The following paper is a summary of the goals, techniques and results of Game Tree Searching by Min / Max Approximation by Ron Rivest. This 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. One of the central ideas of this paper is that by using the generalized mean values to approximate the min and max functions, we can identify in an interesting way that leaf in a game tree upon whose value the 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.

When the tree configuration for the game is small, the tree can be explored completely and thus reach an optimal solution. However most interesting games the game tree is so large that heuristic approximations are needed. We will use a heuristic (aka single static evaluator) to "grow" the search tree one step at a time. At each step a leaf of the current tree is chosen, and the successors of that tip node are added to the tree. Then the values provided by the static evaluator at the new leaves are used to provide new backed-up values to the leaves' ancestors. The 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 "min/max approximation" heuristic is special case of the penalty-based search method, where the penalties are defined in terms of the derivatives of the approximating functions. Although the results of the paper conclude that the approach above is superior to minimax with alpha beta pruning, it does require more CPU time rather than calls to the move operator is the limiting resource in which case minimax search with alpha-beta pruning seems to play better. The implementation of the min/max heuristic worked as follows:

- The 'reverse approximation" was used; the value computed for a node was its true backed-up rain/max value, based on the tree computed so far.
- 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

Experimental results illustrated that with the specified implementation of minimax search with alpha-beta pruning called the move operator approximately 3500 times per second, while our implementation of the min/max heuristic called the move operator approximately 800 times per second. As a result, unlike depth-first search schemes (e.g. minimax search with alpha-beta pruning), penalty-based schemes may not perform well unless they are given a large amount of memory to work with.

The results of the paper indicate that the scheme outlined in the paper outplays alpha-beta with iterative deepening, when both schemes are restricted to the same number of calls to the move operator.