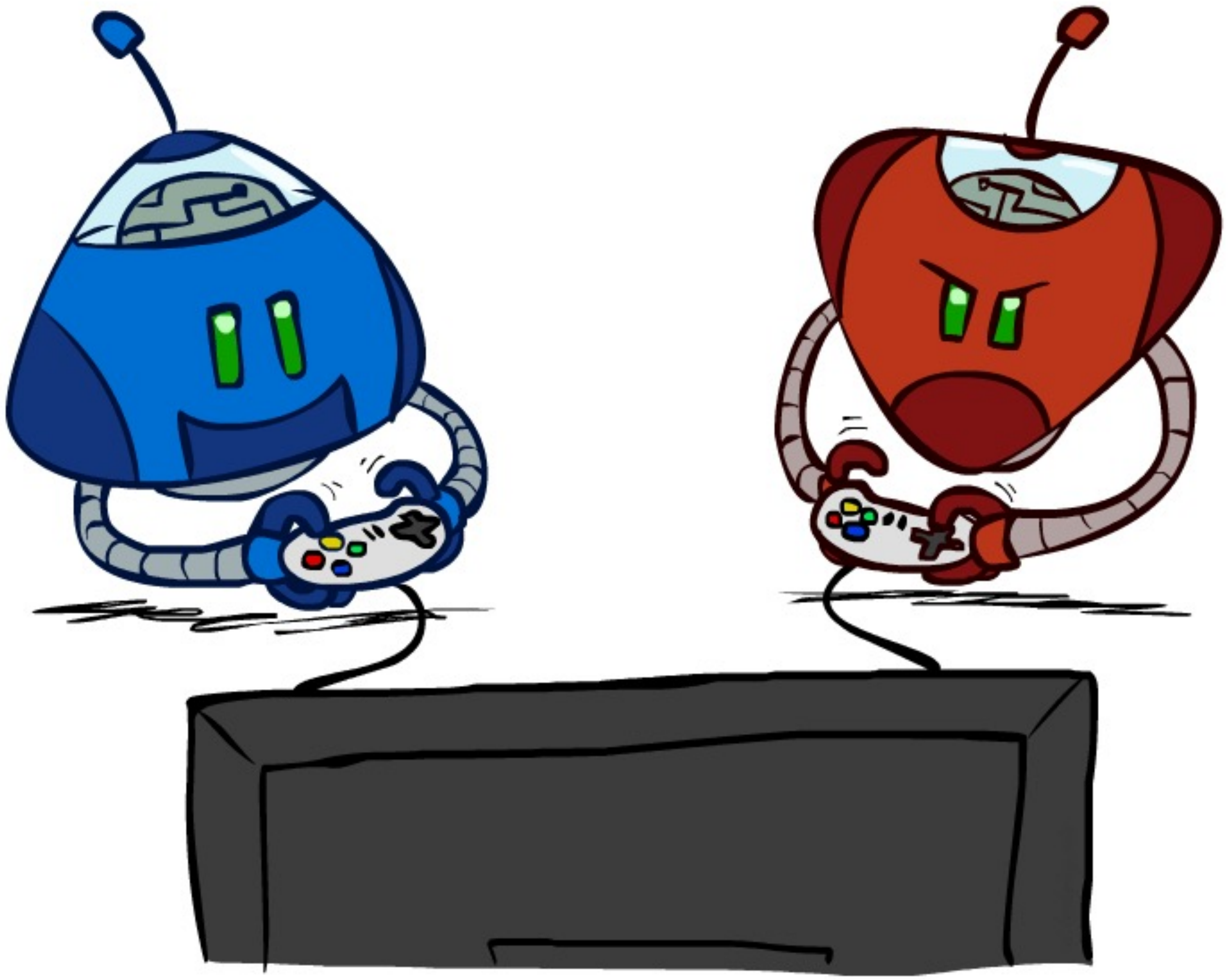
Ve492: Introduction to Artificial Intelligence

Multi-agent search; Games with Chance; Decision Theory



Paul Weng

UM-SJTU Joint Institute

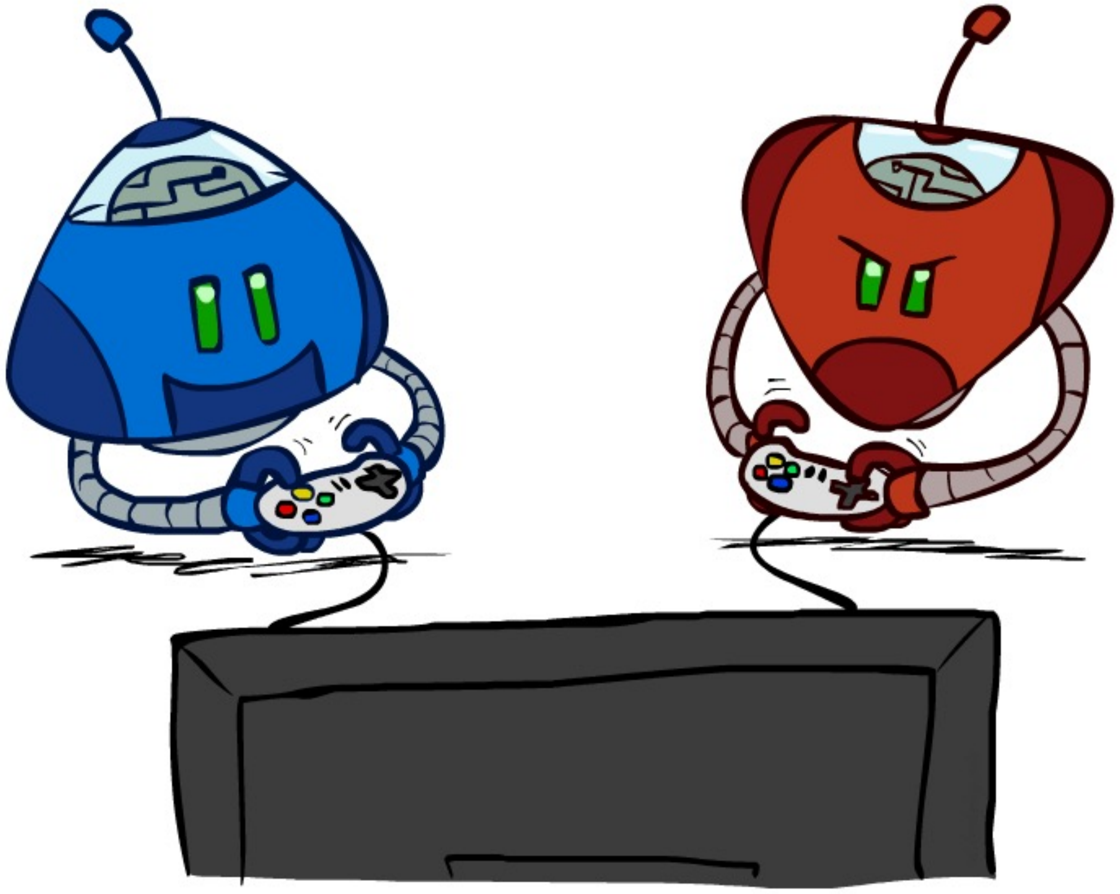
Slides adapted from http://ai.berkeley.edu, AIMA, UM, CMU

