<u>This lecture will be recorded!!!</u>

Welcome to

DS595
Reinforcement Learning

Prof. Yanhua Li

Time: 6:00pm –8:50pm W
Zoom Lecture
Spring 2022

Last Lecture

- What is reinforcement learning?
- Difference from other AI problems
- Application stories.
- Topics to be covered in this course.
- Course logistics

Reinforcement Learning
What is it?

Reinforcement learning (RL) is an area of machine learning concerned with how software agents ought to take actions in an environment to maximize some notion of cumulative reward.

1. Model **2. Value function** **3. Policy**

(From Wikipedia)

RL involves 4 key aspects

1. Optimization.

- Goal is to find an optimal way to make decisions, with maximized total cumulated rewards

2. Exploration.

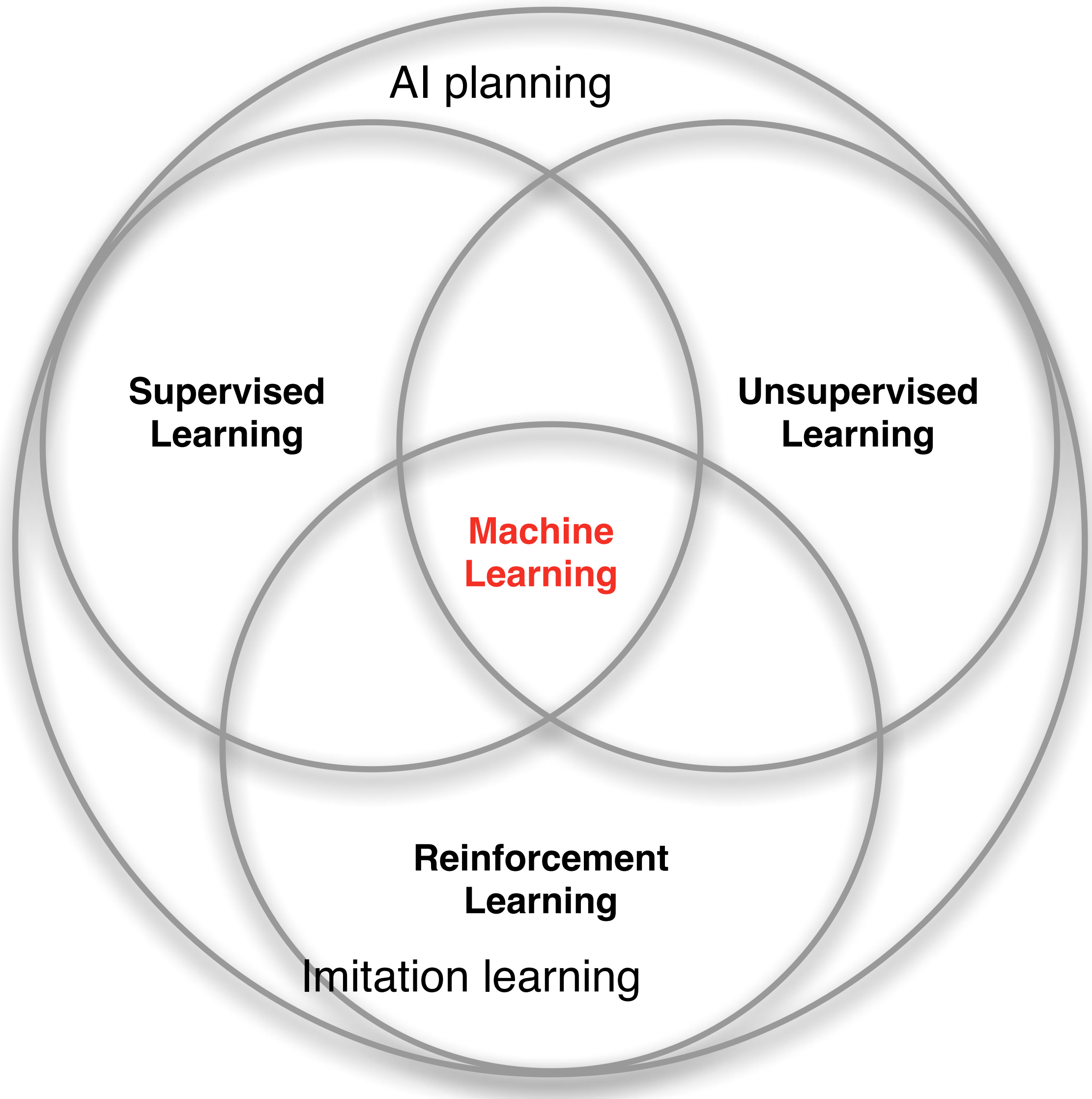
2. Generalization.

