Design and Implementation of the EinStein Würfelt Nicht System Based on Multi Attributes Evaluation

He Sun[1,a] , Fa Shao[1], , Shuqin Li[2,b]

[1]Beijing Key Laboratory of Internet Culture and Digital Dissemination Research, Beijing Information Science & Technology University, Beijing, China

[2]Beijing Information Science and Technology University, Beijing, China

[a]E-mail: lishuqin_de @126.com, [b]E-mail:s_hehe@126.com

Keywords: Computer game; Evaluation method; Monte Carlo; EinStein würfelt nicht(EWN); Artificial Intelligence; Multi attributes

Abstract: EinStein würfelt nicht (EWN), just beginning in China recently, has little relative research on it with the computer fighting algorithm. This paper designs and implements an EWN computer game system. The Monte Carlo algorithm, a large number of sets are simulated by rules, reduce random by dice generated. A situation evaluation method is proposed based on multi attributes: win rate, special position, remaining chesses and probability of being wiped out. The multi attributes evaluation played with static and single attribute which only use Monte Carlo algorithm, and the first one turns to be more effective and practical in experiments.

Introduction

The computer game, also called machine game, is a challenging, vibrant research field, which is also an important direction of the Artificial Intelligence(AI)[1]. Its main content is to make the computer be able to play the game - playing chess – like a human being. Considering that it is not only simple, convenient, economic and contains rich connotation and logical thinking changes, it's thought to be a good approach to study the human mind, and thus has drawn the attentions of foreign scholars.

A complete computer game system includes board representations, move generator, search engine and evaluation functions[2]. The Board representation refers to the description of chess layout during the game, including the locations of chesses and blanks. Move Generator means the possible moves against a specific layout, which requires a good understanding of the rules and full consideration of every feasible ways of walking. The search engine's goal is to find the optimal move in all viable moves, so it is the best performance simulation of human thinking. The evaluation function, the premise of search algorithm, is used to evaluate the game, and will have a direct impact on the accuracy of the search results.

The computer game is widely carried out in recent years in China. In 2006, China Association for artificial intelligence (CAAI) machine game professional committee is established, and held the first national championships in the same year in Beijing (CCGC). Since then, an annually nationwide university computer game competition has been held every year under the name of the committee. It is considered to have a very significant active role in cultivating the interest, abilities of scientific research, maybe consciousness and innovation spirits of the students. At present the game contains 12 sub parts, namely Connect6, military chess, Surakarta, EWN and so on. The EWN is introduced in 2012, so rare researches on it can be seen in China.

This paper gets started from the evaluation function, and applies Monte Carlo algorithm to the EWN game. Several factors are considered together to improve the performance of the system, which includes the results of numerous simulation games, the positions of the chess pieces, the numbers of remain pieces, all dead three pieces of other attributes.

Einstein Würfelt Nicht Introduction

EinStein würfelt nicht(EWN), is launched in 2004 by Ingo Althöfer, Math professor of a German central town[3]. It is included in Olympia computer game competition[4], China National College computer game series[5]. The rules of EWN are listed as follows:

- The game is played on a square board with 5×5 grid. The upper left corner for the red start zone. The lower right corner for blue start zone. As shown in Fig.1.
- Each player has six cubes, numbered one to six. During setup, each player can arrange the cubes as he or she sees fit within the triangular area of their own color.
- The players take turns rolling a six-sided dice and then moving the matching cube. If the matching cube is no longer on the board, the player moves a remaining cube whose number is next-highest or next-lowest to the rolled number. For example, number 4 and 5 of the pieces have been removed, if the dice show for 4, you can walk around number 3 or 6 pieces.
- Red pieces move direction is right, down and lower down; the blue side pieces move direction for the left and upper left. They can move a grid.
- Any cube which already lies in the target square is removed from the board.
- The object of the game is to either for a player to get one of their cubes to the far corner square in the grid (where their opponent started) or to remove all of their opponent's cubes from the board.
- Each set of each side for 4 minutes, lose on time; each game the two sides against up to 7 set. Turn the upper hand(player A one four five upper, player B two three six seven upper). The middle of two set without rest, to win the 4 set for the winner.

