UCT Algorithm in Imperfect Information Multi-Player Military Chess Game

Jiajia Zhang [1], Xuan Wang[2], Jing Lin[3], Zhaoyang Xu[4]

[1 2 3 4]Intelligence Computing Research Center Harbin Institute of Technology Shenzhen Graduate School

Abstract

UCT (Upper Confidence Bound Apply to Tree) is a new algorithm applies to huge branches mini-max game tree search which gains a great success on a Go program called MoGo. This paper proposes an imperfect information multi-player Military Chess system basing on UCT algorithm which also adopts Monte-Carlo sampling and some other classic game algorithms (alpha-beta pruning, TD (λ) Learning and history heuristic). The realization and modification of UCT for Military Chess application are also explained in this paper which improves significantly the performance of the system. Basing on the results of the experiments, the performance of modified UCT for imperfect information game system is discussed.

Keywords: computer game, huge branches, UCT, game tree search, Monte-Carlo, Military Chess

1. Introduction

One of the ways games are classified is whether or not they are perfect or imperfect information games [1]. In an imperfect information game, the players have non-singleton information set which means they have only partial knowledge about the state of the game.

Many researches are done regarding the imperfect information game systems, basing on Monte-Carlo approaches [1] and some other relevant algorithms (like alpha-beta pruning). Some successful game systems are realized basing on the above approaches. The bridge and heart game of Alberta University [2] is a good example of this.

However, when the branches size of the game tree (such as Go) becomes huge [3], the formal approaches can not do a satisfying work any more. A new algorithm UCT (Upper Confidence Bound Apply to Tree) was reported by Levente Kocsis and Csaba Szepesvari in 2006 which is an extended application of the Monte-Carlo algorithm [4]. Sylvain Gelly (University of South Paris) and Yizao Wang (Paris institute of technology) developed a Go program called MoGo basing on this algorithm [5]. MoGo has gained a great success in games with professional human Go players and CGOS [6]. As a typical imperfect information game, Military Chess has many similar characters as Go. Thus, a new system is developed basing on UCT to improve the performance of the Military Chess game system.

The main contribution of the system that is presented is the realization and modification of formal UCT which makes it more suitable for the characters of Military Chess application and can work together with the existing Monte-Carlo sampling algorithm.

This paper is organized as follows. Section 2 briefly introduces related work about the algorithms used in an imperfect information game. Section 3 gives a primary introduction of the Military Chess game. And then, a specific description is presented about the algorithm used and the realization details of our game system basing on UCT algorithm. Section 4 analyzes the performance of the new

system that is tested locally and on the internet. And finally, section 5 contains our conclusion.

2. Monte-Carlo Sampling and UCT

In imperfect information game, none of the players in the game have the whole knowledge about the current game state. In order to solve such problem in practice, one has to either resort to special cases, or fall back heuristics which is created by Monte-Carlo sampling [1].

2.1. Monte-Carlo sampling

Monte-Carlo sampling is an algorithm that deals with imperfect information games. When evaluating the choices of one player, the Monte-Carlo method randomly guesses the unseen pieces of the other players. The game is then searched using min-max game tree as if these pieces are known to each player. As demonstrated in Fig. 1, the square represents the max node and the circle represents the min node. Only the root node is the random (the diamond) and all the other nodes are board states with perfect information. The above process is done a number of times and the move that is best on average is chosen. The number of iterations to do is specified by the user. A state of perfect information is sampled each time and game tree search is applied to that state. After numbers of iterations, the move which is chosen the most frequently is chosen as the final best move.

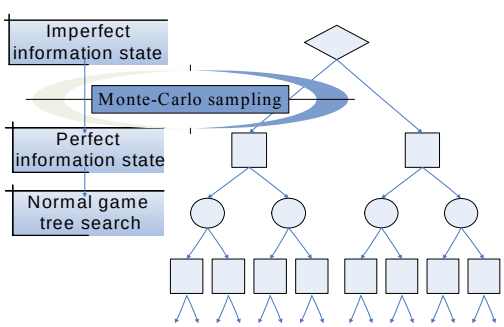

Fig. 1: Game Tree of Imperfect Information using Monte-Carlo Sampling

2.2. The UCT algorithm

The algorithm UCT (Upper Confidence Bound Apply to Tree) is the extension of UCB1 [4] to mini-max tree search. UCB1 is an algorithm that is an approach to solve bandit problem.