- Programming all possibilities is not possible.

4. Delayed consequences

$5 $20

Branches of Machine Learning



From David Silver's Slides

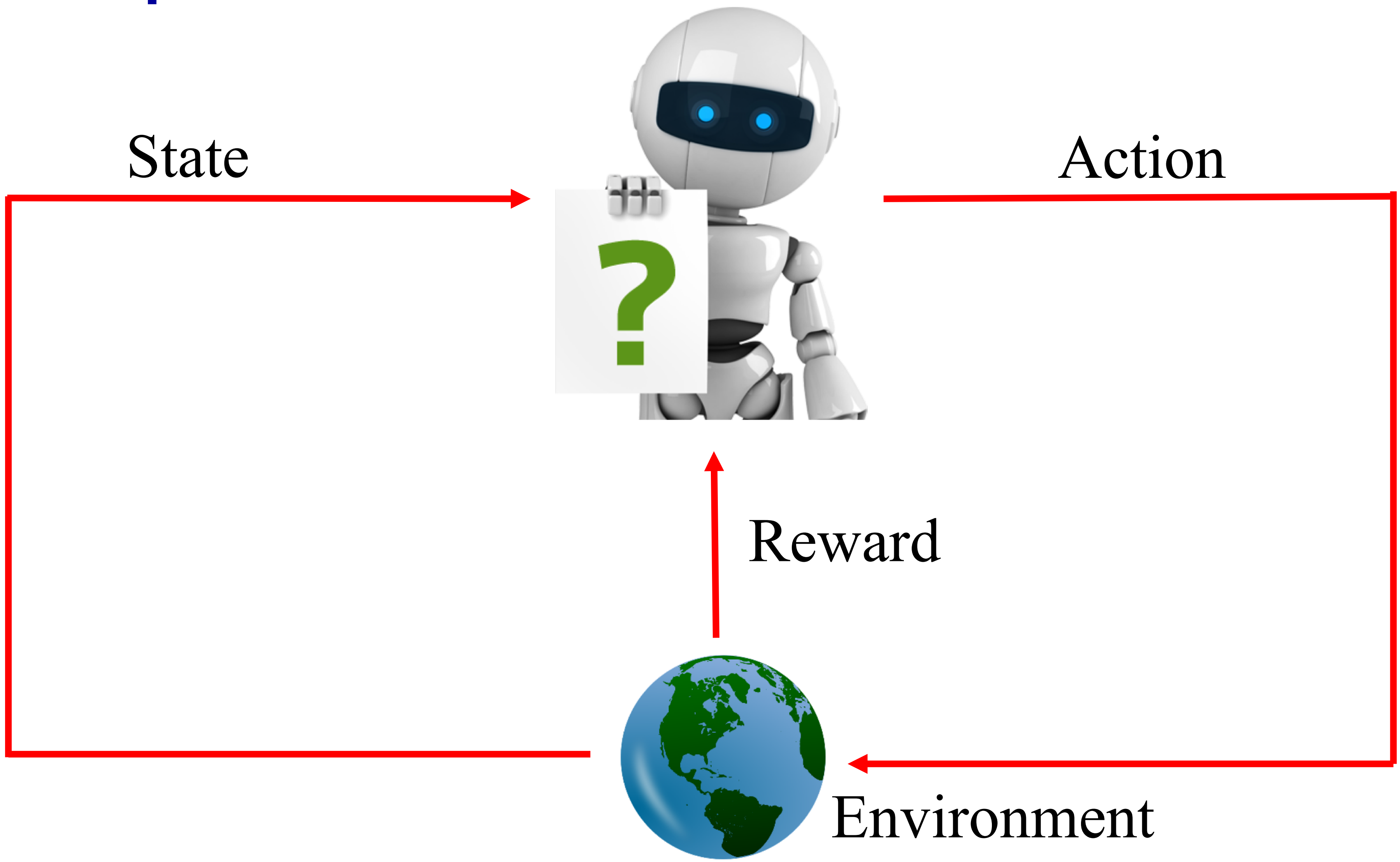
Today's topics

- Reinforcement Learning Components
  - Model, Value function, Policy
- Model-based Planning
  - Policy Evaluation, Policy Search
- Project 1 demo and description.