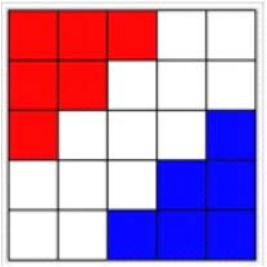

Figure 1. EWN board

The Design And Implementation Of Ewn Computer Game System

EWN is comprised of several main components: chessboard representation, the move generator, the search engine and the evaluation function. The chessboard represents the foundation of computer decision; the move generator is used to generate all possible ways; the search engine searches for a best way from all the possible ways started from current situation; the evaluation function is to evaluate a situation, to assess the situation merits.

Board representation

To make a computer play chess, the first step is formalize the chess board and layout. In here a 5x5 chess board is given to illustrate the formalization.

Define a 5*5 matrix to represent the board information, data structure with a two-dimensional array to record the information of the board. Array subscript represents the chessboard lattice, array value state chess pieces represent lattice, the board is defined as ChessBoard[5][5], as shown in Fig. 2. Among them, the number 11-16 of red pieces 1-6, digital 21-26 said the blue party pieces 1-6, number 0 means without the piece.

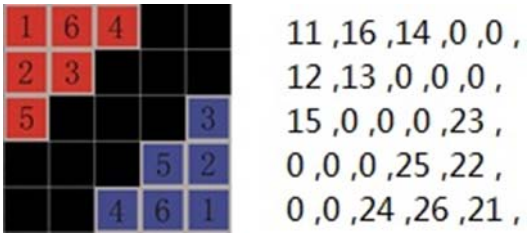

Figure 2. Board representation

Move Generation

Move generation is a component to generate all possible legal walking methods. The rules are quite simple. Dice to determine to go to pieces. Each piece has up to three ways of moving. In this paper, red chess as an example. Move were lower (Down) and right (Right), the lower right (Lower down). If the selected pieces that the right side of the chessboard, it can only go down. By the same token, the selected pieces at the bottom of the pieces can only go to the right.

EWN move generation steps:

Step1: The number of rolling the dice is equal to the number of red pieces. Go to step3;

Step2: The pieces that search greater than the number of rolling the dice, and less than the number in board of red pieces. If there were no pieces can go, go to step4;

Step3: Legal move from the pieces is saved in the multi tree by the leaf node. Go to step5;

Step4: Game over. Computer of red pieces is lost the game;

Step5: Move is over;

Move generation program flowchart as shown in Fig. 3.

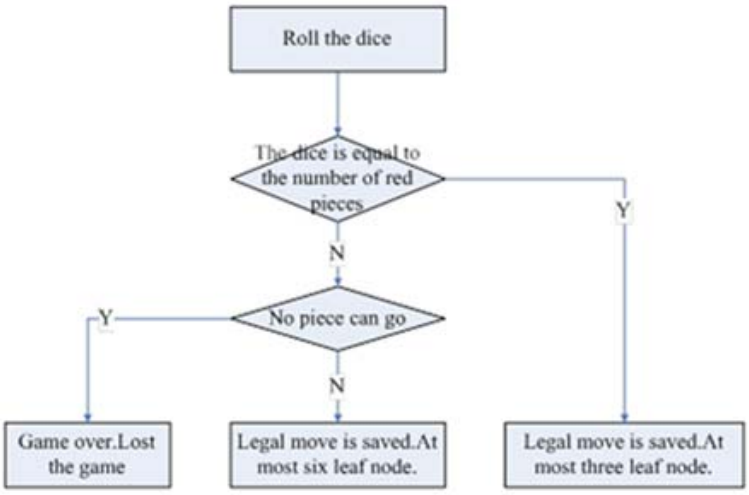

Figure 3. Move generation

Pseudo code algorithm is described as follow:

Input: Current board tree, the number of rolling the dice

Output: Subtree list

```
List<ManyTreeNode>getSubtree(Tree troot,int rollNum){
    List<ManyTreeNode>tlist=new ArrayList<ManyTreeNode>();
    int[] runNum = SelectNum(rollNum);
    if runNum.length == 0
            return null;
    else
            tlist = searchMove(troot,runNum);
    return tlist;
}
```

int[] SelectNum(int):return a array that the number of red pieces to go by input number

List<ManyTreeNode>searchMove(Tree,int[]): return a list through the root and the move pieces.

