Expectimax for Pacman

Results from playing 5 games

<table>
<thead>
<tr>
<th></th>
<th>Minimizing Ghost</th>
<th>Random Ghost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimax Pacman</td>
<td>Won 5/5</td>
<td>Won 5/5</td>
</tr>
<tr>
<td></td>
<td>Avg. Score: 493</td>
<td>Avg. Score: 483</td>
</tr>
<tr>
<td>Expectimax Pacman</td>
<td>Won 1/5</td>
<td>Won 5/5</td>
</tr>
<tr>
<td></td>
<td>Avg. Score: -303</td>
<td>Avg. Score: 503</td>
</tr>
</tbody>
</table>

Pacman used depth 4 search with an eval function that avoids trouble
Ghost used depth 2 search with an eval function that seeks Pacman
Expectimax Search

- Chance nodes
  - Chance nodes are like min nodes, except the outcome is uncertain
  - Calculate expected utilities
  - Chance nodes average successor values (weighted)
- Each chance node has a probability distribution over its outcomes (called a model)
  - For now, assume we're given the model
- Utilities for terminal states
  - Static evaluation functions give us limited-depth search

Expectimax Quantities
Expectimax Pruning?

Expectimax Evaluation

- Evaluation functions quickly return an estimate for a node’s true value (which value, expectimax or minimax?)
- For minimax, evaluation function scale doesn’t matter
  - We just want better states to have higher evaluations (get the ordering right)
  - We call this insensitivity to monotonic transformations
- For expectimax, we need magnitudes to be meaningful
Mixed Layer Types

- E.g. Backgammon
- Expectimimimax
  - Environment is an extra player that moves after each agent
  - Chance nodes take expectations, otherwise like minimax

ExpectiMinimax-Value(state):

\[
\begin{align*}
\text{if state is a MAX node then} & \quad \text{return the highest ExpectiMinimax-Value of Successors(state)} \\
\text{if state is a MIN node then} & \quad \text{return the lowest ExpectiMinimax-Value of Successors(state)} \\
\text{if state is a chance node then} & \quad \text{return average of ExpectiMinimax-Value of Successors(state)}
\end{align*}
\]

Stochastic Two-Player

- Dice rolls increase \( b \): 21 possible rolls with 2 dice
  - Backgammon \( \approx 20 \) legal moves
  - Depth \( 2 = 20 \times (21 \times 20)^3 = 1.2 \times 10^9 \)
- As depth increases, probability of reaching a given search node shrinks
  - So usefulness of search is diminished
  - So limiting depth is less damaging
  - But pruning is trickier…
- TDGammon uses depth-2 search + very good evaluation function + reinforcement learning: world-champion level play
- 1\textsuperscript{st} AI world champion in any game!
Multi-Agent Utilities

- Similar to minimax:
  - Terminals have utility tuples
  - Node values are also utility tuples
  - Each player maximizes its own utility
  - Can give rise to cooperation and competition dynamically...

Maximum Expected Utility

- **Principle of maximum expected utility:**
  - A rational agent should choose the action which maximizes its expected utility, given its knowledge

- **Questions:**
  - Where do utilities come from?
  - How do we know such utilities even exist?
  - Why are we taking expectations of utilities (not, e.g. minimax)?
  - What if our behavior can’t be described by utilities?
Utilities: Unknown Outcomes

Going to airport from home

- Take freeway
  - Clear, 10 min
  - Arrive early
- Take surface streets
  - Traffic, 50 min
  - Arrive late
  - Clear, 20 min
  - Arrive on time

Preferences

- An agent chooses among:
  - Prizes: $A$, $B$, etc.
  - Lotteries: situations with uncertain prizes

\[ L = [p, A; (1-p), B] \]

- Notation:
  - $A \succ B$ \quad $A$ preferred over $B$
  - $A \sim B$ \quad indifference between $A$ and $B$
  - $A \preceq B$ \quad $B$ not preferred over $A$
Rational Preferences

- We want some constraints on preferences before we call them rational
  \((A \succ B) \land (B \succ C) \Rightarrow (A \succ C)\)