A K-armed bandit problem [5] is described to find the strategy for a bandit to choose his K arms to get the max reward. The problem is defined by random variables $X_{i,n}$ for $1 \le i \le K$ and $n \ge 1$, where each i is the index of an arm that the bandit can choose. Successive chooses the arm i yield rewards $X_{i,1}, X_{i,2}, \ldots$ which are independent and identically distributed according to a certain but unknown law with unknown expectation μ_i. Algorithm chooses the next arm depending on the obtained results of the previous plays. Let $T_i(n)$ be the number of times arm i has been chosen after n plays. In works of Auer and Al [7], algorithm UCB1 is given, which ensures the optimal arm is chosen exponentially more often than any other arms uniformly when the rewards are in [0,1]. Note:

$$\overline{X}_{i,s} = \frac{1}{s}\sum_{j=1}^{s} X_{j,r}^{2}, \overline{X}_{i} = \overline{X}_{i,T_{i(n)}} \quad (1)$$

$$V_j(s) = (\frac{1}{s}\sum_{r=1}^{s} X_{j,r}^{2}) - \overline{X}_{j,r}^{2} + \sqrt{\frac{2\log n}{s}} \quad (2)$$

Let (2) be an estimated upper bound on the variance of arm j, which is suggested in Auer and Al's work [7]. Then there is a new value to maximize:

$$\overline{X}_j + \sqrt{\frac{\log n}{T_j(n)} \min\{\frac{1}{4}, V_j(T_j(n))\}} \quad (3)$$

The policy maximizing (3) named UCB1-TUNED. The process of the algorithm work is firstly plays each arm once for initialization,

and then iteratively play arm *j* that maximizes (1) until the end condition appears which is set by player.

UCT [4] [9] is the extension of UCB1 to mini-max tree search. The idea is to consider each node as an independent bandit, with its child nodes as independent arms. Instead of dealing with each node once iteratively, it plays sequences of bandits within limited time, each beginning from the root of the game tree and ending at one leaf.

In the problems of mini-max tree search, what is looked for is often the optimal branch at the root node. However, in the condition of imperfect information game, the usual case is that the game tree has a very large depth and a huge branching factor. Thus, it is usually too difficult to find the optimal branch with in a limited time. Considering about this, it is sometimes acceptable if one branch with a reward near to the optimal one is found.

In this sense, UCT outperforms alpha-beta search for at least three major advantages. First, it works in an anytime manner. The algorithm can be stop at any moment, and its performance can be somehow good and this is not the case of alpha-beta search. Fig. 2 show if the alpha-beta algorithm is stopped prematurely, some moves at first level has even not been explored. So the chosen move may be far from optimal. Of course iterative deepening can be used, and solve partially this problem. Still, the anytime property is stronger for UCT. And it is easier to finely control time in UCT algorithm.

Second, UCT is robust as it automatically handles uncertainty in a smooth way. At each node, the computed value is the mean of the value because each child weighted by the frequency of visits. Then the value is a smoothed estimation of max, as the frequency of visits depends on the difference between the estimated values and the confidence of these estimates. Then, if one child-node has a much higher value than the others, and the estimate is good, this child-node will be explored much more often than the others, and then UCT selects most of the time the 'max' child node. However, if two child-nodes have a similar value, or a low confidence, then the value will be closer to an average.

Third, the tree grows in an asymmetric manner. It explores more deeply the good moves. What is more, this is achieved in an automatic manner. Fig. 3 gives an example. Fig. 2 and Fig. 3 compares clearly the explored tree of two algorithms within limited time.