Announcements

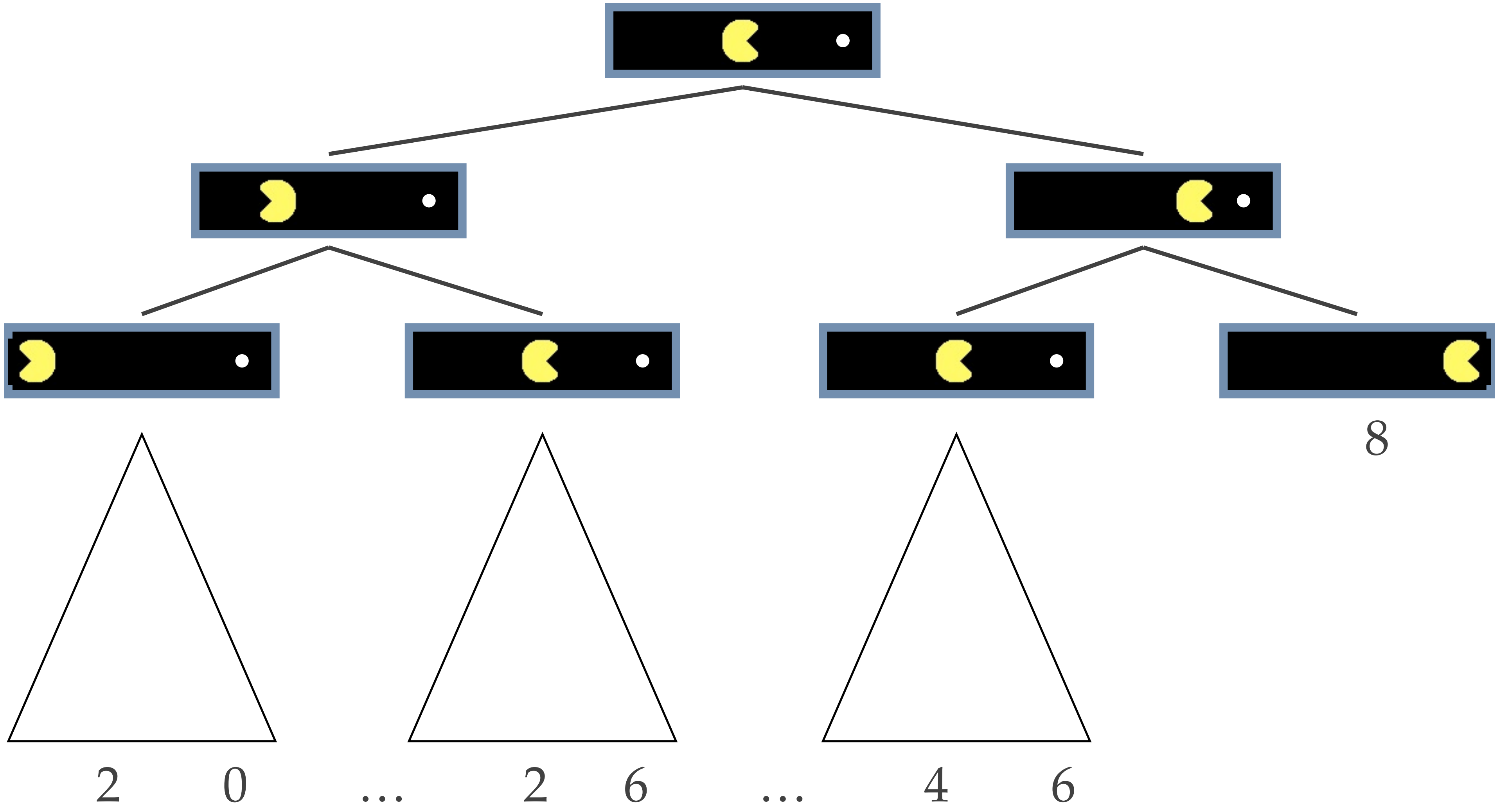
- Project 1: Search
  - Due Wed. June 3 at 11:59pm.
  - Solo or in group of two. For groups of two, both of you need to submit your code into JOJ!
- Homework 2: Multi-agent search
  - Release Wed., May 26, due Wed, June 2 at 11:59pm.
- Make-up classes
  - May 31 => June 18, 2pm-3:40pm, E2-206
  - June 2 => July 16, 2pm-3:40pm, E2-206

Outline

- Multi-agent search
- Games with chance
- Decision Theory

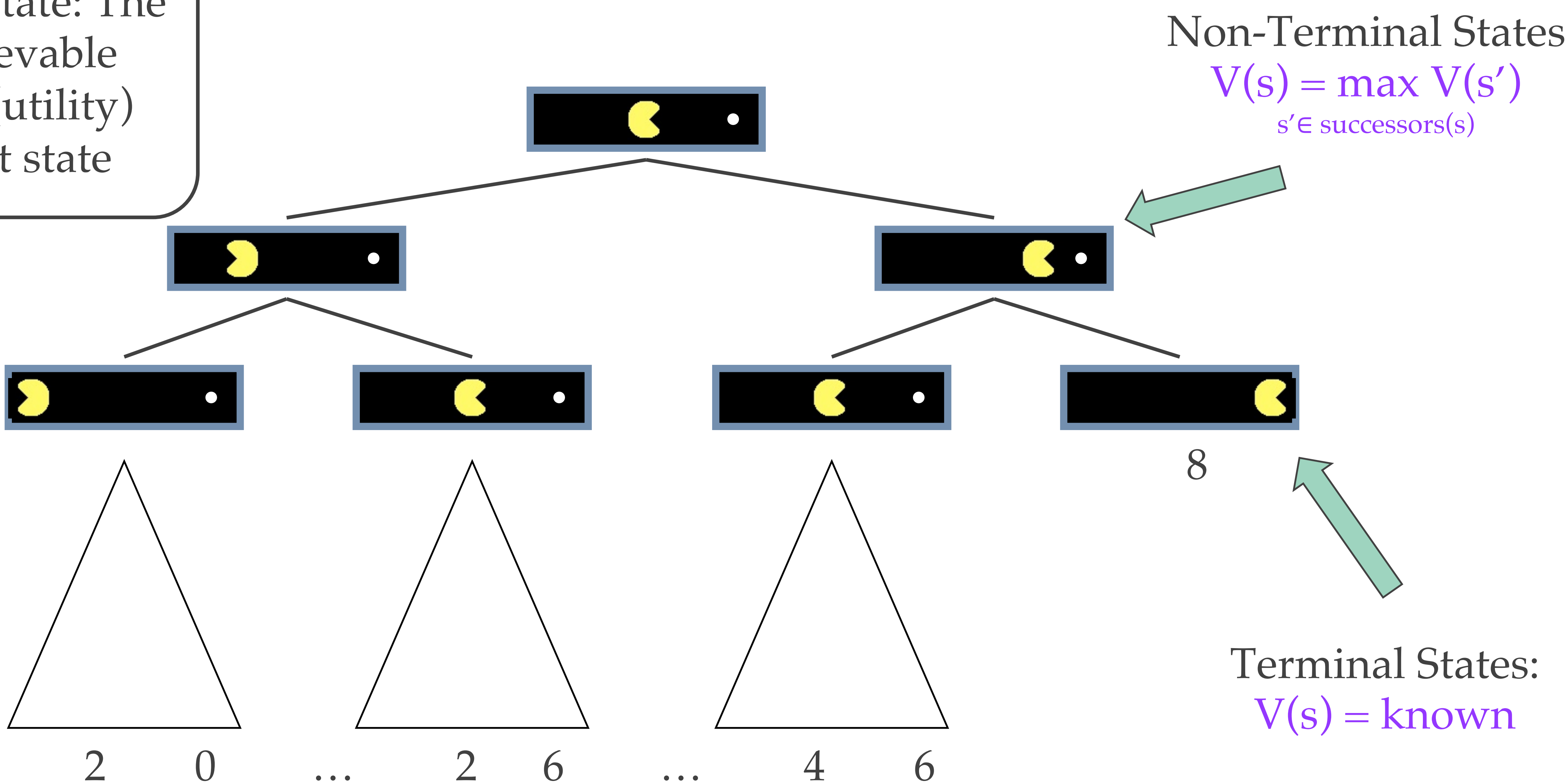


Single-Agent Trees

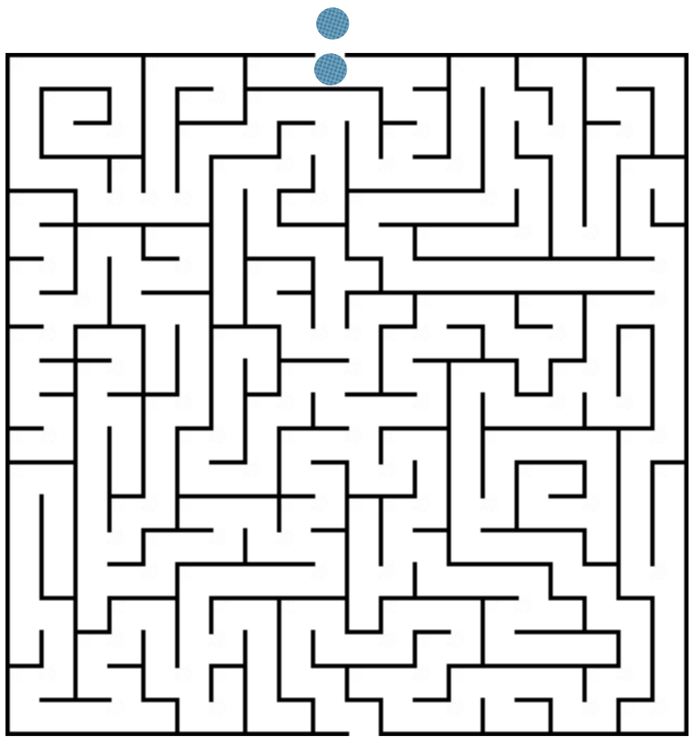


Value of a State

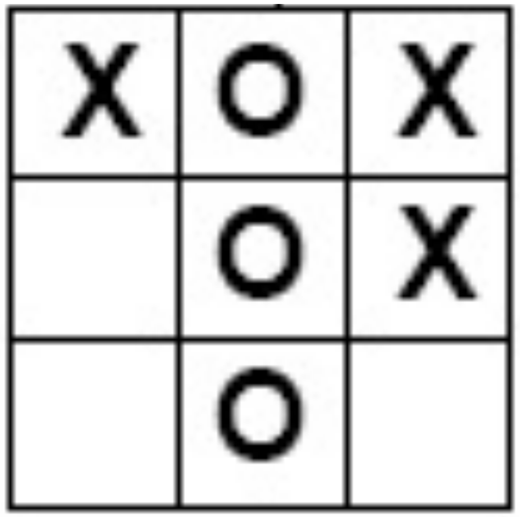
Value of a state: The best achievable outcome (utility) from that state