Monte Carlo Algorithm

Randomness takes a significant role in EWN, because `Throwing the Dice`, a random selection procedure, has to be taken before each step to determine the next chess number. This article tries to reduce the influences of randomness through the simulation of a large amount of random events. The Monte Carlo method is introduced to evaluate the outcome of each simulation, and thus ensures the validity of move.

The Monte Carlo(MC) algorithm, which is applied in the EWN game, can be described as follows: Simulate tens of thousands of games to the end, count the win rates of each method, then find the

method with highest win rates and entitle it as the best one. A simulation based on huge amount of random events will guarantee the validity of results. The MC method can evaluate a random game easily, by counting the win rates of each possible way and finding the way with highest rates.

Because the EWN uses a 5x5 board, a game takes 19 steps at average, indicating the depth of game tree is 19. So the outcome of a game can be decided when 10 steps have been taken by one side. It means the computer will get the calculated results quickly duration a simulation. An experiment with 600,000 simulations will cost 42.631s for each game at average. It is fast enough to complete the match in specified time.

As shown in Fig. 4, is Monte Carlo(MC) algorithm in application of EWN. EWN's opening and dice is six, create child node by move generation. Through the MC algorithm will outcome feedback to the process on a node.

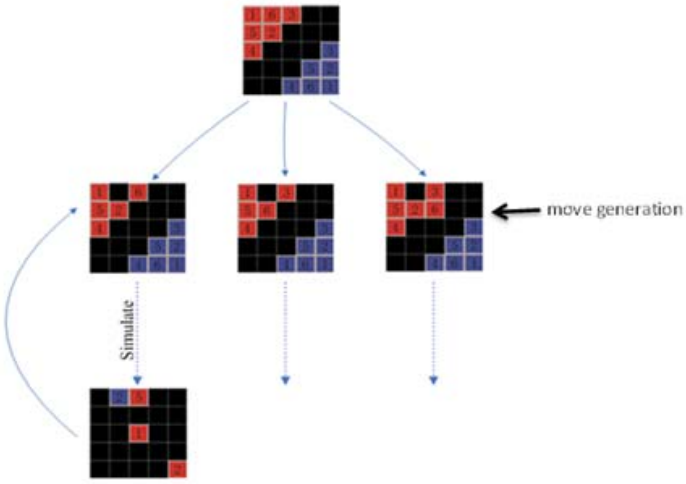

Figure 4. Monte Carlo Evaluation

MC simulation algorithm and formula is as follow:

Result of set: $X_{a,i} \in \{1,0\}$(win or loss) (1)

Winning: $\overline{X} = \frac{1}{a}\sum_{i=1}^{a} X_{a,i}$ (2)

In the formula (1) and (2), a means the times of the MC simulation. X is outcoming(win 1, lose 0). i is current set. $\overline{X}$ means the winning of player.

The design and implementation of function multi attributes evaluation

At present, there are mainly two kinds of method in EWN to evaluate the situation.

One is called static evaluation [6], mainly based on the position information of the existing chesses to judge the merits during a game. Its advantage is fast, with definite walking ways. The disadvantage is the search is endless, and is difficult to verify if the choice of position is good or bad.

Another is the evaluations based on MC algorithm. The MC algorithm cares only the outcome after the implementation of the situation, called as Single Attribute Evaluation in this paper. This method is more accurate than the static evaluation method. However, cases we have no pieces to go and lost games happen occasionally in experiment when the win rate is taken as the only factors into consideration. Less chesses brings more flexibility, which in turn leads to a higher win rate and more likeness to be annihilated. A large number of experiments show that it is very stable for both offensive and defensive when only three chesses are left.

For example, in the EWN game appears in the process as shown in Fig. 5. The dice number is two, according to the playing rules, there are three moves (Down, Right, Lower down). Based on evaluation method of single attribute, namely only consider the high winning percentage. It would go lower down, but it is easy to be eat by blue. And it is more likely to be wiped out by blue.