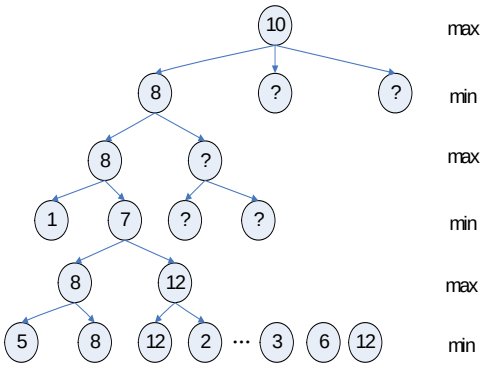

Fig. 2: Alpha-beta search with limited time

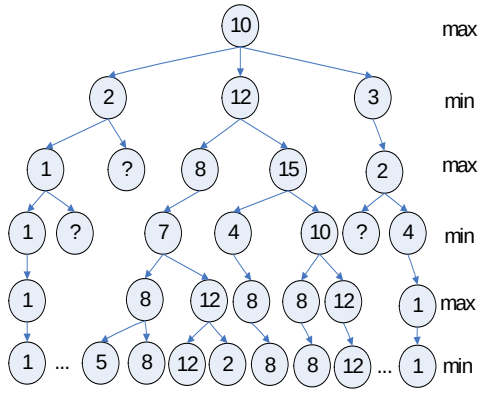

Fig. 3: UCT search with limited time

3. Military Chess Game System with UCT

In this section, the Military Chess game system is presented called SiGuoJunQi, which is based on UCT algorithm. First, a brief introduction about the Military Chess game will be provided.

3.1. Introduction about Military Chess

The Military Chess, which is also called Kriegspiel, has been a very popular game in China for many years, especially with the development of internet. Basing a statistic conclusion of Tencent Company [8], which is one of the most famous internet game operators in China, the number of Military Chess online players exceeds 300,000 at the same time.

The rules of the Military Chess are quite complex. Simply speaking, the basic process of the game is to move pieces, attack opponents' pieces and finally occupy the position of enemy's flag. Maybe it sounds quite like Chess but there are at least three main differences between them as following.

First, each player has an associate who sits in the opposite and two opponents sits aside, like Bridge so that collaboration should be considered in the game. Second, the pieces which have a certain military rank can only destroy those with lower ranks. Take the piece which has the rank of Major General for example, it has the power to destroy most of the other pieces when they conflicts except the pieces of General and Lieutenant General. The General, which is the highest rank piece on the board, can only be destroyed by Main, or perish together with Bomb or another General. Last but most important, player can not see the ranks of other players' pieces, even though it belongs to his associate. As the result, players have to judge the ranks of other pieces by the information appear in the process of game, and sometimes can only take a guess. This is the most difficult and complex point of the development of Military Chess game system.

3.2. Realization of System with UCT

In this section our system SiGuoJunQi is presented, which is a Military Chess game system using UCT algorithm. Fig. 4 illustrates the main components of the system.

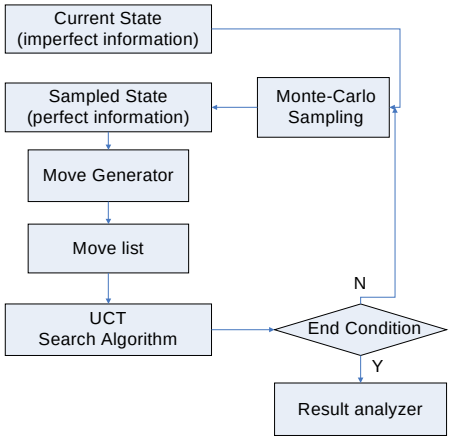

Fig. 4: The main components and flow chart of Military Chess game system

In the process of game, when it is the agent's turn to move, the current state that agent faces is an imperfect information state. Monte-Carlo sampling process transforms the imperfect information state to perfect information state. In the following, basing on the sampled perfect information, move generator module generates moves list and build the game tree. And then, UCT algorithm is used searching the optimal move from the game tree. When a move has been chosen, the system check the end condition, which could be set for a limit of running time or a limit times for system's iteration. If the end condition has not been fulfilled, the system comes to the next iteration. That means Monte-Carlo progress will sample from the current state to generate another state with perfect information. And the following step is as illustrated above. Otherwise, system will collected all the chosen moves in before iterations and result analyzer will take an analysis about them and finally choose the optimal move.

