Proceedings of the 2007 IEEE Symposium on Computational Intelligence and Games (CIG 2007)

Discovering Chinese Chess Strategies through Coevolutionary Approaches

C. S. Ong, H. Y. Quek, K. C. Tan and A. Tay
Department of Electrical and Computer Engineering
National University of Singapore
ocsdrummer@hotmail.com, {g0500073, eletankc, eletaya}@nus.edu.sg

***Abstract*—Coevolutionary techniques have been proven to be effective in evolving solutions to many game related problems, with successful applications in many complex chess-like games like Othello, Checkers and Western Chess. This paper explores the application of coevolutionary models to learn Chinese Chess strategies. The proposed Chinese Chess engine uses alpha-beta search algorithm, quiescence search and move ordering. Three different models are studied: single-population competitive, host-parasite competitive and cooperative coevolutionary models. A modified alpha-beta algorithm is also developed for performance evaluation and an archiving mechanism is implemented to handle intransitive behaviour. Interesting traits are revealed when the coevolution models are simulated under different settings - with and without opening book. Results show that the coevolved players can perform relatively well, with the cooperative model being best for finding good players under random strategy initialization and the host-parasite model being best for the case when strategies are initialized with a good set of starting seeds.**

Keywords: Coevolution, Evolutionary Algorithms, Chinese Chess, Game Strategies, Opening Book

I. Introduction

Chinese Chess [1] is one of the most widely played strategy board games worldwide. A two-player, zero-sum game with a complexity level similar to Western Chess [1], Chinese Chess is beginning to gain popularity among researchers in the field of Artificial Intelligence. With the defeat of World Chess Champion, Garry Kasporov, by IBM's "Deep Blue" in 1997, it is believed that Chinese Chess will be the next chess-like game which a program will defeat a human top player. Even so, computer Chinese Chess [2] has yet to reach a level that is on par with Grandmasters. Existing research is mainly centered on incrementing search depths with efficient pruning strategies, preprogramming larger opening, mid-game and endgame databases and using more learning heuristics, all of which entails expert knowledge. Such methods simply produce programs that play the game in a way human experts think is best and this might stifle the computer's potential to play even better games. It will be more interesting to allow the computer player to learn effective chess strategies freely as it not only removes any dependency on expert knowledge, but is also able to reveal certain traits of effective strategies that no known mathematical model is able to unveil as yet. In this aspect, evolutionary approaches have been applied to discover chess-like game strategies by virtue of their ability to explore complex problems [3] by selecting optimal solutions through natural selection and reproduction, similar to how living organisms evolved through a competitive process where the weaker perish and fittest survive. This process not only offers a viable means to find good solutions to complex problems, but is also able to do so with minimal expert knowledge.

Coevolutionary algorithm [4] is a branch of Evolutionary Algorithm (EA) that is widely used to solve problems where an objective measure to guide the search process is extremely difficult to devise. Unlike most EAs, coevolutionary methods use a subjective instead of objective fitness function. Fitness of individuals is assessed via interaction with similar species within the population or with other species from one or more populations [5]. There are two main variants of coevolution: competitive and cooperative. In the prior, fitness of an agent is based on direct contest with other agents. In the latter, agents share the rewards and penalties of successes and failures [6]. Hillis's seminal paper on coevolving sorting networks and data sets in a predator-prey model opened the research for competitive coevolution [7] while studies on cooperative coevolution was initiated by Potter and De Jong [8] when they developed a relatively general framework for static function optimization and neural network learning [9]. The research on coevolution in chess-like games dates back a decade ago, with one of the earlier works on the coevolution of backgammon strategy [10]. Fogel and Chellapilla [11] also used coevolution to evolve neural networks for playing Checkers. Their work was extended by Chong, Tan and White [12] to learn Othello strategies. In coevolving neural network for Go, Lubberts and Miikkulainen utilized a Symbiotic Adaptive Neuro-Evolution method [13] to evolve two populations that challenge each other [14]. In one of the first attempts to coevolve a Western Chess game, Fogel and his team also successfully applied evolutionary methods with three neural networks [15].

In this paper, the application of coevolutionary models to discover Chinese Chess strategies is studied. While existing works hinge mostly on exploring competitive coevolutionary models [10]–[12], [14], [15], this paper also looks into the use of cooperative models. In total, three distinct models - the single-population competitive, host-parasite competitive and cooperative coevolutionary models are explored. A modified alpha-beta algorithm is developed for performance evaluation while an archiving mechanism is implemented to handle intransitive behaviour within the game. Simulation results reveal the relatively good performance of evolved players and interesting insights to the varied capability of each model in different scenarios. Organization of the paper is as follows: Section II presents the development of the Chinese Chess engine. Section III discusses the implementation of the various coevolution models. Section IV and V look at issues in performance evaluation and onset of intransitive phenomenon in the actual game play while Section VI focuses on the simulation results, observations and discussions. Section VII concludes the paper with a general summary and some possible future works.

