343H: Honors AI

Lecture 7: Expectimax Search

2/6/2014

Kristen Grauman

UT-Austin

Slides courtesy of Dan Klein, UC-Berkeley

Unless otherwise noted

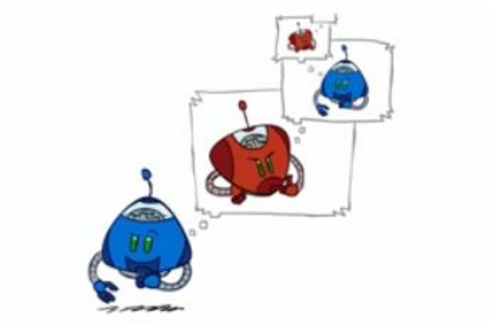
Announcements

- PS1 is out, due in 2 weeks

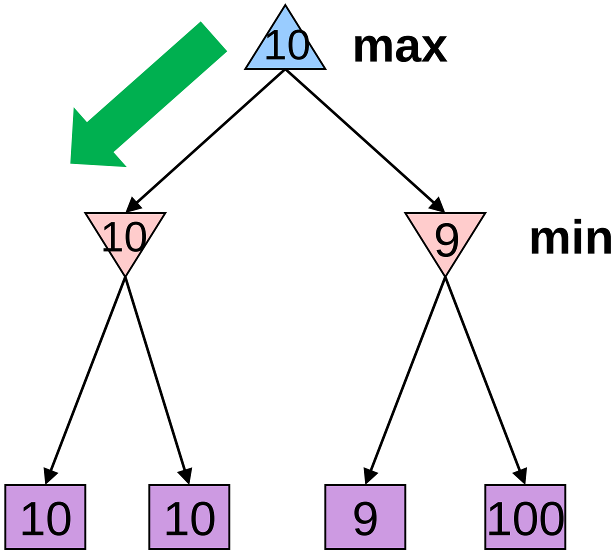
Last time

- Adversarial search with game trees
  - Minimax
  - Alpha-beta pruning

Key ideas



- Now we have an adversarial opponent, must reason about impact of their actions when computing value of a state
- Game trees interleave “MIN” nodes
- Minimax algorithm to select optimal action
- Alpha-beta pruning to avoid exploring entire tree
- Evaluation function + cutoff test (or iterative deepening) to deal with resource limits.

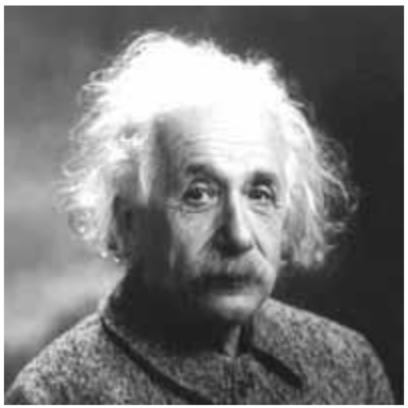


Today

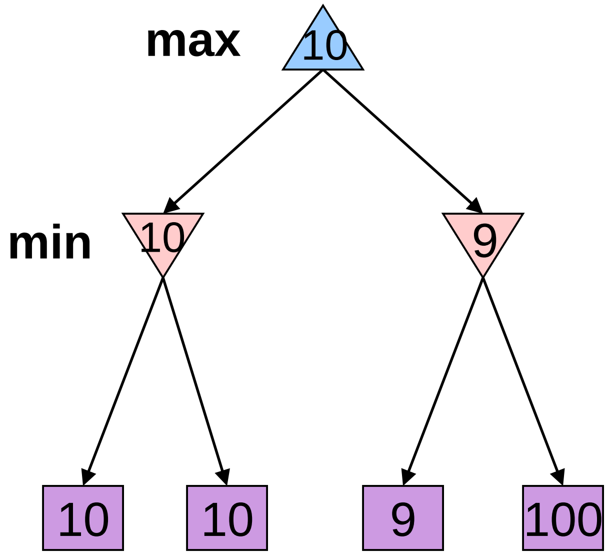
- Search in the presence of uncertainty

Worst-case vs. Average-case

But what about…



Optimal against a perfect player.



