Game Tree Searching by Min/Max Approximation by Ron Rivest

The paper presents a way to identify which node to explore further, so that the time spent evaluating the tree is spent exploring the path that has the most effect on the root node. This is in contrast to the mini-max and alpha-beta pruning with iterative search.

In an iterative search as we had done in our course all the nodes at a certain depth are expanded. The presented method shows a way to attribute penalties to paths so that a path will be continued to be explored as long as it is the path with the least penalty. That means a particular path may be explored to a greater depth on one call to the move method. As the tree is continued to be expanded, the penalties are recalculated which may cause another path to have a lower penalty and at that time that path will be expanded and so on. One great advantage of this method is that it can greatly reduce the horizon effect – the case where there is a possibility that one of the paths could have led to a win faster but it was not favored as it was not explored to the required depth.

The calculation of the weights and penalties remind me of the way error is back propagated in Back Propagation Neural Networks.

The major disadvantage to the method is that the way the weights are calculated involves calculating generalized p-means which has computationally higher overhead. So this method performs better than minimax with alpha-beta for the same number of calls to the move routine (number of nodes explored) but when there is a time limit, the higher computational cost puts it at a disadvantage. The author has experimental results from 1000 games of Connect-Four to back up these claims, which is a very reasonable sample set.

In my personal opinion this approach is more reasonable and akin to how a logical brain works and hopefully if we can develop approximation functions that are reasonable and fast or if the computers get fast enough such method can be used in many real world scenarios.