II. Developing The Chinese Chess Engine

A. Representation

The basic rules of Chinese Chess are straightforward and can be found easily on the internet[1]. Before embarking on the learning process, a Chinese Chess engine needs to be devised and tested to ensure that all piece movements and game rules are strictly adhered to and the program contains no logical bug that would otherwise be detrimental to the decision made by the computer player when intelligence is incorporated. The proposed engine is a simple program implementing basic rules of the game. The boardstate, denoted a 10 by 9 matrix (Fig.1), is the most important information that determines a player's next move. The boardstate matrix represents the positions of all pieces on board, with a nonzero number assigned uniquely to each piece type. One player will have its pieces represented by the positive values of these numbers while the other by the negative equivalent. A "0" indicates an unoccupied position.

$$\begin{pmatrix} 5 & 4 & 3 & 2 & 1 & 2 & 3 & 4 & 5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 & 0 & 0 & 0 & 6 & 0 \\ 7 & 0 & 7 & 0 & 7 & 0 & 7 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -7 & 0 & -7 & 0 & -7 & 0 & -7 & 0 & -7 \\ 0 & -6 & 0 & 0 & 0 & 0 & 0 & -6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -5 & -4 & -3 & -2 & -1 & -2 & -3 & -4 & -5 \end{pmatrix}$$

Fig. 1. Boardstate matrix showing positions of all pieces at the start of game

Chess Piece	Piece Value
King	6000
Advisor	120
Elephant	120
Horse	270
Rook	600
Cannon	285
Pawn	30

Fig. 2. Piece values used by the unevolved strategy

From the boardstate matrix, two sources of information that ultimately guide the move decision are the piece and position values. The piece value is numerically assigned to each chess piece in accordance to its relative importance in the game. The values used by the unevolved strategy (Fig.2) in the program are adapted from Yen et al [1]. The position value is assigned to each piece at each possible position of the board based on the strategic advantage of capturing or moving a piece into that position. Each piece's position values are represented by a unique Positional Value Table (PVT). The opponent's PVT is simply a 180° rotation of the player's PVT. Assessing the worth of a move is done via an evaluation function, based on the linear combination of both the player's piece and position advantage. As piece advantage often plays a more important role, a positive (>1) weighting term is usually tagged to it.

B. Planning move decision

The program uses an alpha-beta search algorithm [16] for searching the associated game tree of each board position to determine the next best move that the computer player should execute. It is essentially a tree searching technique that builds on the basis of the minimax algorithm. With efficient pruning [17], the alpha-beta search algorithm presents a much more effective method to obtain the next move in a much reduced computational time. The ply depth of search is set at 4 (looks two moves ahead for each player) for reasonable execution time. A move ordering mechanism is also employed to further reduce the computation time. At intermediate play, quiescence searching is implemented to reduce the "horizon effect" – inability of the computer player to "see" beyond the search window. Quiescence search performs selective exploration and Beal defined it as the idea of expanding the search just enough to avoid any tactical disruption [18]. While there can be many different forms of disruptions, the chess program effectively considers the immediate capture of pieces as the only form of tactical disruption for simplicity.

C. Program Testing and integration

Through a user-friendly GUI (Fig.3) designed in Microsoft Visual C++ MFC Application Wizard, the chess engine was tested with human players of moderate standard to ensure an error free program. Integration of the engine was subsequently done to allow two computer players to play against each other.

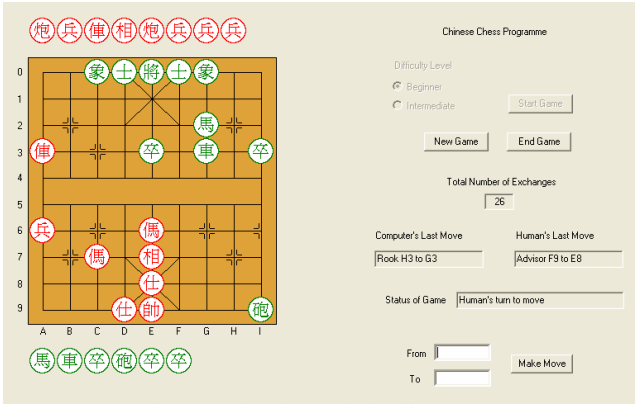

Fig. 3. GUI showing a typical midgame state

III. Coevolution Implementation

A. Coevolution Representation

The two types of parameters used for the simulations are (i) the piece values of all pieces and weight term attached to the piece advantage (Parameter Set A) and (ii) the piece values of all pieces and position values of all the powerful pieces (rook, cannon and horse) (Parameter Set B). Piece parameters of all chess pieces are represented with the exception of king, which has a non-evolvable value of 6000. Two distinct parameters are also assigned to pawn to capture the changes in its relative importance (e.g. value) before crossing river (pawnOne) and after crossing river (pawnTwo). Due to the symmetry of the PVT, a total of 50 position values will be evolved for each powerful piece. A binary-coded representation is adopted for coevolution. Piece values assume integer values from 1 to 1023 (represented as 10-bit genes), while position values take on even integer numerals from -24 to 38 (represented as 5-bit genes). Binary tournament selection is used in combination with uniform crossover as well as a mutation operator with probability that is dependent on the bit position and generation number. An elitism scheme ensures that the best individual is kept in the population from time to time. To reduce chances of encountering premature convergence and also to encourage the search for multiple niches, a simple niching function is also implemented to preserve and maintain diversity in the population of chess players over the generations.