Imperfect adversaries

Factors of chance

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 = traffic level
  - Outcomes: T in {none, light, heavy}
  - Distribution: P(T=none) = 0.25, P(T=light) = 0.50, P(T=heavy) = 0.25



- 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=8am) = 0.60
  - We'll talk about methods for reasoning and updating probabilities later

Reminder: Expectations

- The expected value of a function is its average value, weighted by the probability distribution over inputs

- Example: How long to get to the airport?
  - Length of driving time as a function of traffic:

    L(none) = 20, L(light) = 30, L(heavy) = 60 min

E[L(T)] = L(none)*P(none) + L(light)*P(light) + L(heavy)*P(heavy)

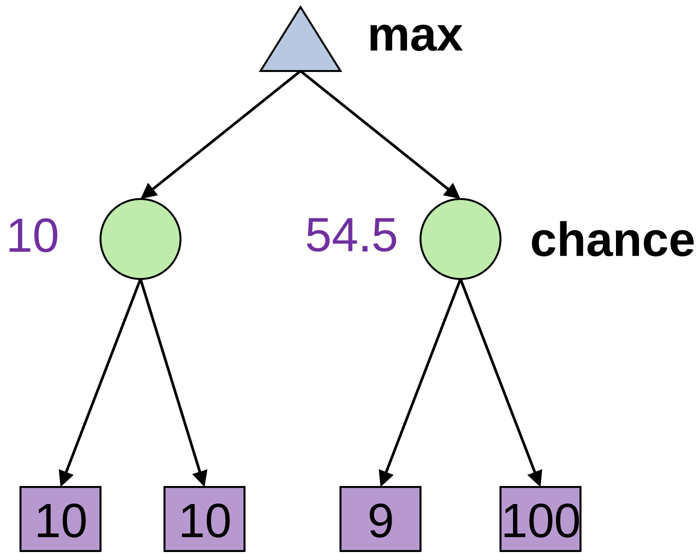
E[L(T)] = (20 * 0.25) + (30 * 0.5) + (60 * 0.25) = 35 minutes

Expectimax search

- Why wouldn't we know what the result of an action will be?
  - Explicit randomness: rolling dice
  - Unpredictable opponents: ghosts respond randomly
  - Actions can fail: when moving a robot, wheels could slip

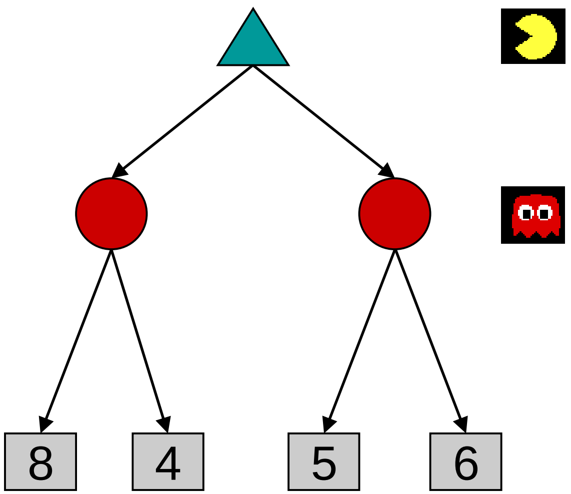
- Values should now reflect average-case outcomes, not worst-case (minimax) outcomes

- **Expectimax search:** compute average score under optimal play
  - Max nodes as in minimax search
  - Chance nodes, like min nodes, except the outcome is uncertain
  - Calculate expected utilities
  - I.e. take weighted average (expectation) of values of children



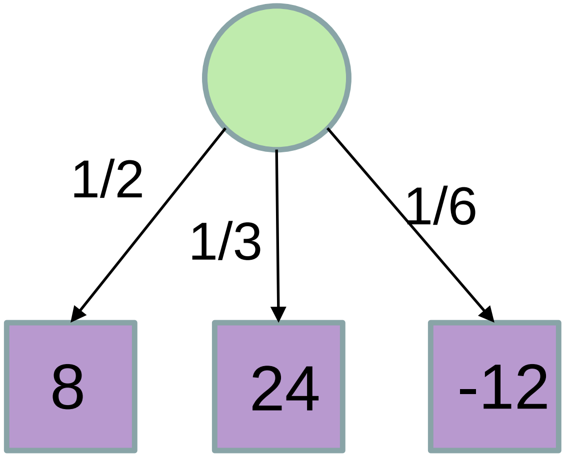
Expectimax Pseudocode

```
def value(s)
    if s is a terminal node return utility(s)
    if s is a max node return maxValue(s)
    if s is an exp node return expValue(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') for s' in successors(s)]
    return expectation(values, weights)
```



