Reminder: Technology Policy

No technology except for first four rows of left and right sides
Jupyter Notebooks Postponed

“Primer”: Friday 5pm in Gates G01

Postponed – TBA

Covers:

- Jupyter Notebooks and .ipynb files
- Google Colab (online Jupyter Notebook environment)
- Useful Python features

Bring your laptop
This Friday’s Lecture Preponed

To be given
Wednesday, February 12 3:30-4:30
Gates G01

Will be recorded
and available on course website
(must watch between Wed and following Mon)
This Friday’s Lecture Preponed

To be given
Wednesday, February 12 2:30-3:30
Gates G01

Will be recorded
and available on course website
(must watch between Wed and following Mon)

(Please come to the live version!)
Recap

Game tree concept
Recap

Game tree concept
Recap

Game tree concept
Terminal nodes have “ground truth” values
(examples: \(+1, 0, -1\), \(+\infty, -\infty\))
Recap

Game tree concept
Terminal nodes have “ground truth” values
(examples: {+1, 0, -1}, {+\infty, -\infty})
Recap

Game tree concept
Terminal nodes have “ground truth” values
(examples: {+1, 0, -1}, {+∞, -∞})

Internal nodes have values based on values of descendants
Minimax Value of a Game

My Turn
\[ V(s) = \text{Max of successors} \]

My Opponent’s Turn
\[ V(s) = \text{Min of successors} \]

Continued to terminal nodes
Recap

Game tree concept

Terminal nodes have “ground truth” values
(examples: \{+1, 0, -1\}, \{+\infty, -\infty\})

Internal nodes have values based on values of descendants
Recap

Game tree concept
Terminal nodes have “ground truth” values
(examples: {+1, 0, -1}, {+∞, -∞})

Internal nodes have values based on values of descendants

Can stop early and use heuristic evaluation functions
Minimax Value of a Game

My Turn
\[ f(s) = \text{Max of successors} \]

My Opponent’s Turn
\[ f(s) = \text{Min of successors} \]

Use heuristic evaluation function
Minimax Value of a Game

My Turn
\[ f(s) = \text{Max of successors} \]

My Opponent’s Turn
\[ f(s) = \text{Min of successors} \]

Use heuristic evaluation function

Tree diagram showing the minimax values and decision points for the game.
Recap

Game tree concept
Terminal nodes have “ground truth” values
(examples: {+1, 0, -1}, {+∞, -∞})

Internal nodes have values based on values of descendants

Can stop early and use heuristic evaluation functions
Recap

Game tree concept
Terminal nodes have “ground truth” values
(examples: \{+1, 0, -1\}, \{+\infty, -\infty\})

Internal nodes have values based on values of descendants

Can stop early and use heuristic evaluation functions
“Minimax Search”
Minimax Algorithm

\textbf{maxturn}(s,ops):
\begin{align*}
\text{if } \text{cutoff}(s) \text{ then return } f(s) \\
\text{else} \\
\quad \text{val} &\leftarrow -\infty; \\
\quad \text{foreach } o \in \text{ops} \\
\quad \quad \text{val}' &\leftarrow \text{minturn}(\text{apply}(s,o),\text{ops}); \\
\quad \quad \text{if } \text{val}' > \text{val} \text{ then} \\
\quad \quad \quad \text{val} &\leftarrow \text{val}'; \\
\quad \quad \quad \text{bestop} &\leftarrow o; \\
\quad \text{return val}
\end{align*}

\textbf{minturn}(s,ops):
\begin{align*}
\text{if } \text{cutoff}(s) \text{ then return } f(s) \\
\text{else} \\
\quad \text{val} &\leftarrow +\infty; \\
\quad \text{foreach } o \in \text{ops} \\
\quad \quad \text{val}' &\leftarrow \text{maxturn}(\text{apply}(s,o),\text{ops}); \\
\quad \quad \text{if } \text{val}' < \text{val} \text{ then} \\
\quad \quad \quad \text{val} &\leftarrow \text{val}'; \\
\quad \quad \quad \text{bestop} &\leftarrow o; \\
\quad \text{return val}
\end{align*}