[1] Xiangqi Club. "CXQ Chinese Chess Rules".
< http://www.clubxiangqi.com/rules/ >

B. Competitive Fitness Sharing

The notion of competitive fitness sharing was first proposed by Rosin and Belew [19] when they attempted to apply a new method of fitness sharing [3] to host-parasite coevolution. In the scheme, each parasite is treated as an independent resource to be shared by those hosts in the population that can defeat it. The purpose of such a scheme is to reward hosts that are able to defeat parasites few other hosts can defeat. This reduces the probability of discarding hosts that can only defeat a below-average number of parasites but could, nonetheless, defeat powerful parasites. In coevolving Chinese Chess strategies, this concept of competitive fitness sharing is adapted for the single-population model. The general idea is to reward players that could defeat powerful opponents which few others in the population can defeat. For an individual x_i the shared fitness value obtained for defeating another individual x_j, is given by:

$$F_{i,j} = \frac{1}{N_j} \tag{1}$$

where N_j is the number of individuals that defeated x_j in the round-robin competition. This equation well resembles the one proposed by Rosin and Belew. If individual x_i happens to draw with x_j, the shared fitness value obtained by x_i is given by the new proposed equation:

$$F_{i,j} = \frac{D}{D \times N_{D,j} + W \times N_{W,j}} \tag{2}$$

where D is the points awarded to each player for a draw game, W is the points awarded to the winning player when the game has a clear winner while $N_{D,j}$ and $N_{W,j}$ are the number of players that drew with and won player x_j in the round-robin competition respectively. A player can thus also be rewarded for managing a draw with players that few others can draw or defeat. The shared fitness of an individual x_i competing with an individual x_j is thus summarized as follows:

$$F_{i,j} = \begin{cases} \dfrac{1}{N_j} & \text{, if } x_i \text{ defeats } x_j \\ \dfrac{D}{D \times N_{D,j} + W \times N_{W,j}} & \text{, if } x_i \text{ draws with } x_j \\ 0 & \text{, if } x_i \text{ loses to } x_j \end{cases} \tag{3}$$

The total competitive shared fitness of an individual over all its competitors in the population would be given by:

$$\sum_{n \in X, n \neq i} F_{i,n} \tag{4}$$

where X is the set of indices for all individuals. Tournament selection will then be conducted based on the shared fitness value obtained for each individual in the population instead of the original fitness value.

C. Coevolution Models

1) Single-Population Competitive Coevolution

In the single-population coevolution model, a population of players engages in a round-robin competition with one another in each generation. Points are awarded after each game based on the following scheme: 2 points for the winning player, 1 point for each player in a draw game and 0 point for the losing player. The subjective fitness of each player is then computed by the sum total of all the points obtained by competing with every other player in the competition.

2) Host-parasite Competitive Coevolution

In this model, two populations (host and parasite) of players evolve independently. Players from the host population will play against those from the parasite population and vice versa. The algorithm selects the 10 best parent players in the previous generation from each population and places them in the evaluation pool to compete with players in the opposing population (Fig.4).

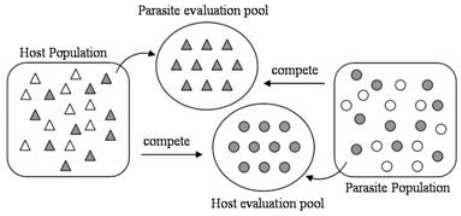

Fig. 4. Host-Parasite model showing evaluating pools

In subjective fitness evaluation, two distinct point systems are formulated to evolve different traits in the two coevolving pools. Parasite players are heavily penalized for not winning a game while host players suffer huge penalties for losing one. Such a scheme signals to the parasite population that winning a game is paramount and to the host population that losing a game is detrimental. The assignment schemes developed for both the host and parasite populations are shown in Table I.

TABLE I
ASSIGNMENT SCHEME IN HOST-PARASITE MODEL

Population	Win	Draw	Lose
Parasite	5 points	0 point	0 point
Host	5 points	5 points	0 point

The above schemes tend to drive parasite players towards developing strong offensive traits, while host players towards acquiring strong defensive traits. By presenting a recurrent pool of strong defensive players for the parasites and strong offensive players for the hosts, an ensuing arms race would accelerate the learning process and players would be able to improve faster and better under cross-population competition.

3) Cooperative Coevolution

In cooperative coevolution (Fig.5), a problem is solved by piecing together solutions of subproblems that define the main problem. Upon decomposition, each component is assigned to a unique subpopulation and evolved independently of other subpopulations. Since any given individual from a particular subpopulation denotes only a component of potential solution to the main problem, collaborators are selected from the other subpopulations to represent the remaining components. The individual is then combined with its collaborators to form the complete solution for fitness evaluation [20].

In cooperatively coevolving Chinese Chess strategies, two subpopulations are used. Subpopulation 1 contains individuals representing the piece values while subpopulation 2 contains those representing the position values. The best strategy in the previous generation of each subpopulation is chosen as the

collaborator for the accompanying subpopulation. In the first generation, collaborators are chosen after a pre-initialization phase (Fig.6). After which, the sub-chromosome representing piece values of the best strategy is chosen as the collaborator for subpopulation 2 while that representing position values of the best strategy is likewise chosen to be the collaborator for subpopulation 1. Similarly, individuals are ranked subjectively based on how well they can cooperate with the collaborators.

