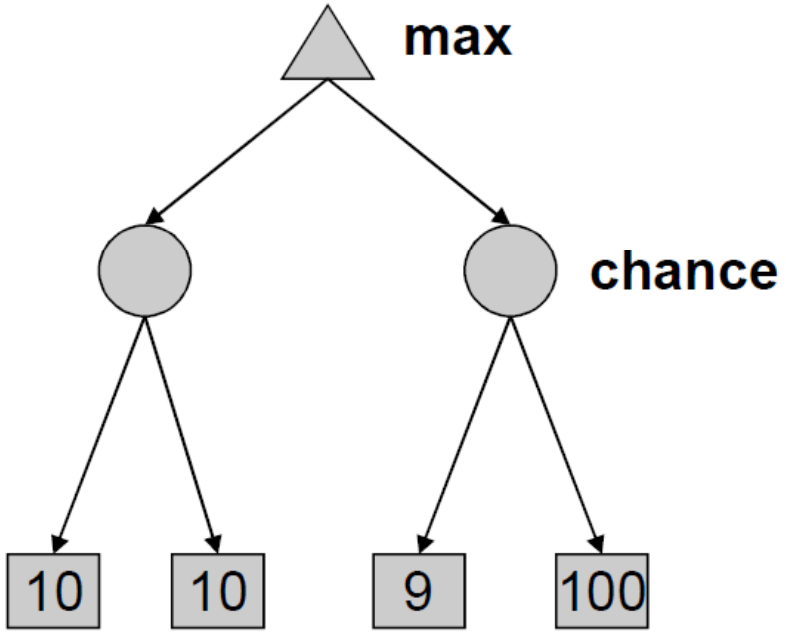
Expectimax Search

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** to maximize average score
  - 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



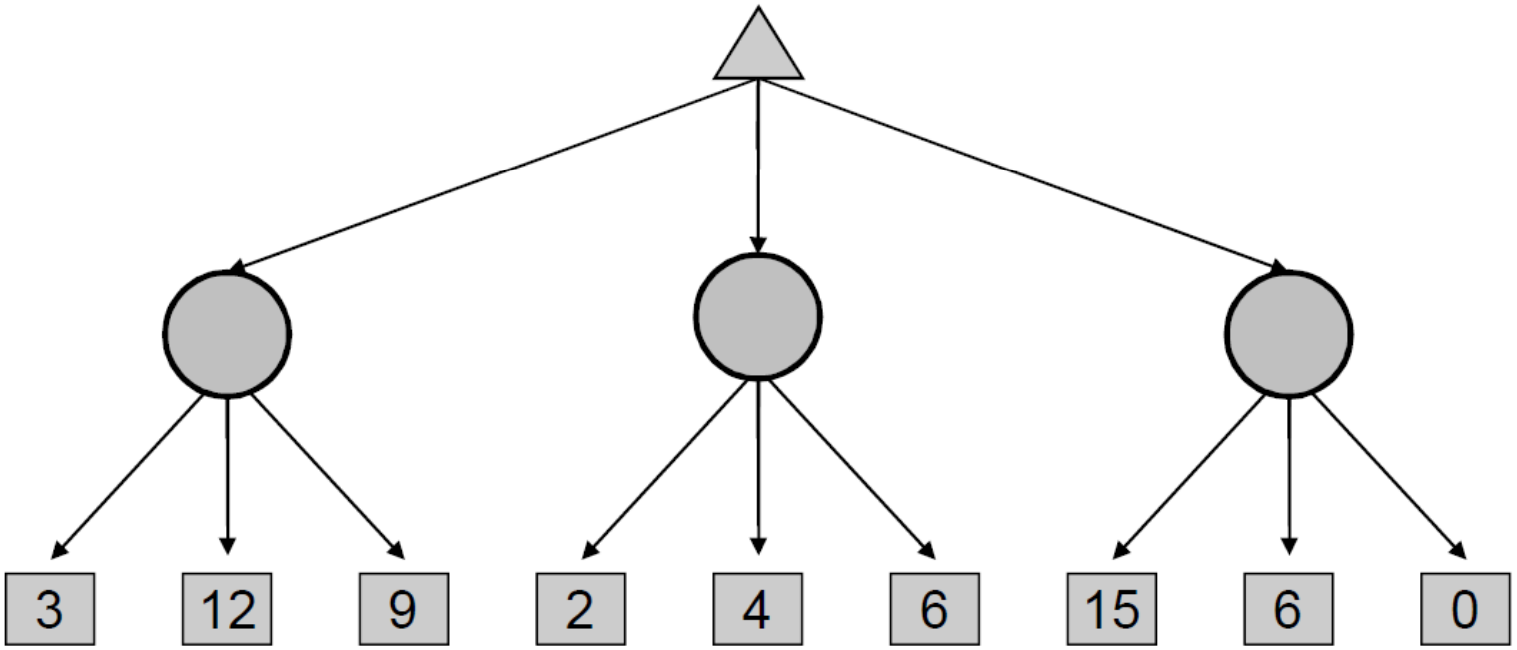
- The underlying problems are formalized as **Markov Decision Processes**

Properties of Expectimax

- The other node is not adversarial, also not in your control.
  we just do not know what is going to happen.
- States now have **Expectimax Values**
- Using expectimax is not 100% safe, you might loose.
- Usually chance nodes are governed by probabilities (instead of just selecting min) and average is weighted by those probabilities.
  - In many games the probability is uniform, flipping a coin, rolling a dice.
  - In real world probabilities governing outcome reflects our knowledge and expectations about the world problems.

