Recap: Resource Limits

- Cannot search to leaves
- Depth-limited search
  - Instead, search a limited depth of tree
  - Replace terminal utilities with an eval function for non-terminal positions
- Guarantee of optimal play is gone
- Replanning agents:
  - Search to choose next action
  - Replan each new turn in response to new state

Evaluation for Pacman

\[ Eval(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s) \]

Iterative Deepening

Iterative deepening uses DFS as a subroutine:
1. Do a DFS which only searches for paths of length 1 or less. (DFS gives up on any path of length 2)
2. If "1" failed, do a DFS which only searches paths of length 2 or less.
3. If "2" failed, do a DFS which only searches paths of length 3 or less.
   …and so on.

This works for single-agent search as well!

Why do we want to do this for multiplayer games?

α-β Pruning Example

α-β Pruning

- General configuration
  - \( \alpha \) is the best value that MAX can get at any choice point along the current path
  - If \( n \) becomes worse than \( \alpha \), MAX will avoid it, so can stop considering \( n \)'s other children
  - Define \( \beta \) similarly for MIN
**α-β Pruning Pseudocode**

```plaintext
function MAX-VALUE(state) returns a utility value
if TERMINAL-TEST(state) then return UTILITY(state)
for a ∈ SUCCESSORS(state) do v ← MAX(a, MIN-VALUE(a))
return v

function MIN-VALUE(state, a, β) returns a utility value
input: state, current state in game
a, the value of the best alternative for MAX along the path to state
β, the value of the best alternative for MIN along the path to state
if TERMINAL-TEST(state) then return UTILITY(state)
for a ∈ SUCCESSORS(state) do v ← MAX(a, MIN-VALUE(a, a, β))
if v ≥ β then return v
α ← MAX(a, β)
return v
```

**α-β Pruning Properties**

- Pruning has no effect on final result
- Good move ordering improves effectiveness of pruning
- With “perfect ordering”:
  - Time complexity drops to O(b^m/2)
  - Doubles solvable depth
  - Full search of, e.g., chess, is still hopeless!
- A simple example of metareasoning, here reasoning about which computations are relevant

**Expectimax Search Trees**

- What if we don’t know what the result of an action will be? E.g.,
  - In solitaire, next card is unknown
  - In minesweeper, mine locations
  - In pacman, the ghosts act randomly
- Can do expectimax search
  - Chance nodes, like min nodes, except the outcome is uncertain
  - Calculate expected utilities
  - Max nodes as in minimax search
  - Chance nodes take average (expectation) of value of children
- Later, we’ll learn how to formalize the underlying problem as a Markov Decision Process

**Maximum Expected Utility**

- Why should we average utilities? Why not minimax?
- Principle of maximum expected utility: an agent should choose the action which maximizes its expected utility, given its knowledge
- General principle for decision making
- Often taken as the definition of rationality
- We’ll see this idea over and over in this course!
- Let’s decompress this definition…

**Reminder: Probabilities**

- A random variable represents an event whose outcome is unknown
- A probability distribution is an assignment of weights to outcomes
- Example: traffic on freeway?
  - Random variable: T = whether there’s traffic
  - Outcomes: T in {none, light, heavy}
  - Distribution: P(T=none) = 0.25, P(T=light) = 0.55, P(T=heavy) = 0.20
- Some laws of probability (more later):
  - Probabilities are always non-negative
  - Probabilities over all possible outcomes sum to one
- As we get more evidence, probabilities may change:
  - P(T=heavy) = 0.20, P(T=heavy | Hour=3am) = 0.60
  - We’ll talk about methods for reasoning and updating probabilities later

**What are Probabilities?**

- Objectivist / frequentist answer:
  - Averages over repeated experiments
  - E.g., empirically estimating P(rain) from historical observation
  - Assertion about how future experiments will go (in the limit)
  - New evidence changes the reference class
  - Makes one think of inherently random events, like rolling dice
- Subjectivist / Bayesian answer:
  - Degrees of belief about unobserved variables
  - E.g., an agent’s belief that it’s raining, given the temperature
  - E.g., pacman’s belief that the ghost will turn left, given the state
  - Often learn probabilities from past experiences (more later)
  - New evidence updates beliefs (more later)
Uncertainty Everywhere

• Not just for games of chance!
  • I’m snuffling: am I sick?
  • Email contains “FREE!” is it spam?
  • Tooth hurts: have cavity?
  • 60 min enough to get to the airport?
  • Robot rotated wheel three times, how far did it advance?
  • Safe to cross street? (Look both ways!)

• Why can a random variable have uncertainty?
  • Inherently random process (dice, etc)
  • Insufficient or weak evidence
  • Ignorance of underlying processes
  • Unmodeled variables
  • The world’s just noisy!

• Compare to fuzzy logic, which has degrees of truth, or rather than just degrees of belief

Reminder: Expectations

• Often a quantity of interest depends on a random variable

• The expected value of a function is its average output, weighted by a given distribution over inputs

• Example: How late if I leave 60 min before my flight?
  • Lateness is a function of traffic:
    L(none) = -10, L(light) = -5, L(heavy) = 15
  • What is my expected lateness?
    • Need to specify some belief over T to weight the outcomes
    • Say P(T) = {none: 2/5, light: 2/5, heavy: 1/5}
    • The expected lateness:

Expectations

• Real valued functions of random variables:
  \[ f : X \rightarrow R \]

• Expectation of a function of a random variable
  \[ E_{P(X)}[f(X)] = \sum_X f(x) P(x) \]

• Example: Expected value of a fair die roll

<table>
<thead>
<tr>
<th>X</th>
<th>( f(x) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( \frac{1}{6} )</td>
</tr>
<tr>
<td>2</td>
<td>( \frac{1}{6} )</td>
</tr>
<tr>
<td>3</td>
<td>( \frac{1}{6} )</td>
</tr>
<tr>
<td>4</td>
<td>( \frac{1}{6} )</td>
</tr>
<tr>
<td>5</td>
<td>( \frac{1}{6} )</td>
</tr>
<tr>
<td>6</td>
<td>( \frac{1}{6} )</td>
</tr>
</tbody>
</table>

= 3.5

Utilities

• Utilities are functions from outcomes (states of the world) to real numbers that describe an agent’s preferences

• Where do utilities come from?
  • In a game, may be simple (+1/-1)
  • Utilities summarise the agent's goals

• Theorem: any set of preferences between outcomes can be summarized as a utility function (provided the preferences meet certain conditions)

• In general, we hard-wire utilities and let actions emerge (why don’t we let agents decide their own utilities?)