Multi-Agent Applications



Collaborative Maze Solving



Adversarial

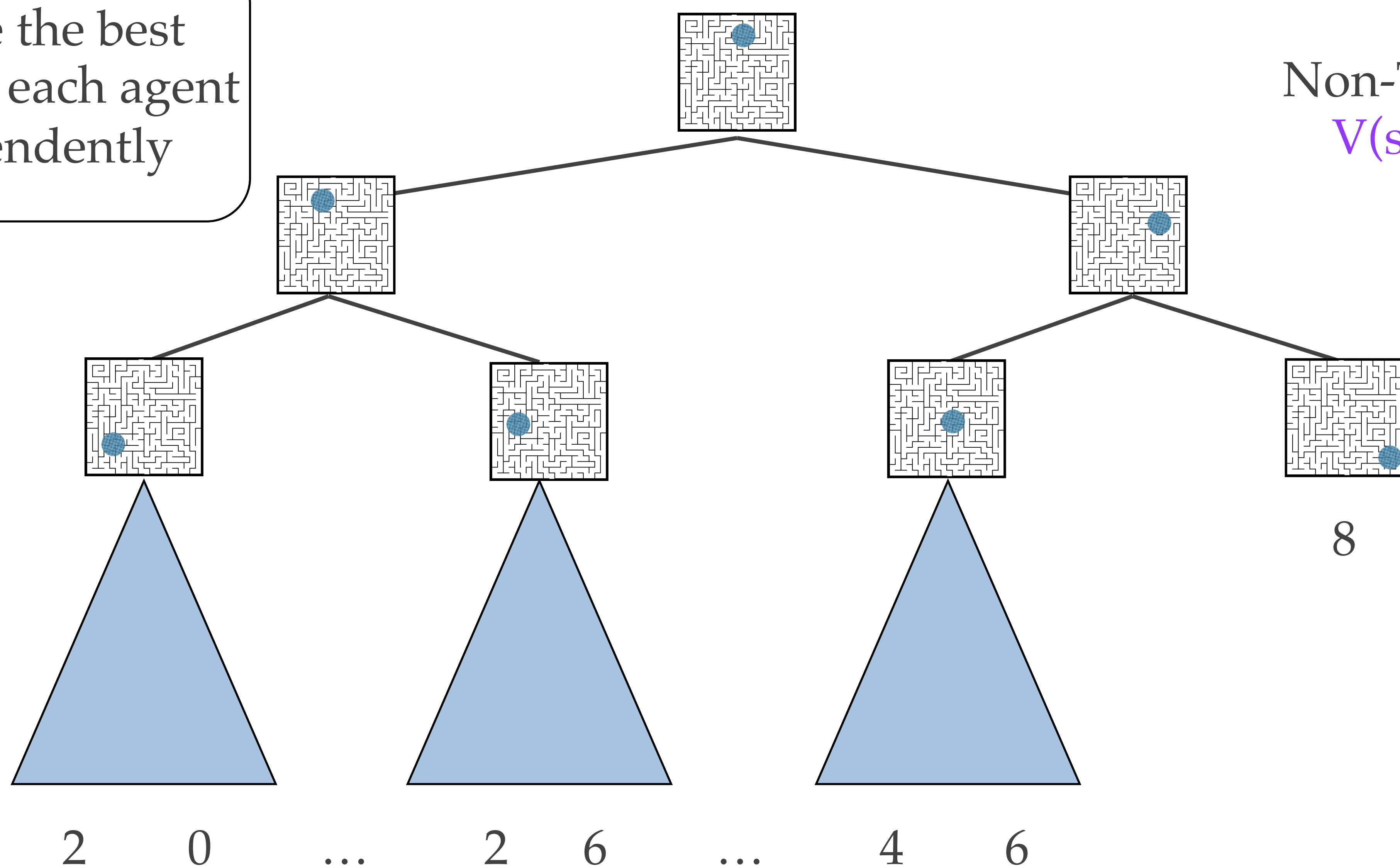
Team: Collaborative
Competition: Adversarial

- How could we tackle multi-agent problems?
    - Depends on problem assumptions

Idea 1: Independent Decision-making

- Each agent plans their own actions separately from others => Many single-agent trees

Choose the best action for each agent independently



Non-Terminal States:

$$V(s) = \max_{s' \in \text{successors}(s)} V(s')$$

Idea 2: Joint State/Action Spaces

- Combine the states and actions of the N agents
- Search looks through all combinations of all agents' states and actions
- Think of one brain controlling many agents

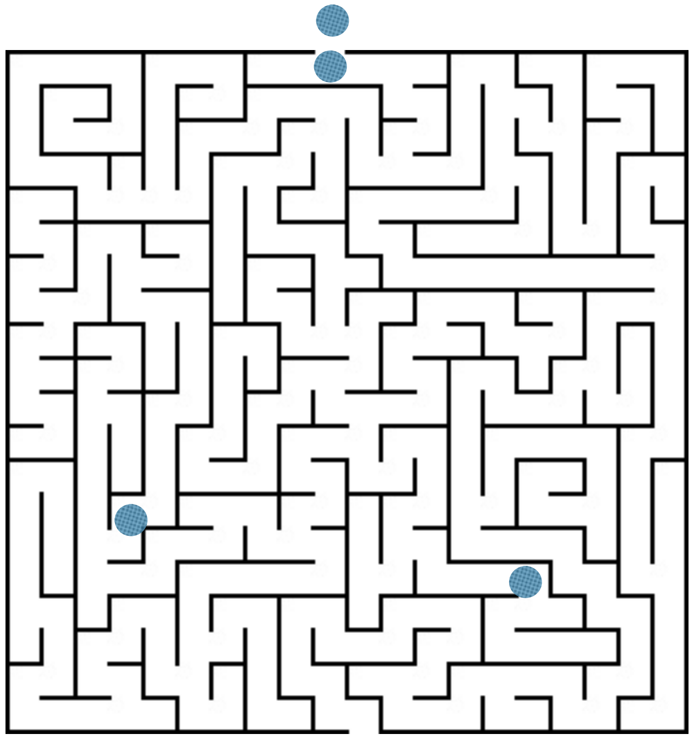
What is the size of the state space?

What is the size of the action space?

What is the size of the search tree?

$S_0 = (S_0^A, S_0^B)$

$S_K = (S_K^A, S_K^B)$



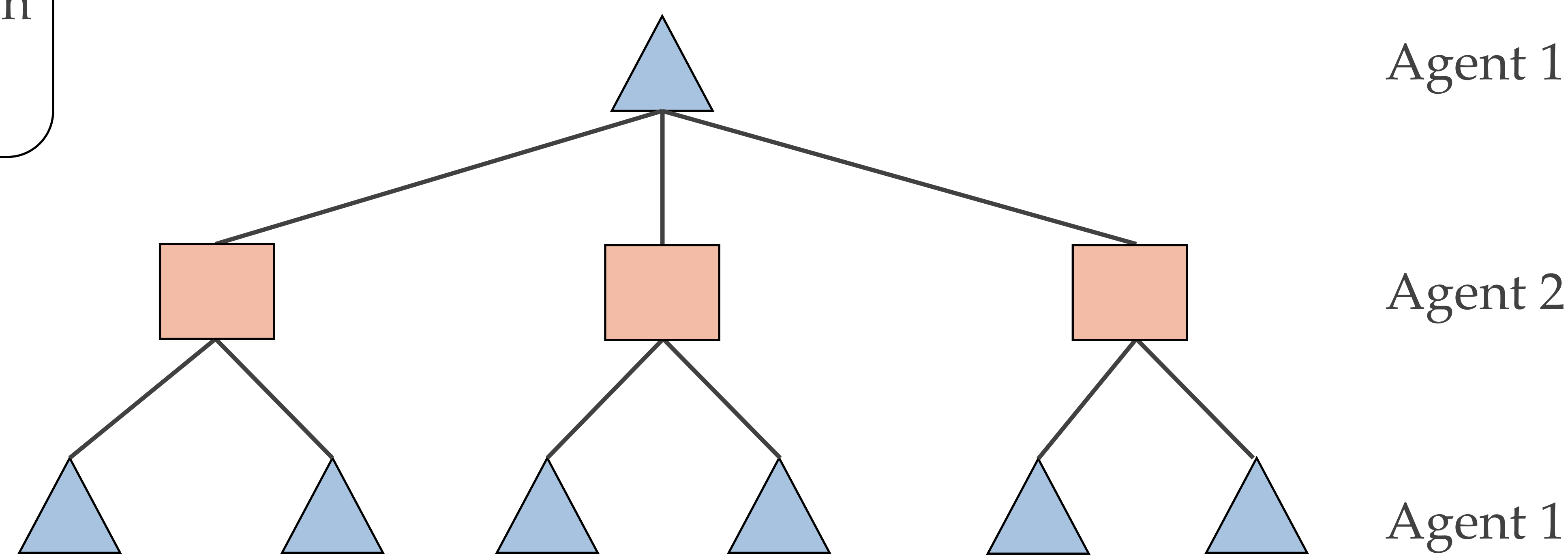
Idea 3: Coordinated Decision Making

- Each agent proposes their actions and computer confirms the joint plan
- Example: Autonomous driving through intersections