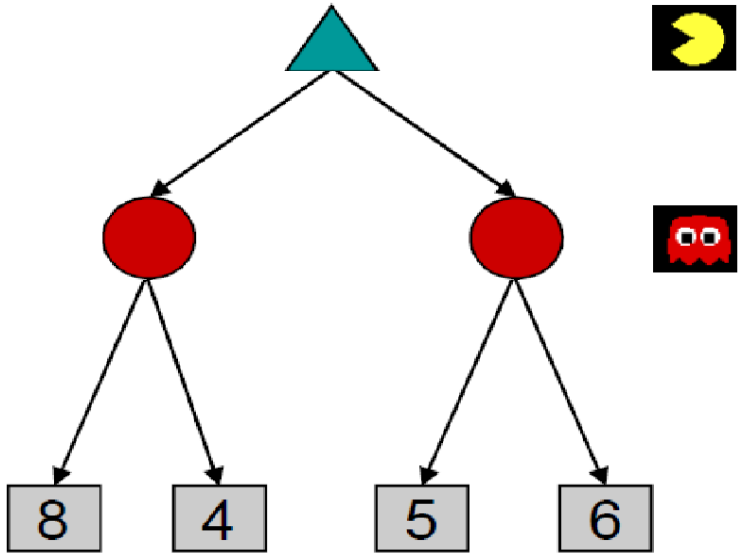
Expectimax Example

- Max (the playing agent) wants the action with maximum expected utility.
- Assume uniform probability



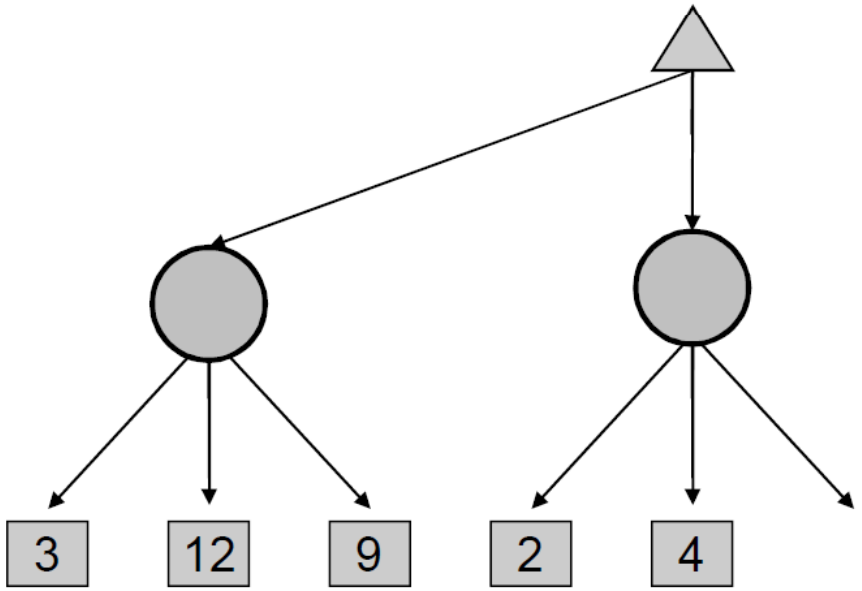
Expectimax Pseudocode

```
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)
```

