The Design and Optimization of Connect6 Computer Game System

Chang Liu，Bingke Wu，Sichen Wu

School of Computer & Communication Engineering, University of Science & Technology Beijing, Beijing 100083

E-mail: ustb_txliuchang@163.com

Abstract:

Computer game, a new field of artificial intelligence, as the name suggests, is to make the computer learn to think and play chess games like human beings. As one of the important research field of the artificial intelligence, computer game, which is considered as the touchstone of the artificial intelligence, has brought many important methods and theories to the field.Connect6, is a newly introduced game recent years, the research of game technology and algorithm on which is still remained relatively little. In this paper, I put forward a design and optimization of Connect6 computer game system , including technology of separating interface from computer kernel ,move generator, improvement of search strategy and system optimization based on threat. These technologies are all new explorations to Connect6 .The experiments and tests prove that our optimized program has advantages .And these technologies have helped us win the first prize in the 2013 National Undergraduate Computer Game Competition.

Key word: Connect6,separating interface from computer kernel,system optimization based on threat.

1.Introduction

Computer game, as the name suggests, is to make the computer learn to think and play chess games like human beings. As one of the important research field of the artificial intelligence, computer game, which is considered as the touchstone of the artificial intelligence, has brought many important methods and theories to the field..The well known success of the Deep blue chess indicated that the dream of a computer overcoming the top most human being chess master has become into reality[1].

Connect6, is a newly introduced game recent years, The professor Wu in Chiao Tung University in Taiwan proposes this new chess facing the fact that the forerunner has the advantage in five-in-a-row [2].The research of game technology and algorithm on which is still remained relatively little. The game platforms that we can find on the Internet nowadays are mostly played between human beings or some stupid robots programs. Connect6 has a simple rule, but a quite complicated game tree which is

almost as huge as the one of Go, that is the most reason of the low intelligence of the software found on the Internet. The traditional Connect6 game program usually consist of the following four parts: chessboard show, move generator, situation evaluation, and search strategy. [2]Chessboard show, and move generator are basic technologies while situation evaluation and search strategy are key factors which make computers have ability to think.

In this paper, I put forward a design and optimization of Connect6 computer game system, including technology of separating interface from computer kernel , move generator, improvement of search strategy and system optimization based on threat. These technologies are all new explorations to Connect6 .The experiments and tests prove that our Optimized program has advantages .And these technologies have helped us win the first prize in the 2013 National Undergraduate Computer Game Competition.

2. Key technologies and optimization of Connect6 System

2.1technology of separating interface from computer kernel

In order to reduce the amount of program development and focus our study mainly on the Connect6's kernel of search and valuation, the whole game system's interface and kernel of search and valuation are completely separate structure. In the process of implementation, our program uses the existing common CCGC （ Chinese Computer Games Conferences ） communication interface. The game software framework is shown as Figure1.

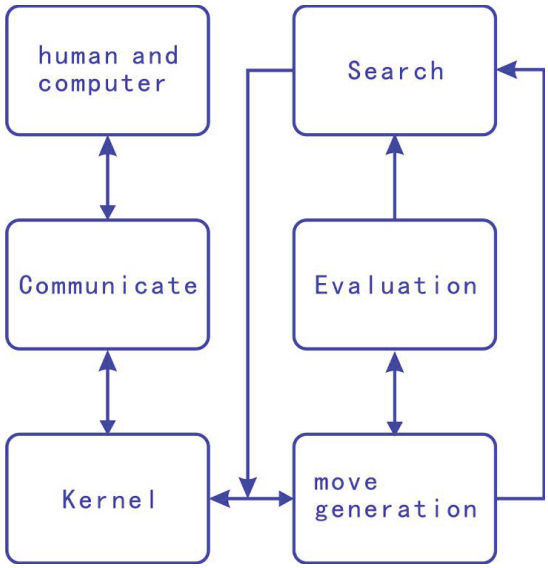

Figure 1

In the interface program, we can select whether the chessman is black or white.by operation. At the same time, the interface processing program will send an instruction "new black" for Black and send an instruction "new white" for White.

Later after each side makes a move, the interface processing program will send an instruction "move XXXX". Among this instruction, XXXX means Coordinates of the two current chessmen.

2.2Move generator

The main function of this part of the program is actually to let the computer program judge the range of the next chessman after seeing the chessboard and the current chessmen. In the design of the move generator, the program uses traditional chessboard scanning method, which is similar to five-in-a-row.

However, if a traditional n×n chessboard is scanned, the efficiency is too low. Assuming human chess player using black chessman, as the upper hand, falls the chessman in the central board, in accordance with traditional scanning methods, the program uses two "for" loops to enumerate coordinates without chessmen. Besides the fact that two chessmen are fallen at a time should be considered, the range that the next chessman can go is considerably huge, shown as $n^2 \times (n-1)2$.

