COMS W4701: Artificial Intelligence

Lecture 4a: Sequential Decision Problems

Tony Dear, Ph.D.
Department of Computer Science
School of Engineering and Applied Sciences

Today

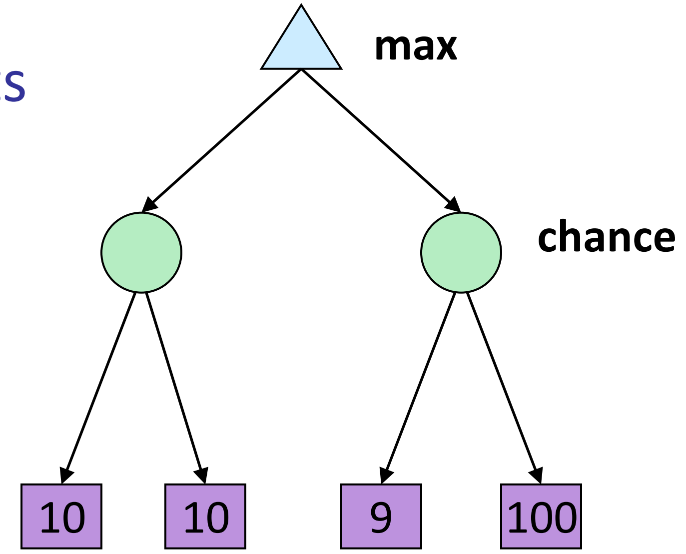
- Stochastic problems
- Markov decision processes
- Utilities and discounting
- Values and policies

Stochastic Problems

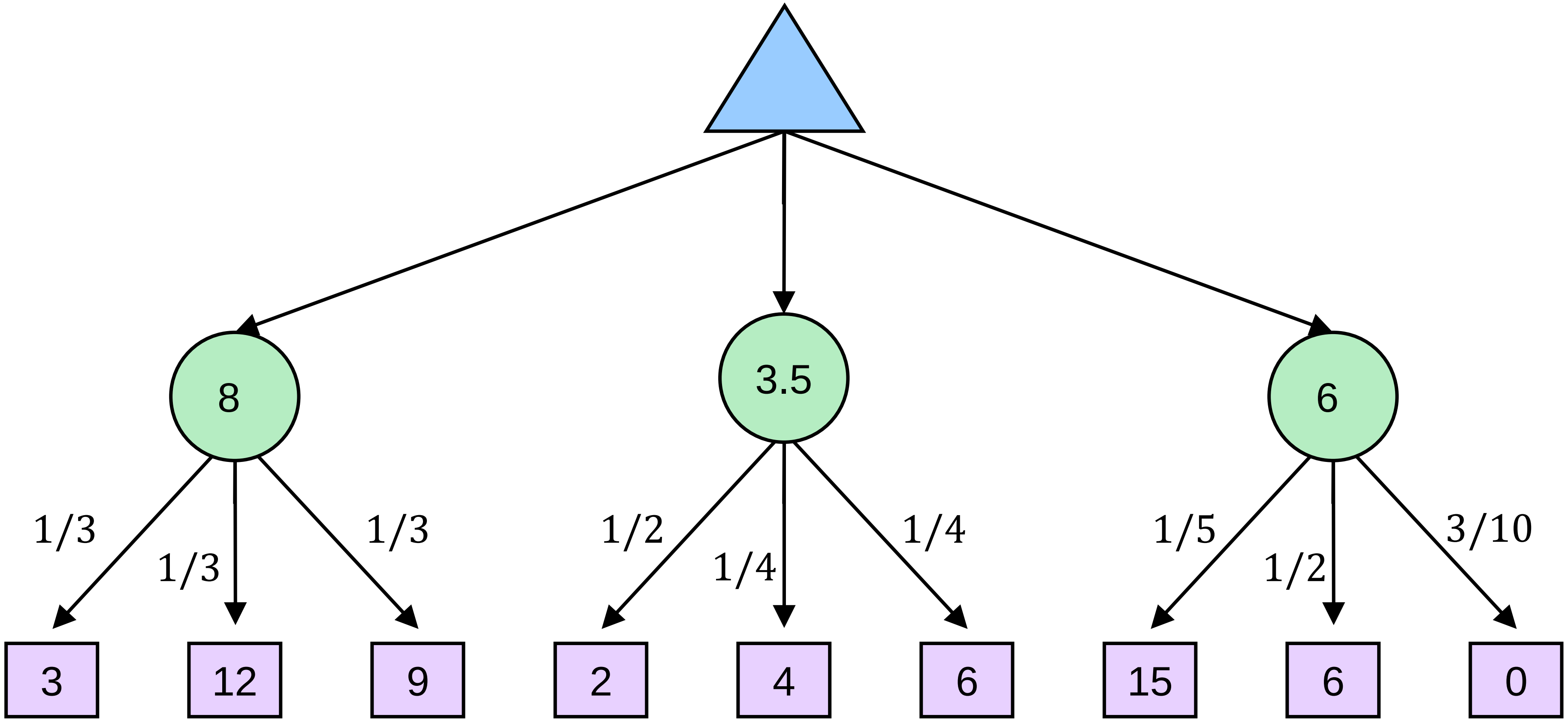
- Games like chess and go are *deterministic*: Each action yields a known resultant state
- Other games and decision-making problems are *stochastic*
- Opponents playing suboptimally or randomly; random game elements (e.g., dice)

