Computer Games Are an Efficient Tool for Event Game Theory

XU Xinhe, Wang Hao, XU Changming

***Abstract*— The game theory includes the static and dynamic one. In the dynamic game theory, the continuous game means the game variables (decisions) of game model are continuous. The differential game theory is very useful and important to the continuous game and has made significant theoretic and application results. For the discrete game variables of dynamic games, the game is defined as event game by Prof. XU Xinhe in reference [1] since the systems belong to discrete event dynamic system (DEDS). The basic model frame is also given in the reference for the perfect information event game system. Because of the huge game tree model, the analytic solution cannot be deducted and the various search solution could be used. Therefore the computer game theory and algorithms becomes an efficient tool for event game theory. This paper give the model of event game theory first, then analyzes problems and difficulties to the model solution. The fundamentals and methodologies of computer game are summarized and efficient tool to the event game could be detected. How to develop the event theory and tools, some application problems are also introduced last.**

Key words: computer games; discrete event dynamic system; event game theory, game tree model, search algorisms

I. Introduction

The game problems are everywhere. The small thing like children playing game and the huge thing like diplomatic affairs, not only the economic, political, military problems, but also the daily life is connected with game problems. But game theory was established only for about one hundred year [2].

Game system can be divided into static and dynamic system. The static one is synchronous and one-off. Such as Rock-Paper-Scissors, Prisoner's Dilemma, Boxed Pigs, bid and bidding. And the dynamic one is nonsynchronous and by multi turns.

By the point view of system theory, the dynamic system whether the system states are vary with respect to time can be divided into Continuous Variable Dynamic System (CVDS) and Discrete Event Dynamic System (DEDS) and the combination of both systems is Hybrid Dynamic System (HDS). In the CVDS the state is driven by time and the mathematical model is differential equations or difference equations which cannot model the DEDS, since the state is discrete and the state evolution is driven by event. Time can be used to describe the evolution, but not the driven factor. [3]

It is easy to see, for the dynamic game system the decision variables also can be divided into continuous and discrete one [1]. The foregut can be called as continuous game system and the latter one as event game system. The decision variable of foregut is continuous and latter one is discrete and state is driven by event.

The continuous dynamic game system can be described by differential equations or difference equations. As about this kind of system, American mathematician R. Isaacs and Scientist Avner Friedman put forward the differential game theory which provides the system solution [4]. A series of theoretic and application achievements have been made.

For the event game system, such as chess, the efficient modeling method is used to game tree. In general speaking, the tree scale is very huge and no analytic method to solve it and there are not many references on it. Therefore the paper tries to find the main way and method to solve it based on the summation of the achievement of computer games.

For computer games the main solving methods of game tree are searching, various searching algorithms, including Mini-max search, Alpha-Beta search, Iterative Deepening Search, Monte-Carlo Tree Search, Threat Space Search, Proof Number Search and so on. Aim at the concrete problem, a series efficient algorithms have been proposed by combination and extension of the above algorithms, and made very significant achievements in many board games. Searching algorithms become the main backbone of computer game methodology. Therefore it is not difficult to clear that computer games could be the efficient tool for event game theory.

There 7 sessions in this paper which introduces respectively typical background and model frame of event game system, analysis the difficulties of the model solving, summarizing the achievement of fundamental principle and methodology of computer games, argumentation of solving way and main methods, indicating computer games are an efficient tool of event game theory, as well as pointing out the facing problems solving event game system.

II. Typical Background of Event Game

For control system, various dynamic systems are

Manuscript received on October 17, 2010.

This work was supported by National Nature Science Foundation of China under Grant 60774097

XU Xinhe is with the school of information science and engineering, Northeastern University, Shenyang, CO 110819, China (corresponding author, fax: +86-024-23899982; phone: +86-024-83683989; e-mail: xuxinhe@ise.neu.edu.cn).

WANG Hao, was with the school of information science and engineering, Northeastern University, Shenyang, CO 110819, China (e-mail: leo_waiy@163.com).

XU Changming is with the department of electronics and information, Northeastern University at Qinhuangdao, CO 066004, China (e-mail: changmingxu@gmail.com).

distinguished by the state variable. If the state variable is continuous the system is Continuous Variable Dynamic System or CVDS in short. Common nature systems such as physical variable (velocity, force, energy, temperature…) system, chemical variable (PH value, concentration…) and common engineering systems belong to CVDS. If the state variable is discrete the system is Discrete Event Dynamic System or DEDS in short. A majority of artificial system, such as serving system, transportation system, manufacturing system etc. possess DEDS characters. Many decision- making systems related with social problems should belong to DEDS.