Expectimax: computing expectations

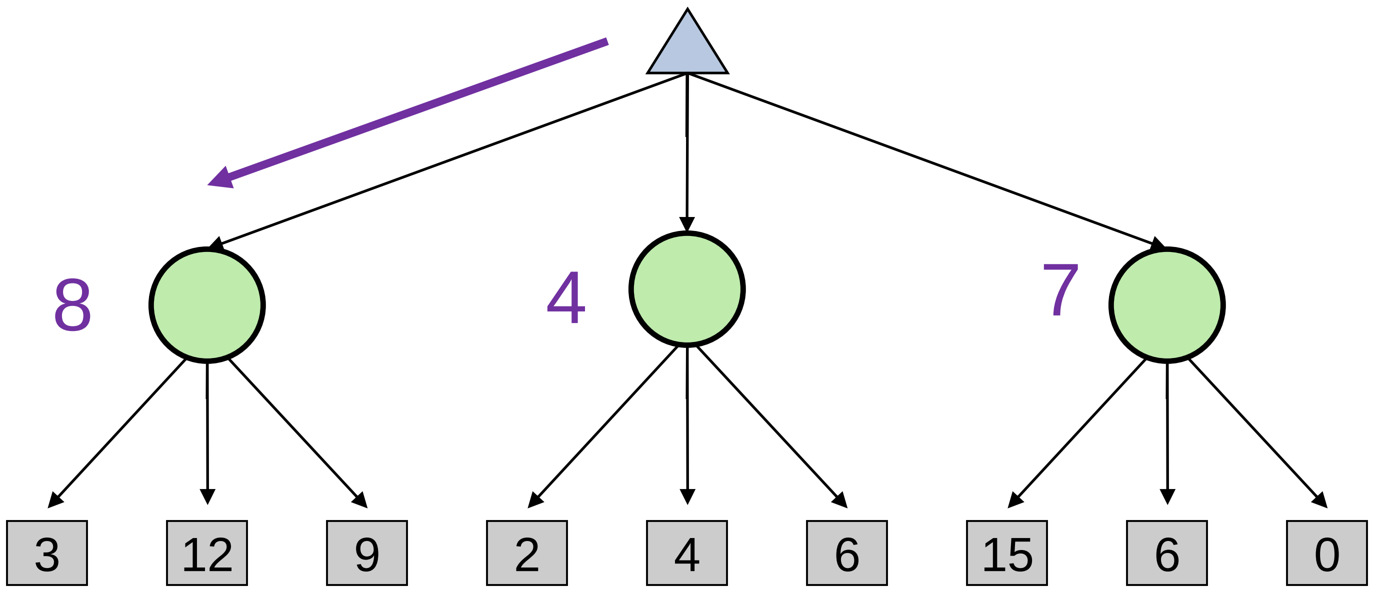
```
def exp-value(state):
   initialize v=0
   for each successor of state:
         p = probability(successor)
         v += p * value(successor)
   return v
```



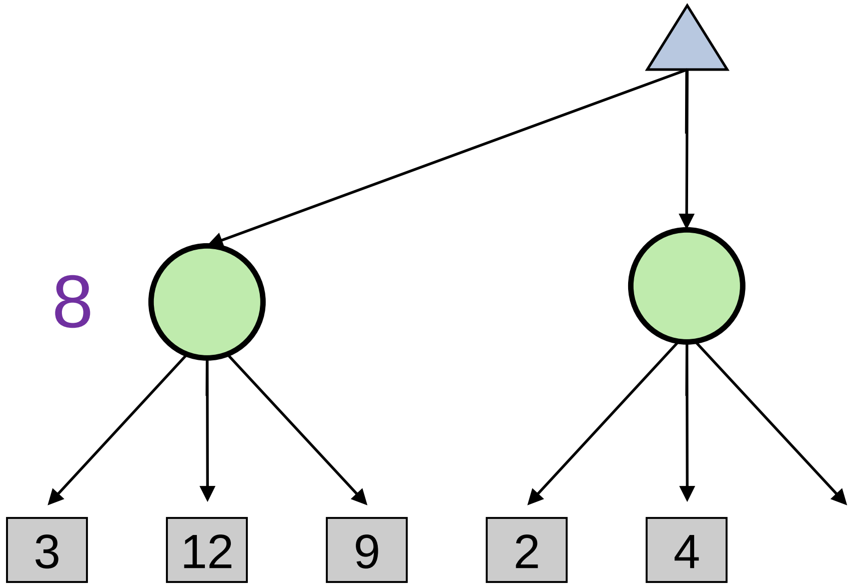
$$v = (1/2)(8) + (1/3)(24) + (1/6)(-12) = 10$$