In the Connect6, n is equal to 19.Consequently the range is up to 10 to the power of 10.

As the move generator can be called a large number of times throughout the search process. complexity of the move generator and the number of candidate points can directly affect the game tree width and the times of valuation function to be called. So the method above is unacceptable.

In so many candidate points, apparently we can find many safe points. For example, some empty corner points are unnecessary to consider, and points too far away from the current point can not be considered.

To reduce the number of candidate points on the chessboard, our program applies only to unoccupied points within three distance units to every occupied point on the chessboard (i.e. unoccupied points within every 7*7 square)as shown in Figure 2 .In the red range of 7 × 7 square ,some points also can not be considered. Our program searches in four directions shown in Figure 3. In this way, the number of test points reduces from 48 in the range of 7 × 7 to 24.

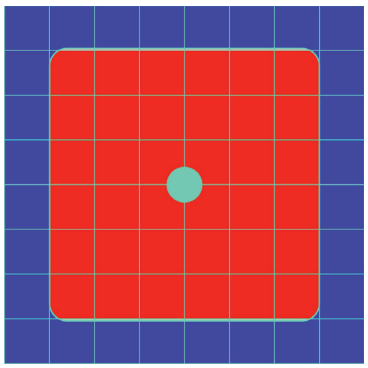

Figure 2

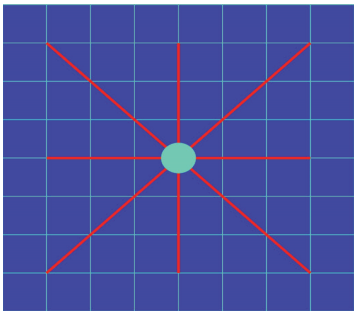

Figure 3

2.3 Improving of search Algorithm

There is a relatively independent method in connect 6,called VCF(Victory of Continuous Four) .This method is extremely offensive and This method is extremely aggressive in which it constantly makes Continuous Four to take the initiative until it has three threats making the opponent lose. To some extent, whether the VCF search is reasonable will determine the overall level of the program.

That is similar to Differentiated search that we propose. Differentiated search means the program distinguish between different move generations and pay attention to some important move generations. The program makes search depth different according to the Classification of the move generation.

For already classified move generation, Differentiated search is performed as the following steps:

1) Utilize different search Algorithms, such as Figure4

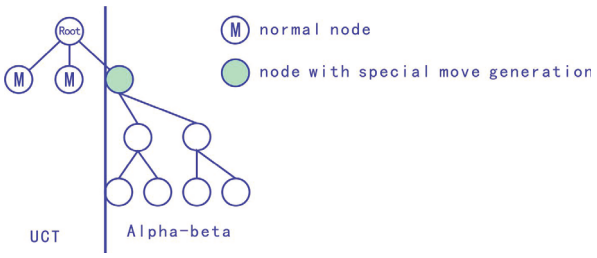

Figure 4

2) Utilize different search depth.

Our aim is to make branches of many nodes searched little, and branches of few nodes searched more. That only needs the function is monotonically decreasing.This is shown in Figure 5:

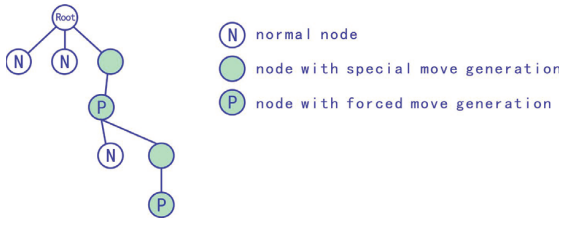

Figure 5

2.4The optimization method based on the threat value

It's quite important to form the chess type based on threat value, so we should pay special attention to the shape of this kind while specific valuation function scans the board at the same time.

The method is described as follows: while evaluation function scans the board, the program counts the chess type which has potential threat and will be a threat in the next step. Once finding such kind of chess type, the corresponding counter adds 1, and the current threat type which can win in the next step or can be formed as threat moves will be stored in an array or in the stack. Once the condition is met, the next candidate landing points will be generated directly from the array or the stack, in other word, instead of calling the CreatePossibleMove function in the original search engine in the NegaScout function, it will produce the next step from the winning move array, which makes the program more aggressive.

