Multimedia Game Based Fitness Function Optimization in Evolutionary Search Process

Dharm Singh
College of Technology and Engineering, MPUAT
Udaipur, India

Thaker Chirag S
Research Scholar, Faculty of Engg.
SGVU, Jaipur

Shah Sanjay M
Research Scholar, Faculty of Engg.
SGVU, Jaipur

ABSTRACT

At the leading edge of Artificial Intelligence, machine learning game applications use a combination of various algorithms and different types of information. Searching the large space of solutions in depth leads to better solution. In checker board game next move of disc is important to defeat the opponent. Different selection strategy can be employed to select best next move. In this paper, we present comparative performance of roulette wheel selection and tournament selection method. The focus of this paper is to incorporate systematic game playing approach by analyzing game of checkers. Expert game players reveal three major playing strategies to make game winning moves. The game moves are divided into three stages opening game, middle stage and endgame. An evolutionary program plays game of checkers with an intention to build resilient middle stage and a set of predefined rules are incorporated to make calculated moves in an endgame. The paper is organized into the sections of Introduction, Introduction to Checkers, Game Complexity and Genetic Algorithm. The last three sections are Implementation, Result Analysis, Conclusion and references.

Keywords-Checkers,Evolutionary Algorithm, Genetic Algorithm, Fitness, Roulette Wheel Selection

1. INTRODUCTION

Game playing is the one of the research area of Artificial Intelligence from its foundation. Games require refined intelligence in a well-defined problem where success is easily measured. Games have therefore proven to be important field for studying problem solving techniques. Many games have many possibilities for a human to understand before making any valid-sharp move at any given stage. Since past few decades of research area of computers games, many powerful learning methods have evolved to use knowledge and search to make game playing decision on the board. The old techniques of artificial intelligence work well with games and to large extents such techniques were developed, tested and improvised for such games [1],[2]. Any board game in general and Game of Checkers in particular will definitely use the AI research through its collected results. This paper present a more human like approach to game playing by using genetic approach. While developing game, the most important part is to make a move like a human player through collected results. Fitness function is used to decide which the best move. Various selection methodologies can be employed to select best parents to generate best offspring based on fitness value. Two methods were tested Roulette Wheel and Tournament Selection method. Checkers game can be solved using brute-force approach and another approach is using optimizing through genetic algorithm. The game learning domain and programs for making move are primarily an optimization problem. The game playing programs have two major areas for the consideration:

- Decision complexity, the difficulty of making correct move decisions,
- Space complexity, the size of the search space[3]

2. INTRODUCTION TO CHECKERS

There are many versions played worldwide. Checkers is a game for two players. It is played on an 8x8 checkered board, with a dark square in each player's lower left corner. (Figure-1) Pieces move only on dark squares which are numbered. Numbers are used to record the moves, for example, if Red moves from square 9 to square 13, then it is recorded as: 9-13 Each player controls its own army of pieces (men). The player who controls Red pieces moves first. The pieces (also known as 'men') are arranged as shown on the left. The goal in the checkers game is either to capture all of the opponent's pieces or to blockade them. If neither player can accomplish the above, the game is a draw.

