

EEL5840: Elements of Machine Intelligence
SubjuGator

Adversarial Search

- Alpha-Beta Procedure
 - The same kind of savings can be achieved even when none of the positions in the search tree represents a win for either player
 - We perform DFS and whenever a tip node is generated, the static evaluation function is computed with backed-up values

5

EEL5840: Elements of Machine Intelligence
SubjuGator

Adversarial Search

- Alpha-Beta Procedure
 - Reductions in search effort are therefore achieved by keeping track of bounds on backed-up values. In general, as successors of nodes are given backed-up values, the bounds can be revised
 - The alpha value of MAX nodes (including the start node) can never decrease
 - The beta values of MIN nodes can never increase

- Search can be discontinued below any MIN node having a beta value less than or equal to the alpha value of any of its MAX node ancestors. The final backed-up value can be set to this beta
- Search can be discontinued below any MAX node having an alpha value greater than or equal to the beta value of any of its MIN node ancestors. Set the final backed-up value to this alpha.

6

EEL5840: Elements of Machine Intelligence
SubjuGator

Adversarial Search

- Alpha-Beta Procedure
 - During search, alpha and beta values are computed as follows
 - The alpha value of a MAX node is set equal to the current largest final backed-up value of its successors
 - The beta value of a MAX node is set equal to the current smallest final backed-up value of its successors
 - When search is discontinued under rule 1, we say that an alpha-cutoff has occurred. Similarly, when search is discontinued under rule 2, we say that a beta-cutoff has occurred.
 - Employing alpha-beta search always results in finding a move that is as good as the move that would have been found by the simple minimax method searching to the same depth.
 - Alpha-beta finds the same move usually after much less search.

7

EEL5840: Elements of Machine Intelligence
SubjuGator

Adversarial Search

- Alpha-Beta Procedure

$AB(n; \alpha, \beta)$

- If n at depth bound, return $AB(n)$ =static evaluation at n . Otherwise let $n_1, \dots, n_k, \dots, n_b$ be the successors of n in order, set $k \leftarrow 1$, and if n is a MAX node; go to step 2, else go to step 2'
- Set $\alpha \leftarrow \max[\alpha, AB(n_k; \alpha, \beta)]$ 2.' Set $\beta \leftarrow \min[\beta, AB(n_k; \alpha, \beta)]$
- If $\alpha \geq \beta$, return β , else continue 3.' If $\beta \leq \alpha$, return α , else continue
- If $k=b$, return α ; else $k \leftarrow k+1$ go to step 2 4.' If $k=b$, return β ; else $k \leftarrow k+1$ go to step 2'

Begin with $AB(s; -\infty, +\infty)$ and $\alpha < \beta$ throughout

8

EEL5840: Elements of Machine Intelligence
SubjuGator

Adversarial Search

- Alpha-Beta Procedure Example

University of Florida
EEL 5840 - Class 427 - Fall 2009
© Dr. A. Armitage Armitage

9

EEL5840: Elements of Machine Intelligence
SubjuGator

Adversarial Search

Consider the following game tree in which the static scores (in parentheses at the tip nodes) are all from the first player's point of view.

- Assuming that the first player is the maximizing player, what move should the first player choose?
- Assuming that the first player is the minimizing player, what move should the first player choose?
- What nodes would not need to be examined in part (a) using the alpha-beta algorithm - assuming that the nodes are examined in left-to-right order?
- What nodes would not need to be examined in part (b) using the alpha-beta algorithm - assuming that the nodes are examined in right-to-left order?
- Is the first player's move in parts (a) and (c) or in parts (b) and (d) different? Explain.

Part (a) A chooses C
Part (b) A chooses D
Part (c) Do Not Evaluate {O,U,X,Y, and K}
Part (d) Do Not Evaluate {V,R,S,H,P,M,L and E}
Part (e) Alpha-Beta and Minimax produce the same results

University of Florida
EEL 5840 - Class 427 - Fall 2009
© Dr. A. Armitage Armitage

10

EEL5840: Elements of Machine Intelligence
SubjuGator

Adversarial Search

- Search Efficiency of the Alpha-Beta Procedure
 - In order to do alpha-beta search, at least some part of the search tree must be generated to maximum depth, which implies some form of DFS.
 - The number of cut-offs that can be made during a search depends on the degree to which the early alpha and beta values approximate the final backed-up values.
 - If a tree has depth d and b successors, it will have b^d tip nodes
 - Suppose that an $\alpha\beta$ procedure generated successors in the order of their true backed-up values—the lowest valued successors first for MIN nodes and the highest valued successors first for MAX nodes (this order is unknown at the time of successor generation). Slagle et al and Knuth et al

$$N_d = 2b^{d/2} - 1 \text{ for even } d \text{ or } N_d = b^{(d+1)/2} + b^{(d+1)/2} - 1 \text{ for odd } d$$

Alpha-Beta with perfect ordering reduces the effective branching factor from b to approximately \sqrt{b} . Since this cannot really be achieved [Pearl 1984] shows that it is more like $b^{4/3}$

University of Florida
EEL 5840 - Class 427 - Fall 2009
© Dr. A. Armitage Armitage

11

EEL5840: Elements of Machine Intelligence
SubjuGator

The End!

University of Florida
EEL 5840 - Class 427 - Fall 2009
© Dr. A. Armitage Armitage

12