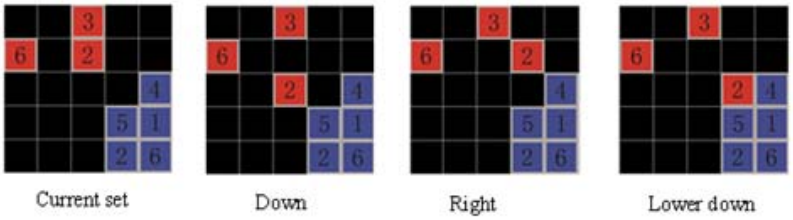

Figure 5. The move of dice is two

Through the analysis of Monte Carlo algorithm in the application of EWN in the process of evaluation, When in calculating the evaluation value, not only consider the winning, but also consider the special position , remain pieces, wipe out the remaining piece rate three features of the game the role of assessment. The following is the improvement of the evaluation function, named multi attribute evaluation function. The steps of the algorithm:

1) The establishment of the root by the current situation, and generating all sub tree nodes.
2) Monte Carlo evaluation in the sub nodes calculate each node record X;
3) Through the multi attribute evaluation algorithm formula, calculated for each node UValue;
4) Select the Max UValue sub node for the current move;

Among them, the multi attributes evaluation algorithm formula:

$$\text{UValue}=\begin{cases} \bar{X}-\frac{\beta}{\gamma}*\text{C} & (\text{remainR}<4, remainB>3) \\ 0.9 & (distance(num)=1) \\ \bar{X} & \end{cases} \quad (3)$$

In the formula above, β represent rounds that we have no pieces to go and lost games; γ means rounds that our opponent win games; C is set to 11 in the experiment as weight; remainR means remain pieces of the red, remainB means remain pieces of the blue; distance(num) represent the number's steps of distance to the end; For instance, distance(3)=1 represent choose the number 3 to the end of the distance is one step; There is only one step away from the end of the situation, set the UValue is 0.9. To prevent escaping eaten by the other side, one more step.

Pseudo code algorithm to calculate UValue:

Input: The number of simulation, Subtree node

Output: UValue

```
float calUValue(int countSimulate,Node SubNode){
  If distance(select piece)==1
    UValue= 0.9;
  else{
    for i=0 to countSimulate{
            result = Simulate(SubNode);
            if red==result
                    redwin++;
            else blue==result
                    bluewin++;
    }
    X=redwin/countSimulation;
    If redNum<4&&blueNum>3
    {
        If blueWin == 0
            UValue= X;
        else
            UValue= X –alldead/bluewin*13;
    }
    else
            UValue= X;
```

```
    }
    return UValue;
}
```

int Simulate():Simulation on a round, return outcome; “alldead” means rounds that we have no pieces to go and lost games.

Result

The development of system takes JAVA as the programming language, as it is Object-Oriented, well readable, and easy for maintenance. And JAVA supports visual user interface, which brings an easy operation experience for users. The development tools is Eclipse, the experimental system runs on PCs with CPU i5, 2.5G and 4G memory.

Two experiments are introduces below.

Experiment one: multi attributes evaluation and single attribute evaluation in contrast to the go chess.

The situation of Fig. 5 in Section 3.4, author compare experiment of two kinds of methods for multi attributes evaluation and single attribute evaluation. The experiment results as shown in Table1. Single attribute evaluation will choose to lower down, this is more likely to be wiped out by blue. Multi attributes evaluation select right move. It is much better to avoid lose the game that all pieces eat by blue.

TABLE I. WINNING PERCENTAGE IN FIG. 6

	Down Winning	**Right Winning**	**Lower Down Winning**
Value of single attribute	0.74383336	0.73785335	0.787125√
Value of multi attributes	-0.12824881	-0.12230843√	-0.30451018

Different move’s winning rates in two kinds of methods.

Experiment two: multi attributes, static and single attribute evaluation comparison of the effect.

The evaluation function is applied to the EWN game system in this paper. And the evaluation of single attribute and multi attributes evaluation, as well as in Ref.[6] static evaluation play each other. In the single attribute evaluation and multi attributes evaluation set simulate value of 600000, each of the program is divided into the upper hand and back hand, playing chess in the 30 sets. The experimental results are shown in table 2. And three strategies of winning in Fig. 6.

TABLE II. EXPERIMENTAL RESULTS

Upper VS Back	***Score***
Single attribute VS Static	21:9
Static VS Single attribute	12:18
Multi attributes VS Static	18:12
Static VS Multi attributes	11:19
Single attribute VS Multi attributes	14:16
Multi attributes VS Single attribute	17:13

Three different kinds of strategies play against each other. Upper is go first. And Back is go second.

