CSC242: Intro to AI

Lecture 8
Quiz 2 Review
TA Help Sessions (v2)

- Monday & Tuesday: 17:00–18:00, Hylan 301
- Doodle poll signup before 16:00
- Link on BB:
  [http://www.doodle.com/xgxcbxn4knks86sx](http://www.doodle.com/xgxcbxn4knks86sx)
Stochastic and Partially Observable Games
EISBot
Overview

Ben Weber,
Michael Mateas, Arnav Jhala
Expressive Intelligence Studio
UC Santa Cruz

http://eis.ucsc.edu/EISBot
Review...
Games ≠ Toy Problems

\[ 9! = 362880 \]

\[ 35^{100} = 10^{154} \]

\[ 2 \times 10^{170} \]
Games

• “Require the ability to make some decision even when calculating the *optimal* decision is infeasible”

• “Penalize inefficiency severely”
# Types of Games

<table>
<thead>
<tr>
<th>Deterministic (no chance)</th>
<th>Nondeterministic (dice, cards, etc.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perfect information (fully observable)</td>
<td>Imperfect information (partially observable)</td>
</tr>
<tr>
<td>Zero-sum (total payoff the same in any game)</td>
<td>Arbitrary utility functions</td>
</tr>
</tbody>
</table>
Minimax Algorithm

\[
\text{MINIMAX}(s) = \begin{cases} 
\text{UTILITY}(s) & \text{if } \text{TERTINAL-TEST}(s) \\
\max_{a \in \text{ACTIONS}(s)} \text{MINIMAX}(\text{RESULT}(s, a)) & \text{if } \text{PLAYER}(s) = \text{MAX} \\
\min_{a \in \text{ACTIONS}(s)} \text{MINIMAX}(\text{RESULT}(s, a)) & \text{if } \text{PLAYER}(s) = \text{MIN}
\end{cases}
\]
Minimax Summary

• Computes the optimal move assuming opponent also plays optimally (i.e., worst-case outcome)

• Explores game tree depth-first all the way to terminal states (end of game)

• Backs up utility values through alternating MIN and MAX (what’s best for me is worst for you, and vice-versa)
Heuristic Minimax

• Cutoff search before reaching terminal nodes (time, depth, “quiescence”)

• Use heuristic evaluation function to estimate state utility

• Backs up utility values through alternating MIN and MAX (what’s best for me is worst for you, and vice-versa)
Heuristic Minimax

\[ H\text{-MINIMAX}(s) = \begin{cases} h(s) & \text{if CUTOFF-TEST}(s) \\ \max_{a \in \text{ACTIONS}(s)} \text{MINIMAX}(\text{RESULT}(s, a)) & \text{if PLAYER}(s) = \text{MAX} \\
\min_{a \in \text{ACTIONS}(s)} \text{MINIMAX}(\text{RESULT}(s, a)) & \text{if PLAYER}(s) = \text{MIN} \end{cases} \]
Alpha-Beta Summary

- Easy bookkeeping modification of basic MINIMAX algorithm
- Not hard to come up with “useful” node orderings
- Even random gets you 33% deeper search
- Works with other ways of improving game tree search
Stochastic and Partially Observable Games
### Types of Games

<table>
<thead>
<tr>
<th>Deterministic (no chance)</th>
<th>Nondeterministic (dice, cards, etc.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perfect information (fully observable)</td>
<td>Imperfect information (partially observable)</td>
</tr>
<tr>
<td>Zero-sum (total payoff the same in any game)</td>
<td>Arbitrary utility functions</td>
</tr>
</tbody>
</table>
Non-deterministic (Stochastic) Games

• A player’s possible moves depend on chance (random) elements, e.g., dice
Stochastic Games
Stochastic Games

• A player’s possible moves depend on chance (random) elements, e.g., dice

• Can’t build game tree since don’t know what future legal moves will be
A game tree is shown, with nodes labeled with 'MAX', 'CHANCE', and 'MIN'. The terminal nodes are labeled with payoffs: 2, -1, 1, -1, 1.
\[ P(H) = 0.5 \quad H \quad T \quad P(T) = 0.5 \]
\[ P(H) = 0.5 \quad \text{and} \quad P(T) = 0.5 \]
$P(H) = 0.5$  
$P(T) = 0.5$
\[ P(H) = 0.75 \quad \begin{array}{c}
+1
\end{array} \]

\[ P(T) = 0.25 \quad \begin{array}{c}
-1
\end{array} \]
\[ P(H) = 0.75 \quad \text{H} \quad +0.5 \quad \text{T} \quad P(T) = 0.25 \]
Expectation

• Weighted average of possibilities

• Sum of the possible outcomes weighted by the likelihood of their occurrence

• What you would expect to win in the long run
Expecti-Minimax

- Same as MINIMAX for MIN and MAX nodes
- Same backing up utilities from terminal nodes
- Take expectation over chance nodes
- Weighted average of possible outcomes
Expecti-Minimax

\[
\text{EMinimax}(s) = \begin{cases} 
\text{Utility}(s) & \text{if Terminal-Test}(s) \\
\max_a \text{EMinimax} (\text{Result}(S, a)) & \text{if Player}(s) = \text{max} \\
\min_a \text{EMinimax} (\text{Result}(S, a)) & \text{if Player}(s) = \text{min} \\
\sum_r P(r) \text{EMinimax} (\text{Result}(S, r)) & \text{if Player}(s) = \text{chance}
\end{cases}
\]
Expecti-Minimax