Idea 4: Alternate Searching One Agent at a Time

- Search one agent's actions from a state, search the next agent's actions from those resulting states , etc…

Choose the best cascading combination of actions



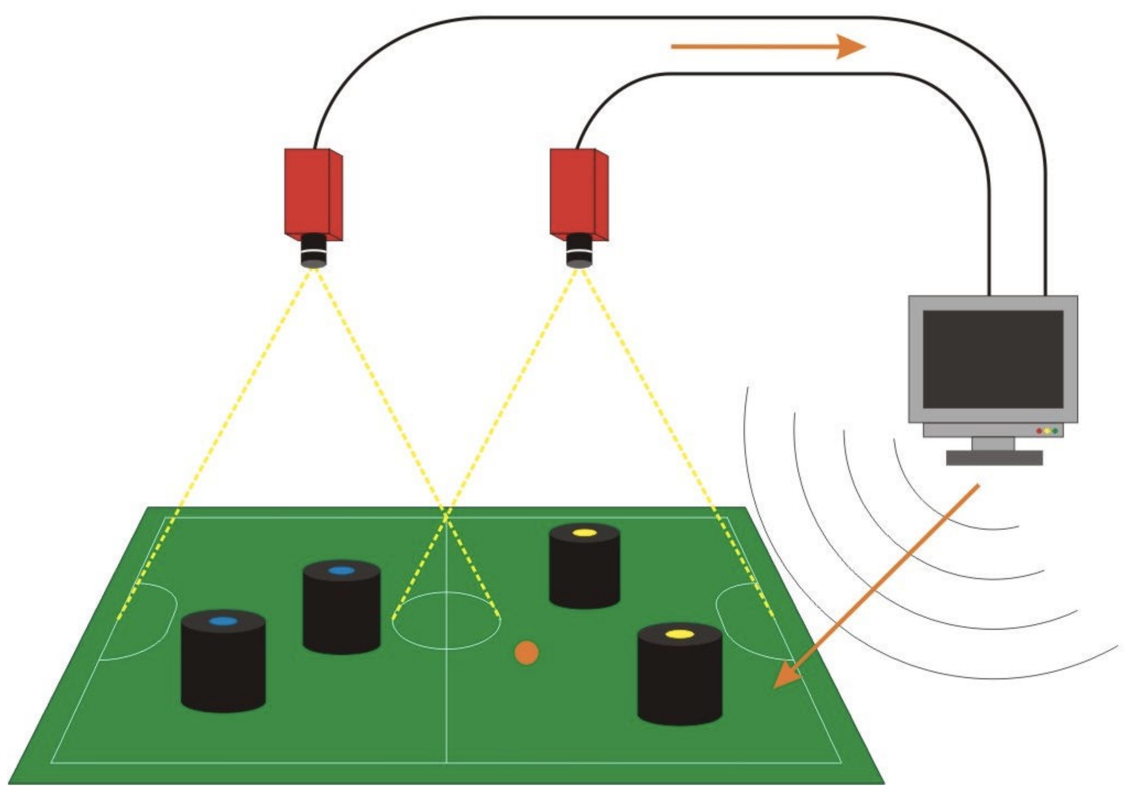
What is the size of the state space?

What is the size of the action space?

What is the size of the search tree?

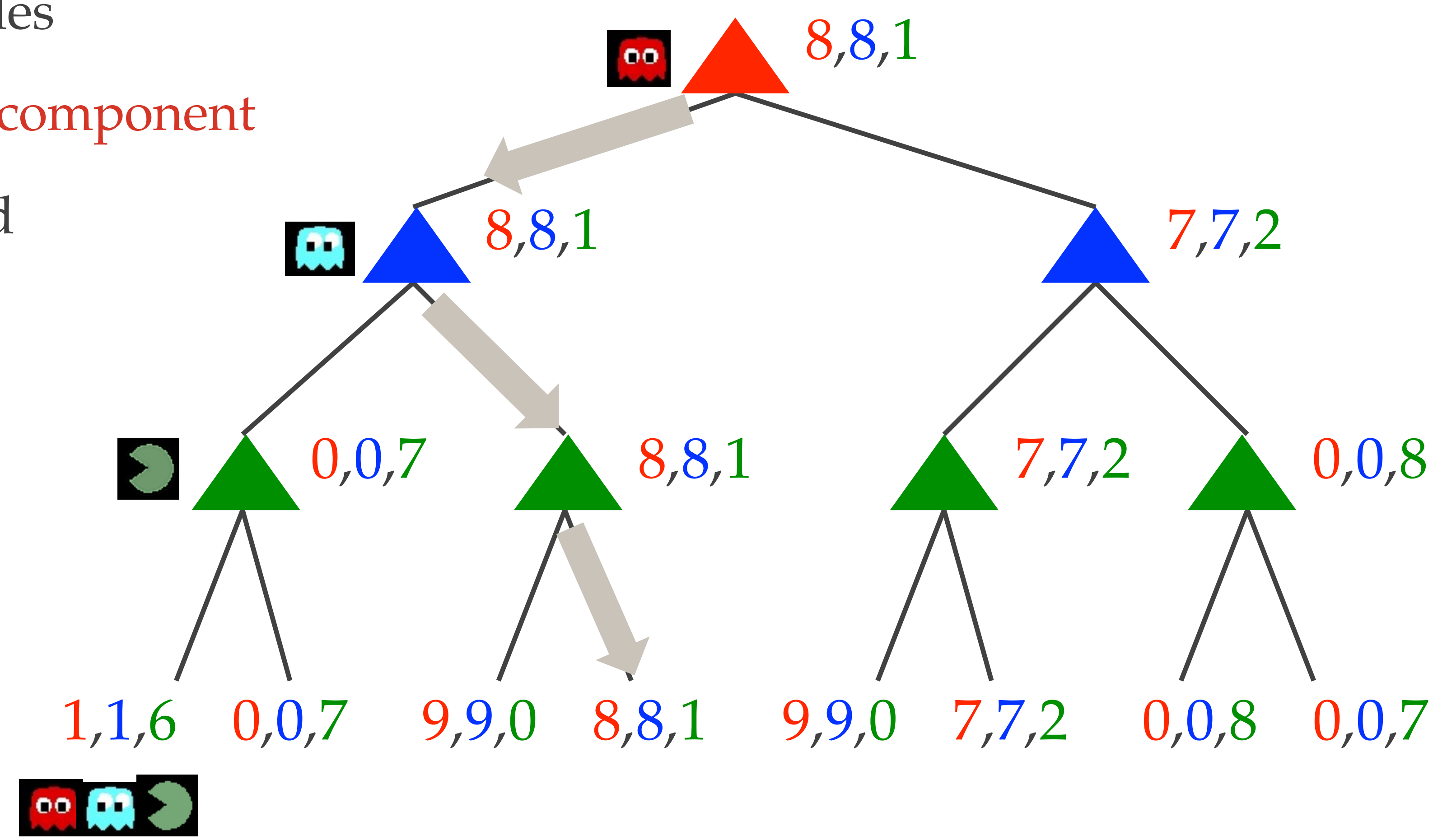
Minimax Search with Two Teams

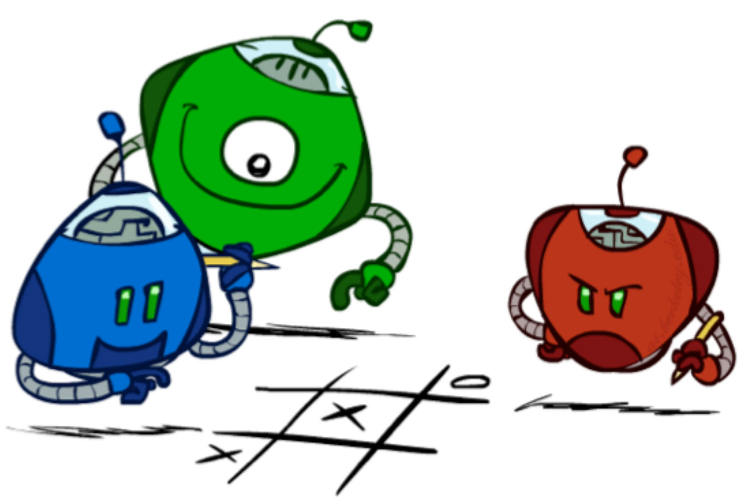
- Joint State / Action space and search for our team
- Adversarial search to predict the opponent team
- Example: Small Size Robot Soccer