Expectimax Example

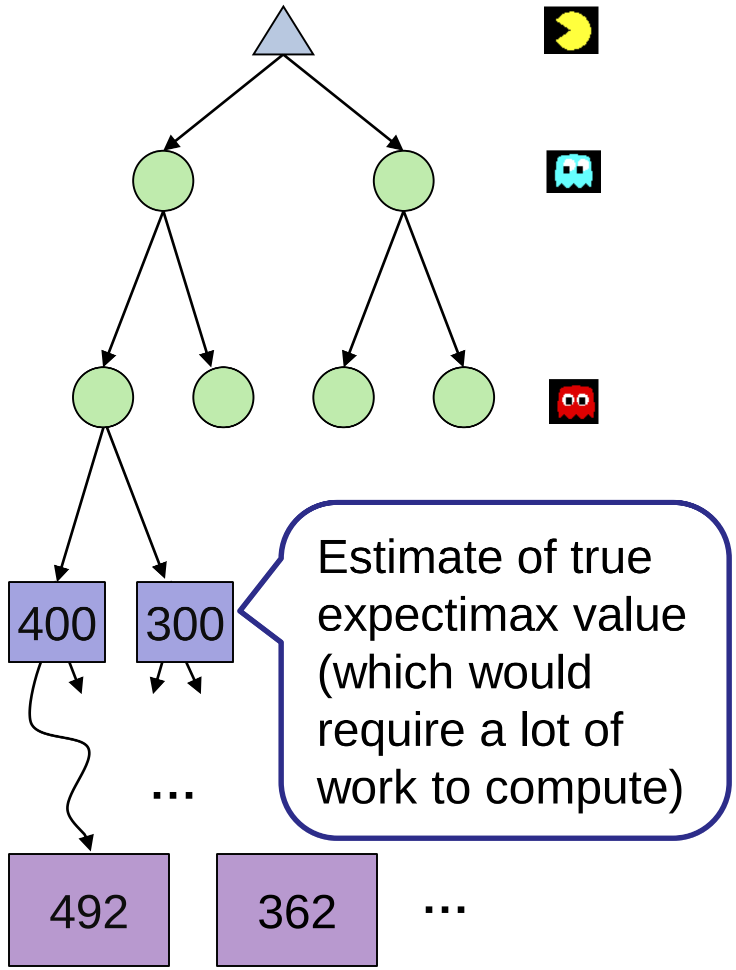
Suppose all children are equally likely



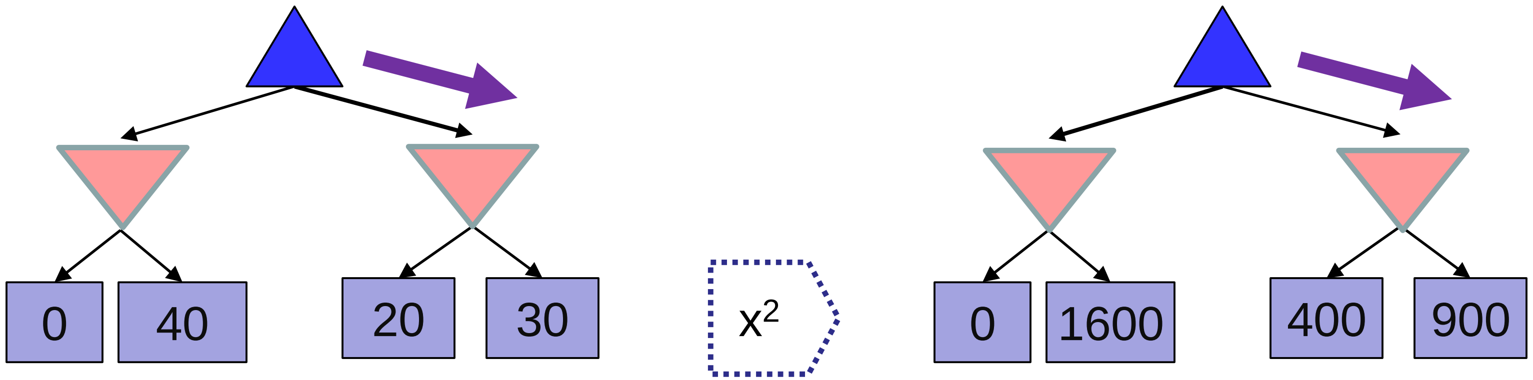
Expectimax Pruning?



Depth-Limited Expectimax

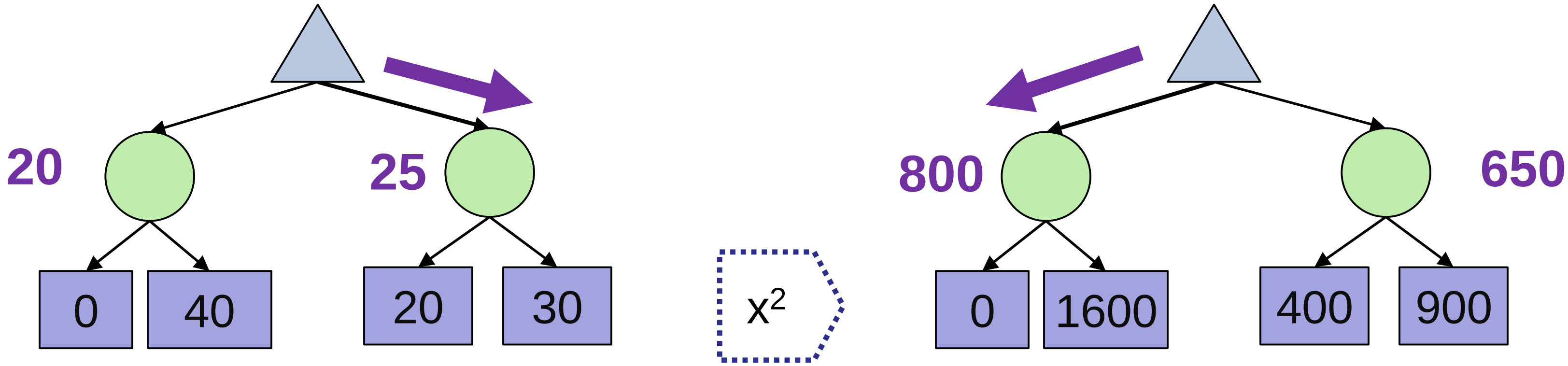


What Utilities to Use?