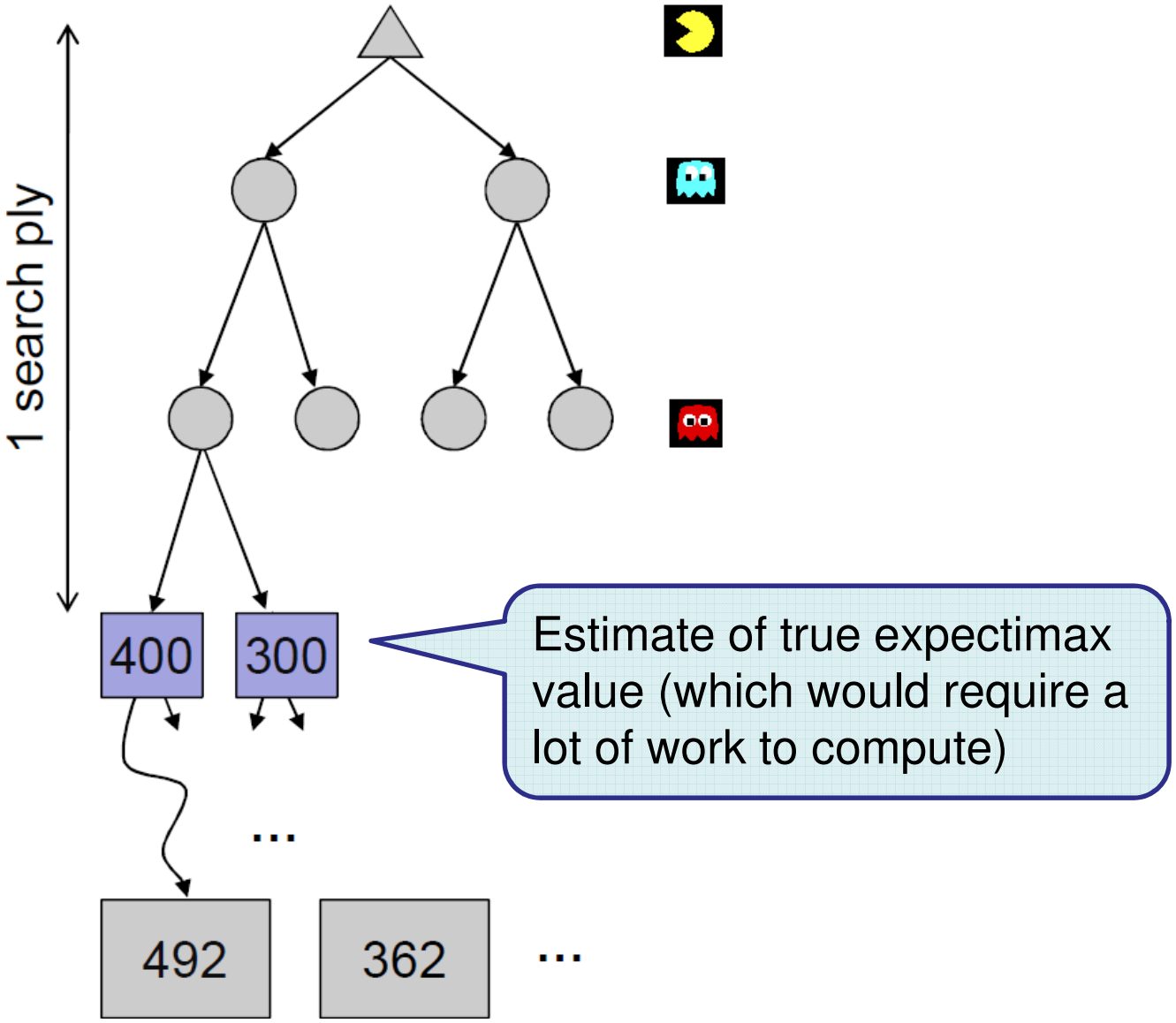


Expectimax Pruning

- There is no pruning for expectimax
  - there is no concept of "optimal play" by adversary, it is just unknown,
  - No matter what you have seen so far, the content of unexplored children could change expectimax value remarkably
- Thus, expectimax is slow
- Strategies exist to speed up



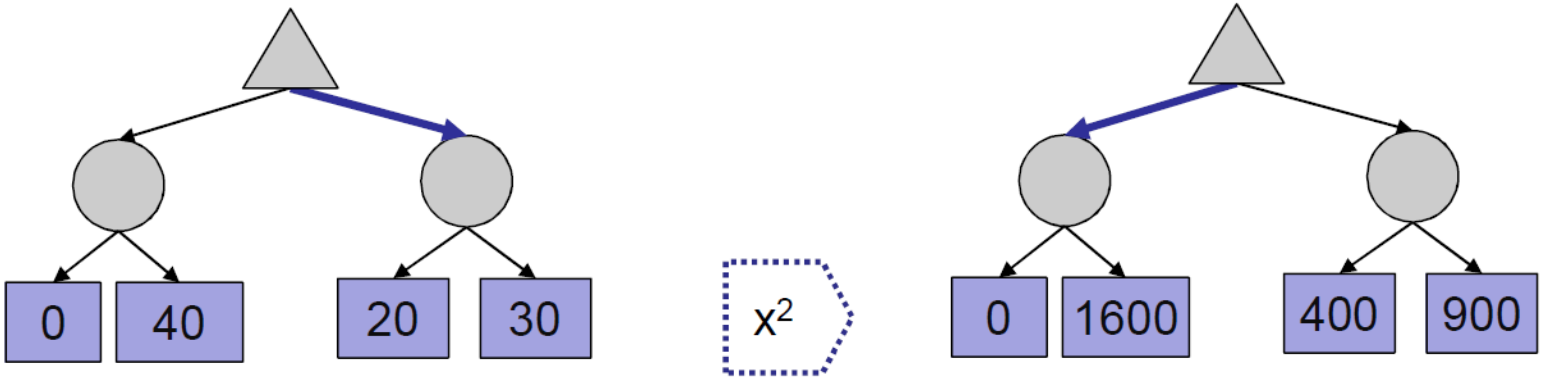
Depth-limited Expectimax



What Utilities to use?