Generalized minimax

- 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…



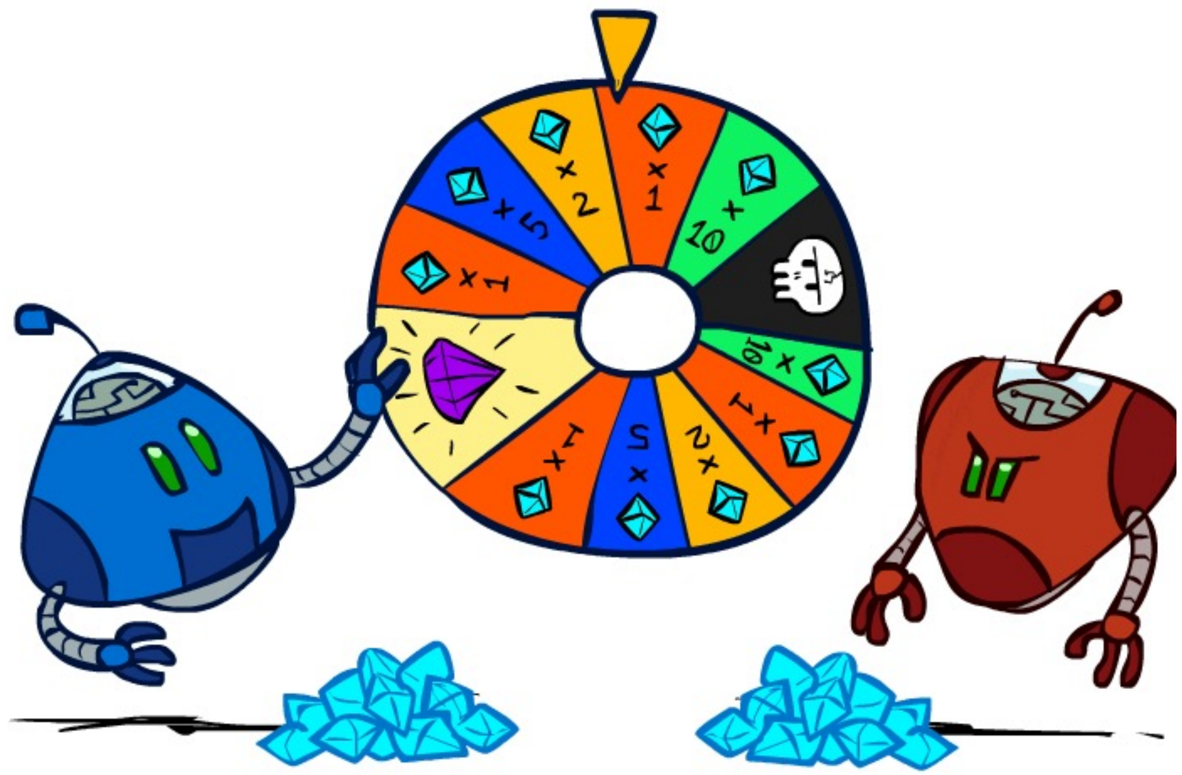


Three-Person Chess

From Wikipedia

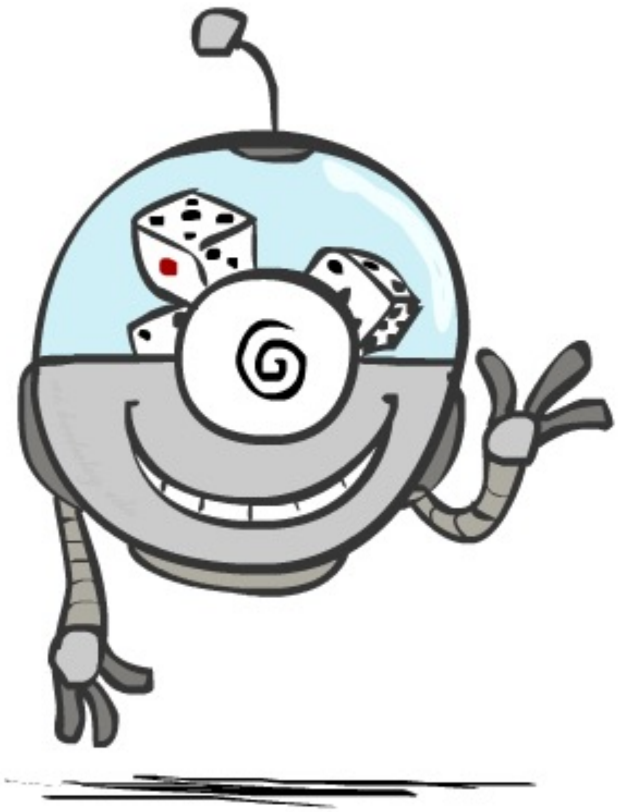
Games with Chance

Search with Random Outcomes



Games with Chance

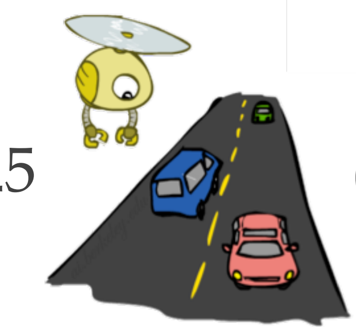
Probabilities



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) = 0.50, 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) = 0.25, P(T=heavy | Hour=8am) = 0.60
  - We'll talk about methods for reasoning and updating probabilities later


0.25


0.50


0.25

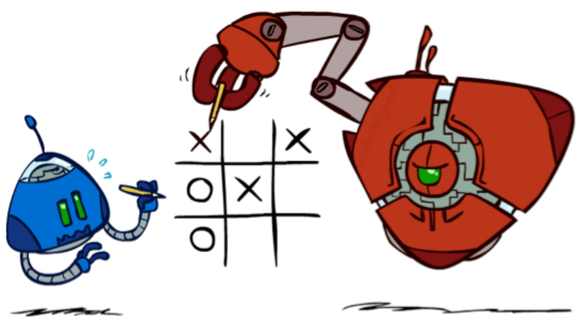
Reminder: Expectations

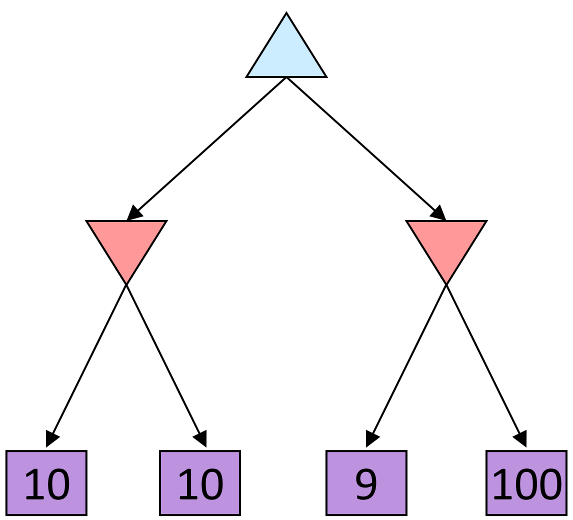
- The expected value of a random variable is the average, weighted by the probability distribution over outcomes

- Example: How long to get to the airport?