- For **minimax**, terminal 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

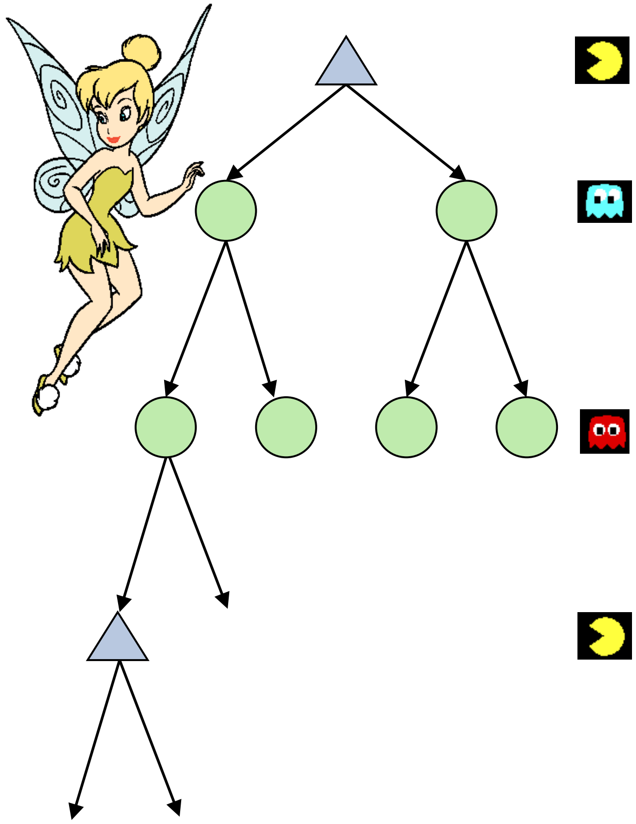
What Utilities to Use?



- For **expectimax**, we need *magnitudes* to be meaningful

What Probabilities to Use?

- 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 chance 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



Having a probabilistic belief about an agent's action does not mean that agent is flipping any coins!

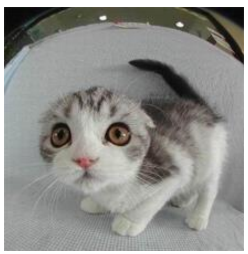
Dangers of optimism and pessimism

Dangerous optimism
Assuming chance when the world is adversarial

Dangerous pessimism
Assuming the worst case when it's not likely

Adapted from Dan Klein

World Asssumptions

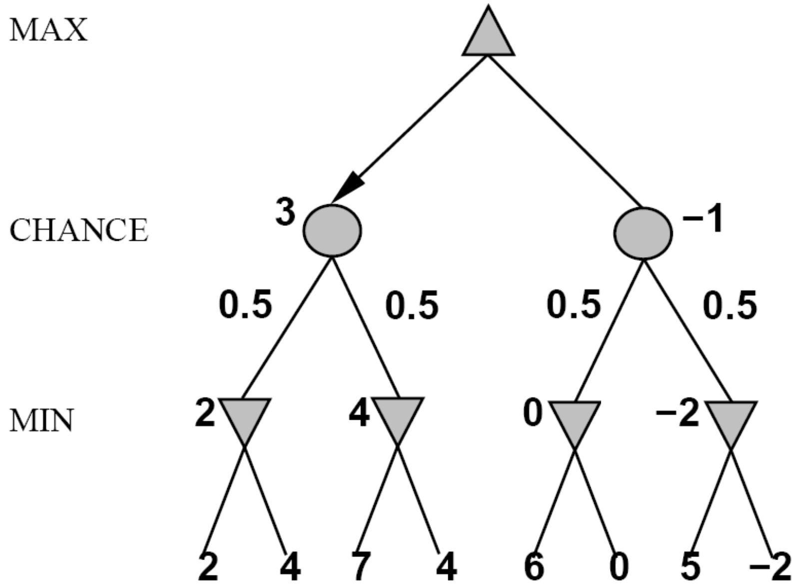



	Adversarial Ghost	Random Ghost
Minimax Pacman	Won 5/5 Avg. Score: 483	Won 5/5 Avg Score: 493
Expectimax Pacman	Won 1/5 Avg. Score: -303	Won 5/5 Avg. Score: 503