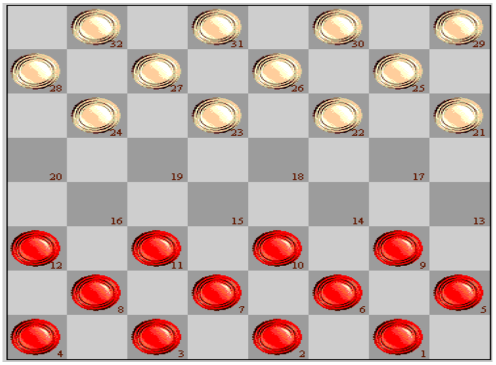

Fig.1. Checkers Initial Board

2.1 History of The Game

Checkers, as it is known in Great Britain, has ancient roots. It is thought that the earliest form of checkers was a game discovered and in an archeological dig at Ur in Iraq. Carbon dating makes it appear that this game was played around 3000 B.C. However, the game used a slightly different board, a different number of pieces and no one is quite certain of the exact rules.In ancient Egypt a game called Alquerque, which had a 5X5 board was a common and much played game. Historians have traced it as far back as 1400 B.C. It was a game of such popularity that it was played all over the western world for thousands of years.Around 1100 a Frenchman got the idea of playing the game on a chess board. This meant expanding the number of pieces to 12 on a side. It was then called "Fierges" or "Ferses". It was soon found that making jumps mandatory made the game more challenging. The French called this version "Jeu Force". The older version was considered more of a social game for women and was called "Le Jeu Plaisant De Dames". Now the rules for checkers were set and the game was exported to England and America. In Great Britain the game was called "Draughts". Books were written on the game in Spain as early as the mid-1500s and in England a mathematician name William Payne wrote his own treatise on Draughts in 1756. [4]

2.2 Types Of Checkers

There are more than 150 variants found worldwide. Some of them are

2.2.1 *International Draughts*

In the International draughts variety of checkers, the game is played on a 10x10 board with 20 checkers pieces being given to each player. In this variant, kings are allowed to move across several squares just as long as the squares are open. This rule is also commonly known as "flying kings". If any player has the option to take more than one path to jump and capture his or her opponent's checkers pieces, he or she must take the option that will result in the capture of the most checkers pieces. If any checkers piece lands in the king’s row during a jump, it must proceed with another capture backward if the option is available. If the move does not end with the checker in the king row, it will not be crowned even though it has passed through that row.

2.2.2 *Canadian Checkers*

This variety of checkers is played on a 12x12 board with 30 checkers pieces given to each player. However, the rules are the same as those of International draughts.

2.2.3 *Brazilian Checkers*

This type of checkers is played on an 8x8 board. Again the rules are just the same as International draughts.

2.2.4 *Italian Checkers*

This checkers type is played on an 8x8 board, with the main difference from British-American checkers being that regular checkers pieces are not allowed to capture kings.

2.2.5 *Chinese checkers*

Although it shares a similar name, Chinese checkers is not actually a checkers variety, and is played on a star-shaped board with marbles or pegs. [5]

3. GAME COMPLEXITY

The game of checkers has roughly 500 billion- billion possible positions (5 × 1020). The task is very daunting to solve the game, determining the finishing result in a game with no error made by either of the player. Since last three decades, almost incessantly, dozens of computers have been working on solving Game of Checkers, applying state-of-the-art soft computing based techniques to improve the learning process [6]. Game of Checkers represents the most computationally challenging game to be solved to date. Evolutionary Learning challenges in Game of Checkers are:

- The space to be searched is huge. It is estimated that there are up to 5X1020 possible positions that can be searched. So any search algorithm based method which is based on exhaustive search for the problem space is infeasible.
- The search space volume is not smooth and straight forward. An evaluation function’s parameters which is feature construction based are highly inter-dependent. In some cases increasing the values of optimization parameters will result in a worse performance, but many a times the controlled set of evolutionary parameter is also increases performance, then an improved overall performance would be obtained.
- The problem is not well understood by researchers. Even though all top performing programs parameters are hand tuned by their program designers, finding the best value for each parameter is mostly based on operational genetic alternatives [7][8].

3.1 Moves in checkers game

Starting with Red, the players take turns moving one of their own pieces. A 'piece' means either a 'man'--an ordinary single checker or a 'king' who is what a man becomes if it reaches the last rank (see kings). A man may move one square diagonally only forward--that is, toward the opponent--onto an empty square. Thus, on the diagram on the right-hand side, the red pieces can move 12-16, 11-16 or 11-15. Similarly, the white pieces can move 24-20, 24-19 or 23-19. (Figure -2).

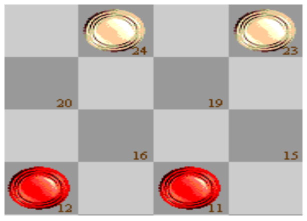

Fig.2. Moves in checkers game

3.2 Captures ('Jumps')

Checkers rules state that captures or 'jumps' are mandatory. If a square diagonally in front of a man is occupied by an opponent's piece, and if the square beyond that piece in the same direction is empty, the man may 'jump' over the opponent's piece and land on the empty square.