3.3. Modification of UCT for Military Chess Application

Comparing with the formal realization of UCT application, the Military Chess has some special characters, which requires modification of UCT algorithm.

An important character of Military Chess that conflicts the formal UCT process is the uncertainty of game state transmission. In formal UCT system, the Monte-Carlo sampling happen when the nodes are created and the nodes' values is computed. Fig. 5 shows the process of creating a new node.

As one of the advantages of UCT shows, the more one branch be chosen, the more samplings are implemented on it. That is why UCT is robust. However, with the different board situation of Military Chess game, same move may leads to different results which mean the uncertainty of game tree creation. Thus, Military Chess can not follow the process strictly. As the Fig. 6 shows, the Monte-Carlo sampling process is moved up before UCT search to transmit the imperfect information state to perfect information, which ensures the certainty of game tree.

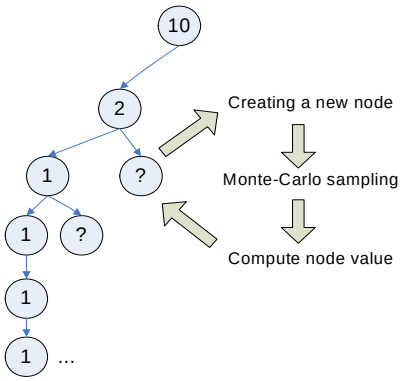

Fig. 5: formal Monte-Carlo sampling happens in UCT process

However, in this case, every branches of the game tree will implements the same sampling times, which conflicts the advantage of asymmetric search of UCT algorithm. As the solution, 20 times of Monte-Carlo sampling will be taken on the chosen five branches for the purpose of ensure the confidence of chosen moves. And the evaluate function will analyze the 20 results of samplings to calculate an average value for the high frequency explored nodes.

In the modified UCT program, five times of Monte-Carlo sampling is taken to create certain game trees, which means five optimal moves are chosen in different game trees. Next, 20 times Monte-Carlo sampling is taken on the five chosen moves for the smooth and the confidence of the values. And then, system analyzes the collected results and some other influential factors to choose the final optimal move.

A Military Chess game system is developed which is called SiGuoJunQi. The game interface is shown in Fig. 7 and Fig. 8 shows the ranks of pieces on the board. The system can operate a game both on single computer and on the internet.

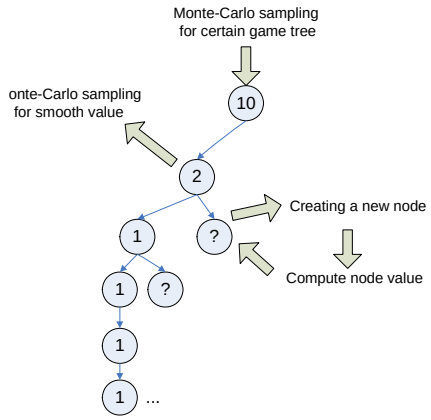

Fig. 6: The Modified UCT Process

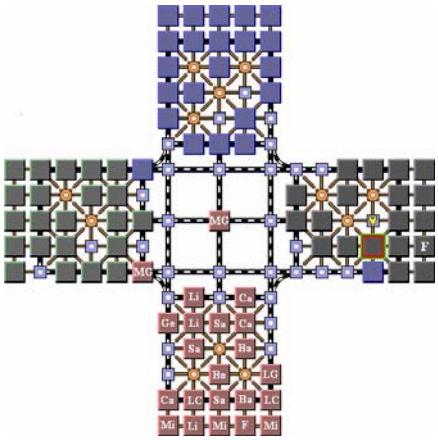

Fig. 7: the Interface of SiGuoJunQi

Rank of Piece	Abbreviation on Board
General	Ge
Lieutenant General	L．G
Major General	M．G
Brigadier General	B．G
Colonel	Co
Lieutenant Colonel	L．C
Captain	Ca
Lieutenant	Li
Sapper	Sa
Bomb	Bo
Landmine	Mi
Flag	F

Fig. 8: Ranks of Pieces on the Board

4. Experiments and Performance

In this section, experimental results will be listed to show the performance of the UCT algorithm. Being one of the earliest groups study on imperfect information Military Chess game, there is no parallel competitor that can be tested against our program. However, with the popularity of Military Chess in China, the human game has reached quite a high level on it.