- Stochastic elements can be modeled by *expected* outcomes
- We can assign **expectiminimax values** to stochastic elements

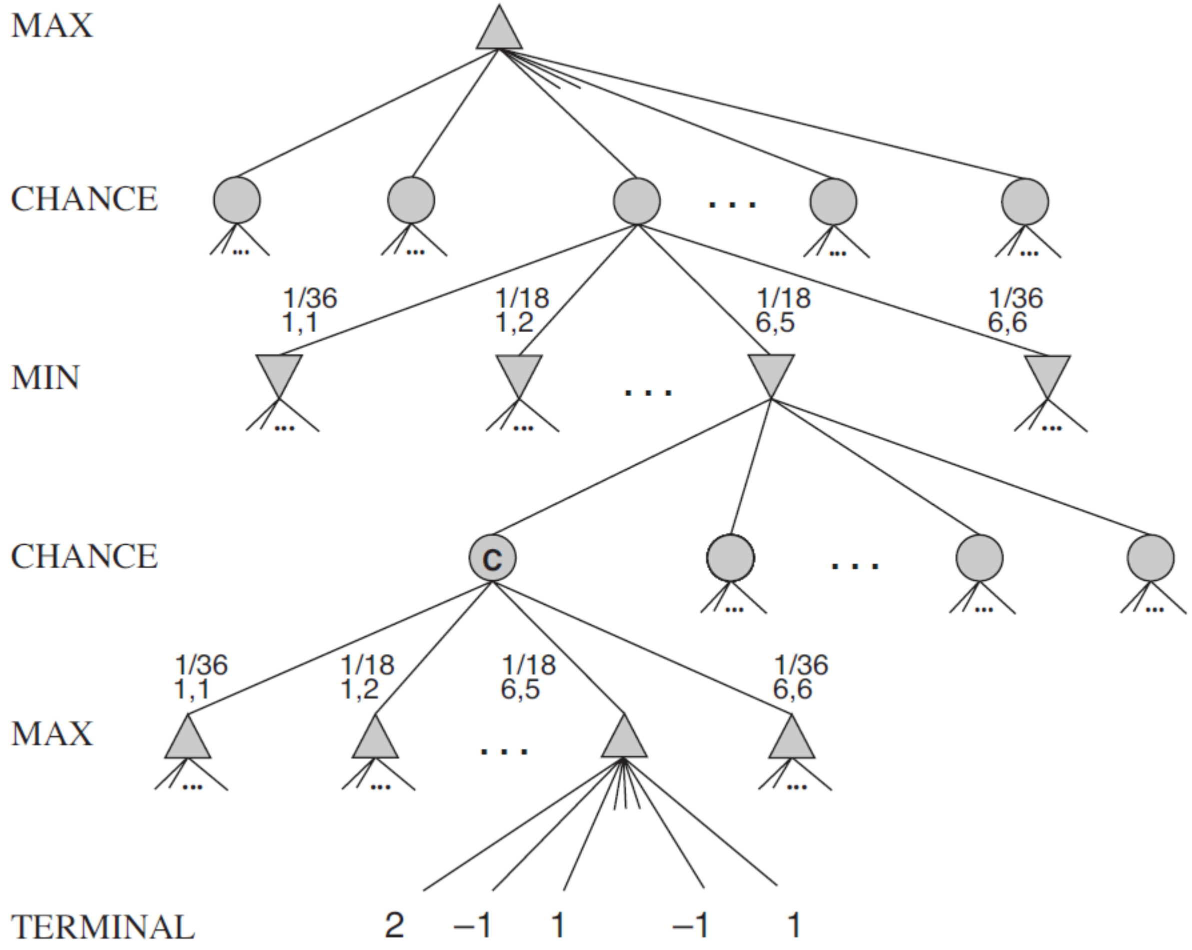
$$\textsc{Expectiminimax}(s) = \begin{cases} \textsc{Utility}(s) & \text{if } \textsc{Terminal-Test}(s) \\ \max_a \textsc{Expectiminimax}(\textsc{Result}(s,a)) & \text{if } \textsc{Player}(s) = \textsc{max} \\ \min_a \textsc{Expectiminimax}(\textsc{Result}(s,a)) & \text{if } \textsc{Player}(s) = \textsc{min} \\ \sum_r P(r)\textsc{Expectiminimax}(\textsc{Result}(s,r)) & \text{if } \textsc{Player}(s) = \textsc{chance} \end{cases}$$