CVDS and DEDS are quite different kinds of system. The foregut could be modeled by differential or difference equation, but the latter one could not. DEDS is driven by event, in which the essentially discrete states and events act and react each other in certain rule, resulting in the evolution of system state [3]. This kind of system can be modeled with Petri Net, Automata, event graph, Mini-max algebra and so on.

It is deferent from the control system and common decision-making system, game system is also a decision-making system, but the agents who make decision are not only 1, but 2 or more. In this kind of game system, every player chooses the rational action and consequence of strategy in the benefit restriction between people. The agent chooses the optimal sequence of strategy to maximum his (her) benefit.

If every agent (player) chooses the decision variable in a finite strategy set, obviously this system has the characters of discrete event. If this game progresses consecutive, the system is a dynamic game system and belongs to an event game system.

There are many examples of event game system. Such as debate system, diplomatic system and so on. The typical system of event game is the card and board games. From the point of view of game theory, card and board games are divided by two kinds. The first one is the games of complete information like chess, I-go etc. in which each player has complete information about the strategies of opponent and himself on and before, and they can also deduce the strategies the opponent will act. The second one is the games of incomplete information like card and mah-jong. The solution of these kinds of games has osculation with probability, but the first one not.

Something should be pointed almost all the games are dynamic games except some games by single player. And the players act by turn. All the games with two players are zero-sum game [5].

Card and board games are always consist of two parts, position and moves. Moves are the basic factor of system evolution. The transform of position are all occurred on the discrete time. The position can be transferred only when the moves are happening. Otherwise the position holds the line, and they have the inherence property of discontinuous. Both of move set and position set are discrete. According to the analysis upwards, we think the card and board game systems are the DEDS. Moves are the events making the system evolvement.

War system is another event game system. Of cause it is a more complicated one since chess and I-Go is the abstraction of war. Now that the chess and I-Go belongs event game system, war system obviously has the characters of event game system.

III. Model Frame of Event Game Theory

Event game system is a kind of discrete dynamic decision-making process, where there are two non-cooperative intelligent agents. It has following features.

In one point, the state of event game system is changing on the discrete time and driven by event. The other time it will hold the line. Additionally, event is the main body of event game system, and the event set constitutes of decision-making processes restricted each other by (two) players. In addition, the analysis aim of event game system is to find out the best strategy (action) of every intelligent agent in the situation of restricted strategies. So that, driven by the series of events, the system is able to bring the most benefits, or most avoid the loss, to every intelligent agent [1].

Based on these characters, An event game system can be defined as seven-tuple [6]:

$$E = (P, Q, \Sigma, R, \delta, q_0, F) \tag{3-1}$$

Where

1. P: player $P = \{P_1, P_2\}$. Both of them are intelligent and active decision-maker. They will choose their actions according to the state of the system to gain the maximum payoffs of themselves;

2. Q: limited state set of game system;

3. Σ : The strategy set composed by the limited strategies (actions) subset of both sides. $\Sigma = \{\Sigma_1, \Sigma_2\}$. It is the function of the state Q.

4. R: Game rules including the action playing sequence, the style and content of the information released etc.

5. δ : State transfer function. It defines the state evolution rule in the effect of participant's moves. So $\delta : Q \times \Sigma \to Q$;

6. q_0: The initial state of the system. And $q_0 \in Q$;

7. F: The payoffs of the players. It connects with the current state q_k of the system. So $F(q_k) = \{F_1(q_k), F_2(q_k))\}$.

IV. Analyzing the Difficulties of Event Game Model Solving

As the representative of board game, Chinese chess belongs to DEDS. It is not hard to give the dynamic model of Chinese chess according to Event Game Theory [7]. Where move is strategy of player. Even though every turn only one move is taken, player should consider all valuable moves and forward extend several steps. Game player must extend a massive game tree with root up and leaves below, as shown in Figure I. Compared with normal decision-making tree, it has not only one main decision-making agent, but two opposite sides, red and black. In the figure I, quadrate node represents the position of red's turn; round node represents the position of black's turn. Obviously, the lines between quadrate and round node represent moves. On condition that one excellent player can look forward 4 steps, which means he can extend the game tree 4 depth in his mind.

