Game Tree Searching by Min / Max Approximation*

* Goals or techniques introduced

Goals:

The Goal of this paper used a method, "min/max approximation," attempts to focus the computer's attention on the important lines of play. The key idea is to approximate the "min" and "max" operators with generalized mean-value operators. The combinatorial explosion of possibilities in a game such as chess tax our most powerful computers, and even special-purpose hardware soon reaches its limits the most careful organization and allocation of computational resources is needed to obtain expert-level play.

Techniques introduced:

Techniques such as alpha-beta pruning and its successors have been essential in reducing the computational burden of exploring a game tree. Still, new techniques are needed. Nau et al., after much experimentation with existing methods, assert that "A method is needed which will always expand the node that is expected to have the largest effect on the value." This paper suggests such a method.

* Paper Results :

This paper demonstrating some initial experimental results that "min/max approximation," can produce play superior to that produced by minimax search with alpha-beta pruning, for the same number of calls to the underlying "move" operator. However, when CPU time rather than calls to the move operator is the limiting resource, minimax search with alpha-beta pruning seems to play better.

TABLE 2. Experimental results

Resource bound per turn	MM wins	AB wins	Ties
1 second	41	46	11
2 second	40	42	16
3 seconds	36	44	18
4 seconds	39	52	7
5 seconds	30	55	13
Total	186	239	65
1000 moves	47	35	16
2000 moves	50	35	13
3000 moves	42	47	9
4000 moves	49	42	7
5000 moves	61	31	6
Total	249	190	51

This Paper note that the implementation of minimax search with alpha-beta pruning called the move operator approximately 3500 times per second, while our implementation of the min/max heuristic called the move operator approximately 800 times per second. When special-purpose hardware is used, or when the move

operator is expensive to implement, the move-based comparison would be more relevant. For software implementations more development of the min/max approach is needed to reduce the computational overhead per call to the move operator.