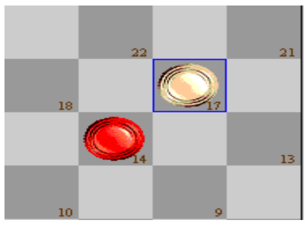

Fig.3. Capture in checkers

The opponent's piece is captured and removed from the board. Thus, on the diagram on the right-hand side red can 'jump' 14-21, leaving square (where white man used to stand) 17 empty. Similarly, if it were white turn to move, the white man could 'jump' over its red counterpart 17-10, leaving square 14 empty.(Figure-3) If in the course of single or multiple jumps the man reaches the last rank, becoming a king, the turn shifts to the opponent. No further 'continuation' jump is possible.

3.2.1 The kings

When a single piece reaches the last rank of the board by reason of a move, or as the completion of a 'jump', it becomes a king; and that completes the move, or 'jump' A king can move in any direction and 'jump' in any direction one or more pieces, as the limits of the board permit. The king can only jump diagonally over one adjacent piece at a time, in any of the four diagonal directions. Multiple jumps are possible.

4. GENETIC ALGORITHM

There are many different variants of Genetic Algorithms, but in the basics, these algorithms share the same philosophy. In a population of individuals, the strong ones survive longer than the weak ones: survival of the fittest. This causes a rise in the overall fitness of the population. Based on the fitness value of the individuals, the weak ones are terminated. The strong ones are chosen to reproduce themselves by using recombination and/or mutation on them. Recombination is an action that is applied to two or more of the selected candidates (called parents). It will result in one or more new candidates (children). Mutation is applied to one candidate and results in one new candidate. Execution of these two operators leads to a set of new candidates that compete with the old ones for a place in the next generation. When this process is repeated, the average fitness of the population will increase until a maximum has been reached. There are two elements that are important in an evolutionary algorithm.

- The variation operators (recombination and mutation) that create diversity. Different techniques can be used for both of them.

-The selection process of which individuals will be terminated, and which will be parents for the next generation. Evolution is a process of adaptation. The fitness function that is used for evaluating the individuals tells us something about the requirements that are needed to survive in the environment. To obtain a higher fitness value, the population needs to adapt increasingly more to the environment. Before different methods and operators are explained, the basic algorithm is discussed.

for each candidate in the population do Initialize candidate with random solution Evaluate each individual end for repeat

- EVALUATE population;
- RECOMBINE pairs of parents;
- MUTATE the resulting children;
- REINSERT new candidates;
- TERMINATE the non-parents;

Algorithm: The evolutionary Process

First, the population needs to be initialized with random candidates. Next, the loop is entered in which the best candidates are selected as parents. After that, the variation operators are applied. First, pairs of parents are recombined. This is a stochastic process. The pieces of the candidates that are recombined are determined randomly. Next, mutation is applied. The parts that are mutated are chosen randomly. The next step in the loop is reinserting the new candidates (the children). Finally, the individuals that were not parents are terminated.

4.1 Fitness Function

Fitness function is the important parameter of the genetic algorithm that defines the fitness of each chromosome where the values of genetic parameters are adapted as the genetic evolution progresses. At ever generation, fitness value of each chromosome is calculated using fitness function. If fitness function of two chromosomes is equal, then the mutation rate is increased, in order to help the genetic evolution get out of issues like local, maxima or local minima whichever is applicable. Once there is an improvement in the overall fitness, the original mutation rate is restored to continue evolution as normal. If the evolution stabilizes, but the fitness does not seem to be improving for several generations and the search does not find any error, new set of initial population is generated using the initial default parameter values and a new randomly generated seed. [9][10]

The static board evaluation function is essentially a weighted-sum of features score based on the various properties of the board. The board features considered when evaluating a terminal board are the usual properties through important by human players such as: number of pieces, mobility count, center-control, advancement of pieces, etc. [11] Thus evaluated score for a board may be viewed as simple linear polynomial, usually represented as follows:

Fitness Scores = (A1 x B1) + (A2 x B2) + ……+ (An x Bn)

Where Fitness Scores is called the static evaluation of a game board configuration The Ai’s are features that play important roles in game-playing strategies The Bi’s are weights that indicated the relative importance of the features. Evaluating Scores, different fitness of the each chromosome is found in the checker game.