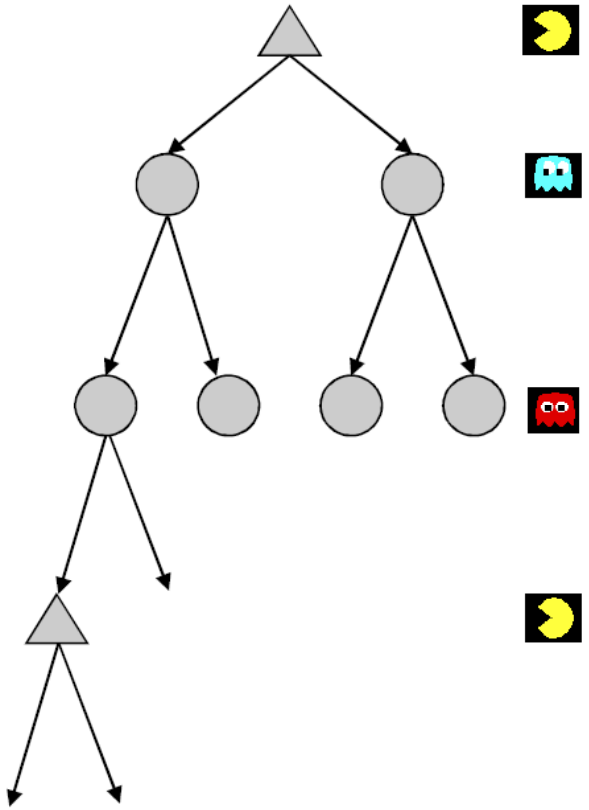
- For minimax, the scale of (terminal) utility function doesn't matter
  - We just want better states to have higher evaluations (get the ordering right)
  - We call this insensitivity to monotonic transformations

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



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 dice)
  - 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.



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

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

Reminder: Expectations

- We can define function $f(X)$ of a random variable X
- 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
  - **What is my expected driving time?**
    - Notation: $\mathcal{E}$[L(T)]
    - Remember, P(T) = {none: 0.25, light: 0.5, heavy: 0.25}
    - $\mathcal{E}$[L(T)] = L(none) * P(none) + L(light) * P(light) + L(heavy) * P(heavy)
    - $\mathcal{E}$[L(T)] = (20 * 0.25) + (30 * 0.5) + (60 * 0.25) = 35

Expectimax for Pacman

- Notice that we've gotten away from thinking that the ghosts are trying to minimize pacman's score

  Instead, the ghosts are now a part of the environment

- Pacman is the only player and try to max its score other events are happening but

  it is pacman's job to predict what other stuff will occur and what probability to assign to it

  Pacman has a belief (distribution) over how ghosts will act

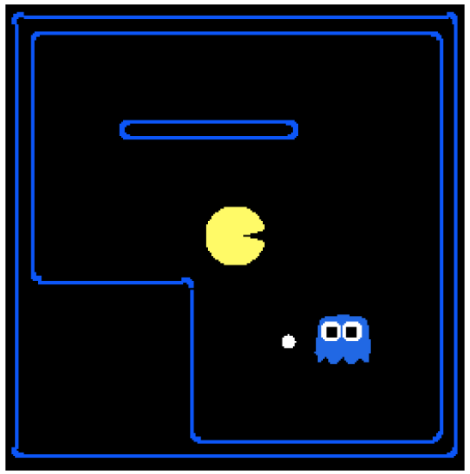
- **Quiz:** Can we see minimax as a special case of expectimax?

World Assumptions

- The type of computation performed for action selection has to match the way we think the world works.

	Minimizing 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

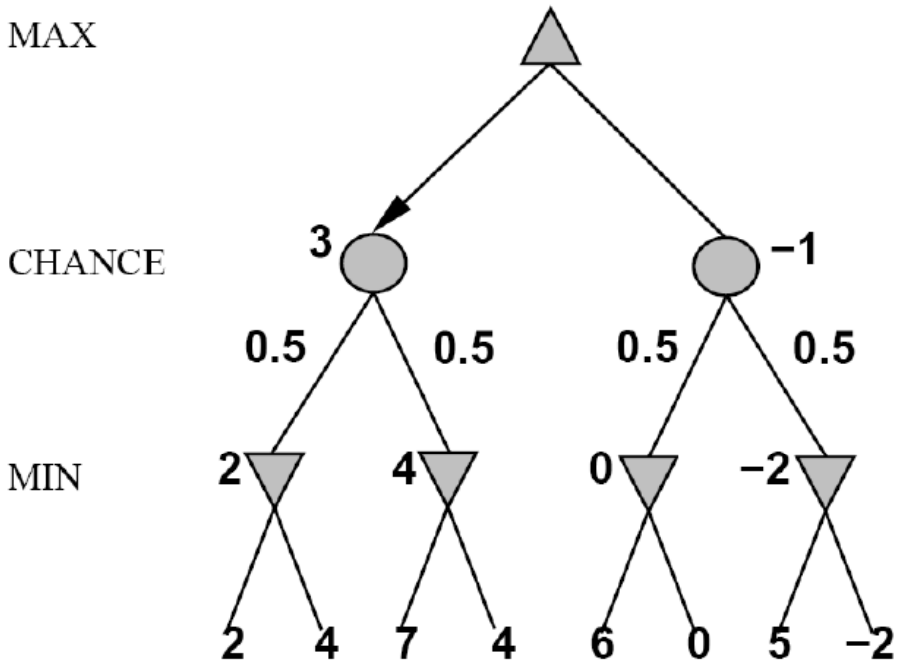
Results from playing 5 games



  - 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 (the dice) is an extra player that moves after each agent
  - Chance nodes take expectations, otherwise like minimax



