Monte-Carlo Tree Search: A New Framework for Game AI

Guillaume Chaslot, Sander Bakkes, Istvan Szita and Pieter Spronck*
Universiteit Maastricht / MICC
P.O. Box 616, NL-6200 MD Maastricht, The Netherlands
{g.chaslot, s.bakkes, i.szita, p.spronck}@micc.unimaas.nl

Abstract

In this paper, we put forward Monte-Carlo Tree Search as a novel, unified framework to game AI, which doesn't require an evaluation function. In the framework, randomized explorations of the search space are used to predict the most promising game actions. We will demonstrate that Monte-Carlo Tree Search can be applied effectively to (1) classic board-games, (2) modern board-games, and (3) video games.

1 Introduction

When implementing AI for computer games, the most important factor is the evaluation function that estimates the quality of a game state. However, building an *adequate* evaluation function based on heuristic knowledge for a non-terminal game state is a domain-dependent and complex task. It probably is one of the main reasons why game AI in complex game-environments did not achieve a strong level, despite intensive research and additional use of knowledge-based methods. Monte-Carlo Tree Search (MCTS), a Monte-Carlo based technique that was first established in 2006, is implemented in top-rated GO programs. These programs defeated for the first time professional GO players on the 9×9 board. The technique can be generalized easily to modern board-games or video games. In the proposed demonstration, we will illustrate that MCTS can be applied effectively to (1) classic board-games (such as GO), (2) modern board-games (such as SETTLERS OF CATAN), and (3) video games (such as the SPRING RTS game).

2 Monte-Carlo Tree Search

Monte-Carlo Tree Search (MCTS) is a best-first search technique which uses stochastic simulations. MCTS can be applied to any game of finite length. Its basis is the simulation of games where both the AI-controlled player and its opponents play random moves, or, better, pseudo-random moves. From a single random game (where every player selects his actions randomly), very little can be learnt. But from simulating a multitude of random games, a good strategy can be inferred. The algorithm builds and uses a tree of possible future game states, according to a four-step mechanism. First, *selection*: while the state is found in the tree, the next action is chosen according to the statistics stored, in a way that balances between exploitation and exploration. On the one hand, the task is often to select the game action that leads to the best results so far (exploitation). On the other hand, less promising actions still have to be explored, due to the uncertainty of the evaluation (exploration). Several effective strategies can be found in Chaslot et al. [2] and Kocsis and Szepesvári [3]. Second, *expansion*: when the game reaches the first state that cannot be found in the tree, the state is added as a new node. This way, the tree is expanded by one node for each simulated game. Third, *simulation*: for the rest of the game, actions are selected at random until the end of the game. Naturally, the adequate weighting of action selection probabilities has a significant effect on the level of play. If all legal actions are selected with equal probability, then the strategy played is often weak, and the level of the Monte-Carlo program is suboptimal. We can use heuristic knowledge to give larger weights to actions that look more promising. Fourth, *backpropagation*: after reaching the end of the simulated game, we update each tree node that was traversed during that game. The visit counts are increased and the win/loss ratio is modified according to the outcome.

*This research was funded by NWO grant No 612.066.406 and grant No 612.006.409.

3 Applications

Classic Board-Games such as two-player deterministic games with perfect information, have been submitted to intensive AI researched. Using the alpha-beta framework, excellent results have been achieved in the game of CHESS and CHECKERS. However, alpha-beta only works well under two conditions: (1) an adequate evaluation function exists, and (2) the game has a low branching factor. These two conditions are lacking in numerous classical board-games (such as GO), modern board-games and video games. As an alternative to alpha-beta, researchers opted the use of MCTS. It has been shown that MCTS is able to use highly randomised and weakly simulated games in order to build the most powerful GO-programs to date. In our demonstration, we will present our program MANGO, which is a top-rated GO program. We will use graphical tools to demonstrate how MANGO focuses its search on the best moves. We will emphasise that MCTS without any expert knowledge can still achieve a reasonable level of play.

Modern board-games are becoming more and more popular since their (re)birth in the 1990's. The game SETTLERS OF CATAN can be considered an archetypical member of the genre. Modern board-games are of particular interest to AI researchers because they provide a direct link between classic (two-player, perfect information) board-games and video games. On the one hand, state variables of most modern board-games are discrete, and decision making is turn-based. On the other hand, the gameplay in modern board-games often incorporates randomness, hidden information, multiple players, and a variable initial setup that makes it impossible to use opening books. In our demonstration, we will show that MCTS outperforms previous heuristic game AI's in SETTLERS OF CATAN, and provides a challenging opponent for humans.

Video games present a complex and realistic environment in which game AI is expected to behave realistically. When implementing AI in video games, arguably the most important factor is the evaluation function that rates the quality of newly generated game AI. Due to the complex nature of video games, the determination of an adequate evaluation function is often a difficult task. Still, experiments performed in the SPRING RTS game have shown that is is possible to generate an evaluation function that rates the quality of game AI accurately before half of the game is played [1]. However, it is desirable that accurate ratings are established even more early, when adaptations to game AI can influence the outcome of a game more effectively. Monte-Carlo simulations provide a powerful means to accurately rate the quality of newly generated game AI, even early in the game. In our demonstration, we will show how we abstract the SPRING RTS game for use of MCTS simulation. The abstraction contains, among others, the position of each unit in the game, and the game strategy employed by all players. We will emphasis that in complex video-games, effective game AI may be established by using MCTS, even with highly randomised and weakly simulated games.

4 Conclusions

In this abstract, we put forward Monte-Carlo Tree Search (MCTS) as a novel, unified framework to game AI. In the framework, randomized explorations of the search space are used to predict the most promising game actions. We state that MCTS is able to use highly randomised and simulated games in order to established effective game AI. In demonstrations, we will show that MCTS can be applied effectively to (1) classic board-games, (2) modern board-games, and (3) video games.

References

[1] S. Bakkes and P. Spronck. *AI Game Programming Wisdom 4*, chapter Automatically Generating Score Functions for Strategy Games, pages 647–658. Charles River Media, Hingham, MA., U.S.A., 2008.

[2] G.-B. Chaslot, J.-T. Saito, B. Bouzy, J. Uiterwijk, and H. van den Herik. Monte-Carlo Strategies for Computer Go. In *Proceedings of the 18th Belgian-Dutch Conference on Artificial Intelligence*, pages 83–90, 2006.

[3] L. Kocsis and C. Szepesvári. Bandit Based Monte-Carlo Planning. In *Machine Learning: ECML 2006, Lecture Notes in Artificial Intelligence 4212*, pages 282–293, 2006.