As the result, our experiment is designed in two parts. One is playing against SiGuoJunQi 2.0 [10], which is the former edition of our system and was the first Military Chess program attends 2nd CCGC (Chinese Computer Game Championship). The other is implemented on the internet and against human player. In the experiments, the time limit is set by 15 seconds for each operation of UCT search.

4.1. Experiments that Play Against SiGuoJunQi 2.0

The first experiment is implemented by playing against SiGuoJunQi 2.0 which also uses the Monte-Carlo sampling but adopts Alpha-Beta pruning and history heuristic for the game tree search algorithm. Fig. 9 shows the wining rate of the program used UCT algorithm, which is marked by edition 3.0, in 100 times of playing against SiGuoJunQi 2.0. The contrast result of the performance between the two programs supports the advantages of UCT algorithm.

4.2. Experiments that Play Against Human on the Internet

Another experiment is on the internet and playing against human players. In the forty rounds of each edition, the computer does not show a satisfying performance comparing with human especially in two primary aspects. One is the computer agent does not have the skills about cooperation with its partner whilst it is one of most important strategies human player considers. The second is about the skills of “series ratiocination”. Experienced human players can get important information from a series performance of their competitor. The skill comes from the collected experience and sometimes from intuition which can not be simulated by computer.

Edition	Win	Draw	Lose	win rate
SiGuoJun Qi 3.0 (UCT)	73	6	21	73%
SiGuoJun Qi 2.0	21	6	73	21%

Fig.9: Experiment result in 100 games between SiGuoJunQi 2.0 and 3.0 (UCT)

However, comparing with the edition 2.0, the edition 3.0 which used UCT application has an obvious progress not only in the win rate, which is shown in Fig. 12, but also in the performance of each step.

There is a serious of values that is used to evaluate the situations during the process of a round play. They are formulated in mainly three parts:

7. The basic value and the flexibility factor of each pieces.
   Flexibility factor means the moveable range and the influence scale of the pieces. The value of each piece on board is the product of the basic value and the value of flexibility factor.
7. The occupation of the crucial position on the board. Totally 36 crucial position is evaluated.
7. The whole estimation of the round. This point includes the gross influence about the round, such as the max piece on the board which is called "order piece" and so on.

There are 77 related factors in all to evaluation the situation of the current step.

Piece	Value	Flexibility	Piece	Value	Flexibility
Ge	500	2.0	Ca	131	0.8
L.G.	400	1.8	Li	104	0.6
M.G.	320	1.6	Sa	163	0.4
B.G.	256	1.4	Bo	200	1.0
Co	204	1.2	Mi	320	1.0
L.C.	163	1.0	F	1000	1.0

Fig. 10: The basic value and flexibility of pieces in the beginning of the game.

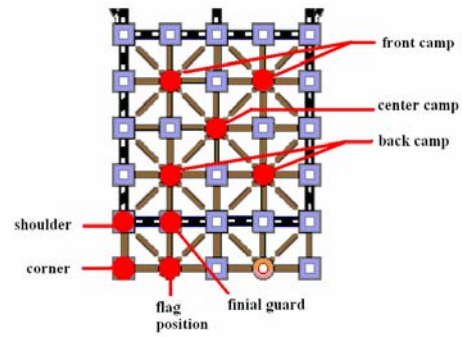

Fig.11: Crucial positions on the board (one side)

Fig.13 shows the two editions' performance of the process in 80 rounds games (40 rounds for each). The board situation is evaluated every 8 steps to illustrate the changing condition and calculate the average of the 40 rounds. The evaluate function analyzes 77 related factors that illustrates the situation of the board.

At least two advantages of the edition 3.0 can be conceded from the figure. One is about the evaluated value of board situation. Generally, the edition 2.0 will fall into an obvious disadvantage position in the second half of the game. However, no matter what the final result is, the edition 3.0 can holds a nearly balance of the round and rarely falls into a terrible inferior position even in the lose rounds. Another is about the shape of the situation curve. The edition 3.0 follows a relative smooth curve comparing with the edition 2.0's which has several rapid declines in the game. The rapid declination of the curve means a terrible lost of a single step. UCT algorithm avoids these by deeper search level and more sampling times on the chosen move to explore the latent dangers in limited time. The result supports that UCT search has a better performance.

