3, pp. 195-209.

Pollack, J. B. and Blair, A. D. (1998) "Co-evolution in the successful learning of Backgammon strategy," Machine Learning, Vol. 32, pp. 225-240.

Rosenbloom, P. S. (1982) "A World-Championship-Level Othello Program," Artificial Intelligence, Vol. 19, pp. 279-320.