Chinese chess game tree is inconceivable huge. If each step has an average of 45 possible moves, each game has an average of 90 steps, so $45^{90} \approx 10^{150}$ positions should be considered from the initial state to the end of the game. Reputedly, the astronomical number is bigger than the number of atoms on the earth. Until the destruction of the Earth, the first step's moves can not be calculated althouth using the fastest computer in the world.

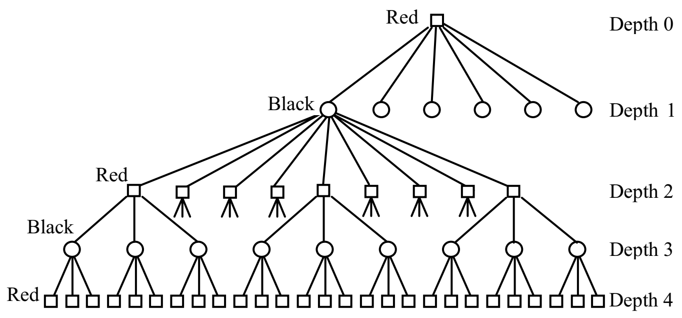

Fig. I depths game tree extended by red side

Inconceivable huge tree of Chinese chess represents the solving difficulty of event game model (game tree) -- dimension disaster and NP hard. Especially in the huge strategy set and large game steps we must investigates proper algorithms in order to achieve ideal results.

V. Principle and Methodology of Computer Games

When playing Chess, both sides generate moves in turn to push the game toward their own favorite and the ultimate winning. Thus, the core element of Chess game is the root move generation, which is a sophisticated mental process. Put it in a simple way, this process is to evolve the chess states by moves and choose current move from positive states. Obviously, the ability to correctly evaluate board state evolved demonstrates the level of the player.

The typical architecture of game-playing software is illustrated in Figure Ⅱ. The figure only shows the relationship between functional units, while in practice computer game program often uses recursion and iterative algorithm to unfold, evaluate and search the game tree.

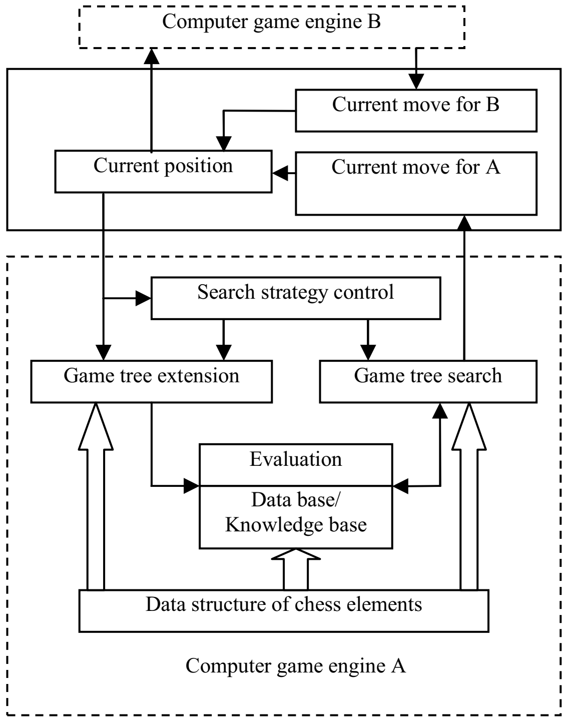

Fig. Ⅱ Software architecture of compute games

To evolve the chess state using moves is to unfold the game tree. It is closely related to core data structure. So, searching algorithm to choose current move from positive state is the key of game software.

Searching is a rather broad concept as well as a widely applicable technique. In many planning and optimization problem such as Linear Programming (LP), Non-linear Programming and Integer programming, Travel Salesman Problem (TSP) and scheduling problem, we tend to use many heuristic searching algorithms if the analytical solution (even the numerical solution) is not available. Examples include Hill-climbing, branch and bound, Tabu Search, Simulated Annealing, Genetic Algorithm and Ant Colony optimization. Although possible searching algorithms are diverse, they have some following common points:

(1). Single decision agent, i.e. unified performance evaluation function.

(2). Explicit objective function and constraints that often could be described by mathematical model.

(3). Most problems are static programming and optimization problem, i.e. single step decision problem.