- For example: an agent with intransitive preferences can be induced to give away all of its money
  - If \(B \succ C\), then an agent with \(C\) would pay (say) 1 cent to get \(B\)
  - If \(A \succ B\), then an agent with \(B\) would pay (say) 1 cent to get \(A\)
  - If \(C \succ A\), then an agent with \(A\) would pay (say) 1 cent to get \(C\)

Rational Preferences

- Preferences of a rational agent must obey constraints.
  - The axioms of rationality:
    - **Orderability**
      \((A \succ B) \lor (B \succ A) \lor (A \sim B)\)
    - **Transitivity**
      \((A \succ B) \land (B \succ C) \Rightarrow (A \succ C)\)
    - **Continuity**
      \(A \succ B \succ C \Rightarrow \exists p [p, A; 1 - p, C] \sim B\)
    - **Substitutability**
      \(A \sim B \Rightarrow [p, A; 1 - p, C] \sim [p, B; 1 - p, C]\)
    - **Monotonicity**
      \(A \succ B \Rightarrow (p \geq q \Leftrightarrow [p, A; 1 - p, B] \succeq [q, A; 1 - q, B])\)

- Theorem: Rational preferences imply behavior describable as maximization of expected utility
MEU Principle

- Theorem:
  - [Ramsey, 1931; von Neumann & Morgenstern, 1944]
  - Given any preferences satisfying these constraints, there exists a real-valued function $U$ such that:
    \[
    U(A) \geq U(B) \iff A \succeq B
    \]
    \[
    U([p_1, S_1; \ldots; p_n, S_n]) = \sum p_i U(S_i)
    \]

- Maximum expected likelihood (MEU) principle:
  - Choose the action that maximizes expected utility
  - Note: an agent can be entirely rational (consistent with MEU) without ever representing or manipulating utilities and probabilities
  - E.g., a lookup table for perfect tic-tac-toe, reflex vacuum cleaner

Utility Scales

- Normalized utilities: $u_+ = 1.0$, $u_- = 0.0$

- Micromorts: one-millionth chance of death, useful for paying to reduce product risks, etc.

- QALYs: quality-adjusted life years, useful for medical decisions involving substantial risk

- Note: behavior is invariant under positive linear transformation
  \[
  U'(x) = k_1 U(x) + k_2 \quad \text{where } k_1 > 0
  \]

- With deterministic prizes only (no lottery choices), only ordinal utility can be determined, i.e., total order on prizes
Human Utilities

- Utilities map states to real numbers. Which numbers?
- Standard approach to assessment of human utilities:
  - Compare a state A to a standard lottery $L_p$ between
    - "best possible prize" $u$, with probability $p$
    - "worst possible catastrophe" $u$, with probability $1-p$
  - Adjust lottery probability $p$ until $A \sim L_p$
  - Resulting $p$ is a utility in $[0,1]$

\[ \text{pay } $30 \sim \] continue as before

\[ 0.999999 \]

\[ \text{instant death} \]

Money

- Money does not behave as a utility function, but we can talk about the utility of having money (or being in debt)
- Given a lottery $L = [p, X; (1-p), Y]$
  - The expected monetary value $EMV(L)$ is $pX + (1-p)Y$
  - $U(L) = pU(X) + (1-p)U(Y)$
  - Typically, $U(L) < U(EMV(L))$: why?
  - In this sense, people are risk-averse
  - When deep in debt, we are risk-prone

- Utility curve: for what probability $p$ am I indifferent between:
  - Some sure outcome $x$
  - A lottery $[p, M; (1-p), 0]$, $M$ large
Example: Insurance

- Consider the lottery $[0.5,$1000; 0.5,$0]
  - What is its expected monetary value? ($500)
  - What is its certainty equivalent?
    - Monetary value acceptable in lieu of lottery
    - $400 for most people
  - Difference of $100 is the insurance premium
    - There’s an insurance industry because people will pay to reduce their risk
    - If everyone were risk-neutral, no insurance needed!