4.2 Selection

Based on the value of the fitness function, a weighted roultte function [Code-1] selects the next best possible move chromosome that will create a new generation and be genetic parents for the next generations. It also allows for some parent with low fitness to go to the next generation; this way chromosome with previously good performance but weak results weak results after the latest genetic modifications, can be maintained in order to possible regain or improve their fitness value.[12][13][14]

Code-1: Function of weighted roulette selection

```
Random r= new Random();
 Double rouletteWheelPointer =
r.nextInt()*Totalfitness;


Int totFitness=0;
  for (idx=0; idx<POP_SIZE &&
rouletteWheelPointer >0; ++idx) {
roulettePointer-=
Arrays.binarySearch(pi,poplation[idx]);
  }
 return population[idx-1];
```

Code-2: Function of tournament Selection

```
Random m_rand=new Random();
String selectedparents[]=new
String[TOTAL_CROSSOVERELEMENTS];
int i=0;
while(i< TOTAL_CROSSOVERELEMENTS)
{
int first_random = m_rand.nextInt(POP_SIZE);
int second_random=m_rand.nextInt(POP_SIZE);
if(first_random > second_random)
selectedparents[i++]=parents[first_random];
else
 selectedparents[i++]=parents[second_random];
  }
 return selectedparents;
```

5. IMPLEMENTATION

The population size was set to 100. The initial values are selected randomly from the 1024 pool size. Fitness proportional selection was employed through the evolution function. Each game is played and the values of the fitness are forwarded to the next generation. The crossover and survivor rate is 89% and 11% for all generations respectively. Mutation was kept as low as 0.1%. The experiment lasted for 5 generations.

6. RESULT ANALYSIS

Figure – 4 show the initial generation when it is populated randomly. The graph shows both are almost equal fitness value for the population

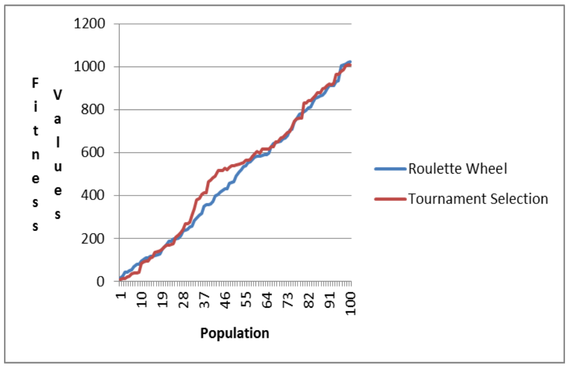

Fig.4. Fitness values of chromosome first generation

Figure-5 shows the initial fitness values of the chromosomes in the population in increasing order.

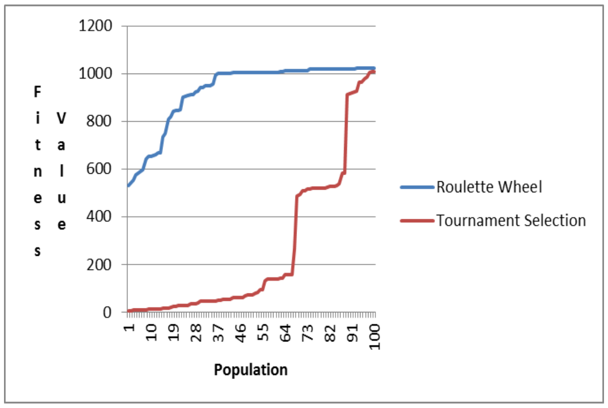

Fig.5. Fitness values of chromosome of fifth generation

When both the selection method applied, the fitness of next generation is observed as shown in figure-5. Each generation is generated using mutation and crossover operator using roulette selection method and tournament selection method. It seems that the roulette wheel selection method has improved in next generation while the tournament selection method has very few fit chromosomes in the next population.

As the population generated by the roulette wheel selection method has stronger fitness values which leads to better move of disc in the board positions which will lead to win the game while the population generated by the tournament selection method has less fit population which may lead to loss of the game.