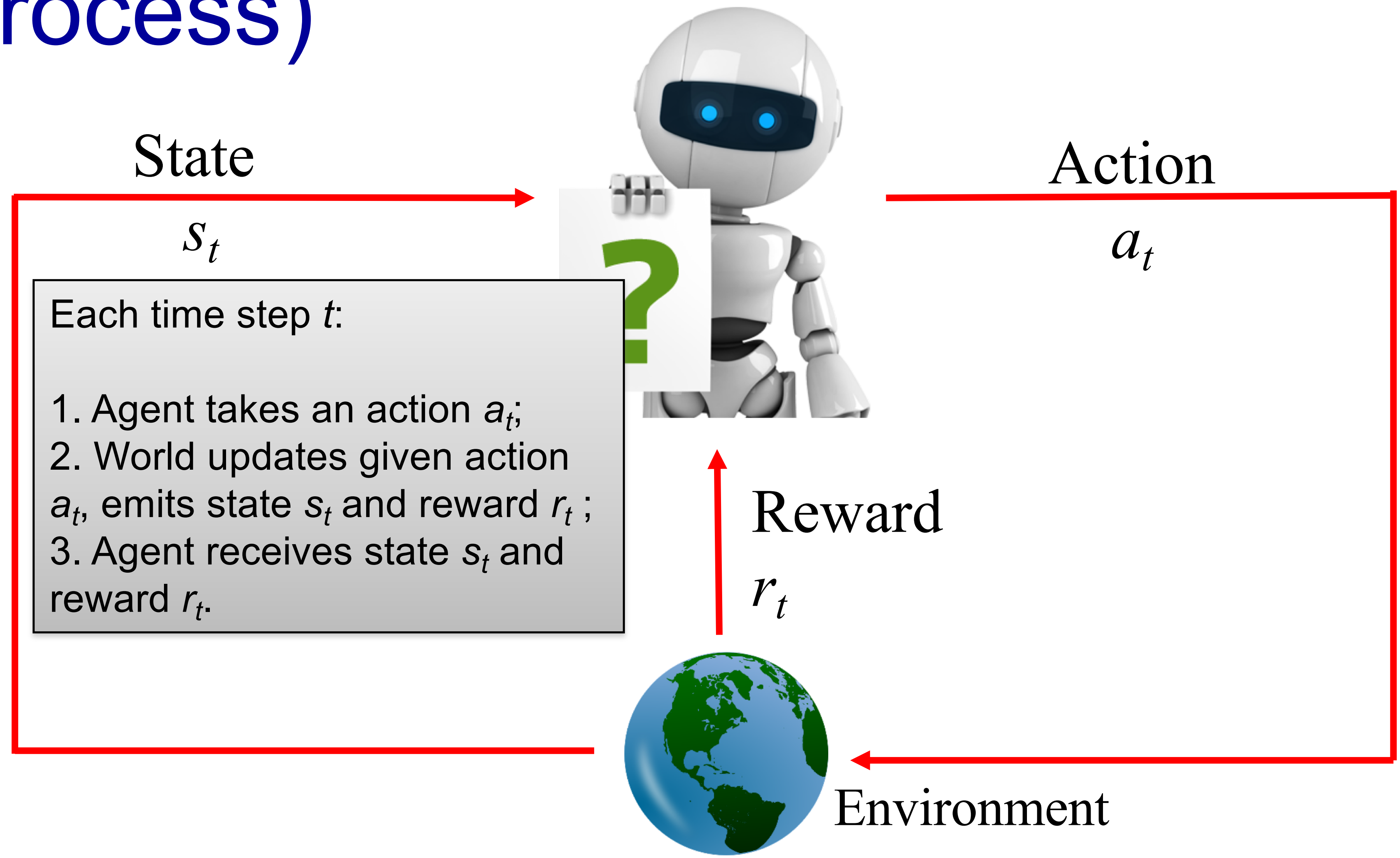
Today's topics

- Reinforcement Learning Components
  - State, History, Markov Property
  - Stochastic vs deterministic model and policy
  - 3 key components: Model, Value function, Policy
- Model-based Planning
  - Policy Evaluation, Policy Iteration, Value Iteration
- Project 1 demo and description.