Initial call:
\begin{itemize}
\item If I go first: \text{maxturn(initial-state,ops)}
\item If opponent goes first: \text{minturn(initial-state,ops)}
\end{itemize}

cutoff(s) could be a depth-bound or something more sophisticated
Recap

Game tree concept
Terminal nodes have “ground truth” values
(examples: \{+1, 0, -1\}, \{+\infty, -\infty\})

Internal nodes have values based on values of descendants

Can stop early and use heuristic evaluation functions
“Minimax Search”
Recap

Game tree concept

Terminal nodes have “ground truth” values
(examples: {+1, 0, -1}, {+∞, -∞})

Internal nodes have values based on values of descendants

Can stop early and use heuristic evaluation functions

“Minimax Search”

Can avoid needless effort: Alpha-Beta Pruning
Alpha-Beta Pruning

This is guaranteed to be a better move than this regardless of what results from the other actions.

My Turn
f(s) = Max of successors

My Opponent’s Turn
f(s) = Min of successors
There is no way to set a value to either of these two states that would change the outcome.
If $\alpha \geq \text{val}$ then there's no need to search further for this node.
Alpha-Beta Pruning: General Case

\[ \beta \text{ is lowest (worst) value of all options available to opponent in response to my move thus far} \]

\[ \text{val is biggest (best) value of all options available to opponent in response to my move thus far} \]

If \( \beta \leq \text{val} \) then there’s no need to search further for this node.
Minimax Algorithm with Alpha-Beta Pruning

\[
\text{maxturn}(s,\text{ops},a,b): \quad \{\text{my turn}\}
\]
\[
\text{if \ cutoff}(s,\text{depth}) \text{ then return } f(s)
\]
\[
\text{else}
\]
\[
\text{val} \leftarrow -\infty;
\]
\[
\text{foreach } o \in \text{ops}
\]
\[
\text{val'} \leftarrow \text{minturn}(\text{apply}(s,o),\text{ops},a,b);
\]
\[
\text{if val'} > \text{val} \text{ then}
\]
\[
\text{val} \leftarrow \text{val'};
\]
\[
\text{bestop} \leftarrow o;
\]
\[
\text{if val } \geq b \text{ then return val;}
\]
\[
a \leftarrow \max(a,\text{val})
\]
\[
\text{return val}
\]

\[
\text{minturn}(s,\text{ops},a,b): \quad \{\text{opponent’s turn}\}
\]
\[
\text{if \ cutoff}(s,\text{depth}) \text{ then return } f(s)
\]
\[
\text{else}
\]
\[
\text{val} \leftarrow +\infty;
\]
\[
\text{foreach } o \in \text{ops}
\]
\[
\text{val'} \leftarrow \text{maxturn}(\text{apply}(s,o),\text{ops},a,b);
\]
\[
\text{if val'} < \text{val} \text{ then}
\]
\[
\text{val} \leftarrow \text{val'};
\]
\[
\text{bestop} \leftarrow o;
\]
\[
\text{if val } \leq a \text{ then return val;}
\]
\[
b \leftarrow \min(b,\text{val})
\]
\[
\text{return val}
\]

Initial call:
- If I go first: \(\text{maxturn}(\text{initial-state},\text{ops},-\infty,+\infty)\)
- If opponent goes first: \(\text{minturn}(\text{initial-state},\text{ops},-\infty,+\infty)\)
Alpha-Beta Pruning: Implementation

- My Turn
  \[ f(s) = \text{Max of successors} \]

- My Opponent’s Turn
  \[ f(s) = \text{Min of successors} \]
Alpha-Beta Pruning: Implementation

My Turn
\[ f(s) = \text{Max of successors} \]

My Opponent’s Turn
\[ f(s) = \text{Min of successors} \]