Time:	20 min		30 min		60 min		
	x	+	x	+	x		35 min
Probability:	0.25		0.50		0.25		

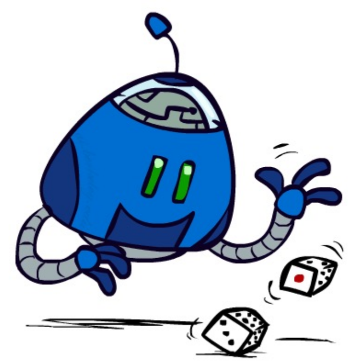
Different Game Trees

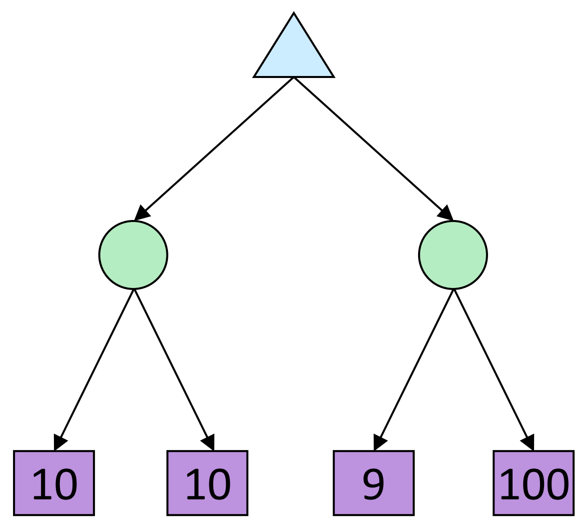




Tictactoe, chess

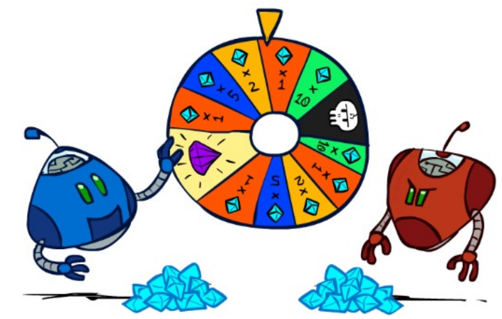
Minimax

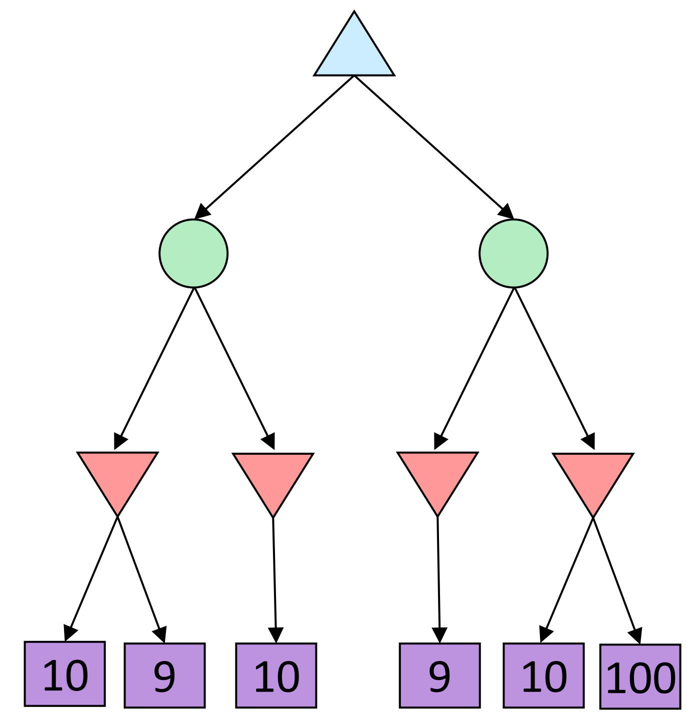




Tetris, investing

Expectimax





Backgammon, Monopoly

Expectiminimax

Minimax

```
function decision(s) returns an action
    return the action a in Actions(s) with the highest
     value(Succ(s,a))
```

```
function value(s) returns a value
    if Terminal-Test(s) then return Utility(s)
    if Player(s) = MAX      then return max_{a in Actions(s)} value(Succ(s,a))
    if Player(s) = MIN      then return min_{a in Actions(s)} value(Succ(s,a))
```

Expectimax

function decision(s) returns an action

return the action a in Actions(s) with the highest value(Succ(s,a))

function value(s) returns a value

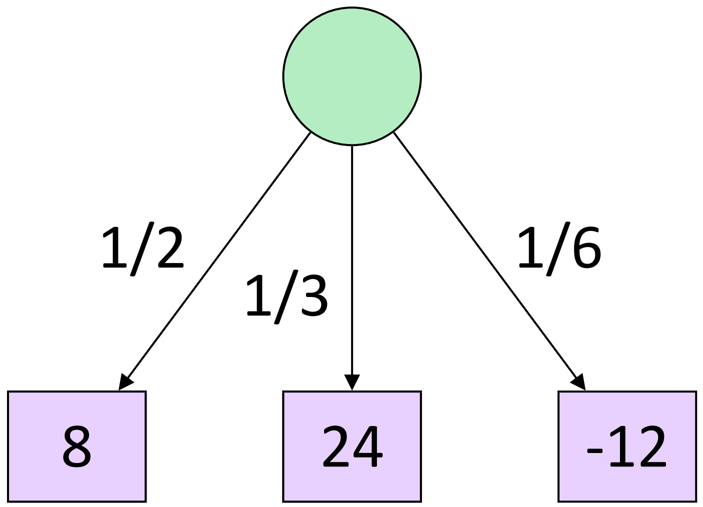
if Terminal-Test(s) then return Utility(s)

if Player(s) = MAX then return $\max_{a \text{ in Actions}(s)}$ value(Succ(s,a))

if Player(s) = CHANCE then return $\text{sum}_{a \text{ in Actions}(s)}$ Pr(a) * value(Succ(s,a))

