Research Review

Game Tree Searching by Min / Max Approximation

Many pruning techniques have been proposed to search game trees efficiently such as the alpha-beta pruning and the use of transposition table. However, these techniques fail to answer the following questions: Given the limited computational resource, which node should be searched first within the sub-trees? A reasonable heuristic is mentioned in the paper published by [Nau et. al 1985]. It suggests that a node with a more significant impact on the values of the concurrent explored sub-tree should be explored with a higher priority.

In this paper, the author proposes a searching algorithm to prioritize the node expansion procedure based on its value impact on the root node. To quantify the impact, the author utilizes $D(x, y)$, the partial derivates of the parent node (x) value with respect to the child node (y) value. A weighting on each edge is calculated based on $-\log D(x, y)$. In this case, a higher weighting value indicates a smaller value of $D(x, y)$. Finally, the penalty from the root to each leaf node of the current explored sub-tree is found by summing over all the edge weights along the path from the root to the leaf. The leaf node with the smallest penalty, i.e. the highest impact to the root node value, is expanded.

One of the advantages of using partial derivates is due to its scalable nature: it can be computed locally between each parent-children pair. Thanks to the chain rule of the derivates, the partial derivates across a path can be computed by using the product of derivates, or the summation of the weighting (due to the log-function), over all the edges along the path. However, it is noted that the min-max function applied is non-differentiable. Therefore, when evaluating the value at each node, the author proposes to use the p-mean value: $M_p(\boldsymbol{a}) = \left(\frac{1}{n}\sum_{i=1}^{n} a_i^p\right)^{1/p}$. The function is differentiable with respect to $\boldsymbol{a}$ and it is also a good approximation of max (min) function if p is selected to be a large positive (small negative) value.

The author uses a connect-four game to evaluate the performance of the proposed algorithm. Two difference resource restrictions are imposed: time and the number of times that the “move” action is called. The minimax search algorithm with alpha-beta pruning is used as an evaluation bench mark. It is observed that when a time limitation is imposed, the minimax search performs better. Within a total of 490 games, the minimax search won the proposed algorithm by 53 games. The reason is because that the proposed algorithm requires additional penalty computation when trying to identify the best node to expand. However, when the number of move calls is limited, the proposed algorithm performs better than the minimax search. Within a total of 490 games, the proposed algorithm won the minimax search by 59 games. As a result, an improvement in the CPU cycle time would essentially make the proposed algorithm a more promising solution.