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

Mixed Layer Types

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

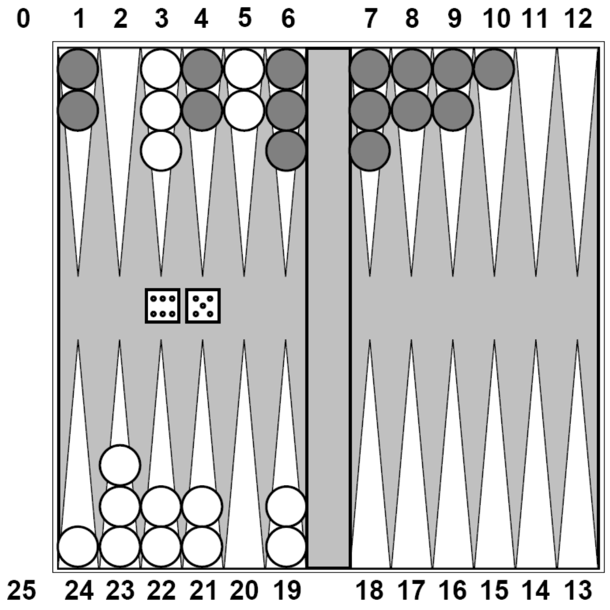


ExpectiMinimax-Value(*state*):

if *state* is a MAX node **then**
  return the highest EXPECTIMINIMAX-VALUE of SUCCESSORS(*state*)
if *state* is a MIN node **then**
  return the lowest EXPECTIMINIMAX-VALUE of SUCCESSORS(*state*)
if *state* is a chance node **then**
  return average of EXPECTIMINIMAX-VALUE of SUCCESSORS(*state*)

Example: Backgammon

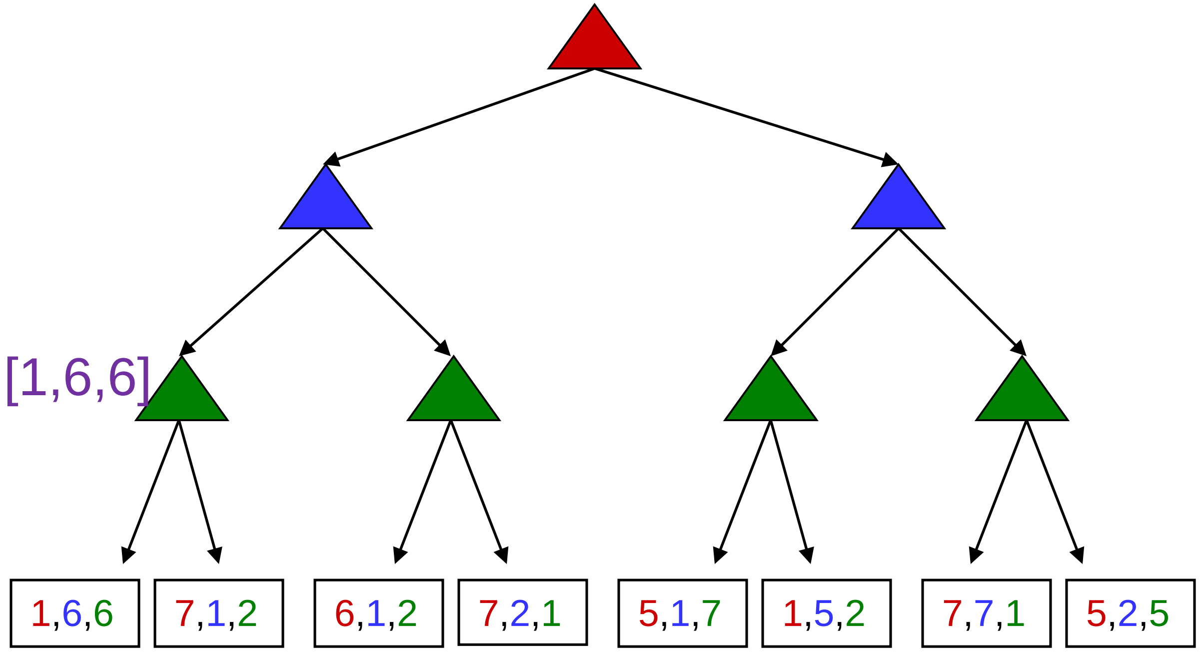
- Dice rolls increase *b*: 21 possible rolls with 2 dice
  - Backgammon ≈ 20 legal moves
  - Depth 2 = 20 x (21 x 20)^3 = 1.2 x 10^9
- As depth increases, probability of reaching a given search node shrinks
  - So usefulness of search is diminished
  - So limiting depth is less damaging
  - But pruning is trickier…
- TDGammon (1992) uses depth-2 search + very good evaluation function + reinforcement learning: world-champion level play
- 1st AI world champion in any game!



