Lecture 14 (α-β Pruning)

1 General Idea - X Version

1. X is MIN or MAX
2. Consider a node n which is being explored currently
3. Let a be *best* value parent of X can get along any choice on current path from the root
4. If value at n becomes *worse* than a, $\sim X$ will not consider this node and hence we don't explore further children of a

2 Algorithm

α: MAX's best option on path to root

β: MIN's best option on path to root

```
def max-value(state, alpha, beta):
    initialize v = -INF
    for each successor of state:
        v = max(v, value(successor, alpha, beta))
        if v >= beta:
            return v
        alpha = max(alpha, v)
    return v
```

```
def min-value(state , alpha, beta):
    initialize v = +INF
    for each successor of state:
        v = min(v, value(successor, alpha, beta))
        if v <= alpha:
            return v
        beta = min(beta, v)
    return v
```

3 Properties

1. Doesn't affect minimax value at root
2. It is a form of meta-reasoning
3. Ordering of nodes matters, but best ordering cannot be found
4. Time complexity: $O(b^{m/2})$ if best ordering used, else $O(b^{3m/4})$ on average

4 Cutting Off Search

1. Time complexity is very large
2. Depth-Limited search is done, use heuristic for non-terminal node at "max" depth
3. Evaluation function is usually weighted sum of features