Research review

Paper: Game Tree Searching by Min / Max Approximation

When performing a tree search using the *minimax* algorithm with *Alpha-Beta pruning* the amount of nodes being pruned depend on the order in which the nodes are visited. Usually the selection of the next child node to explore is done in an arbitrary way and there is no guarantee that the selected node is "the node that is expected to have largest effect on the value".

The key idea presented in the paper is to use the generalized *p-mean* functions, defined as

$$M_p(a) = \left(\frac{1}{n}\sum_{i=1}^{n} a_i^p\right)^{1/p}$$

to approximate the "min" and "max" operators of *minimax* by using the facts that

$$\lim_{p\to\infty} M_p(a) = \max(a_1, ..., a_n)$$

and

$$\lim_{p\to-\infty} M_p(a) = \min(a_1, ..., a_n)$$

This allows to identify "in an interesting way that leaf in a game tree upon whose value the value at the root depends most strongly".

The algorithm presented is an example of a penalty based iterative search heuristic method. This means that a non-negative value, called penalty, is assigned to each node and the search tree is iteratively grown one step a time by expanding the tip node with the least penalty. The estimated value of each node is updated each time the search tree grows by backpropagating the estimated value (using the heuristic function) of the nodes just expanded.

In the algorithm presented the derivatives of the *p-mean* functions are used to calculate the "sensivity" of the root node to changes in the values of each tip node. This sensitivity, in turn, is used to calculate the value of the penalty.

Calculating the generalized p-means is computational difficult beause of the large computational cost involved in taking powers and roots but it may allow an improve in the play because the min/max approximation favors moves whose min/max value can be achieved in several ways over a move whose min/max value can be achieved in only one way.

To avoid the costly calculations, the paper proposes another approach, called the "reverse approximation" idea. Since the main point of using the generalized mean values was for their derivatives, not for the values themselves; and since they are intended to approximate the min/max values, using the appropiate min/max functions may not introduce too much error and reduces the complexity of the calculations.

An implementation of the algorithm using "reverse aproximation" was compared against a straigth forward implementation of *iterative deepening minimax search with alpha-beta pruning* by playing the game *Connect-Four*. Each strategy was allocated a fixed amount of resources to use in computing its move: elapsed CPU time (measured in seconds), and calls to the basic "move" subroutine (measured in thousands of calls).

For each experiment, 49 different starting positions were considered. For each starting position, two games were played--one with alpha-beta (AB) moving first, and one with min/max approximation (MM) moving first. It was recorded how many times each strategy won, and how many ties occured. One experiment was run for each of five posible time bounds (1 second to 5 seconds, in one-second intervals), and for five possible move bounds (1000 moves to 5000 moves, in 1000-move increments). Thus, 490 games were played for each resource bound, and 980 games played altogether.

Based on time usage alone, alpha-beta was reported to be superior to the implementation of the min/max approximation approach. However, when the comparison was done on move-based resource limits min/max approximation was reported as superior.