Computer games belong to the domain of game theory and strategy planning. The problem can be modeled by a strictly competitive, multi-step and dynamic two-person, zero-sum games whose objective function is difficult to describe in a mathematical model. The goal of game tree searching algorithm is to search for optimal routes and currently optimal move (root move) and evolve step by step.

The basic searching algorithms of computer games include the following [8]:

1) Mini-Max Search

Wise enough and desiring victory, players of both sides choose different standard in different level in the game tree (term "Ply"). For nodes on the even Ply, the desirable moves are to find a node with highest evaluation value in all its child nodes, i.e. to perform "MAX search". On the contrary, odd-Ply nodes find the minimum value and perform "Min search". Given evaluation values for the leaf node, the algorithm searches adversely in the bottom-up fashion until it reach the root node. By remembering the child nodes of the evaluation value, we could find a path from the root to the leaf node which is the optimal path as well as the best sequences of moves of both sides. Regularly, it is called Principal Variation (PV) and its root move is what should be currently selected.

2) Alpha-Beta search

Alpha-Beta search is based on depth-first searching algorithm. To reduce the complexity of Mini-Max search, Alpha-Beta pruning technique is commonly used to greatly improve the efficiency. Alpha and Beta values represent for the lower and upper bound on best moves, respectively. The Alpha-Beta value therefore defines the value range of moves and is also called the Alpha-Beta window.

3) Nega-Max Algorithm

It is mentioned earlier that the game tree searching is a kind of "alternative" search. It brings about many troubles to the implementation to conduct different searches in odd and even Plies.

Knuth and Moore fully utilized the inherent nature of "alternative" search and introduced an important Nega-Max Algorithm in 1975 [9]. The main improvement is that the value of parent node is the negative maximum of all the child nodes so that the awkward previous implementation is avoided.

4) Iterative deepening search [9]

Having delved into the α-βgame tree pruning algorithm, we will soon discover that the pruning result is closely related to the order of moves. If we could unfold the search tree along the optimal path, we are definite to get a minimum game tree. However, it is clearly impossible with the objective of searching the optimal path. On the other hand, it is not to imagine that PVD-1 with depth D-1 has the maximum likelihood to be the PV of the game tree with depth D. As a result, people proposed an Iterative Deepening algorithm. An iterative series of 1-ply, 2-ply, 3-ply up to D-1 ply is searched to find PV1, PV2 , PV3 up to PVD-1. Finally we search the game tree of D Ply. At first glance, it seems that the iterative deepening increases the number of searches and waste some time. It then turns out that the effective pruning could greatly reduce the branch factors and makes the search very efficient.

5) Alpha-Beta search for ordering the moves

From basic Alpha-Beta searching algorithm and the minimal game tree analysis, we can easily learn that move ordering is very important in affecting the efficiency of Alpha-Beta pruning. Using some heuristics, we could gain double performance to order the good moves at each node for sooner extension and search. Table I shows several move heuristics that are easy to implement while improve the performance a lot. The advantage is they require no specific domain knowledge and so it is universally applicable to various chesses.

TABLE I
TRANSPOSITION TABLE, HISTORY TABLE AND KILLER MOVE

name	index	content	function	characters	frequency
Transposition Table	hash key of position	best move, evaluation, etc.	avoid researching, heuristic	used at any time	reset in each game
History Table	move	score of move	move ordering	used in different layers	reset after a move
Killer Heuristic	move	move caused pruning	move ordering	used in the same layers	**

6) Window-based Alpha-Beta search

Whenever the initial window of α-β is (-∞, +∞), the pruning for α-β algorithm is safe as it is equivalent to Mini-Max search. As the window grows smaller, the possibility of the node's evaluation value fallen outside the window is higher and the pruning occurs more frequently. If the initial window is small, the pruning rate is higher. However, we get other risk to narrow the window that if the true value falls outside the window, the search is doomed. Based on those features, people proposed yet another series of α-β search algorithm including Fail-soft Alpha-Beta, Aspiration Search, Principal Variation Search (PVS), Minimal Window search, Zero-Window search, MTD(f), etc. Obviously, Zero-width window search is the most efficient but it is used only to confirm the optimality of one certain move as a dual problem has to be answered.

7）Heuristics

Heuristic search uses application-dependant knowledge and information.

To avoid the horizon effect caused by even searching depth, quiescent search is proposed to further searching the deeper Plies if these positions change dramatically. More specifically, the method searches further the child node until all positions are fixed or "dead".