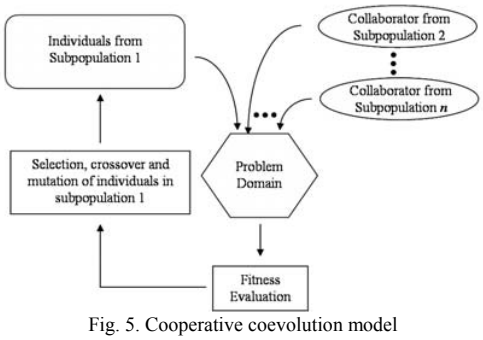

Fig. 5. Cooperative coevolution model

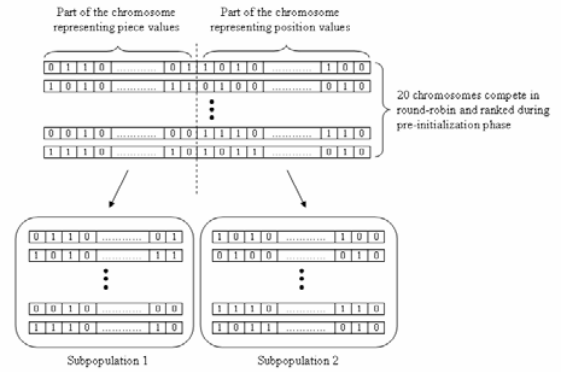

Fig. 6. Obtaining the two subpopulations after pre-initialization

In cooperative coevolution, pressure for improvement may not be as great as in the case of the host-parasite competitive coevolution model. Nevertheless, similar in spirit to the divide and conquer strategy, its attempt to split the complex problem into simpler subtasks to solve them separately might actually prove to be more effective in finding a good set of solutions.

IV. Performance Measurement

A. Subjective versus Objective fitness

During coevolution, the selection process is based on the relative performance of an individual against other members of the population. Such a performance measure is deemed to be a subjective performance measure. In general, Wiegand [20] defines four different types of measures:

Definition 1: Objective measure
A measurement of an individual is *objective* if the measure considers that individual independently from any other individuals, aside from scaling or normalization effects.

Definition 2: Subjective measure
A measurement of an individual is *subjective* if the measure does not consider that individual independently from other individuals.

Definition 3: Internal measure
A measurement of an individual is *internal* if the measure influences the course of evolution in some way.

Definition 4: External measure
A measurement of an individual is *external* if the measure does not influence the course of evolution in any way.

From the definitions, it is clear that traditional EC adopts an objective internal measure to assess fitness while coevolution adopts a subjective internal measure. Whilst the subjective measure is useful for problems where a truly objective fitness measure is absent, there has to be a means of evaluating the performance of the best evolved individual ultimately. It is not suffice to conclude that the best evolved individual in the final generation is good if there is no objective measure to assess its performance. This will require an objective, external measure.

B. Modified Alpha-beta Algorithm

Evaluating the performance of the best evolved individual entails playing 200 games against the test player, which serves as the objective, external measure. The parameters[2] of the test player are taken from [1]. In the original alpha-beta algorithm, the process of pruning has prevented the true minimax values (values as though the minimax algorithm has been used) of all the available next moves to be found, resulting in the inability to probabilistically pick the next move based on all the moves' minimax values. This renders the algorithm inadequate to play 200 non-deterministic games. While the minimax algorithm could allow for the introduction of stochasticity, it is however, also too computationally expensive to be implemented. Thus, alpha-beta algorithm is modified with the notion of pruning threshold (PT). The new algorithm will prune off a branch between a second and third level node (root node represents first level node) when the value passed back from the third level node is less that the α – PT value (instead of α value in the original algorithm). Figs. 7 and 8 depict the differences in pruning (indicated by dotted lines) using the alpha-beta and modified alpha-beta algorithm with PT = 4 respectively.

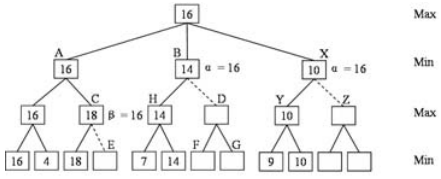

Fig. 7. Tree structure subjected to alpha-beta pruning

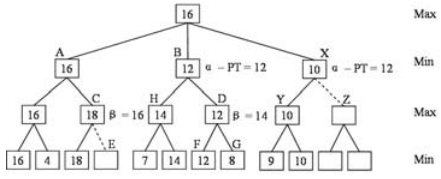

Fig. 8. Tree structure evaluated using modified alpha-beta algorithm

The original alpha-beta algorithm will prune off the branch B-D (since value of B found so far is less than α), rendering node B unsuitable for move decision (since its true minimax

[2] The missing PVTs of the king, advisor and elephant in [1] are consequently filled up with the help of an advanced Chinese Chess player based on observations of the other PVTs

value is unknown). The modified alpha-beta search, however, does not prune off the B-D branch as the value passed back from H is within the PT limit of the α value. Thus, the minimax value of B represents its true minimax value, and can be used for move decision. The X-Z branch is, however, pruned in both cases. In the modified algorithm, the PT value is set to 4, with all moves (subjected to a maximum of 4) evaluated to be within the PT limit of the minimax value of the best move being given equal probability of being chosen. Overall, the modified alpha-beta algorithm presents an elegant way of introducing variations to the game in ways similar to how human plays it. This is because a move made by a human player is often a decision from a set of good moves. Unless a substantial amount of time is spent to analyze these moves, each of these possibly non-dominated moves will very likely be given an equal probability of being selected.

V. Intransitive Behaviour

The concept of intransitive superiority has been a central issue in coevolutionary failure [21]. An intransitive relation *R* occurs when *aRb* and *bRc* does not imply that *aRc*. In chess, it is not difficult to encounter such a situation. By definition, the superiority of players in chess is clearly not transitive [22]. Furthermore, if *cRa* also holds true, this is often known as a circularly superiority or dominance relation. One such classic example will be the "rock, scissors, paper" game. Existence of such intransitive relation in coevolution could lead to the occurrence of cycling – recurrence of previously visited states [23]. The problem of intransitive behaviour occurs in complex games (i.e. chess) due to the multi-dimensional abilities of players. In such games, the relative criticality of a player's characteristics really depends on the nature of the opponent itself and it is always possible that a circularly superiority relationship [22] may just form between three or more players.