Example: Insurance

- Because people ascribe different utilities to different amounts of money, insurance agreements can increase both parties’ expected utility

You own a car. Your lottery: $L_Y = [0.8, $0 ; 0.2, -$200]
i.e., 20% chance of crashing

You do not want -$200!

$U_Y(L_Y) = 0.2U_Y(-$200) = -200$
$U_Y(-$50) = -150$

<table>
<thead>
<tr>
<th>Amount</th>
<th>Your Utility $U_Y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0</td>
<td>0</td>
</tr>
<tr>
<td>-$50</td>
<td>-150</td>
</tr>
<tr>
<td>-$200</td>
<td>-1000</td>
</tr>
</tbody>
</table>
Example: Insurance

- Because people ascribe different utilities to different amounts of money, insurance agreements can increase both parties' expected utility.

<table>
<thead>
<tr>
<th>You own a car. Your lottery:</th>
<th>Insurance company buys risk:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_Y = [0.8, $0; 0.2, -$200]$</td>
<td>$L_I = [0.8, $50; 0.2, -$150]$</td>
</tr>
<tr>
<td>i.e., 20% chance of crashing</td>
<td>i.e., $50 revenue + your $L_Y$</td>
</tr>
<tr>
<td>You do not want -$200!</td>
<td>Insurer is risk-neutral:</td>
</tr>
<tr>
<td>$U_Y(L_Y) = 0.2*U_Y(-$200) = -200$</td>
<td>$U(I) = U(0.8<em>50 + 0.2</em>(-150))$</td>
</tr>
<tr>
<td>$U_Y(-$50) = -150</td>
<td>= $U(10) &gt; U(0)$</td>
</tr>
</tbody>
</table>

Example: Human Rationality?

- Famous example of Allais (1953)
  - A: [0.8,$4k; 0.2,$0]
  - B: [1.0,$3k; 0.0,$0]
  - C: [0.2,$4k; 0.8,$0]
  - D: [0.25,$3k; 0.75,$0]

- Most people prefer B > A, C > D
- But if $U(0) = 0$, then
  - B > A ⇒ $U(3k) > 0.8 U(4k)$
  - C > D ⇒ 0.8 $U(4k) > U(3k)$
Reinforcement Learning

- **Basic idea:**
  - Receive feedback in the form of rewards
  - Agent’s utility is defined by the reward function
  - Must learn to act so as to maximize expected rewards
  - Change the rewards, change the learned behavior

- **Examples:**
  - Playing a game, reward at the end for winning / losing
  - Vacuuming a house, reward for each piece of dirt picked up
  - Automated taxi, reward for each passenger delivered

- First: Need to master MDPs

---

Grid World

- The agent lives in a grid
- Walls block the agent’s path
- The agent’s actions do not always go as planned:
  - 80% of the time, the action North takes the agent North (if there is no wall there)
  - 10% of the time, North takes the agent West; 10% East
  - If there is a wall in the direction the agent would have been taken, the agent stays put
- Big rewards come at the end
Markov Decision Processes

- An MDP is defined by:
  - A set of states \( s \in S \)
  - A set of actions \( a \in A \)
  - A transition function \( T(s, a, s') \)
    - Prob that \( a \) from \( s \) leads to \( s' \)
    - i.e., \( P(s' | s, a) \)
    - Also called the model
  - A reward function \( R(s, a, s') \)
    - Sometimes just \( R(s) \) or \( R(s') \)
  - A start state (or distribution)
  - Maybe a terminal state

- MDPs are a family of non-deterministic search problems
  - Reinforcement learning: MDPs where we don’t know the transition or reward functions

Solving MDPs

- In deterministic single-agent search problem, want an optimal plan, or sequence of actions, from start to a goal
- In an MDP, we want an optimal policy \( \pi^* : S \rightarrow A \)
  - A policy \( \pi \) gives an action for each state
  - An optimal policy maximizes expected utility if followed
  - Defines a reflex agent
Example Optimal Policies

R(s) = -0.01

R(s) = -0.03

R(s) = -0.4

R(s) = -2.0