Jason Asaytuno
Paper Summary
Game Tree Searching by MinMax Approximation
AIND – Advanced Game Playing

In which we expand nodes that affect the value we most care about, the final returned value.

With a search tree, a decision is needed to decide which node and its corresponding rabbit hole to dive into first. In an adversarial search, a node that is most favorable to the player in play is returned. Each node, representing a state of the game, is given a score of favorability, a high score implying a path to success. In this MinMax Approximation paper by Rivest, Rivest proposes to use a sensitivity indicator that tests which nodes are the most influential in deciding the value that is ultimately backed up and returned by the algorithm.

The technique introduced is using a max() or min() function that places the scores on a spectrum. Instead of a max() function that returns true or false if a number is indeed the maximum among its peers, the proposed new max() function would return a small or big value depending if it is the maximum or close to it.

Concretely, the max function is:

$$M_p([]) = avg([]^p)^{\frac{1}{p}}, \text{ where [] is a list[] of values}$$

Which is a specific form of this averaging function $f^{-1}(avg(f([])))$. Why this function is sensitive is because for $\lim_{p\to\infty} M_p([]) = \max([])$ and $\lim_{p\to-\infty} M_p([]) = \min([])$.

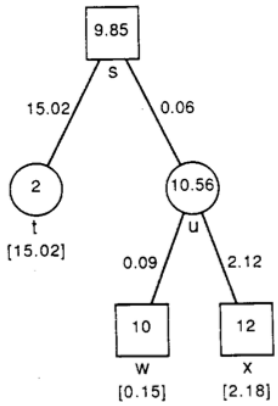

Figure 1. MinMax Approximation

As an example, given a list of sibling nodes with scores [10,21,29,32] and applying the max function with power values of 1, −32, and 32:

$$M_{p=1}([10,21,29,32]) = 23,$$
$$M_{p=-32}([10,21,29,32]) = 10.4,$$
$$M_{p=32}([10,21,29,32]) = 30.7,$$

notice the output is close to the max 32 when p is high and is close to the min 10 when p is low. Given that $M_p([])$ is just a normal function, we can take a partial derivative with respect to an element of the list [] and the derivative is then the sensitivity indicator.

Applying the sensitivity function to the example:

$$\frac{\delta M_{p=32}([10,21,29,32])}{\delta[]} = [2\cdot10^{-16}, 2\cdot10^{-6}, 0.04, 0.80],$$

results in high numbers for $[32,29]$ which is close the max 32; and low numbers for $[10,21]$ which are not close.

This sensitivity indicator is used to assign costs to edges of the graph. The leaf node with least cost from the root is then the node to be expanded.

Tracing Figure 1 in the style of a Minimax algorithm, the leaf nodes w and x are minimized using $M_{p=-10}([10,12]) = 10.56 = u$, the cost between w,u is $-\ln\left(\frac{\delta M_{p=-10}([10,12])}{\delta\, 10}\right) = .09$ using a logarithm to normalize the derivative. From there, nodes t,u are maximized using $M_p([2,10.56])$ and the cost between s,u is calculated. The cost from root node s to w is $.06 + .09 = .15$, which is less then the cost $.06 + 2.12 = 2.18$ to node x, node w is then the node to be expanded.

Rivest performed 1,000 game trials and found it superior to and expanded a two-thirds less nodes then Minimax with alpha-beta pruning when limited by moves. Alpha-Beta search performed better when limited by time. Approximating is processor intensive; Rivest chose to use logarithm lookup tables to reduce some overhead.

Rivest showed a new way of approaching a traditional problem.