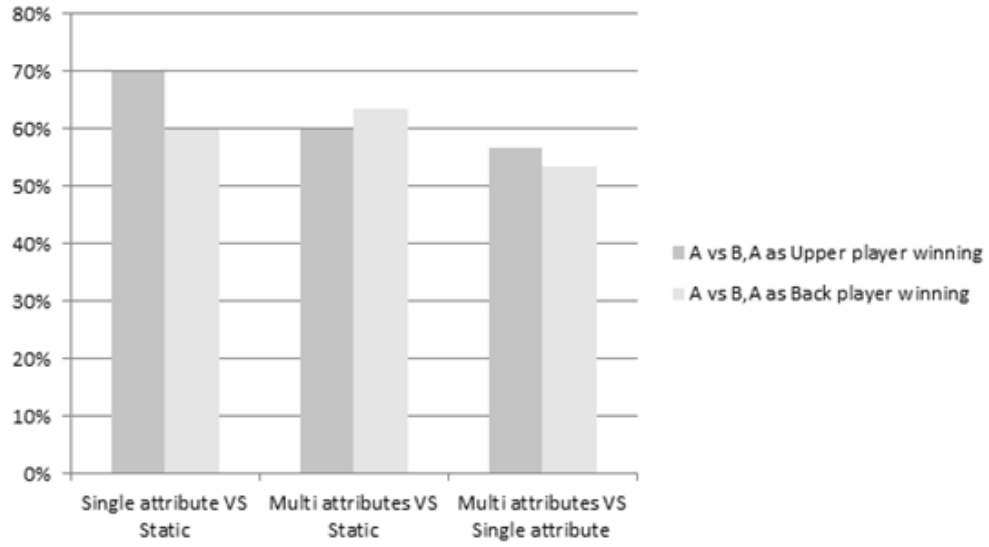

Figure 6. Winning of three strategies play each other

The experimental results through table 2 and Fig. 6 show that multi attributes evaluation is superior to the single attribute evaluation and static evaluation, single attribute evaluation is better than the static evaluation.

Conclusion

The evaluation function is one of the main factors that will decide the outcome of game. This paper introduces an evaluation function, based on MC algorithm, which also considers special positions, remaining chesses and all dead rates. The experiments show this method can greatly improve the likeness to win.

We can join the dice factors during the search process in the future for find pieces on both sides of more easily go. In order to make MC algorithm more efficient, more targeted simulation game, we can make the deeper search in the search process.

Acknowledgment

This work was supported by the Beijing information science and technology university graduate students of science and technology innovation projects, and by Opening Project of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research(ICDD2015), and by the Project of Construction of Innovative Teams and Teacher Career Development for Universities and Colleges Under Beijing Municipality(IDHT20130519).

References

[1] XU Xin-he, DENG Zhi-li, WANG Jiao, els. Challenging issues facing computer game research[J]. CAAI Transactions on Intelligent Systems, 2008,3(4), pp.289-293.

[2] XU Xin-he, WANG Jiao. Key Technologies Analysis of Chinese Chess Computer Game[J]. MINI-MICRO SYSTEMS, 2006,27(6), pp.961-969.

[3] Ingo Althöfer. On the Origins of "EinStein würfelt nicht!". http://www.althofer.de/origins-of-ewn.html.

[4] ICGA Tournaments. http://www.grappa.univ-lille3.fr/icga/games.php.

[5] WTN-EinStein würfelt nicht! Game rules. University Computer Games Championship. http://www.caaigames.net/cg.asp.

[6] ZHOU Wen-min, LI Shu-qin. The Reserch of Static Algorithms in WTN Chess[J]. Computer Knowledge and Technology, 2014,10(5), pp.1027-1031.

[7] Bernd Brugmann, FohringerRing 6. Monte Carlo Go [EB/OL]. [2013-09-30]. http://www.ideanest.com/vegos/MonteCarloGo.pdf.

[8] SHU Kang-yuan, HU Fu-qiao. Design of Chinese Chess Computer Game Engine[J]. Microcomputer Information, 2009, 25(10), pp. 39-41.

[9] KOCSIS L. SZEPESVARI C. Bandit based monte-carlo planning[C]//In 15th European Conference on Machine Learning(ECML), 2006, pp.282-293.

[10] Sylvain Gelly, David Silver. Monte-Carlo Tree Search and Rapid Action Value Estimation in Computer Go. Artificial Intelligence. Volume 175, Issue 11, July 2011, pp.1856-1875.