Research of Game Tree Searching by Min/Max Approximation article

A paper talks about new methods which could help to improve the search of game trees. Minimax and Alpha-Beta pruning are quite usual algorithms used to search game trees, however they are not efficient enough for more complicated games. This article introduces another method - Min/Max Approximation with generalized mean-value operators which is more efficient for complex games tree search.

The Method

Min/Max Approximation with generalized mean-value operators is not only a good approximation of Min/Max functions but it also allows to expand and focus on those nodes which have the biggest influence on the value. The main advantage of Min/Max Approximation is that by calculating derivatives for every node we identify which leafs are going to be expanded. Min/Max Approximation uses an iterative heuristics which works as follows: at each step a tip node or leaf of the current tree is chosen, and the successors of that tip node are added to the tree. Then the values provided but the static evaluator at the new leaves are used to provide new backed-up values to the leave's ancestors.
Exploration of partial trees starts with initialization of a partial tree as well as the value of partial tree. Then, while partial tree is not the same as entire tree and as long as there is some time left for the search, we pick expandable tip of the tree, expand a subtree at that tip and update the value and ancestors. The tip to expand is chosen using penalty based search. The main idea of penalty search is to assign penalty (weights) to every edge in the game tree such that edges representing bad moves would be penalized more than those representing good moves. Then, the penalty of the tip is calculated as the sum of penalties in that subtree. In min/max approximation penalties are calculated using derivatives but the main idea is the same as for penalty search– we are looking for the tip having the lowest penalty.

Method application and results

Min/Max Approximation with generalized mean-value operators was tested against other quite usual technique – Minimax with Alpha-Beta pruning. Both techniques were applied on the game called Connect-Four. Both techniques were tested using different starting positions (seven different positions in total). The results are quite interesting: when the time was used as a limiting factor, Minimax with alpha beta pruning outperformed Min/Max approximation. However, when a move bound was in effect, Min/Max approximation outperformed Minimax. Another interesting thing is that the rate in which Minimax alpha beta pruning called the move factor was more than three times higher than Min/Max heuristic. The conclusion of this experiment is that when special-purpose hardware is used or when move operator is expensive to implement, the move-based comparison would be more relevant to use.

Future improvements

However, even if the results of Min/Max Approximation look very promising, there still is a lot of room for improvements. In some situations, this method has some disadvantages against other algorithms. For example, it requires more memory than depth-first methods, because for Min/Max Approximation to work the tree must be completely stored. What is more, it takes more time and is

less efficient as it takes a lot of time travelling between the root and the leaves while depth first algorithms spend the most of the time near the leaves. Penalty based algorithms also spend some time evaluating non-optimal nodes which makes them less efficient as well.