• More on utilities soon...

Expectimax Search

• In expectimax search, we have a probabilistic model of how the opponent (or environment) will behave in any state
  • Model could be a simple uniform distribution (roll a die)
  • Model could be sophisticated and require a great deal of computation
  • We have a node for every outcome out of our control, opponent or environment
  • The model might say that adversarial actions are likely!

• For now, assume for any state we magically have a distribution to assign probabilities to opponent actions / environment outcomes

Expectimax Pseudocode

```
def value(s):
    if s is a max node return maxValue(s)
    if s is an exp node return expValue(s)
    if s is a terminal node return evaluation(s)

def maxValue(s):
    values = [value(s') for s' in successors(s)]
    return max(values)

def expValue(s):
    values = [value(s') for s' in successors(s)]
    weights = [probability(s, s') for s' in successors(s)]
    return expectation(values, weights)
```

Having a probabilistic belief about an agent’s action does not mean that agent is flipping any coins!
**Expectimax for Pacman**

- Notice that we’ve gotten away from thinking that the ghosts are trying to minimize pacman’s score.
- Instead, they are now a part of the environment.
- Pacman has a belief (distribution) over how they will act.
- Quiz: Can we see minimax as a special case of expectimax?
- Quiz: what would pacman’s computation look like if we assumed that the ghosts were doing 1- ply minimax and taking the result 80% of the time, otherwise moving randomly?
- If you take this further, you end up calculating belief distributions over your opponents belief distributions over your belief distributions, etc…
  - Can get unmanageable very quickly!

---

**Expectimax Evaluation**

- For minimax search, evaluation function insensitive to monotonic transformations
  - We just want better states to have higher evaluations (get the ordering right)

- For expectimax, we need the magnitudes to be meaningful as well
  - E.g. must know whether a 50%/50% lottery between A and B is better than 100% chance of C
  - 100 or -10 vs 0 is different than 10 or -100 vs 0

---

**Mixed Layer Types**

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

If state is a MAX node then return the highest \( \text{EXPECTIMINIMAX-VALUE OF SUCCESSORS}(\text{state}) \)
If state is a MEX node then return the lowest \( \text{EXPECTIMINIMAX-VALUE OF SUCCESSORS}(\text{state}) \)
If state is a chance node then return average of \( \text{EXPECTIMINIMAX-VALUE OF SUCCESSORS}(\text{state}) \)

---

**Stochastic Two-Player**

- Dice rolls increase b: 21 possible rolls with 2 dice
  - Backgammon = 20 legal moves
  - Depth 4 = \( 20 \times (21 \times 20)^3 \approx 1.2 \times 10^9 \)
  - As depth increases, probability of reaching a given node shrinks
    - So value of lookahead is diminished
    - So limiting depth is less damaging
    - But pruning is less possible...
  - TDGammon uses depth-2 search + very good eval function + reinforcement learning: world-champion level play

---

**Non-Zero-Sum Games**

- Similar to minimax:
  - Utilities are now tuples
  - Each player maximizes their own entry at each node
  - Propagate (or back up) nodes from children
  - Can give rise to cooperation and competition dynamically…

---
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
  - \( A \preceq B \): B not preferred over A

Rational Preferences

- Preferences of a rational agent must obey constraints.
- The axioms of rationality:
  - 
    \[ (A \succ B) \lor (B \succ A) \lor (A \sim B) \]
  - Transitivity
    \[ (A \succ B) \land (B \succ C) \Rightarrow (A \succ C) \]
  - Continuity
    \[ A \succ B \succ C \Rightarrow \exists \rho \ni [p, A; (1 - p), C] \sim B \]
  - Substitutability
    \[ A \sim B \Rightarrow \exists \rho \ni [p, A; (1 - p), C] \sim [p, B; (1 - p), C] \]
  - Monotonicity
    \[ A \succ B \Rightarrow (\rho \geq \sigma \Rightarrow [p, A; (1 - p), B] \succeq [p, A; 1 - q, B]) \]
- 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:
    \[ U(A) \geq U(B) \Leftrightarrow A \succeq B \]
    \[ U([p_1, S_1; \ldots ; p_n, S_n]) = \sum p_i U(S_i) \]
- 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 tictactoe, reflex vacuum cleaner

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 \([0,1]\)

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
  \[ U'(x) = k_1 U(x) + k_2 \quad \text{where} \quad k_1 > 0 \]
  - With deterministic prizes only (no lottery choices), only ordinal utility can be determined, i.e., total order on prizes
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-prone, no insurance needed!

Money

- Money does not behave as a utility function
  - Given a lottery L:
    - Define expected monetary value EMV(L)
    - Usually $U(L) < U(EMV(L))$
    - i.e., people are risk-averse
  - Utility curve: for what probability p am I indifferent between:
    - A prize x
    - A lottery [p,$M; (1-p),$0] for large M?
  - Typical empirical data, extrapolated with risk-prone behavior:

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)