Example: Expectiminimax



Example: Backgammon



Sequential Decision Problems

- We can generalize the idea of computing expected values in stochastic games to stochastic, singe-agent, *sequential decision problems*
- There are also typically rewards associated with each action

- Unlike in deterministic environments, a fixed *plan* will not suffice
- An agent following a plan will deviate from the expected state sequence

- Instead, we will want to find a *policy* that specifies an action for each state
- We will do so by solving for state *values* (as in game search)

Markov Decision Processes

- A **Markov decision process (MDP)** is a mathematical model for a sequential decision problem with uncertainty
- *Solving* a MDP problem gives us optimal/rational decisions

- MDP components: State space S and action set $Actions(s)$ for each state
- *Transition function* $T: S \times A \times S \rightarrow [0,1]$, where $T(s, a, s') = \Pr(s'|s, a)$
- *Reward function* $R: S \times A \times S \rightarrow \mathbb{R}$, written as $R(s, a, s')$

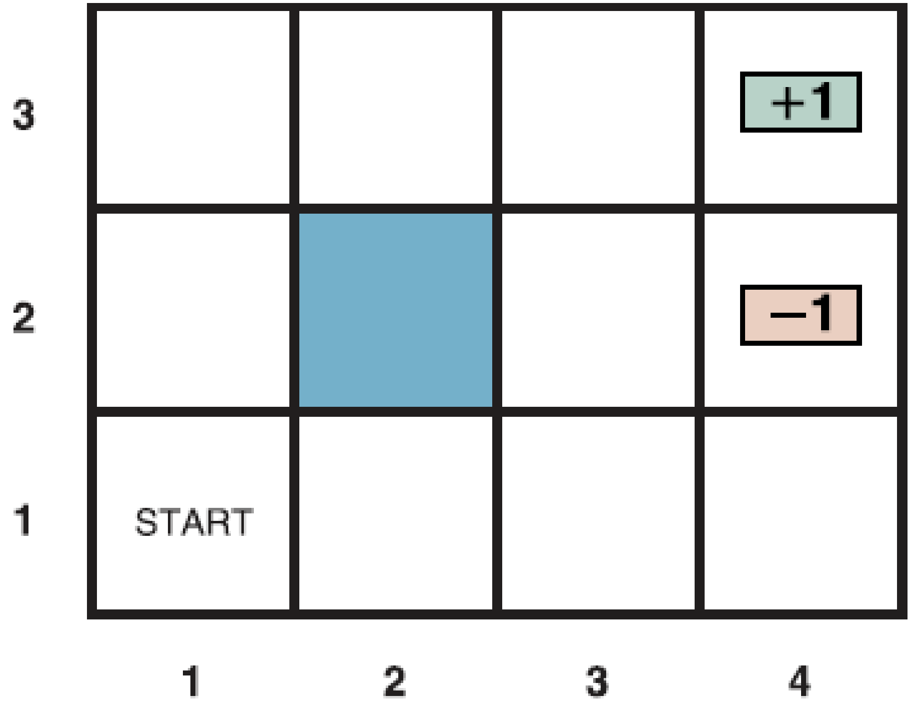
- **Markov property**: Transitions depend on a finitely many previous states