Search extensions, for some important moves, the fixed depth searching strategy could not lead to a robust result. We need to search deeper in addition to the left depth and it fixes the defects of horizon effect.

Null move: the so-called Null-move is to make a passing. In common, to make a move is better than just passing. If the return value of passing is higher than the upper bound of window we could prune this sub-tree immediately. Null move proves to be effective in practice but it is also a double-edge sword in that the entrance condition must be carefully checked.

8) Monte-Carlo based game tree search [10]

Monte-Carlo Tree Search (MCTS) is a kind of Best-first search algorithm and is more suitable for game tree with big branching factor.

Monte-Carlo simulation is to start from one position and generate moves randomly. There is a metaphor that let two fools play chess without violating the rules and surely without any strategies. As a result, there will be a winner and if we let millions of fools play this chess, then statistically we could trust the moves with the highest winning possibility.

The Monte-Carlo tree search is to extend the game tree by iteration. On one hand, we have to use domain knowledge to give more chances to nodes with high winning rate. While on the other hand, we also have to explore those sibling nodes with lower winning rate and access rate. This trade-off between exploitation and exploration shows itself in the policy of moves selection function of UCT algorithm. The basic four stages of the algorithm include selection, expansion, simulation and back-propagation. [11]

One valiant advantage of UCT search is that the search could end at any time and return a result, which means that at any moment UCT gets a relatively optimal result.

9）TSS- Threat Space Search [12]

Threat is defined as a board state that one could win by one move. The threaten moves are the moves that could lead to a threat condition. If one continuously generates threaten moves to push the opponent to defense passively, the moves will be ideal. In the game of Connect6, the threaten-based moves strategy is commonly adopted. Because of the big branching factor of the Connect6, normal α-β algorithm could only search fairly shadow Plies of the game tree. In contrast, TSS strategy could rapidly search tens of Plies and is able to discover solution deep inside the tree.

10）PN- Proof Number Search [13]

PN search is proposed against a kind of AO-Tree. The AO-tree is a binary tree, i.e. the evaluation value of nodes range from True or False. In the game theory, the problem we consider is whether we could win or not, that is the True value of one Boolean condition.

PN algorithm select one optimal moves from several applicable moves, so it is also called Best-first search. The foothold point is how to search deeper along a rather sparse branch in order to reach the find result. So, two variables are added to the nodes:

PN searching algorithm then computes the cost of search according to the value of pn and dn and choose the orientation of extension to reach deeper Plies.

PN search is especially applicable to the final stage of chess as well as those TSS-strategy chess. When solving for some theoretical value of game trees, PN search performs better then α-β search. PN search will use history heuristic to decide which nodes to extend and make the search go to the direction of lower cost. In addition, the game tree of PN is sparer than Alpha-Beta's and it holds no reliance to specific domain knowledge. Application has been seen in Awari, Chess, Checkers, Go, etc.

Above all, no matter what searching algorithm is, it has clear logic and no complicated computation. Given the powerful performance of recursion and iteration, the program of every kind of searching algorithm is simple. However, the specific search plan is impossible without hard work. Each step of the scheme is related to the move generation and nodes evaluation, which is tied with different rules of chesses. As a result, we have to consider thoroughly the detail of software, as is shown in Figure II.

VI. Solving the Event Game System

Through tough research of more than half century, computer games have been made very significant achievement. In the face of various inconceivable huge tree models, most of the computer programs won the matches against world champion of many board games. It means the fundamental principle and methodology of computer game is the efficient tool to solve the board games and suitable to the model frame of event game theory. So it is an efficient tool to solve the problems of event game theory.

We must know that the practical event game system is more complicated than board game system. For example, the war simulation in which there are many synchronous events, each move does not only related one fight units, but various combination of fight units. So the branch of game tree enlarges one or two rank of power. There is no a certain capture rule of fight and the result of fight possesses uncertain and stochastic character. The two players do not give moves by turn and move number of players is not symmetric. In a word, practical game tree is very complicated. Under this kind of situation analytic method is impossible and searching method is only way to solve the problem which important factors are to master more professional and expert knowledge, combining the practical circumstances, selecting and applying proper algorithms.

Summing up, more problems surmounting board games in solving event game systems are as follows:

1) Extra large state space and strategy space

The move of Chinese chess includes 4 parts: piece moved, from, to, piece captured. In war system fight unit is not only one, positions of departure and arrival are multifarious and various, fight results are uncertain and stochastic. So the branch of game tree is very big.