ExpectiMinimax-Value(*state*):

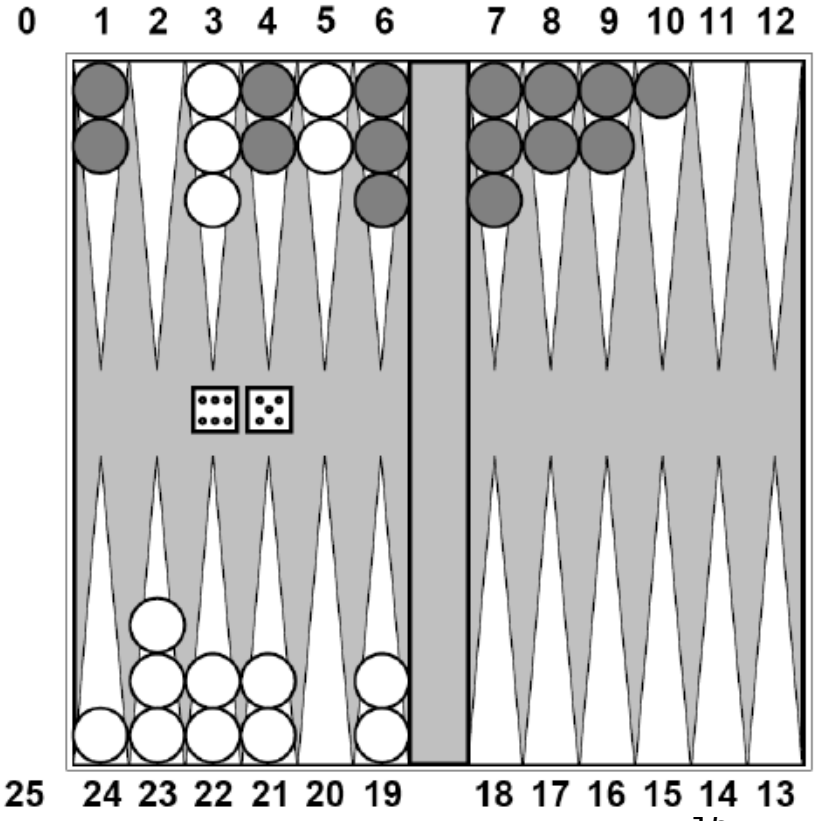
if *state* is a MAX node **then**
    return the highest EXPECTIMINIMAX-VALUE of SUCCESSORS(*state*)
if *state* is a MIN node **then**
    return the lowest EXPECTIMINIMAX-VALUE of SUCCESSORS(*state*)
if *state* is a chance node **then**
    return average of EXPECTIMINIMAX-VALUE of SUCCESSORS(*state*)