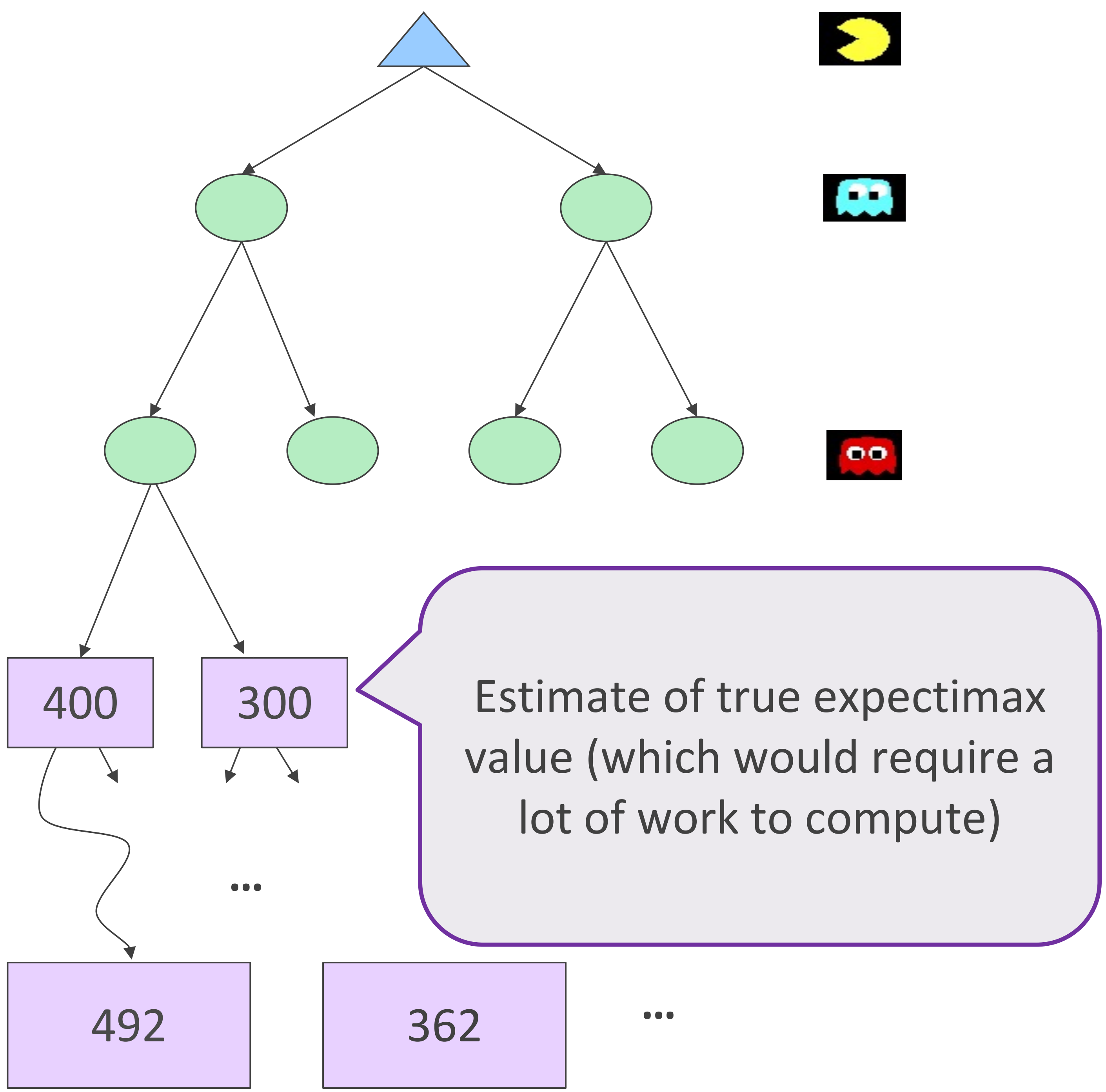
Expectimax Pseudocode

$\text{sum}_{a \text{ in Outcome}(s)} \Pr(a) * \text{value}(\text{Succ}(s,a))$

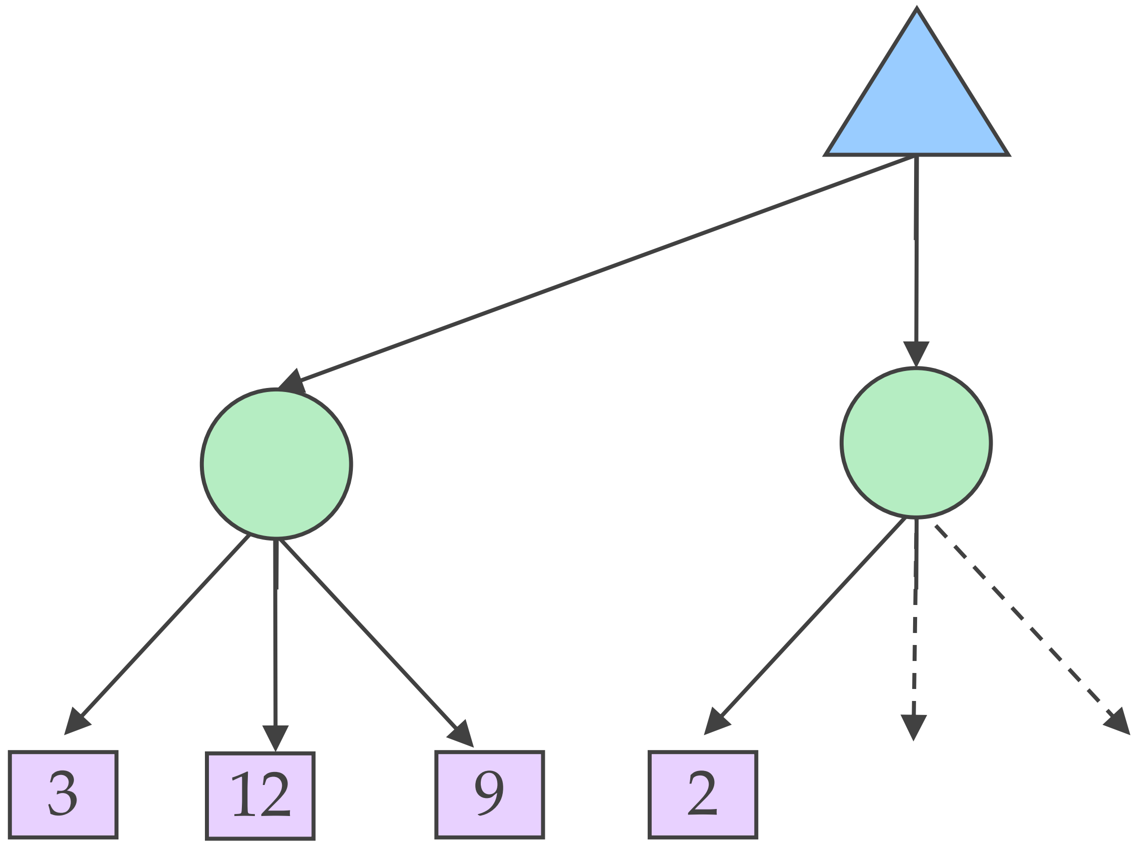


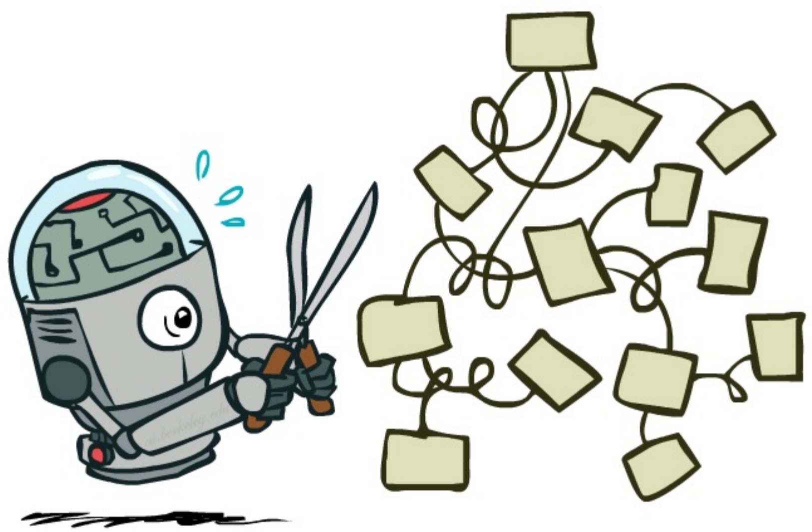
v = (1/2) (8) + (1/3) (24) + (1/6) (-12) = 10

Depth-Limited Expectimax



Expectimax Pruning?





Expectiminimax

function decision(s) returns an action

return the action a in Actions(s) with the highest value(Succ(s,a))

function value(s) returns a value

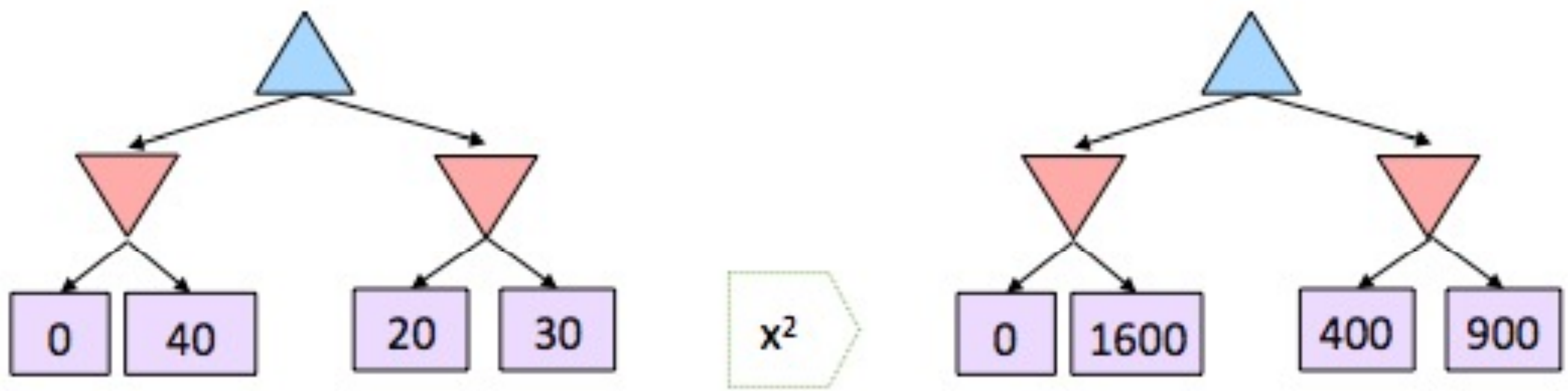
if Terminal-Test(s) then return Utility(s)

if Player(s) = MAX then return $\max_{a \text{ in Actions}(s)}$ value(Succ(s,a))

if Player(s) = MIN then return $\min_{a \text{ in Actions}(s)}$ value(Succ(s,a))

if Player(s) = CHANCE then return $\text{sum}_{a \text{ in Actions}(s)}$ Pr(a) * value(Succ(s,a))

What Values to Use?

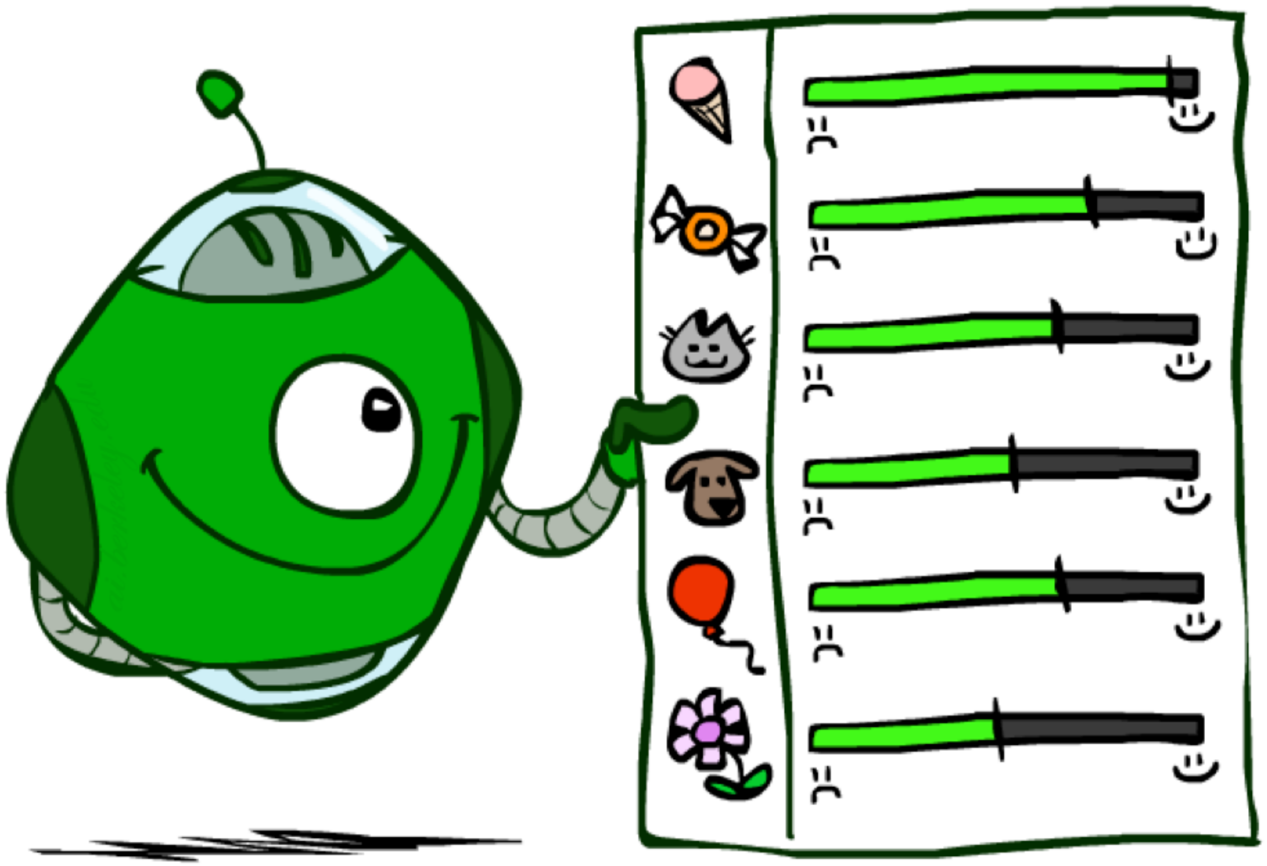


x>y => f(x)>f(y)

f(x) = Ax+B where A>0

- For worst-case minimax reasoning, evaluation function scale doesn't matter
  - We just want better states to have higher evaluations (get the ordering right)
  - Minimax decisions are ***invariant with respect to monotonic transformations on values***
- Expectiminimax decisions are ***invariant with respect to positive affine transformations***
- Expectiminimax evaluation functions have to be aligned with actual win probabilities!

