CSE 40171: Artificial Intelligence

Adversarial Search: Game Trees, Expectimax; Partial Observability
Homework #3 has been released. It is due at 11:59PM on 9/28.
Upcoming Talk on AI:

Terry Boult, University of Colorado, Colorado Springs
“The Deep Unknown”
9/27 @ 3:30PM, DBART 138

Dr. Boult explores how deep networks fail on various types of unknown inputs and how to address some of the problems of open set deep networks.
Quiz 1 will be given on 9/28
(Review will be on Weds.)
Start thinking about your group for the class project (details coming 10/1)
Stochastic Games
What we’ve assumed thus far…
What if uncertain outcomes are controlled by chance, and not an adversary?
Expectimax Search

Why wouldn’t we know what the result of an action will be?

- Explicit randomness: rolling dice
- Unpredictable opponents: the pacman ghosts respond randomly
- Actions can fail: when moving a robot, wheels might slip

Values should now reflect average-case (expectimax) outcomes, not worst-case (minimax) outcomes

Slide credit: Dan Klein and Pieter Abbeel, UC Berkeley CS 188
Expectimax search: compute the average score under optimal play

- Max nodes as in minimax search
- Chance nodes are like min nodes but the outcome is uncertain
- Calculate their expected utilities
- i.e., take weighted average (expectation) of children
Demo: Minimax + Alpha-Beta Pruning

https://www.youtube.com/watch?v=_bEQJKXZ1-U
Demo: Expectimax

https://www.youtube.com/watch?v=ilxr3lAbpkw
Expectimax Pseudocode

```python
def value(state):
    if the state is a terminal state: return the state’s utility
    if the next agent is MAX: return max-value(state)
    if the next agent is EXP: return exp-value(state)

def max-value(state):
    initialize \( \nu = -\infty \)
    for each successor of state:
        \( \nu = \max(\nu, \text{value(successor)}) \)
    return \( \nu \)

def exp-value(state):
    initialize \( \nu = 0 \)
    for each successor of state:
        \( p = \text{probability(successor)} \)
        \( \nu += p \times \text{value(successor)} \)
    return \( \nu \)
```

Slide credit: Dan Klein and Pieter Abbeel, UC Berkeley CS 188
Expectimax Pseudocode

```python
def exp-value(state):
    initialize \( v = 0 \)
    for each successor of state:
        \( p = \text{probability}(\text{successor}) \)
        \( v \leftarrow v + p \times \text{value}(\text{successor}) \)
    return \( v \)
```

\[
v = (1/2) \times 8 + (1/3) \times 24 + (1/6) \times (-12) = 10
\]
Expectimax Example
Expectimax Pruning?
Depth-Limited Expectimax

Estimate of true expectimax value (which would require a lot of work to compute)
The Choice of Evaluation Function is Important

Image credit: Russell and Norvig
Partially Observable Games
Fog of War
“War is the realm of uncertainty; three quarters of the factors on which action in war is based are wrapped in a fog of greater or lesser uncertainty. A sensitive and discriminating judgment is called for; a skilled intelligence to scent out the truth”.

- Carl von Clausewitz
Example: Battleship
Chess Variant: Kriegspiel
Belief States

Initially, White’s belief state is a singleton because Black’s pieces haven’t moved yet.

After White makes a move and Black responds:

- White’s belief state contains 20 positions
- Because Back has 20 replies to any white move
KRK Endgame

Image credit: Russell and Norvig
Card Games
Naive Assumption: Card Games are Just Like Dice Games

**Algorithm:** consider all possible deals of the invisible cards; solve each one as if it were a fully observable game.

Then choose the move that has the best outcome averaged over all of the deals.

Assume that each deal $s$ occurs with probability $P(s)$.
Naive Assumption: Card Games are Just Like Dice Games

\[
\arg \max_x \sum_s P(s) \text{MINIMAX}(\text{RESULT}(s, a))
\]

Run exact \text{MINIMAX} if computationally feasible

Otherwise run \text{H-MINIMAX}
But the number of deals is very large

Monte Carlo Approximation: instead of adding up all the deals, take random sample of $N$ deals

The probability of deal $s$ appearing in the sample is proportional to $P(s)$:

$$\arg\max_x \frac{1}{N} \sum_{i=1}^{N} \text{MINIMAX} (\text{RESULT}(s_i, a))$$
Averaging Over Clairvoyance

**Day 1:** Road $A$ leads to a heap of gold; Road $B$ leads to a fork. Take the left fork and you’ll find a bigger heap of gold, but take the right fork and you’ll be run over by a bus.
Averaging Over Clairvoyance

**Day 2:** Road $A$ leads to a heap of gold; Road $B$ leads to a fork. Take the right fork and you’ll find a bigger heap of gold, but take the left fork and you’ll be run over by a bus.
Averaging Over Clairvoyance

**Day 3:** Road $A$ leads to a heap of gold; Road $B$ leads to a fork. One branch of the fork leads to a bigger heap of gold, but take the wrong fork and you’ll be hit by a bus. Unfortunately you don’t know which fork is which.
Averaging Over Clairvoyance’s Answer

Day 1: $B$ is the right choice

Day 2: $B$ is the right choice

Day 3: $B$ is still the right choice