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

- Estimate of true expectimax value (which would require a lot of work to compute)

Expectimax Pruning?

- 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

Expectimax Evaluation
Mixed Layer Types

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

```
ExpectMinimax-Value(state):
    if state is a MAX node then
        return the highest ExpectMinimax-Value of Successors(state)
    if state is a MIN node then
        return the lowest ExpectMinimax-Value of Successors(state)
    if state is a chance node then
        return average of ExpectMinimax-Value of Successors(state)
```

Stochastic Two-Player

- Dice rolls increase b: 21 possible rolls with 2 dice
  - Backgammon = 20 legal moves
  - Depth 2 = 20 x (21 x 20)^2 = 1.2 x 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
  - 1st 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 chose 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
    - Traffic, 50 min
  - Take surface streets
    - Clear, 20 min

Preferences

- An agent chooses among:
  - Prizes: A, B, etc.
  - Lotteries: situations with uncertain prizes
  - \( L = [p, A; (1 - p), B] \)

- Notation:
  - \( A \succ B \)  A preferred over B
  - \( A \sim B \)  indifference between A and B
  - \( A \succeq B \)  B not preferred over A
Rational Preferences

- We want some constraints on preferences before we call them rational.

\( (A > B) \land (B > C) \Rightarrow (A > C) \)

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

Preferences of a rational agent must obey constraints.
- The axioms of rationality:
  - Ordering
    \( (A > B) \lor (B > A) \lor (A \sim B) \)
  - Transitivity
    \( (A > B) \land (B > C) \Rightarrow (A > C) \)
  - Continuity
    \( A > B > C \Rightarrow \exists p \in [0,1] \text{ such that } A \sim [p \cdot A; 1-p \cdot C] \sim B \)
  - Substitutability
    \( A \sim B \Rightarrow [p \cdot A; 1-p \cdot C] \sim [p \cdot B; 1-p \cdot C] \)
  - Monotonicity
    \( A > B \Rightarrow (p > q \Rightarrow [p \cdot A; 1-p \cdot B] > [q \cdot A; 1-q \cdot B]) \)
- 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:
    \[
    U(A) \geq U(B) \Leftrightarrow A \succeq B \\
    U([p_1, S_1; \ldots; p_n, S_n]) = \sum_i p_i U(S_i)
    \]

- 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: \( u_0 = 1.0, u_1 = 0.0 \)
- 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 \( L_0 \) between
    - ‘best possible prize’ \( u \), with probability \( p \)
    - ‘worst possible catastrophe’ \( u \) with probability 1-p
  - Adjust lottery probability \( p \) until \( A \sim L_0 \)
  - Resulting \( p \) is a utility in \([0,1]\)

  Pay \$30 ~ continue as before
  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 \( EMV(L) = p \cdot X + (1-p) \cdot Y \)
  - \( U(L) = p \cdot U(X) + (1-p) \cdot U(Y) \)
  - 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 \( [p, X; 1-p, Y], M \) 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!
  \(U_Y(L_Y) = 0.2*U_Y(-$200) = -200\)
  \(U_Y(-$50) = -150\)

Example: Insurance

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

  Insurance company buys risk:
  \(L_I = [0.8, $50; 0.2, -$150]\)
  i.e., $50 revenue + your \(L_Y\)
  Insurer is risk-neutral:
  \(U_I(L_I) = U(0.8*50 + 0.2*(-150))\)
  \(= U($10) > U($0)\)

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 \(\Rightarrow U($3k) > 0.8 U($4k)\)
  - C > D \(\Rightarrow 0.8 U($4k) > U($3k)\)

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, 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

- \( R(s) = -0.03 \)
- \( R(s) = -0.01 \)
- \( R(s) = -0.4 \)
- \( R(s) = -2.0 \)