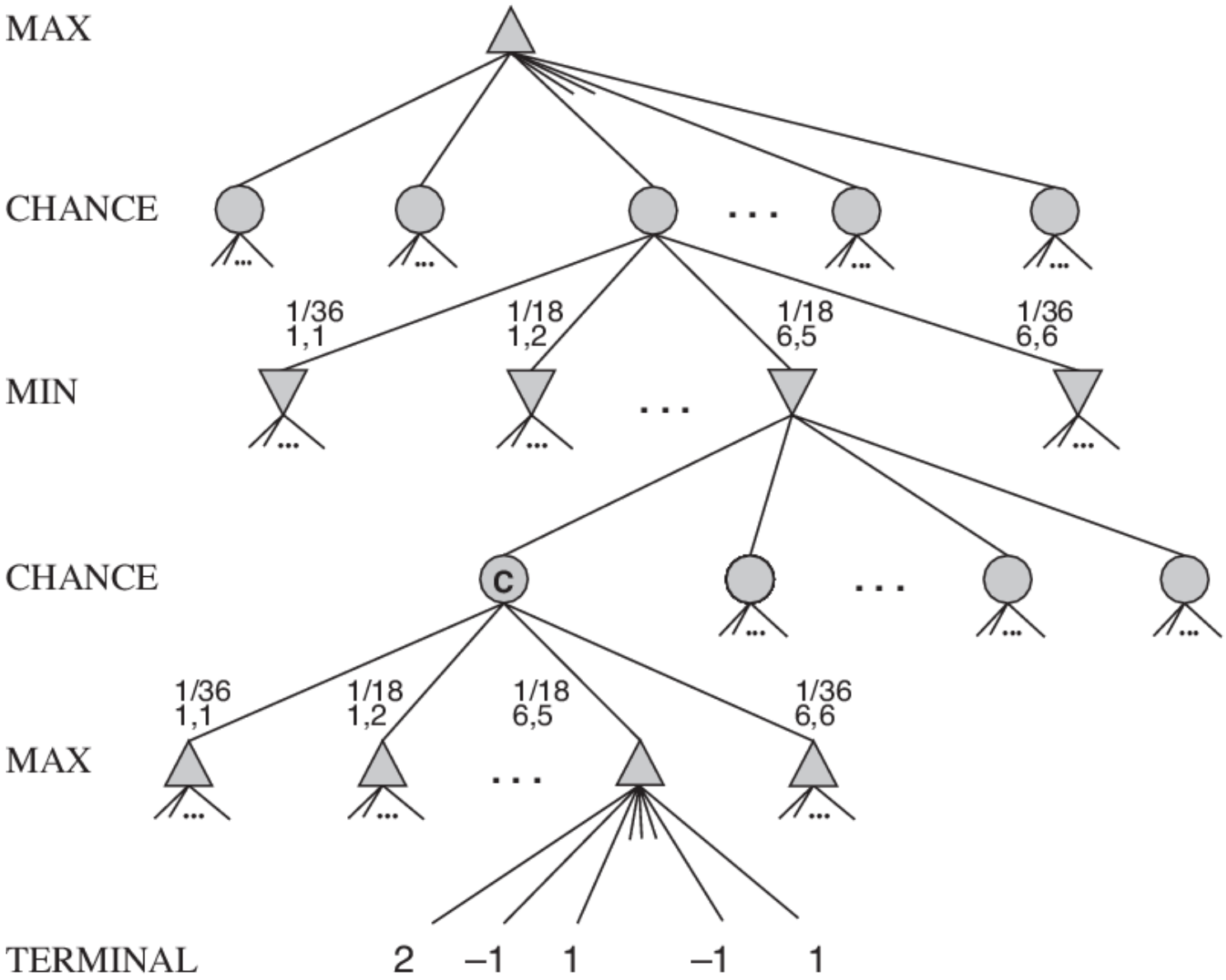
Stochastic Two-Player

- Dice rolls increase *b* to **21** possible distinct outcomes with 2 dice
  - Backgammon ≈20 legal moves (actions)
  - Depth_2 search tree has: $20 \times (21 \times 20)^3 = 1.2 \times 10^9$ nodes

- 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
- First AI world champion in any game!

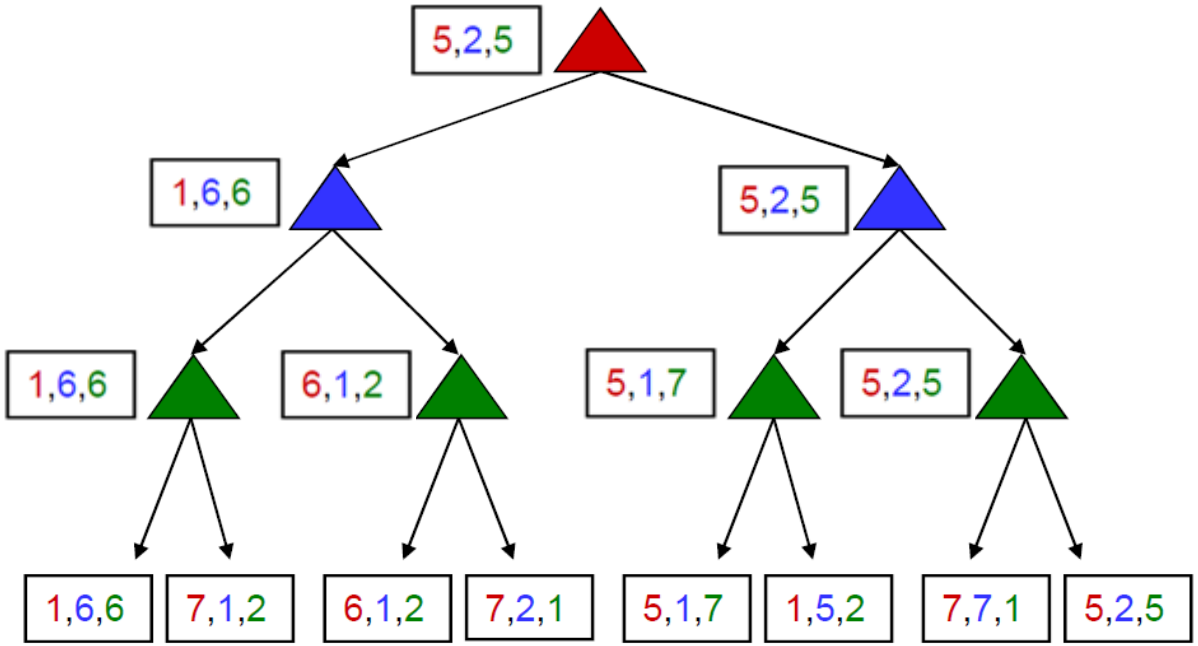



MAX
CHANCE
1/36
1,1
1/18
1,2
1/18
6,5
1/36
6,6
MIN
C
CHANCE
1/36
1,1
1/18
1,2
1/18
6,5
1/36
6,6
MAX
TERMINAL
2
–1
1
–1
1

Multi-Agent Utilities

General case, not zero-sum, similar to minimax:

- Terminals have utility vectors
- Node values are also utility vectors
- Each player maximizes its own utility in the vector
- Can give rise to cooperation and competition dynamically…