The problem of cycling arising from intransitive superiority has been thought of as a possible obstacle that could prevent coevolution from being a reliable problem solving technique [23]. In fact, "cycling problem, like the local minima problem in gradient-descent methods…, is an intrinsic problem of coevolution that cannot be eliminated completely" [24]. While there have been approaches to reduce the problem of cycling, this paper does not attempt to reduce intransitive behaviour. Rather, a simple archiving scheme has been implemented to prevent the loss of good individuals that could achieve a high objective fitness, due to the effect of cycling. In this paper, the archive and elite players are different, unlike in conventional EAs. The archive player, at any one time, is the player with the best objective fitness, while the elite player is the player with the best subjective fitness in a particular generation. Archiving ensures that the player that performs best against the objective test evaluator is preserved as the coevolution process continues to be guided by the players' interactions. To prevent losing the genetic materials of the archive player, the archive player is re-introduced into the population after every generation as long as it is not the elite player. Niching helps to maintain high population diversity and reduces the likelihood of premature convergence, which can occur due to repeated re-introduction of the same archive player back into the mating pool, should the archive player remain unchanged for a few generations. Through this setup, the archive player will represent the best evolved player at the end of the entire coevolution process.

VI. Simulation Results

Due to computation limitations, simulations are performed without the use of quiescence searching. For each experiment, five independent runs are simulated. Hypothesis testing is then conducted at the 5% level of significance to determine if the evolved player could match the standard of the test player ($Z > -1.645$) or outperform it ($Z > 1.645$). All runs are simulated with an initial population of 20 players over 50 generations.

A. Evolving Parameter Set A with random initialization

In this set of simulations, parameter set A is evolved with random initialization using the single-population competitive coevolution model. Table II shows the performance of the best evolved player against the test player over 200 games.

TABLE II
Single-Population Coevolution with Random Initialization

Simulation	Wins	Draws	Losses	%Wins	Z-value
1	114	24	62	64.77%	3.919
2	107	19	74	59.12%	2.454
3	121	14	65	65.05%	4.105
4	93	41	66	58.49%	2.141
5	110	25	65	62.86%	3.402
Average	109.0	24.6	66.4	62.06%	3.204

Based on the Z-values obtained, all the best evolved players manage to outperform the test player in all runs. The objective and subjective fitness traces of the best player (averaged over all the five runs) are plotted in Fig.9 for every generation.

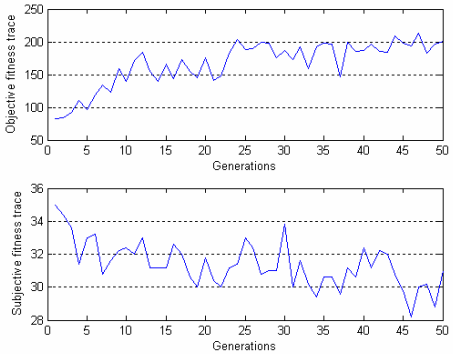

Fig. 9. Objective and subjective fitness trace of best player

An overall rise in objective fitness trace over 50 generations indicates that coevolution drives the best individual to achieve better and better performance over time. Fluctuation in fitness level, despite the introduction of elitism, reveals the existence of intransitivity as discussed previously. The subjective fitness plot, on the other hand, shows a decreasing trend as the generation progresses. This is a logical expectation as the overall standard of players in the population has improved over the generations. The best player will find more difficulty to score points against their peers in the later generations as compared to the earlier generations. However, unlike the elite player, objective fitness of the archive player depicts a non-decreasing trace (Fig.10), signifying an improvement in performance over the generations.

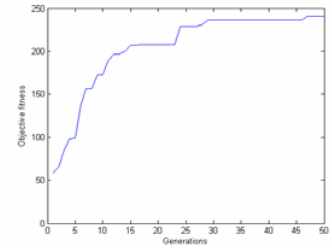

Fig. 10. Objective fitness trace of archive player

Table III shows the piece values of the best evolved player in all the five runs. From a completely random set of piece parameters, the coevolutionary algorithm recognizes the rook as the most important piece in the Chinese Chess by assigning the highest piece value to it across all runs. The algorithm also values the cannon higher than the horse in all runs, similar to the piece values provided by Yen et. al. [1]. While this is a disputable case (some expert may value a horse more than a cannon), the algorithm actually acknowledges that the cannon is a relatively more important piece than the horse for the type of tree structure that the computer uses to find the best move. The simulation results also indicate that a pawn after crossing river (pawnTwo) is valued very much higher than one before crossing (pawnOne). In most runs, pawnTwo's values are even higher than the advisor and elephant. This reflects the importance of pawnTwo due to its increased mobility and the threat it can pose to the opponent's king. In all, the learning capability of coevolution is revealed through the evolved parameter values, which is somewhat similar to how human players would perceive a good set of parameter values to be.

TABLE III
EVOLVED PIECE VALUES

Piece	Simulations				
	1	2	3	4	5
Advisor	111	24	10	189	310
Elephant	372	51	337	220	185
Horse	539	291	460	585	428
Rook	1018	983	997	971	931
Cannon	853	970	887	656	788
PawnOne	71	27	6	15	115
PawnTwo	392	208	336	220	354

B. Evolving Parameter Set B with random initialization

After obtaining good performance from evolving parameter set A, the number of evolvable parameters is now increased to 157 in parameter set B. Tables IV – VI depict the performance of the best evolved player against the test player (in terms of objective fitness) over 200 games for all the three coevolution models. The results illustrate that the host-parasite competitive and cooperative coevolution models yield better performance than the single-population coevolution model in general while the cooperative model is able to evolve the best players when the population starts off with a randomized set of parameter values. A discussion of this will be done in section D.