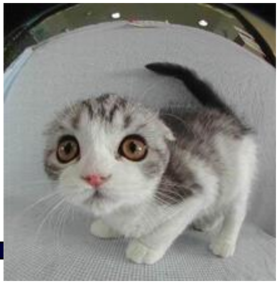
Multi-Agent Utilities

What if the game is not zero-sum, or has multiple players?

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



Maximum Expected Utility



- Why should we average utilities? Why not minimax?

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

Utilities

Kristen Grauman

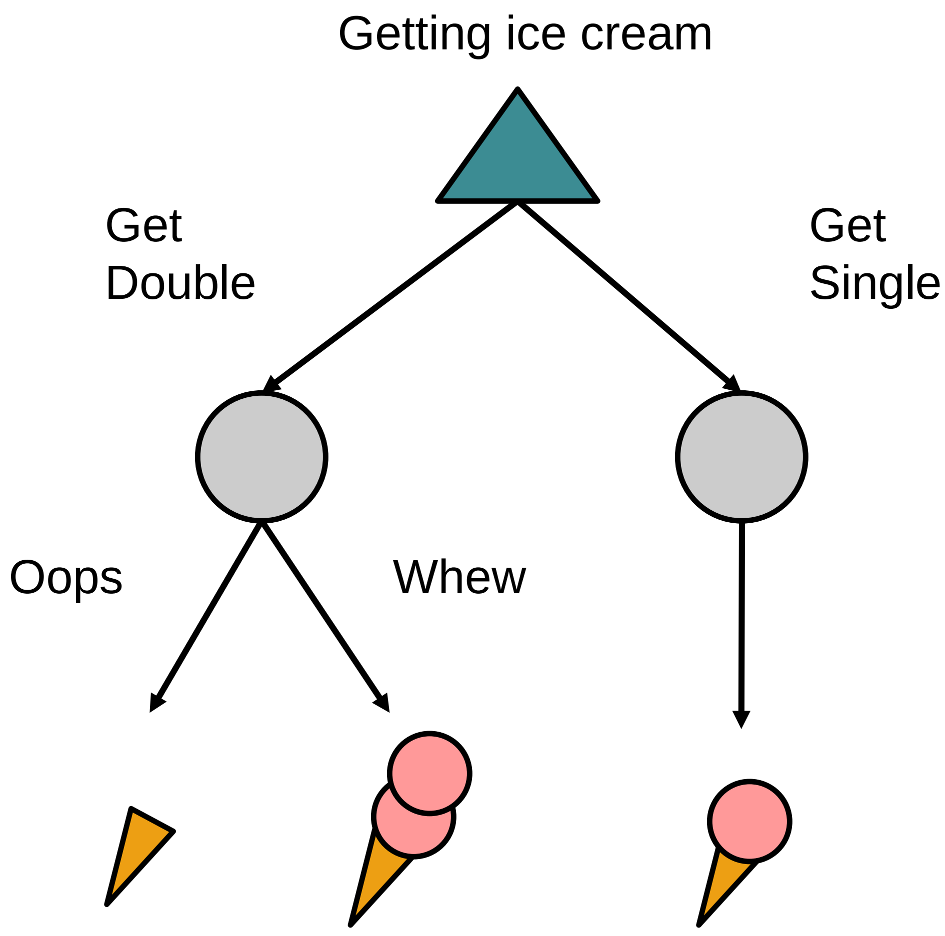
Utilities

- 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 summarize the agent's goals
  - Theorem: any "rational" preferences can be summarized as a utility function

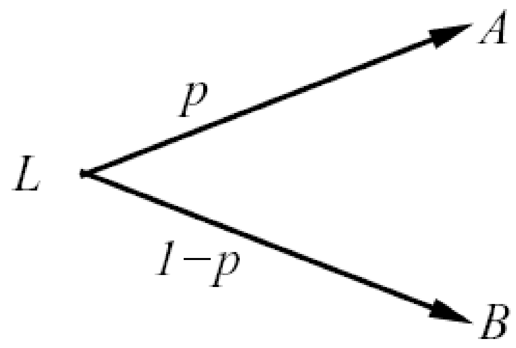
- We hard-wire utilities and let behaviors emerge
  - Why don't we let agents pick utilities?
  - Why don't we prescribe behaviors?

