Research Review

Game Tree Searching by Min / Max Approximation

# Summary of the Papers Goals and Techniques

The goal of this paper is to suggest a method that will always expand the node that is expected to have the largest effect on the game.

To achieve the goal of always expanding the node that is expected to have the largest effect on the game, the papers author suggests creating a static evaluation function that allows for evaluating a node at any depth. The static evaluation function uses the state of a node to produce a heuristic value that guides the search towards branches that are expected to produce the biggest value, without requiring the search reach the bottom of the tree. This heuristic value can offer huge computational savings for games that require large search trees. This heuristic value is then backed up the tree.

This approach also offers the advantage of allowing for iterative deepening search. Since nodes can be evaluated at any depth by the static evaluation function, this approach can return a meaningful heuristic value for any depth. This is a huge advantage in situations where real time responses are required, as it guarantees a decision will be made within the time limit, while allowing it to try to go as deep as possible in the tree to try to make the best choice it can.

# Summary of the Papers Results

While this approach performed better than Minimax search with alpha-beta pruning when number of calls are limited, the author admits that Minimax with alpha-beta seems to perform better than this approach when the limiting factor is CPU time.

The author suggests that the reason for the for this approach losing to Minimax with alpha-beta when CPU time is limited is that it is likely less efficient. Depth-first search algorithms, such as Minimax with alpha-beta will spend most of their time near the leaves, looking for the best result, which becomes much more efficient when paired with pruning. Whereas approaches such as the once presented will spend a lot of time running between the leaves and the root.