2) Considering various impersonality circumstances

In war system the transportation ability and cooperation of fight units must be considered. In diplomatic activities the international environment and condition should be involved.

3) Events firing synchronously

Not only the events of one player, but also the events of both sides are firing synchronously.

4) No symmetry of supplying strategy by two players

It is possible that one player supplies strategies continuously, but not by turn according to the rule of board game.

5) Randomicity of payoff function

Win-loss-draw rule of war system possesses randomicity. So in war simulation uses the dice to represents the stochastic property.

6) Formulization, modelization and modulization

In many circumstances the strategy set is hard to expresses in formulization. For example, the talking points of argumentation, results of activity and passivity, and very hard to find a number for quantification.

7) Consider two sides' problems of macro and micro tactic

For example, if there many bottle fronts, the selection of bottle front is the first important. So the problems become more complicated.

8) Knowledge base and inference engine

Using the opening book and endgames of board game is far not enough for war system. More cases, experiences and lectures, large scale knowledge base and inference engine are needed.

9) Visualization of game result

Existing search strategy has close relation with evaluation, also with data and logical value. Therefore the quantification and the comparability of each result needs to consider.

VII. Conclusion

It should see that the related event game theory's research is preliminary. And it merely sums up a kind of system as the discrete event dynamic system, has given the formalized description, pointed out the solution direction. But this also has many insufficiencies. For example, it has not considered the multi-person game question, not included non-complete information situation. Therefore it has the very big expansion space. The key lies in first needs to break through the complete information definite game question.

It also has many kinds of models and the solution methods regarding the discrete event dynamic system. Regarding the event game system it is quite suitable to be the model and method of finite-state machine, certainly also needs the further expansion. Because the existing finite-state machine model faces the sole decision-making agent, does not consider the decision-making mechanism of non-definite question. However the event game system's existence is objective, this kind of question's solution needs the theory and the methodology achievement. This article has given the foundation and the direction of further studies. May expect that in the near future a big progress will be made.

References

[1] XU Xinhe, ZHENG Xinying. "Card & Board Games and Event Game Theory". Control and Decision, Vol.22, No.7: pp. 787-790, Jul 2007.

[2] XU Xinhe, WANG Yan, LIU Jihong, ZHANG Xuefeng. "Analysis on the Achievement Milestones and Limitations of Game Theory". *In Proceedings of Chinese Control and Decision Conference*, Jul 2-4. 2008, Yantai China.

[3] ZHENG Dazhong, ZHAO Qianchuan. *Discrete Event Dynamic Systems*. Beijing: Tsinghua University Press, 2001.

[4] ZHANG Siying. *Differential Game*. Beijing: Science Press, 1987

[5] LI Guang-jiu. *The Foundational Lectures of Game Theory*. Beijing: Chemical Industry Press, 2005.

[6] Christos G. Cassandras, Stéphane Lafortune. *Introduction to Discrete Event Systems (Second Edition)*. Springer Science+Business Media, LLC, New York, NY 10013, USA, 2008.

[7] XU Xinhe, WANG Jiao. "Key Technologies Analysis of Chinese Chess Computer Game". Mini-Micro Systems, 2006,27(6): 961-969

[8] XU Xinhe, XU Changming. "Summarization of Fundamental and Methodology of Computer Games". Progress of Artificial Intelligence in China (2009), pp. 748-759.

[9] R E Korf, "Depth-first iterative-deepening: An optimal admissible tree search". Artificial Intelligence, 1985, vol. 27, no.1, pp. 97-109.

[10] Bernd Brugmann, Fohringer Ring 6. Monte Carlo Go [Online]. Available: http://www.ideanest.com/vegos/ MonteCarloGo.pdf

[11] Guillaume Chaslot, Sander Bakkes, Istvan Szita and Pieter Spronck. "Monte-Carlo tree search: A new framework for game AI", i*n Proc. 4th Artificial Intelligence and Interactive Digital Entertainment Conf.*

[12] J. Pearl, "Asymptotic Properties of Mini-Max Trees and Game Searching Procedures", Artificial Intelligence, 1980, vol. 14, no. 2, pp. 113-138.

[13] Allis L.V, Meulen M. van der, Herik, H.J. van den. "Proof-Number Search". Artificial Intelligence, 1994, vol. 66, no.1, pp. 91-124.