Gridworld Example

- States: All grid cells except for (2,2)

- Actions: North, south, east, west
- Available in most states, except *terminal* states (4,2) and (4,3)

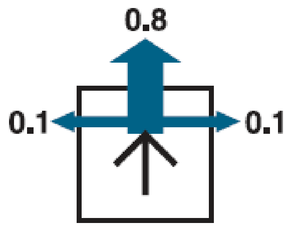
- Reward function: ± 1 for entering respective terminal states; **living reward** received for all other transitions



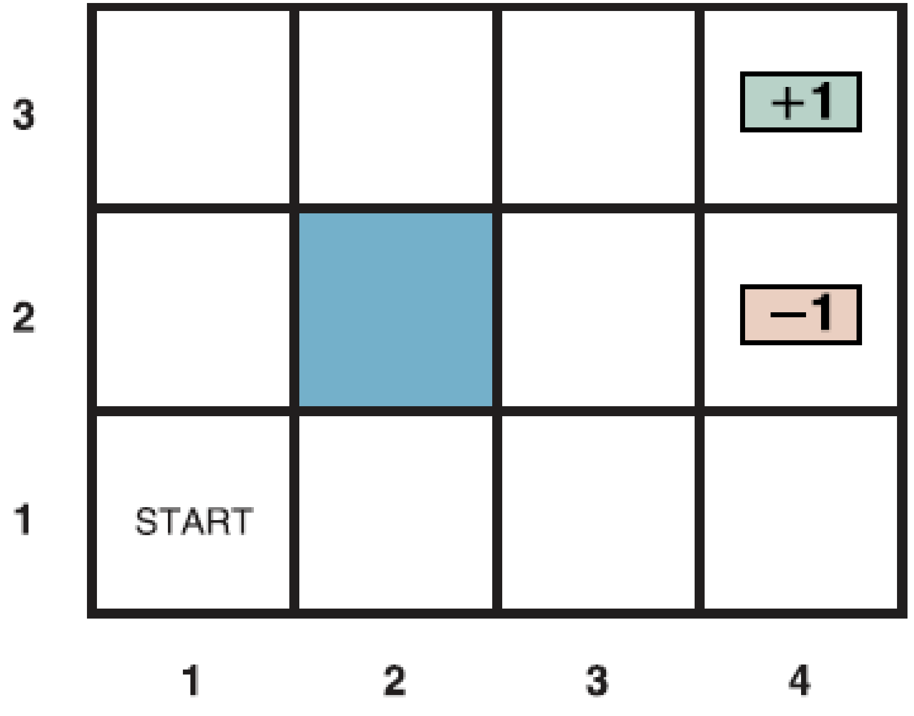
Gridworld Example

- Transition function: Agent ends up in the "expected" successor state *most* of the time

- Small probability that the agent moves "sideways" relative to expected successor



- If the successor state is outside gridworld limits, agent simply remains in original state



MDPs in Practice

- Agriculture
  - S: Soil condition and precipitation forecast. A: Whether or not to plant a given area.
- Water resources and energy generation
  - S: Water levels and inflow. A: How much water to use to generate power.
- Inspection and maintenance
  - S: System age and probability failure. A: Whether to test / restore / repair a system.
- Inventory
  - S: Inventory levels and commodity prices. A: How much to purchase.
- Finance and investment
  - S: Holding or capital levels. A: How much to invest.