TABLE IV
SINGLE-POPULATION MODEL WITH RANDOM INITIALIZATION

Simulation	Wins	Draws	Losses	%Wins	Z-value
1	90	26	84	51.72%	0.455
2	66	46	88	42.86%	-1.773
3	48	69	83	36.64%	-3.058
4	51	49	100	33.77%	-3.988
5	60	29	111	35.09%	-3.900
Average	63	43.8	93.2	40.02%	-2.453

TABLE V
HOST-PARASITE MODEL WITH RANDOM INITIALIZATION

Simulation	Wins	Draws	Losses	%Wins	Z-value
1	72	44	84	46.15%	-0.961
2	55	53	92	37.41%	-3.052
3	87	36	77	53.05%	0.781
4	60	60	80	42.86%	-1.690
5	66	48	86	43.42%	-1.622
Average	68	48.2	83.8	44.58%	-1.309

TABLE VI
COOPERATIVE MODEL WITH RANDOM INITIALIZATION

Simulation	Wins	Draws	Losses	%Wins	Z-value
1	79	49	72	52.32%	0.570
2	87	40	73	54.38%	1.107
3	87	52	61	58.78%	2.137
4	76	40	84	47.50%	-0.632
5	86	33	81	51.50%	0.387
Average	83	42.8	74.2	52.89%	0.714

Performance of the best evolved players in all five runs pale in comparison to those in Table II due to the huge increase in the number of parameters. Nevertheless, the overall rise in the objective fitness (Fig.11) is an indication that the learning process is actively taking place throughout the generations.

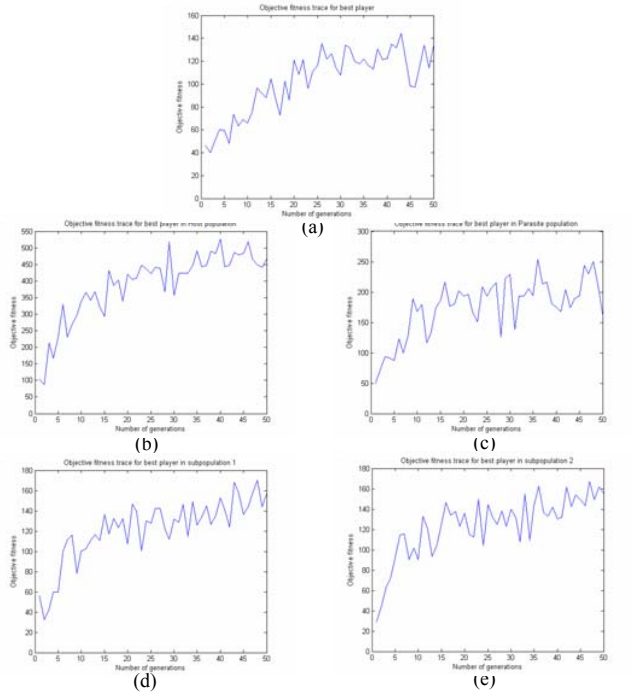

Fig. 11. Objective fitness trace of best player in (a) single-population model, (b) host population (host-parasite model), (c) parasite population (host-parasite mode), (d) subpopulation 1 (cooperative model) and (e) subpopulation 2 (cooperative model)

C. Evolving Parameter Set B with starting seeds

While it is clear from the fitness traces in the previous section that learning is taking place, performance of the best evolved player still shows the inability of coevolution to find good solutions with random parameter values. To assist the coevolution process, a set of starting seed values is now used as initialization centres to create the starting population. Such attempt will possibly place the initial individuals at the near optimal regions. Tables VII - IX shows the performance of the best evolved players in all the three coevolution models.

TABLE VII
SINGLE-POPULATION MODEL WITH STARTING SEEDS

Simulation	Wins	Draws	Losses	%Wins	Z-value
1	86	45	69	55.48%	1.365
2	93	43	64	59.24%	2.314
3	85	54	61	58.22%	1.986
4	74	62	64	53.62%	0.851
5	78	61	61	56.12%	1.442
Average	83.2	53	63.8	56.54%	1.592

TABLE VIII
HOST-PARASITE MODEL WITH STARTING SEEDS

Simulation	Wins	Draws	Losses	%Wins	Z-value
1	79	59	62	56.03%	1.432
2	113	28	59	65.70%	4.117
3	105	37	58	64.42%	3.681
4	103	38	59	63.58%	3.457
5	100	41	59	62.89%	3.252
Average	100	40.6	59.4	62.52%	3.189

TABLE IX
COOPERATIVE MODEL WITH STARTING SEEDS

Simulation	Wins	Draws	Losses	%Wins	Z-value
1	94	35	71	56.97%	1.791
2	98	38	64	60.49%	2.671
3	81	62	57	58.70%	2.043
4	85	41	74	53.46%	0.872
5	101	35	64	61.21%	2.880
Average	91.8	42.2	66	58.17%	2.052

Results show that starting seeds are effective in assisting coevolution to find much better players. Most are now able to surpass the test player. There is also an overall improvement in average performance of the best evolved player in each model compared to the prior case. While the simulation with random initialization seems to suggest that the cooperative model can produce the best player amongst the three models, results in this section seem to indicate that the host-parasite model is able to produce the best performing player instead.