7. CONCLUSION

The move of disc in the checker game is highly depends on the fitness function. The analysis of the fitness values is driving force of the move optimization. The evolutionary process of selection-crossover-mutation for the fitness function using genetic algorithm gives some rearrangement of the fitness values to produce excellent moves against the opponent. Comparative result analysis for fitness value using roulette wheel selection method and tournament selection clearly shows that former method has generated very fit population compare to tournament selection method as shown in figure 5. So roulette method leads to proper move of the disc on the board against the opponent.

Genetic algorithm provides excellent path to explore the search space through the fitness value.

8. REFERENCES

[1] Hong, J.-H. and Cho, S.-B. (2004). Evolution of emergent behaviors for shooting game characters in robocode. In Evolutionary Computation, 2004. CEC2004. Congress on Evolutionary Computation, volume 1, pages 634–638, Piscataway, NJ. IEEE.

[2] J. Clune. Heuristic evaluation functions for general game playing. In Proc. of AAAI, 1134–1139, 2007.

[3] S.M.Shah, C.S.Thaker and Dr. Dharm Singh " Performance Improvement in Game Playing using Evolutionary Computation by Large Search Space Exploration " at International Conference on ETNCC 2011 at MPUAT, Udaipur on 22-24 April 2011 Xplore Digital Object Identifier:10.1109/ETNCC.2011.5958504)

[4] http://www.indepthinfo.com/checkers/history.shtml

[5] http://www.checkerslounge.com/varieties.html

[6] Osman, D. Mańdziuk, J.: Comparison of Tdleaf(λ) and Td(λ) Learning In Game Playing Domain PAL, N. R., ET AL, eds: 11th int conf. on neural inf. Proc (ICONP 2004), Calcutta, India. Volume 3316 of INCX, SPRINGER (2004) 549 {554

[7] Osman, D., Ma¶ndziuk, J.: TD-GAC: Machine Learning experiment with give-away checkers. In Drami¶nski, M., Grzegorzewski, P., Trojanowski, K., Zadro_zny, S., eds.: Issues in Intelligent Systems. Models and Techniques. Exit (2005) 131{145

[8] Pollack, J.B., Blair, A.D., Land, M.: Coevolution of a backgammon player. In Langton, C.G., Shimokara, K., eds.: Proceedings of the Fifth Arti¯cial Life Conference, MIT Press (1997) 92{98

[9] Play java dot com is available at http://www.playjava.com/checkers_game_online.html

[10] Shah, S.M.; Thaker, C.S.; Singh, D.; Multimedia based fitness function optimization through evolutionary game learning ,Emerging Trends in Networks and Computer Communications (ETNCC), 2011 International Conference on Publication Year: 2011 , Page(s): 164 - 168

[11] G. Kendall and G. Whitwell. An evolutionary approach for the tuning of a chess evaluation function using population dynamics. In Proceedings of the 2001 Congress on Evolutionary Computation, pages 995–1002. IEEE Press, World Trade Center, Seoul, Korea, 2001.

[12] Chisholm, K.J.; Bradbeer, P.V.G.; Machine learning using a genetic algorithm to optimize a draughts program board evaluation function Evolutionary Computation, 1997. IEEE International Conference on Publication Year: 1997, Page(s): 715 – 720

[13] Adriana Elena Chis, Vane a Chiprianov and Daniel Cernea 3C Checkers Expert System A Comparative Study of Search Depth and Expert Knowledge Infuence on Neural Network Performance, January 2007. http://www.cernea.net/wp-content/uploads/2010/03/3C_article.pdf

[14] Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Pub. Co. (1989)

[15] S. Y. Chong, D. C. Ku, H. S. Lim, M. K. Tan, and J. D. White, "Evolved neural networks learning Othello strategies," in Proc. Congr. Evol.Comput., vol. 3, 2003, pp. 2222–2229.

[16] R. Fortman, Basic Checkers (http://home.clara.net/davey/basicche.html, 2007).

[17] Seo, Y.G., Cho, S.B., Yao, X.: Exploiting coalition in co-evolutionary learning. In:Proceedings of the 2000 Congress on Evolutionary Computation. Volume 2., IEEE Press (2000) 1268{1275