Utilities: Uncertain Outcomes



Preferences

- An agent must have preferences 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

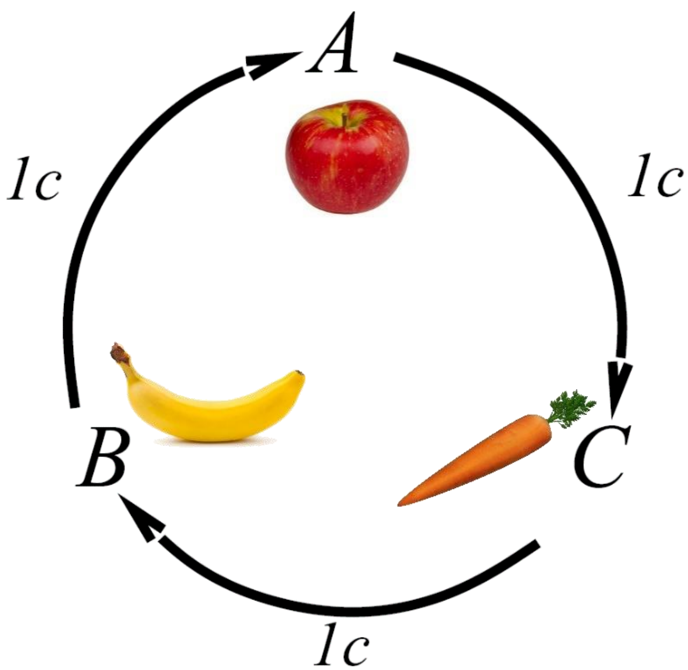
Rational Preferences

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

Axiom of transitivity

$$(A \succ B) \wedge (B \succ C) \Rightarrow (A \succ C)$$

- For example: an agent with intransitive preferences can be induced to give away all of its money
  - If B > 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



Rational Preferences

- Preferences of a rational agent must obey constraints.
  - The axioms of rationality:

Orderability

$$(A \succ B) \vee (B \succ A) \vee (A \sim B)$$

Transitivity

$$(A \succ B) \wedge (B \succ C) \Rightarrow (A \succ C)$$

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:

$$U(A) \geq U(B) \;\Leftrightarrow\; A \succeq B$$

$$U([p_1, S_1;\; \ldots\;;\; p_n, S_n]) = \sum_i p_i U(S_i)$$

  - i.e., values assigned by U preserve preferences of both prizes and lotteries!

- Maximum expected utility (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

Utility Scales, Units

- 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

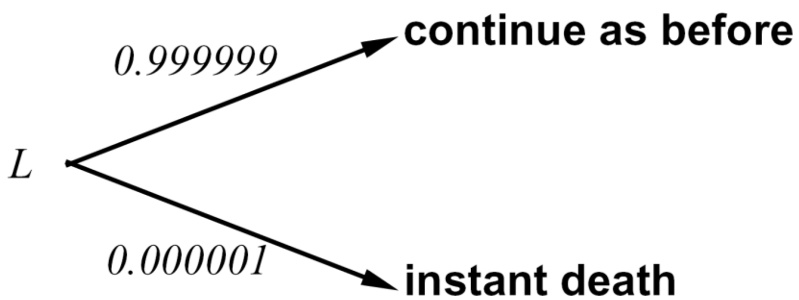
$$U'(x) = k_1 U(x) + k_2 \quad \text{where } k_1 > 0$$

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

Eliciting 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 ~ L_p
  - Resulting p is a utility in [0,1]

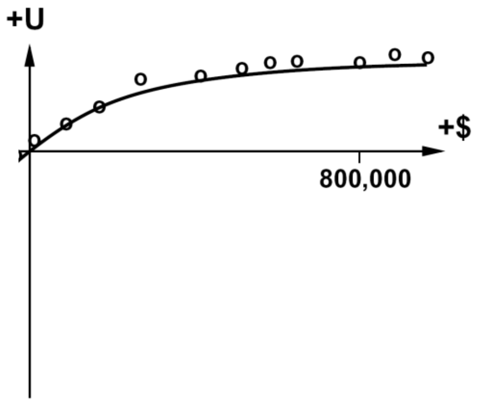
pay $30 ~



Money

- Money <u>does not</u> 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), $Y]
  - The expected monetary value EMV(L) is p*X + (1-p)*Y
  - U(L) = p*U($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

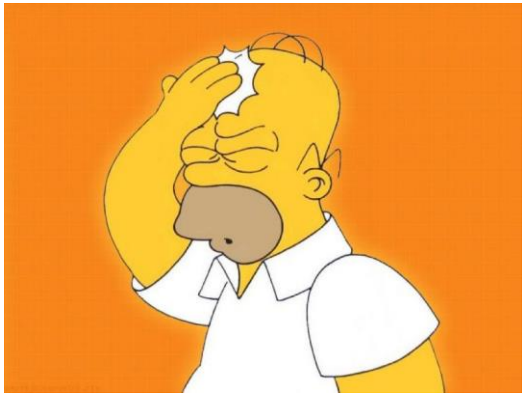
Example: Insurance

- 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: 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)

Summary

- Games with uncertainty
  - Expectimax search
  - Mixed layer and multi-agent games
  - Defining utilities
  - Rational preferences
  - Human rationality, risk, and money

- Next time: Probability