$O(b^m n^m)$
Stochastic Games

- Expectation to handle uncertainty and randomness
- For example in poker: “Pot Odds”
Partial Observability

- Some of the state of the world is hidden (unobservable)
- There is some uncertainty about the state of the world
Partially-Observable Games

• Some of the state of the game is hidden from the player(s)

• Interesting because:
  • Valuable real-world games (e.g., poker)
  • Partial observability arises all the time in real-world problems
Partially-Observable Games

• Deterministic partial observability
• Opponent has hidden state
• Battleship, Stratego, Kriegspiel
Partially-Observable Games

- Deterministic partial observability
  - Opponent has hidden state
  - Battleship, Stratego, Kriegspiel
- Stochastic partial observability
  - Missing/hidden information is random
  - Card games: bridge, hearts, poker (most)
Stochastic Partially Observable Games
<table>
<thead>
<tr>
<th>Hand</th>
<th>Frequency</th>
<th>Approx. Probability</th>
<th>Approx. Cumulative</th>
<th>Approx. Odds</th>
<th>Mathematical expression of absolute frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Royal flush</td>
<td>4</td>
<td>0.000154%</td>
<td>0.000154%</td>
<td>649,739 : 1</td>
<td>( \binom{4}{1} )</td>
</tr>
<tr>
<td>Straight flush (excluding royal flush)</td>
<td>36</td>
<td>0.00139%</td>
<td>0.00154%</td>
<td>72,192.33 : 1</td>
<td>( \binom{10}{1} \binom{4}{1} - \binom{4}{1} )</td>
</tr>
<tr>
<td>Four of a kind</td>
<td>624</td>
<td>0.0240%</td>
<td>0.0256%</td>
<td>4,164 : 1</td>
<td>( \binom{13}{1} \binom{12}{1} \binom{4}{1} )</td>
</tr>
<tr>
<td>Full house</td>
<td>3,744</td>
<td>0.144%</td>
<td>0.170%</td>
<td>693.2 : 1</td>
<td>( \binom{13}{1} \binom{4}{3} \binom{12}{1} \binom{4}{2} )</td>
</tr>
<tr>
<td>Flush (excluding royal flush and straight flush)</td>
<td>5,108</td>
<td>0.197%</td>
<td>0.367%</td>
<td>507.8 : 1</td>
<td>( \binom{13}{5} \binom{4}{1} - \binom{10}{1} \binom{4}{1} )</td>
</tr>
<tr>
<td>Straight (excluding royal flush and straight flush)</td>
<td>10,200</td>
<td>0.392%</td>
<td>0.76%</td>
<td>253.8 : 1</td>
<td>( \binom{10}{1} \binom{4}{5} - \binom{10}{1} \binom{4}{1} )</td>
</tr>
<tr>
<td>Three of a kind</td>
<td>54,912</td>
<td>2.11%</td>
<td>2.87%</td>
<td>46.3 : 1</td>
<td>( \binom{13}{1} \binom{4}{3} \binom{12}{1} \binom{4}{2} )</td>
</tr>
<tr>
<td>Two pair</td>
<td>123,552</td>
<td>4.75%</td>
<td>7.62%</td>
<td>20.03 : 1</td>
<td>( \binom{13}{2} \binom{4}{2} \binom{11}{1} \binom{4}{1} )</td>
</tr>
<tr>
<td>One pair</td>
<td>1,098,240</td>
<td>42.3%</td>
<td>49.9%</td>
<td>1.36 : 1</td>
<td>( \binom{13}{1} \binom{4}{2} \binom{12}{3} \binom{4}{3} )</td>
</tr>
<tr>
<td>No pair / High card</td>
<td>1,302,540</td>
<td>50.1%</td>
<td>100%</td>
<td>0.95 : 1</td>
<td>( \left( \binom{13}{5} - 10 \right) \binom{4}{5} - 4 )</td>
</tr>
<tr>
<td>Total</td>
<td>2,598,960</td>
<td>100%</td>
<td>100%</td>
<td>1 : 1</td>
<td>( \binom{52}{5} )</td>
</tr>
</tbody>
</table>
Weighted Minimax

- For each possible deal $s$:
  - Assume $s$ is the actual situation
  - Compute Minimax or H-Minimax value of $s$
  - Weight value by probability of $s$
  - Take move that yields highest expected value over all the possible deals
Weighted Minimax

$$\arg\max_a \sum_s P(s) \text{Minimax(RESULT}(s, a))$$
Weighted Minimax

\[
\arg\max_a \sum_s P(s) \text{Minimax}(\text{Result}(s, a))
\]

\[
\binom{26}{13} = 10,400,600
\]

\[
\binom{47}{25} = 1.48338977 \times 10^{13}
\]
Monte Carlo Methods

- Use a “representative” sample to approximate a large, complex distribution
Monte Carlo Minimax

\[ \arg\max_a \frac{1}{N} \sum_{i=1}^{N} \text{Minimax}(\text{Result}(s_i, a)) \]
Summary

• Non-deterministic games

• Expecti-MINIMAX: Compute expected MINIMAX value over chance nodes

• Partially observable games

• Weighted MINIMAX: Compute expected value over possible hidden states

• Naive approaches impractical (but stay tuned)
For Next Time:
AIMA 6.0-6.4