Game Tree Searching by Min / Max Approximation

The paper introduces a new technique for searching in game trees, based on the idea of approximating the min and max operators with generalized mean-value operators. These are good approximators to the min/max operators, but have continuous derivatives with respect to all arguments. This allows to define the “expandable tip upon whose value the backed-up value at the root most heavily depends” in a non-trivial manner. This tip is the next one to be expanded using that heuristic.

One of the ideas of the paper is that by using generalized mean values to approximate the min and max functions, we can identify that leaf in a game tree upon whose value at the root depends most strongly. This is done by taking derivatives of the generalized mean value functions at each node and using the chain rule. This leaf will be the one to expand next.

Considering there’s a huge game tree the heuristic estimation is needed. The paper uses the concept of iterative heuristics, which “grow” the search tree one step at a time. At each step a tip node (or leaf) of the current tree is chosen, and the successors of that tip are added to the tree. Then the values provided by static evaluator at the new leaves are used to provide new backed-up values to the leaves’ ancestors. Tree grown by an iterative heuristic need not be of uniform depth: some branches may be searched to a much greater depth than other branches.

The experimental results demonstrated in the paper show that the approach can produce play superior to that produced by minimax search with alpha-beta pruning, for the same number of calls to the underlying “move” operator. However, when CPU time rather than calls to the move operator is the limiting resource, minimax search with alpha-beta pruning seems to play better.

The game of Connect-Four was chosen as a basis for experiments. The implementation of the move and unmove operators also implemented the static evaluation function. This static evaluation function was used by all playing strategies, so the differences in playing ability would not be due to differences in static evaluators. By convention, Black is Max and Red is Min. The static evaluator returned integers in range 1 to 1023 where 1 is denoted a win by Red and 1023 by Black. Value 512 denotes a middle or neutral value.

In each strategy a fixed amount of resources was allocated to use in computing its move. Two different resource bounds were used: elapsed CPU time (measured in seconds), and calls to the basic “move” subroutine (measured in thousands of calls).

For each experiment they considered 49 starting positions. Each starting position was defined by specifying the first two moves in the game.

The results show that based on time usage alone, the alpha-beta seems to be superior to the implementation of min/max approximation approach.

However, if base the analysis on the move-based resource limits, the story is reversed: min/max approximation is definitely superior.