Udacity – AIND - Advanced Game Playing Research Review

By: Cheryl Miller

"GAME TREE SEARCHING BY MIN / MAX APPROXIMATION"

INTRODUCTION

This research review summarizes the goals and results of the 1988 paper **Game Tree Searching by Min / Max Approximation** by Ronald L. Rivest from MIT. The paper introduces a new (at the time) technique for searching in game trees based on approximating min/max operators with 'generalized mean-value' operations. It compares the performance of this technique against the typical Minimax search with alpha-beta pruning technique.

PAPERS GOALS

The goal of this paper was to develop an alternate technique to Minimax/Alpha-Beta pruning that could decide which leaf node to expand based on the leaf's value having the most impact on the estimated value of the root node using 'Generalized Mean Values' to approximate the min and max functions. They use a 'Penalty Based Heuristic' where penalties on leaf nodes are calculated based on their derivatives and not their values themselves, known as "reverse approximation" using the following equation:

$$w(c) = \log(n) + (p-1) * (\log(\widehat{v_E}(d)) - (\log(\widehat{v_E}(c)))$$

Where:

$\widehat{v_E}$ – is the static evaluation function
c – is the leaf (or tip) node that we are expanding
d -is c's child node
log(n) – was just a constant for this experiment, using a value of 0.05, chosen based on preliminary testing. Performance was sensitive to this constant.
p – is key to calculating the generalized 'p-mean' where large positive or negative values provide a good approximation of max or min values respectively. As p gets large, the heuristic would grow very deep/narrow trees, and corresponds to having a high confidence in the accuracy of values returned. Small p values grow broad trees and a low confidence.

PAPERS RESULTS

For implementing and testing this technique, the author chose the 'Connect-Four' game for the basis of his experiments as it was well-known yet simple and easy to implement. The same static evaluator was used by all game playing strategies so that differences in playing ability would not be due to differences in the static evaluators. He also used resource bounds of elapsed CPU time in seconds and calls to the 'move' function subroutine measured in thousands of calls. The implementation was done in C on a DEC

MicroVax workstation (wow, haven't heard that name in years ☺), with experiments run in parallel on ten workstations.

One experiment was run for each of five possible time bounds (1 – 5 seconds in 1 second intervals), and five possible move function call bounds (1000 – 5000 moves in 1000 move increments). 490 games were played for each resource bound for a total of 980 games altogether.

For the time bounds, the number of distinct positions considered by each strategy was roughly equal. For the move call bounds, the Minimax/Alpha-Beta pruning called the operator about 3500 times per second, while the min/max approximation heuristic called it about 800 times per second. The differences in could be accounted for the inefficiencies of the min/max approximations.

The author found that his technique played superior for the same number of move calls, but the Minimax/Alpha-Beta Pruning played better when CPU was the limiting factor. Keep in mind, this experiment and paper were written almost 30 years ago, ancient history in regards to computing capabilities of today. I assume this is why Minimax/Alpha-Beta pruning are still being taught as a key search algorithm today rather than the author's technique.