Another point of the experiment result is that both the two edition will undeniable fall into inferior position in the end half of the round. That is because with the process of one

round, the advantages of human player, cooperation and series ratiocination, will be more and more important. This is also the one of the future work of us for the purpose of enhancement of our program.

5. Conclusions

The improved performance of the third edition of SiGuoJunQi shows the efficiency of UCT. It has some advantages compares with other game tree search algorithm such as Alpha-Beta pruning and history heuristic, especially when the tree has a huge branching factor and time is limited. The point is that it studies the nodes of tree with an asymmetric manner that concentrate the search resource (time and sampling times) on the moves that has more probability to be optimal.

The attempt of modifications is discussed which makes UCT algorithm more suitable for the Military Chess game system. And the program has gained a primary success in the experiments against computer, and some progress against human players.

Edition	Win	Draw	Lose	win rate
SiGuoJunQi 3.0 (UCT)	9	2	29	22.5%
SiGuoJunQi 2.0	2	3	35	5%

Fig. 12: Experiment result in 80 games of SiGuoJunQi edition 2.0 and edition 3.0 (UCT) against human player on the internet

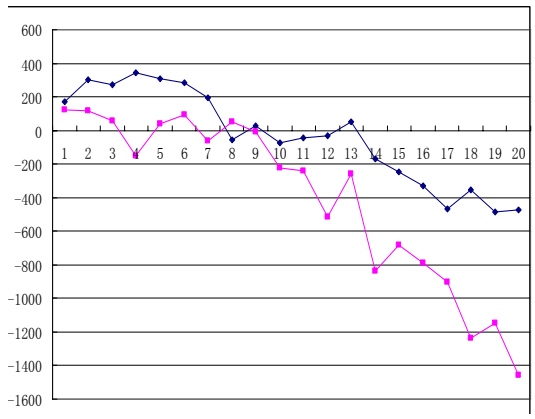

Fig. 13: the situation evaluation of the process of game. The blue curve denotes the edition 3.0 (UCT) and the red curve denotes the edition 2.0

6. Acknowledgement

We would like to thank Ma Xiao for the theory direction for our system. We also appreciate Wang Jinyi for his fundamental work of the SiGuoJunQi system.

7. References

[1] Howard James Bampton, "Solving imperfect information games using the Monte Carlo heuristic [Master thesis]", University of Tennessee, Knoxville, 1994.

[2] Denis Papp, "Dealing with imperfect information in Poker", MSC. Thesis, University of Alberta. 1998:1~2.

[3] B. Bouzy B. Bouzy, "Associating domain-dependent knowledge and monte carlo approaches within a go program", Information Sciences, Heuristic Search and Computer Game Playing IV,Edited by K. Chen, (4):247–257, 2005.

[4] Levente Kocsis and Csaba Szepesvari, "Bandit based Monte-Carlo Planning", *15th European Conference on Machine Learning (ECML)*, pp.282-293, 2006.

[5] Sylvian Gelly, Yizao Wang, Remi Munos and Olivier Teytaud, "Modification of

UCT with patterns in Monte-Carlo Go", University of South Paris and Paris institute of technology,2006.
[6] http://sports.sina.com.cn/go/2007-08-29/11433136461.shtml.
[7] P. Auer, N.Cesa Bianchi and P.Fischer, "Finite time analysis of the multiarmed bandit problem", *Machine Learning*, 47(2/3), pp. 235-256,2002.
[8] http://www.tencent.com/.
[9] Sylvain Gelly and David Silver, "Combining online and offline knowledge in UCT", Univ. Paris Sud, LRI, CNRS, INRIA, France, University of Alberta, Edmonton, Alberta, 2006.
[10] Xuan Wang and Zhaoyang Xu, "TD(Λ) optimization of imperfect information game's evaluation function", Intelligence Computing Research Center Harbin Institute of Technology Shenzhen Graduate School, unpublished.