D. Discussion on general performance of the models

The performance difference between the host-parasite and cooperative model in sections B and C is due to the relative strength of the two models under different conditions. Success of the cooperative model largely hinges on its ability to decompose a complex problem into its simpler subproblems, in particular, coevolving two sets of relatively independent parameters separately while relying on the best evolved player from the other subpopulation as the collaborator to find good solutions. Despite random initialization of parameter values, this collaborative nature between the two subpopulations is exploited to find desirable solutions in the high-dimensional search space through multiple searches, where each deals with a sub-domain with a search space of smaller dimension. In this scenario, the host-parasite model is unable to perform as well, since the random distribution of the initial population across the huge, complex search space has prevented the competing partners in the evaluation pools to be strong enough to push the overall fitness of the population up fast. The pressure for players to improve is virtually not present.

With a good set of starting seeds, the advantage gained by the cooperative model in splitting the problem domain into subproblems is, however, less than the benefit gained by the host-parasite model from having a good set of competing partners. The good initialization of parameters has brought the population into regions where possible good solutions exist. As such, the evaluation pools in the host-parasite model, after some generations, will consist of individuals that are strong in their respective areas: players in the parasite evaluation pool are strong defensive players while those in the host evaluation pool are strong offensive players. This is unlike the random initialization case where players in the evaluation pool might not be very strong. Pressure to improve in simulations with good starting seeds is much greater than those with random initialization for the host-parasite model, due to the escalating arms race between the host and parasite populations.

E. Imparting Opening Book knowledge

In all the earlier simulations, players compete against each other purely based on the piece and position values that they have. Move decisions are made with respect to these values right from the start of the game. Nonetheless, it is known that opening book is an integral part of chess games [25]. A player with no opening book knowledge would have a clear disadvantage when playing against one with wide knowledge of opening book moves. This section thus attempts to evolve players that play with opening book moves where each game starts with one series of opening book moves. As the opening book is merely meant to serve as a guide to assist players in making good starting moves, a maximum use of 10 opening book moves is imposed. This is to prevent players from being too dependent on the opening book and fail to discover good strategies, which is in fact the ultimate aim of coevolution.

Table X shows the average performance of the best evolved players with random initialization in the single-population model. The results indicate a slight dip in performance when compared with the case when no opening book was used. This might mean that the test player could play better with opening book, rendering it more difficult for the evolved players to score a larger number of points against it. Nevertheless, the results do indicate the relatively desirable performance of the single-population model in evolving the two parameter sets.

Tables XI and XII show the piece values of each of the best evolved players from the two sets of simulations. The evolved values seem to differ from those obtained without opening book, with the most obvious difference being the change in relative value of horse and cannon. Previous simulations have produced players that value the cannon more than the horse. In Tables XI and XII, 6 out of 10 simulations have recognized the horse to be a more valuable piece. This is attributed to the opening book moves that bring the horse to strategic positions where its uses would be more appreciated. As most Chinese Chess experts will agree, the horse towards endgame - where number of pieces remaining on board is relatively few, would be more important than cannon, due to its ability to threaten the opponent's king with more ease. At opening game, a horse is seen less important than a canon due to its low 'speed' and difficulty to threaten opponent's pieces far away, especially for a search depth of 4. The large number of pieces on board during early stages of the game also made it easier for cannon mounting to take place. Importance of cannon at the beginning and possibly mid game thus made it a more valuable piece in the opinion of coevolution, if opening book is not used. With

opening book, all games are brought to midgame stage before the evolved parameters are used. At this stage, a horse might sometimes prove more useful than the cannon. As such, some simulations do recognize the importance of horse, leading to a larger number of players which favour horse over cannon.

TABLE X
AVERAGE PERFORMANCE OF BEST EVOLVED PLAYERS (WITH OPENING BOOK)

Simulation	Wins	Draws	Losses	%Wins	Z-value
Parameter Set A	91	37	72	55.84%	1.489
Parameter Set B	59.2	55	85.8	40.82%	-2.151

TABLE XI
EVOLVED PIECE VALUES (OPENING BOOK WITH PARAMETER SET A)

Piece	Simulations				
	1	2	3	4	5
Advisor	195	170	81	136	231
Elephant	17	106	90	100	239
Horse	517	368	316	784	715
Rook	907	862	657	905	1018
Cannon	491	324	312	768	713
PawnOne	118	56	45	80	186
PawnTwo	310	116	140	86	262

TABLE XII
EVOLVED PIECE VALUES (OPENING BOOK WITH PARAMETER SET B)

Piece	Simulations				
	1	2	3	4	5
Advisor	194	125	246	130	238
Elephant	164	275	236	373	81
Horse	526	513	581	501	528
Rook	985	951	895	1015	1001
Cannon	421	626	591	568	612
PawnOne	205	230	167	123	32
PawnTwo	359	417	295	301	238

VII. CONCLUSION

In conclusion, this paper shows the successful application of coevolutionary approaches to discover strategies in Chinese Chess. Interesting traits are revealed when various coevolution models are assessed under different settings, with and without opening book. Results show that the coevolved players can perform relatively well, with the cooperative model being best for finding good players under random strategy initialization and the host-parasite model being best when strategies are initialized with a good set of starting seeds.

