Mini-Max vs Alpha-Beta

Michael Erdmann
15-150
Principles of Functional Programming
April 10, 2018
Mini-Max at a Maxie Node

\[ v = \max\{v_1, \ldots, v_k\} \]
alpha-beta pruning

• Rather than blindly evaluate children of a node, we pass an accumulator argument:

\[(\alpha, \beta)\] (REQUIRE \(\alpha < \beta\))

• \(\alpha\) is the best Maxie can achieve in the tree explored so far.
• \(\beta\) is the best Minnie can achieve in the tree explored so far.

Think of \((\alpha, \beta)\) as an open interval of potential game values that both players would like better than their current best options.
alpha-beta pruning

• Rather than blindly evaluate children of a node, we pass an accumulator argument:
  \((\alpha, \beta)\)  
  (REQUIRE \(\alpha < \beta\))

• \(\alpha\) is the best Maxie can achieve in the tree explored so far.
• \(\beta\) is the best Minnie can achieve in the tree explored so far.

**KEY POINT:** If Maxie sees a move to a node with value \(v \geq \beta\),
then Maxie should stop searching from the current node.

Q: Why?
A: Minnie can prevent the game from reaching the current node.

(Dually, with roles of Maxie and Minnie reversed, now using “value \(v \leq \alpha\”).)
**Alpha-Beta at a Maxie Node**

\[ \alpha_1 = \alpha \]
\[ \alpha_2 = \max(\alpha_1, v_1) \]
\[ \vdots \]
\[ \alpha_i = \max(\alpha_{i-1}, v_{i-1}) \]

\[ v = \max\{v_1, \ldots, v_i\}, \]

with \( i \) the smallest index such that \( v_i \geq \beta \), if that occurs,

and \( i=k \) otherwise.

\[ \alpha_1 = \alpha \]
\[ \alpha_2 = \max(\alpha_1, v_1) \]
\[ \vdots \]
\[ \alpha_i = \max(\alpha_{i-1}, v_{i-1}) \]

\[ v = \max\{v_1, \ldots, v_i\}, \]

with \( i \) the smallest index such that \( v_i \geq \beta \), if that occurs,

and \( i=k \) otherwise.

No need to explore these children
Let’s hide the leaf values until the search needs them.
We’ll label nodes with values $v$ and edges with $(\alpha, \beta)$ intervals and update these dynamically as the search progresses.
Maxie

Minnie

Maxie

\([ (\infty, +\infty) \]

\([ (7, +\infty) \]

7 6 3
\( (\infty, +\infty) \) 

\( (\infty, 7) \)

Maxie

Minnie

Maxie

\( v \geq 9 > 7 = \beta \)
Maxie

Minnie

Maxie

\( (\infty, +\infty) \)

\( (-\infty, 7) \)

\( 7 \)

\( (\geq 9) \)

\( 7 \)

\( 6 \)

\( 3 \)

\( 9 \)

\( 2 \)

\( 4 \)
Maxie

Minnie

Maxie

4

≥9

(≥9)

(4, +∞)

(4, +∞)

(4, +∞)

7 6 3 9 2 4 1 1 3 5

7 6 3 9 2 4 1 1 3 5
Maxie

Minnie

Maxie

\[ (\geq 9) \]

\[ (4, +\infty) \]
Maxie

Minnie

Maxie
The “deep pruning” on the right occurs because at a Minnie level $v \leq 3 < 5 = \alpha$. 
Alpha-Beta saved 11 of 27 node evaluations
Alpha-Beta saved 11 of 27 node evaluations

(Optimal game play indicated by arrows.)