- Many, many more (D. J. White 1993)

Utilities

- The *rewards* of a state/action sequence define its **utility**
- A *rational* agent seeks a state/action sequence that maximizes utility

- Utilities may depend on *timing* of when rewards are received
- Example: Sums of reward sequences $R_1 = (1,1,1)$ and $R_2 = (0,0,3)$ are equal, but R_1 is preferable if rewards *now* are better than rewards *later*

- **Additive discounted rewards** using *discount factor* $0 \leq \gamma \leq 1$:

$$V([s_0, a_0, s_1, a_1, \dots, a_{T-1}, s_T]) = \sum_{t=0}^{T-1} \gamma^t R(s_t, a_t, s_{t+1})$$

Infinite-Horizon MDPs

- The *choice* of γ determines how myopic or forward-looking our agent is
- Usually a parameter that is defined by the problem-solver

- If we have a **finite-horizon** MDP, can use $\gamma = 1$ since the number of rewards is finite
- But **infinite-horizon** MDPs *must* have $\gamma < 1$ to yield well-defined utilities!

- Upper bound on state/action sequence utility:

$$V([s_0, a_0, s_1, \dots]) = \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \leq \frac{R_{\max}}{1-\gamma}$$

Reward Engineering

- Where do rewards and utilities come from in general?
- One source: A reflection of the preferences and goals of the user

- **Utility/reward engineering** can be difficult or even controversial
- People have different utility functions and unobservable constructs

- Maximizing rewards for some may not yield the same outcome for others
- How open and accessible do we make these parameters?

Policies and Value Functions

- Solving MDP means finding a **policy**—mapping from states to actions
- $\pi: S \rightarrow A$ tells agent what to do in any state

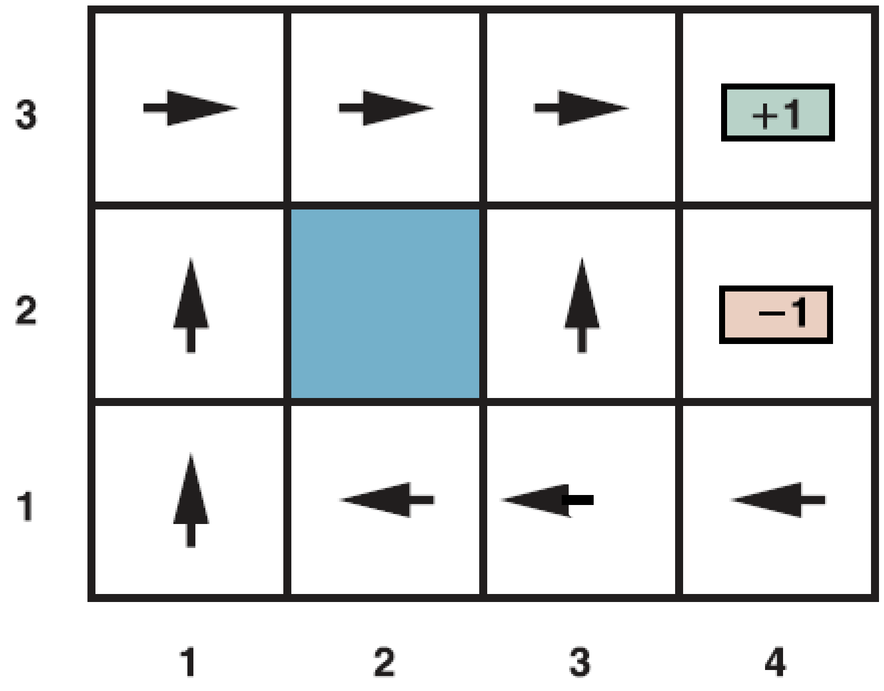
- Policies can be quantified by **value functions**
- $V^{\pi}: S \rightarrow \mathbb{R}$ is *expected* utility of following π starting from given state

$$V^{\pi}(s) = E\left[\sum_{t=0} \gamma^t R(s_t, \pi(s_t), s_{t+1})\right], s_0 = s$$

- **Optimal** policy and value function:

$$\pi^* = \operatorname{argmax}_{\pi} V^{\pi} \qquad V^* = \max_{\pi} V^{\pi}$$

Gridworld Policy and Value Function



3	0.8516	0.9078	0.9578	+1
2	0.8016		0.7003	−1
1	0.7453	0.6953	0.6514	0.4279
	1	2	3	4

$R(s) = -0.04$ for nonterminal states
$\gamma = 1$ (no discounting)