Decision Theory



Decision Theory

- Decision problem:
  - Choose $a \in A$ assuming given preference relation $\succsim$ over A
- Often, choice has uncertain outcomes
  - Probability distribution over outcomes
  - Called lottery in decision theory
- Here, we assume single-agent decision-making
- Which decision criterion should we choose?
  - Descriptive
  - Normative

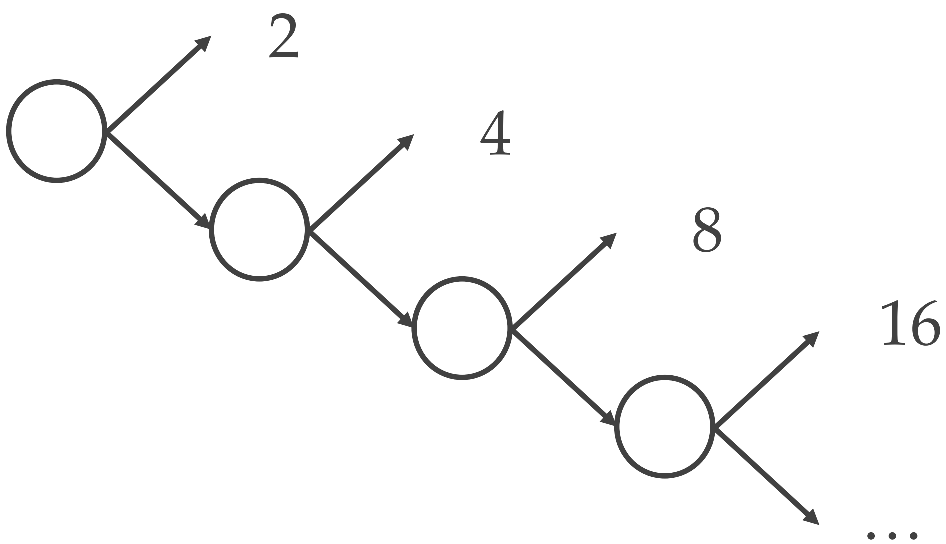
Maximum Expected Utility

- MEU principle:
  - $\max_{p} \sum_{o} p(o) \times U(o)$

- Why is the MEU principle considered rational?
- Where do the utilities come from?
- Where do the probabilities come from?

St Petersburg Paradox

- Game:



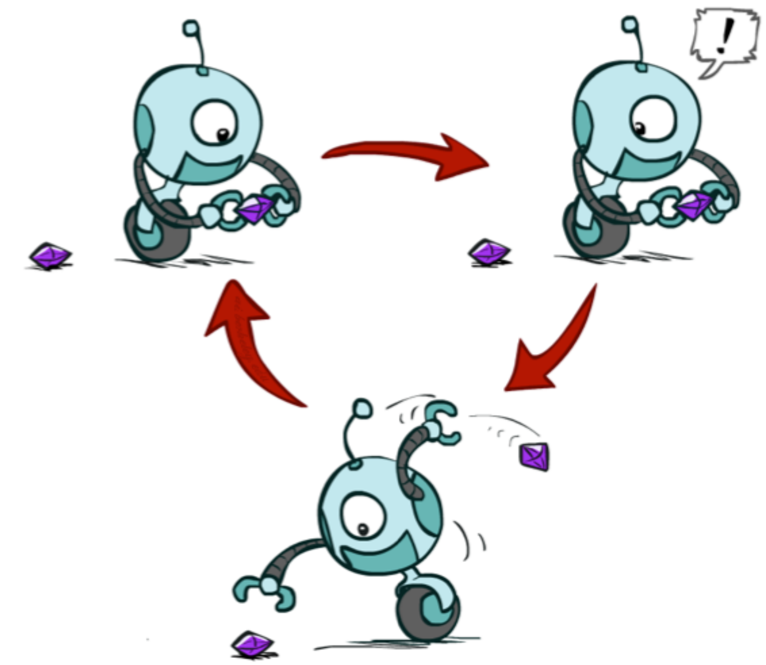
- How much would you pay to play this game?
- Expectation:
- $\mathrm{EU}(L) = \sum_{o \in O} p_L(o) \log(o)$

Axiomatization of MEU

- Decision under uncertainty
  - Outcomes: any consequences from a choice
  - Lotteries: distributions over outcomes
  - Preference relation over lotteries: $\gtrsim$
- Two decision models seen so far: EU and minimax
- Axiomatization of decision model C:
  - If set of conditions on $>$ are satisfied, $L \gtrsim L' \Leftrightarrow C(L) \geq C(L')$

Transitivity

- For any three lotteries, L, L', and L'':
  - $(L \succsim L')$ and $(L' \succsim L'') \Rightarrow (L \succsim L'')$



- Is it a reasonable axiom?

- Money pump argument:
  - If $L' > L''$, then an agent with L'' would pay (say) 1 cent to get L'
  - If $L > L'$, then an agent with L' would pay (say) 1 cent to get L
  - If $L'' > L$, then an agent with L would pay (say) 1 cent to get L''

Axioms of MEU

- Completeness
  - $L \gtrsim L'$ or $L' \gtrsim L$
- Transitivity
  - $(L \gtrsim L')$ and $(L' \gtrsim L'') \Rightarrow (L \gtrsim L'')$
- Independence
  - $(L \gtrsim L') \Rightarrow [p, L; 1\text{-}p, L''] \gtrsim [p, L'; 1\text{-}p, L'']$
- Continuity
  - $(L \gtrsim L' \gtrsim L'') \Rightarrow \exists p, [p, L; 1\text{-}p, L''] \sim L'$

Characterization of MEU

- Theorem [Ramsey, 1931; von Neumann & Morgenstern, 1944; Machina, 1988]
  - $\gtrsim$ satisfies the 4 previous axioms iff there exists a utility function U : O → $\mathbb{R}$ such that
    - $L \gtrsim L' \Leftrightarrow EU(L) \geq EU(L')$
    - $EU(L) = \sum_{o \in O} p_L(o) U(o)$
- If we agree with the 4 axioms, we should apply MEU
- However, most axioms are debatable
- More general axiomatization where probabilities are not assumed to be given (Savage, 1954)
- Decision theory moved to more general notion of rationality

Risk-Sensitive Decision-Making

- Certainty equivalent of lottery L
  - Outcome o_L such that $\mathrm{U}(o_L) = \mathrm{EU}(\mathrm{L})$
- Risk-neutral decision-making
  - $o_L = \sum_{o \in O} p_L(o) \times o$
  - This is the case if U linear
- Risk-averse decision-making
  - $o_L < \sum_{o \in O} p_L(o) \times o$
  - This is the case if U concave
- Risk-seeking decision-making
  - $o_L > \sum_{o \in O} p_L(o) \times o$
  - This is the case if U convex

Preference Elicitation

- Utility function is unique up to a positive affine transformation
- How to specify a utility function for a given decision problem?
  - Assume U is normalized U(o^+)=1 and U(o^-)=0
  - Compare any outcome with binary lotteries:
    - For which p, is this true: o ~ [p, o^+; 1-p, o^-]?
    - The answer gives U(o) = p
  - Extend to a all lotteries

Uncertainty Elicitation

- How to specify a probability distribution for a given decision problem, if unknown?
  - For which o, is this true: [E, o^+; E^c, o^-] ~ [1, o; 0, o^-]
  - The answer gives P(E) = U(o)

Allais Paradox (1953)

- What do you prefer?
  - A: [0.8, $4k; 0.2; $0]
  - B: [1.0, $3k; 0.0; $0]
- What do you prefer?
  - C: [0.2, $4k; 0.8; $0]
  - D: [0.25, $3k; 0.75; $0]
- Usually, B > A and C > D
- However, incompatible with MEU! Assuming U($0)=0:
  - B > A => U($3k) > 0.8 U($4k)
  - C > D => U($4k) > U($3k)

Ellsberg Paradox

- Urn with 30 red balls and 60 other balls, which are either black or yellow.
- What do you prefer?
  - A: [R, $100; B or Y; $0]
  - B: [B, $100; R or Y; $0]
- What do you prefer?
  - C: [R or Y, $100; B; $0]
  - D: [B or Y, $100; R; $0]
- Usually, A > B and D > C
- However, incompatible with MEU!

Summary

- Multi-agent problems can require more space or deeper trees to search
  - Bounded-depth search and approximate evaluation functions
  - Alpha-beta pruning
- Game playing has produced important research ideas
  - Reinforcement learning (checkers)
  - Iterative deepening (chess)
  - Monte Carlo tree search (Go)
  - Solution methods for partial-information games in economics (poker)
- Video games present much greater challenges – lots to do!
  - $b = 10^{500}$, $|S| = 10^{4000}$, $m = 10{,}000$
- Basics of decision theory