Maximum Expected Utility

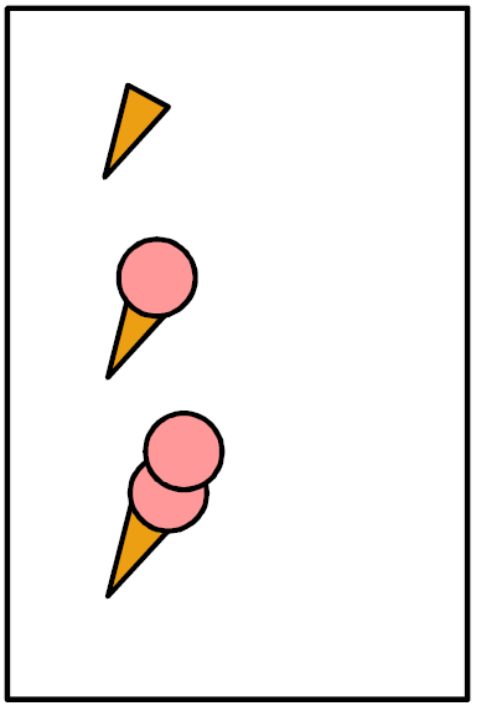
- Why should we average utilities? Why not minimax?

- Principle of maximum expected utility (MEU) :
  - 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?