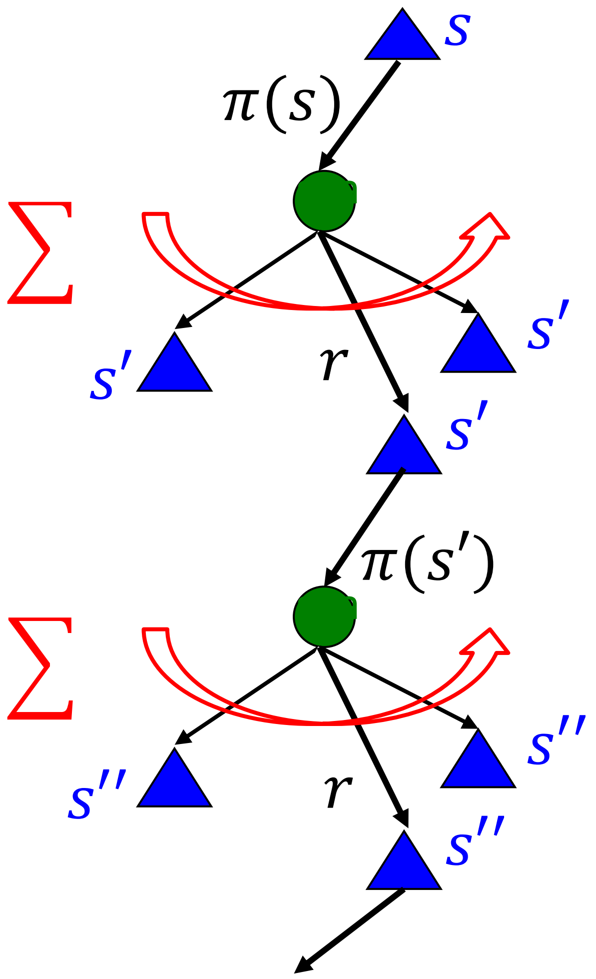
Recursive Relationship

$$V^{\pi}(s) = E\left[\sum_{t=0} \gamma^t R(s_t, \pi(s_t), s_{t+1})\right]$$

- We can rewrite each state value $V^{\pi}(s)$ as a function of (other) *successor state values*

$$V^{\pi}(s) = \sum_{s'} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V^{\pi}(s')]$$

- This is a system of $|S|$ *linear* equations in the $|S|$ unknowns $V^{\pi}(s)$
- Can solve for all state values in $O(|S|^3)$ time



Example: Mini-Gridworld

- Consider a mini-gridworld with states A, B, C
- No terminal states!

+3	-2	+1
A	B	C

- From each state, we can take action L or R

- Reward function: $R(s, a, A) = 3, R(s, a, B) = -2, R(s, a, C) = 1$

- Transition function: $\Pr(\text{intended direction}) = 0.8$, $\Pr(\text{opposite direction}) = 0.2$; $s' = s$ if outside grid boundaries

Example: Mini-Gridworld

$$V^{\pi}(s) = \sum_{s'} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V^{\pi}(s')]$$

+3	-2	+1
A	B	C

- Suppose we are given the policy $\pi(s) = L \; \forall s$
- Suppose we use the discount factor $\gamma = 0.5$

- We can form a system of three equations, one for each state:

$$V^{\pi}(A) = 0.8\big(3 + 0.5V^{\pi}(A)\big) + 0.2(-2 + 0.5V^{\pi}(B))$$
$$V^{\pi}(B) = 0.8\big(3 + 0.5V^{\pi}(A)\big) + 0.2(1 + 0.5V^{\pi}(C))$$
$$V^{\pi}(C) = 0.8\big(-2 + 0.5V^{\pi}(B)\big) + 0.2(1 + 0.5V^{\pi}(C))$$

$$\Rightarrow \quad V^{\pi} = \begin{pmatrix} 4.04 \\ 4.25 \\ .333 \end{pmatrix}$$

Summary

- Stochastic games can be solved by computing expectiminimax values
- Sequential decision problems can be modeled as MDPs
  - Key components: States, actions, transitions, rewards
  - Derived concepts: Utilities, policies, value functions
- Discounting can apply diminishing weights to future rewards and allow utilities of infinite sequences to converge
- Policies and value functions describe what an agent can do