Possible future works can include the formulation of a two-tier coevolution process which first evolve position values that are dependent on two parameters – row and column values, instead of the current 50 independent values. This allows for an easier search of optimal regions before performing fine tuning of solutions by independently evolving the 50 position values. Coevolution can also be used to engage the tuning of two parameter sets – one for midgame and one for endgame (assuming opening game follows the opening book), instead of using only just one set. This allows for the shift in relative importance of different piece values as the game proceeds from midgame to endgame. With the combination of both human expert knowledge and the coevolutionary algorithm's capability to perform good parameter tuning in vastly complex environments, it is hoped that computer Chinese Chess players can eventually acquire the ability to attain grandmaster levels.

REFERENCES

[1] S. J. Yen, J. C. Chen, T. N. Yang, and S. C. Hsu, "Computer chinese chess," *ICGA Journal*, vol. 27, no. 1, pp. 3-18, Mar 2004.

[2] Y. T. Zhang, "Application of artificial intelligence in computer chinese chess," M.Sc. thesis, Department of Electrical Engineering, National Taiwan University, Taiwan, 1981.

[3] D. E. Goldberg., *Genetic Algorithms in Search, Optimisation and Machine Learning*. Addison Wesley, 1989.

[4] P. Darwen and X. Yao, "Coevolution in iterated prisoner's dilemma with intermediate levels of cooperation: Application to missile defense," *International Journal of Computational Intelligence Applications*, vol. 2, no. 1, pp. 83–107, 2002.

[5] C. K. Goh, H. Y. Quek, K. C. Tan and H. A. Abbass, "Modeling civil violence: An evolutionary, multi-Agent, game-theoretic approach," in *Proceedings of the IEEE Congress on Evolutionary Computation,* 2006, Vancouver, Canada, 16-21 July, pp. 6088-6095.

[6] C. H. Yong and R. Miikkulainen, "Cooperative coevolution of multi-agent systems," University of Texas, Austin, USA, Tech. Rep. AI01-287, 2001.

[7] D. Hillis, "Co-evolving parasites improve simulated evolution as an optimization procedure," *Artificial Life II*, *Santa Fe Institute Studies in the Sciences of Complexity*, vol. 10, pp. 313–324, 1991.

[8] M. Potter and K. De Jong, "A cooperative coevolutionary approach to function Optimization," in *Proceedings from the fifth conference on Parallel Problem Solving from Nature*, 1994, pp. 530–539.

[9] M. Potter and K. De Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evolutionary Computation,* vol. 8, no. 1, pp. 1–29, 2000.

[10] J. B. Pollack and A. D. Blair, "Coevolution in the successful learning of backgammon strategy," *Machine Learning*, vol. 32, no.1, pp. 225–240, 1998.

[11] K. Chellapilla and D. B. Fogel, "Evolving neural networks to play checkers without relying on expert knowledge," *IEEE Transactions on Neural Networks*, vol. 10, no. 6, pp. 1382-1391, Nov 1999.

[12] S. Y. Chong, M. K. Tan and J. D. White, "Observing the evolution of neural networks learning to play the game of Othello," *IEEE Transaction on Evolutionary Computation*, vol. 9, no. 3, pp. 240-251, 2005.

[13] D. E. Moriarty and R. Miikkulainen, "Discovering complex Othello strategies through evolutionary neural networks," *Connection Science*. vol. 7, no. 3, pp. 195-209, 1995.

[14] A. Lubberts and R. Miikkulainen, "Co-evolving a Go-playing neural network," in *Proceedings of the Genetic and Evolutionary Computation Conference Workshop Program*, 2001, pp. 14-19.

[15] D. B. Fogel, T. J. Hays, S. L. Hahn, and J. Quon, "A self-learning evolutionary chess program," in *Proceedings of the IEEE International Conference*, vol. 92, no. 12, pp. 1947-1954, Dec 2004.

[16] T. A. Marsland, "Computer chess and search," in *Encyclopedia of Artificial Intelligence*, S. Shapiro, Ed. J. Wiley & Sons, 2nd edition, 1992, pp. 224-241.

[17] S. Russell and P. Norvig, *Artificial Intelligence – A Modern Approach*, *2nd Edition*. New Jersey: Pearson Education Inc, 2003.

[18] D. F. Beal, "A generalized quiescence search algorithm," *Artificial Intelligence*, vol. 43, no. 1, pp. 85-98, 1990.

[19] C. D. Rosin and R. K. Belew, "Methods for competitive co-evolution: Finding opponents worth beating," in *Proceedings of the Sixth International Conference on Genetic Algorithms*, 1995, pp. 373-380.

[20] R. P. Wiegand, "An analysis of cooperative coevolution algorithm," Ph.D. Thesis, George Mason University, Fairfax, VA, 2004.

[21] D, Cliff and G. F. Miller, "Tracking the red queen: Measurements of adaptive progress in co-evolutionary simulations," in *Proceedings of the Third European conference on Artificial Life*, 1995, pp. 200-218, Springer-Verlag, LNCS 929.

[22] R. A. Watson and J. B. Pollack. "Coevolutionary dynamics in a minimal substrate," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2001, pp. 702-709.

[23] E. D. De Jong, "Intransitivity in coevolution," in *Proceedings of the Eighth Conference on Parallel Problem Solving from Nature*, 2004, pp. 843-851.

[24] S. Nolfi and D. Floreano, "Co-evolving predator and prey robots: Do 'arms races' arise in artificial evolution?" *Artificial Life,* vol. 4, no. 4, pp. 311-335, 1998.

[25] M. Buro, "Towards opening book learning," *International Computer Chess Association Journal*, vol. 22, pp. 98–102, 1999.