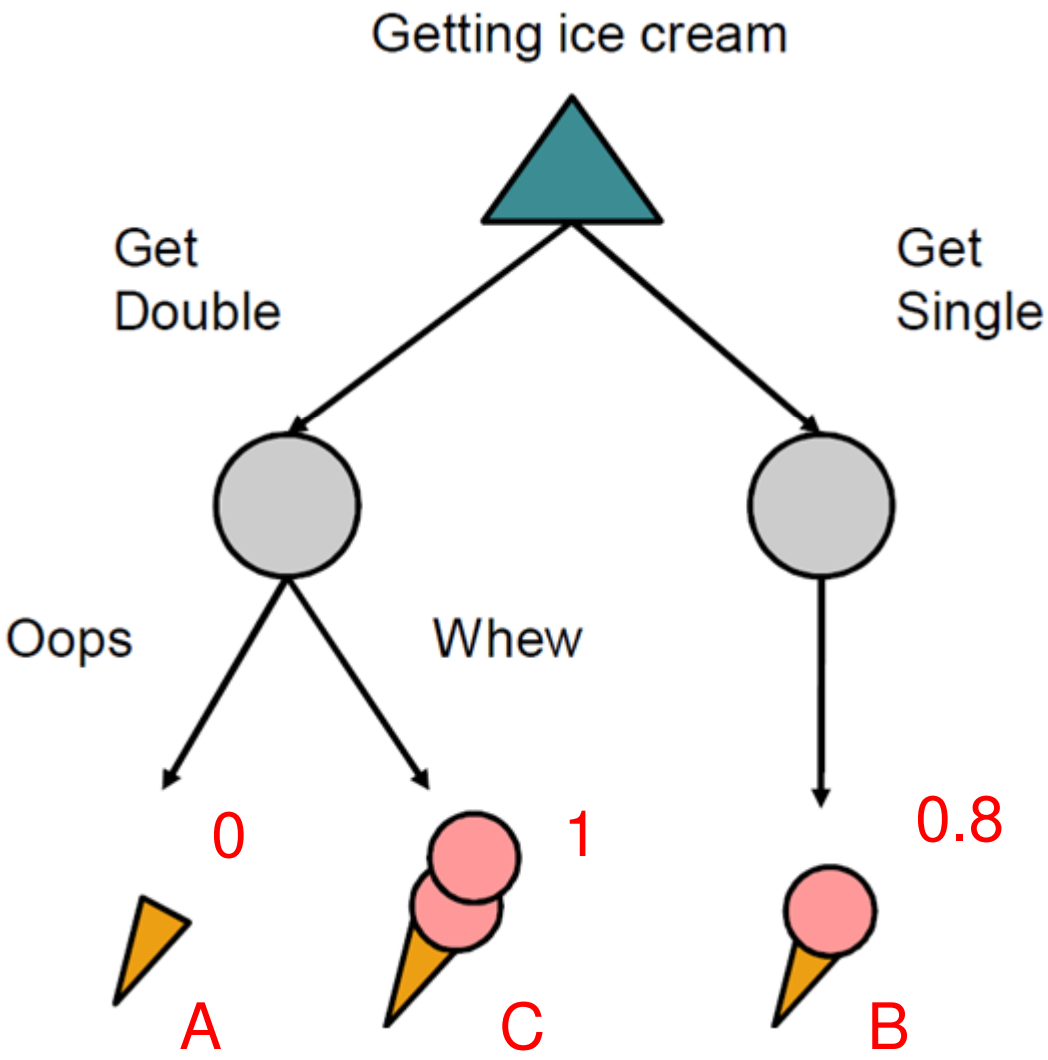
Utilities

- 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



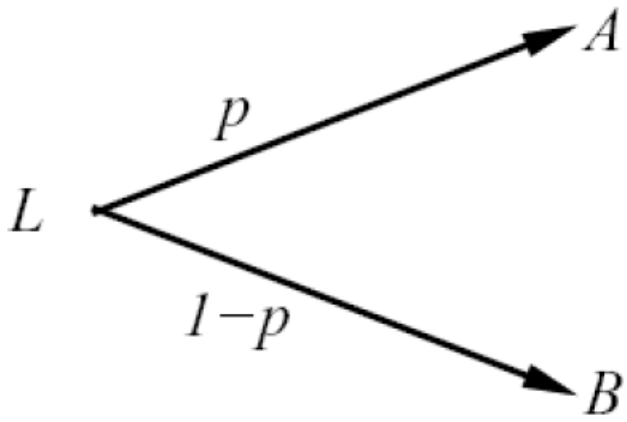
- We hard-wire utilities (based on our preferences) and let behaviors emerge
  - Why don't we let agents pick utilities?
  - Why don't we prescribe behaviors?

Utilities: Uncertain Outcomes



Preferences

- An agent must have preferences among:
  - Prizes: *A*, *B*, etc.
  - Lotteries: situations with uncertain prizes



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

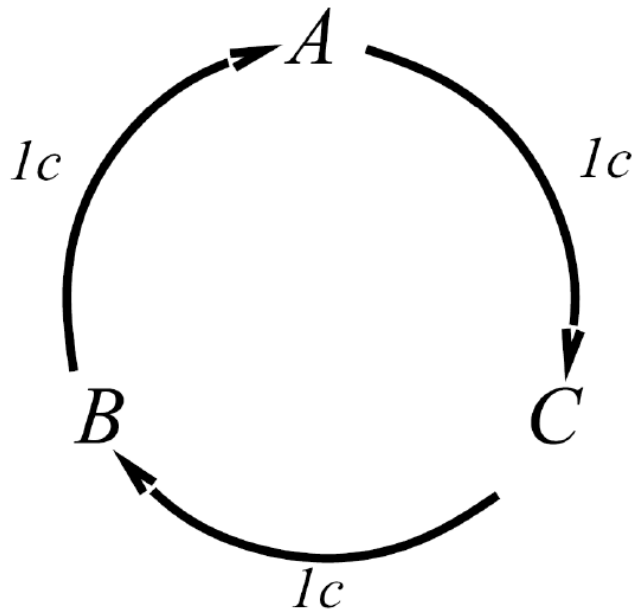
- Notation:

$A \succ B$	*A* preferred over *B*
$A \sim B$	indifference between *A* and *B*
$A \succeq B$	*B* not preferred over *A*

Rational Preferences

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

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



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:

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

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

- 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 tic-tac-toe, reflex vacuum cleaner

Utility Scales

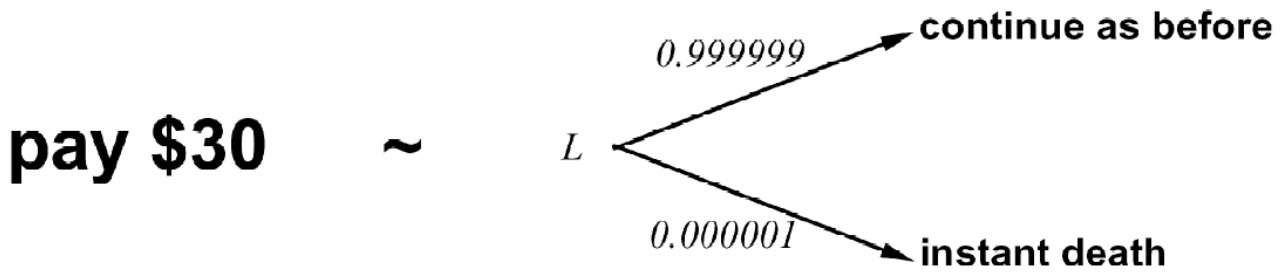
- Normalized utilities: $u_+ = 1.0$, $u_- = 0.0$
- Micromorts: one-millionth chance of death, useful for paying to reduce product risks, etc.
- QALYs: quality-adjusted life years, useful for medical decisions involving substantial risk
- **Note:** behavior is invariant under positive linear transformation

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

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

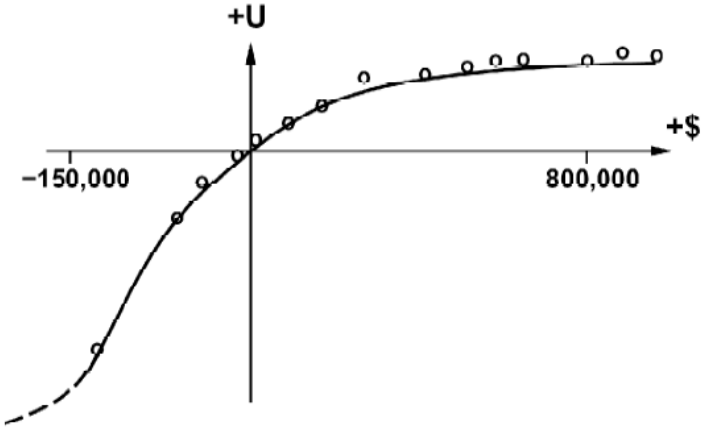
Human Utilities

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



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 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
- 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: 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)