Review of “Game Tree Searching by Min/Max Approximation”

Introduction:

Rivest asserts that game playing capability of computer is still limited by computing powers despite that many brilliant ideas have been introduced to get around this problem, such as Alpha and Beta Pruning. Therefore, Rivest introduces a new method called “Min/Max Approximation” in order to achieve more efficiency in game playing and as a method that “will always expand the node that is expected to have the largest effect on the value.”

Method:

“Min/Max Approximation” is the generalized mean value of possible moves, and is considered more appropriate for “sensitivity analysis” than min or max functions because it always has continuous derivatives to work with. Having continuous derivatives is significant to the idea of “Min/Max Approximation” because by taking derivatives of the generalized mean value functions (as penalties), it is possible to identify which leaf in a tree that the root depends on most strongly. However, in implementation, computing generalized means does face a few challenges including being computationally expensive as a result of taking powers and roots. Rivest proposes two potential solutions. One is to “ use the min/max approximation ~E(s), i.e. selecting the child of the root with maximum ~E(c) instead of the child with maximum bE(c).” The other is to skip computing the generalized mean values, because the having generalized mean values is to get the derivatives but not the values themselves.

Experiment:

Connect Four is chosen for the experiment, and 2 different techniques are tested. 49 different starting positions are used for each experiment, and different time limits and move limits are chosen in each experiment as well. One technique is a minimax search wit Alpha and Beta Pruning, and the second technique is “min/max approximation” with Alpha and Beta Pruning. Rivest implemented “min/max approximation” following the second approaches that he proposes above to address computational limitations. Furthermore, “the penalty on an edge was computed to be 0.05 plus the absolute value of the difference between the natural logarithm of the value of the node and the natural logarithm of the value of his "best" sibling (the one with the best backed-up score. ” Rivest admits that the constant influences the performance largely and “more search is needed to make the computation of penalties more robust”.

Result:

On time-based resource limits, minimax with Alpha and Beta Pruning outperforms the “min/max” approximation approach, but move-based resource limits, the result is quite the opposite. The result is largely because of the higher computational expensiveness of the “min/max” approximation approach. For example, while minmax with Alpha and Beta Pruning called move operator about 3,500 times per second, “min/max approximation” did only 800 times per second. Rivest admits later in the discussion section that penalty-based schemes like “min/max approximation” do seem inefficient and do not perform well unless they are given large enough memory.