Summary: Game Tree Searching by Min/Max Approximation

This paper introduces an iterative method for searching min/max game trees based on approximating the "min" and "max" operators by generalized mean-valued operators. This game tree searching method performed better than minimax search with alpha-beta pruning, when the same number of calls to move routine was used.

The paper discuses one of the major reasons to use generalized means as they are more suitable for a "sensitivity analysis" than the min or max functions. The paper finds an interesting way to search game tree by taking the derivates of the generalized mean value functions at each node and using the chain rule.

The paper also talks about difference between a static evaluation function vs iterative evaluation function. Static evaluation function approximates a heuristic score by estimating "static" features of the current configuration that can be analyzed without further expansion of the search tree. Iterative heuristics grow the search tree one step at a time, and the values provided by the static evaluator at the new leaves are used to provide new backed-up values to the leaves ancestors.

The paper introduces a general method for choosing which node to expand in an iterative method. The method assigns a non-negative "penalty" or weight to every edge in the game tree such that the edges represented bad moves are penalized more than edges representing good ones. The idea is to expand the tip node which the least penalty. Hence, the min/max approximation is a penalty-based scheme. The min/max approximation method is a special case of the penalty-based search method, where the penalties are defined in terms of the derivatives of the approximating functions.

Compared to minimax search with alpha-beta pruning, min/max approximation method is more computationally intensive as we are calculating the generalized p-means. Another approach called "reverse approximation" is an idea where the generalized mean values computation is skipped altogether, and instead use the approximate min or max values.

The performance of the min/max approximation method was tested against minimax with alph-beta pruning method by playing connect-four matches using the two competing methods, and looking at the win, loss, and ties percentages. For each experiment 9 different starting position were considered. Based on the time usage alone, alpha-beta method seemed to be superior than mini/max approximation method. However, if the comparison is based on move-based resource limits, the min/max approximation method did better. For a move bound constraint, both methods considered roughly equal number of distinct position, but the number was higher for alpha-beta method for time bound constraint.

Finally, the paper discusses several general features of penalty-based schemes. First, penalty-based schemes require the tree being explored be stored, hence penalty-based schemes may not perform well in a memory constrained situation. Second, the penalty-based schemes are geared towards improving the value of the estimate at the root, than towards selecting the best move to make from the root. Third, the penalty-based schemes require that a node be expanded by generating and evaluating all of the node's successors. Fourth, the penalty-based schemes may appear not optimal compared to depth-first schemes, as the penalty-based schemes spend large amount of time traversing back and forth between root and leaves of the tree. Lastly, penalty-based schemes spend some time evaluating non-optimal lines of play.