We can find that, in this method ,the main change is in the evaluation function. In details, when evaluation function scans and analyses the model, it also needs to analyses the kind of chess type which can win in the next step, such as live five, live four, sleep five, sleep four ,as well as the chess type which has potential threat and can form two threat values in the next step, such as live three, sleep three, live two. It is the most urgent problem that how it can dynamically recognize a chess type and analyze the next step which may generate the threat in the search process. But as the evaluation function in the search process of the game tree will be called in many times, so this method must be simple, as far as possible in less time。

The procedure uses an array called LineRecord to store the model analyzing of the current coordinate point, and the corresponding elements of the array are corresponding values for the coordinates which have own models, however, the corresponding elements of coordinates LineRecord without falling on the chessboard have no value. We can use this array to store moves which can threaten the next step, then they'll

be separately counted and stored in a separate stack or array.

```
if (RightEdge - LeftEdge == 4)     // Five chess pieces of the same color
connected
{
    if (RightEdge < GRIDNUM && Line[RightEdge+1] == NOSTONE)
    {
        if (LeftEdge && Line[LeftEdge-1] == NOSTONE)
        {
            LineRecord[AnalysePosition] = FIVE;   //live five
            LineRecord[LeftEdge-1] = IMPORTANT;
        }
        else
            LineRecord[AnalysePosition] = SFIVE; //dead five
        LineRecord[RightEdge+1] = IMPORTANT;
    }
    else if (LeftEdge && Line[LeftEdge-1] == NOSTONE)
    {
        LineRecord[AnalysePosition] = SFIVE;      //dead five
        LineRecord[LeftEdge-1] = IMPORTANT;
    }
    return LineRecord[AnalysePosition];// return to the current position of
chess-type analysis
```

Here are the codes that can make statistics of moves which form the threat in the next step using the LineRecord array:

Referring to the above code, it can achieve statistics of all the chess type which can generate threat in the next step. Based on the full use of the LineRecord array, it realizes statistics of all kinds of chess types which can generate the threat value in the next step, based on process analysis of the original model, by adding the minimum amount of code. All marked IMPORTANT moves are the points which can win or force rival to defense with two pieces in the next step.

Through the method described above, we can see that this is not only in the process of valuation function optimization, but in the process of search engines optimization.

3. Comparative Experiment

In order to test chess and the optimization effect of Connect 6 program adding the optimization method, we conduct a group of contrast tests.

The experiment was divided into two groups. Group 1 is using the optimized program while group 2 used not -optimized program. The two groups' opponent is the same. Each group carries out 20 independent experiments， in 10 of which our program is on the offensive with the black chess and in other 10 times is on defensive position with the white . We respectively count the times of the victory and the average search time per step. Statistical results can be seen in Figure 6 and Figure 7.

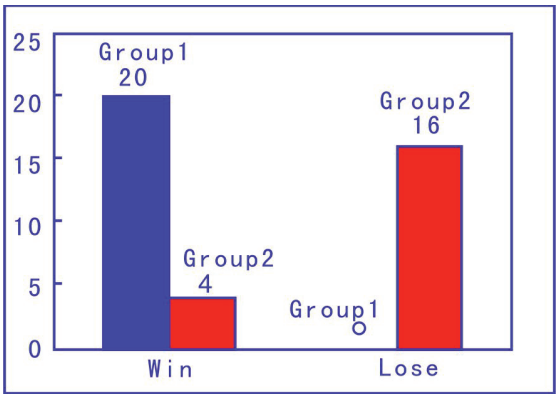

Figure 6

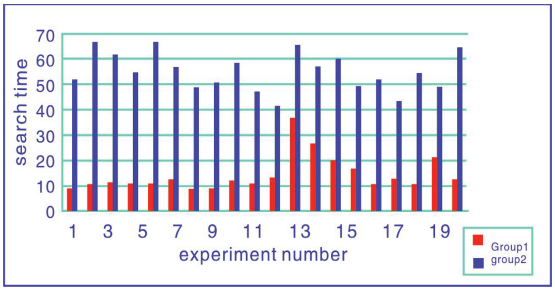

Figure 7

Seeing from the Figure 6, Group 1 obtains all 20 victories while Group 2 only wins 4 times and loses 16 times. This shows that the optimized procedure of the above is more likely to win. And in the actual test, it is found that, compared with the Group 2, the Group 1 program chooses points more reasonably and is more aggressive in attacking. Seeing from the Figure 7, comparing the two groups according to the average search time per step, the result is very obvious that Group 2 takes three times longer than Group 1 program.

Therefore, the optimized procedure in the game tree search process is more quick and effective, and optimization methods above can make the program focus on the threatening points our part and the opponent. In the end, we win the first prize

in the 2013 National Undergraduate Computer Game Competition.

5 CONCLUSION AND OUTLOOK

These achieved results show that: the proposed technology of separating interface from computer kernel ,move generator, improvement of search strategy and system optimization based on threat. do have good results. The improved program enables us to have won the first prize in the 2013National Undergraduate Computer Game Competition.

REFERENCES

[1] Zhang Ming-liang, Wu Jun,Li Fan-zhang Design of evaluation-function for computer gobang game system. Journal of Computer Applications.2012,32(7) : 1969 -1972.

[2]Wu Cheng Yi a new challenge game-connect6[J].2006China computer Game seminar．pp：7—16