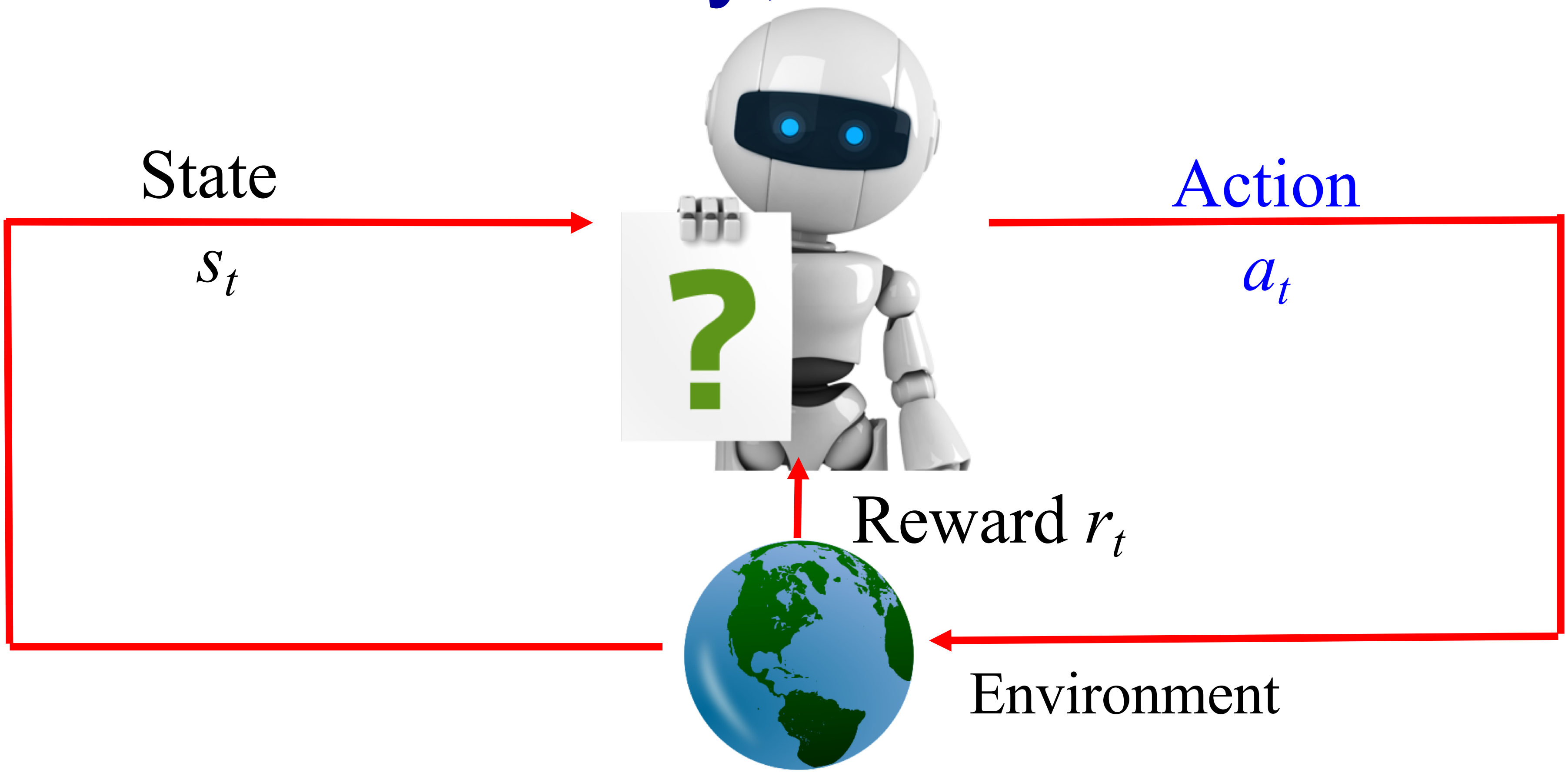
Reinforcement Learning Components



Agent-Environment interactions over time (sequential decision process)



Interaction history, Decision-making



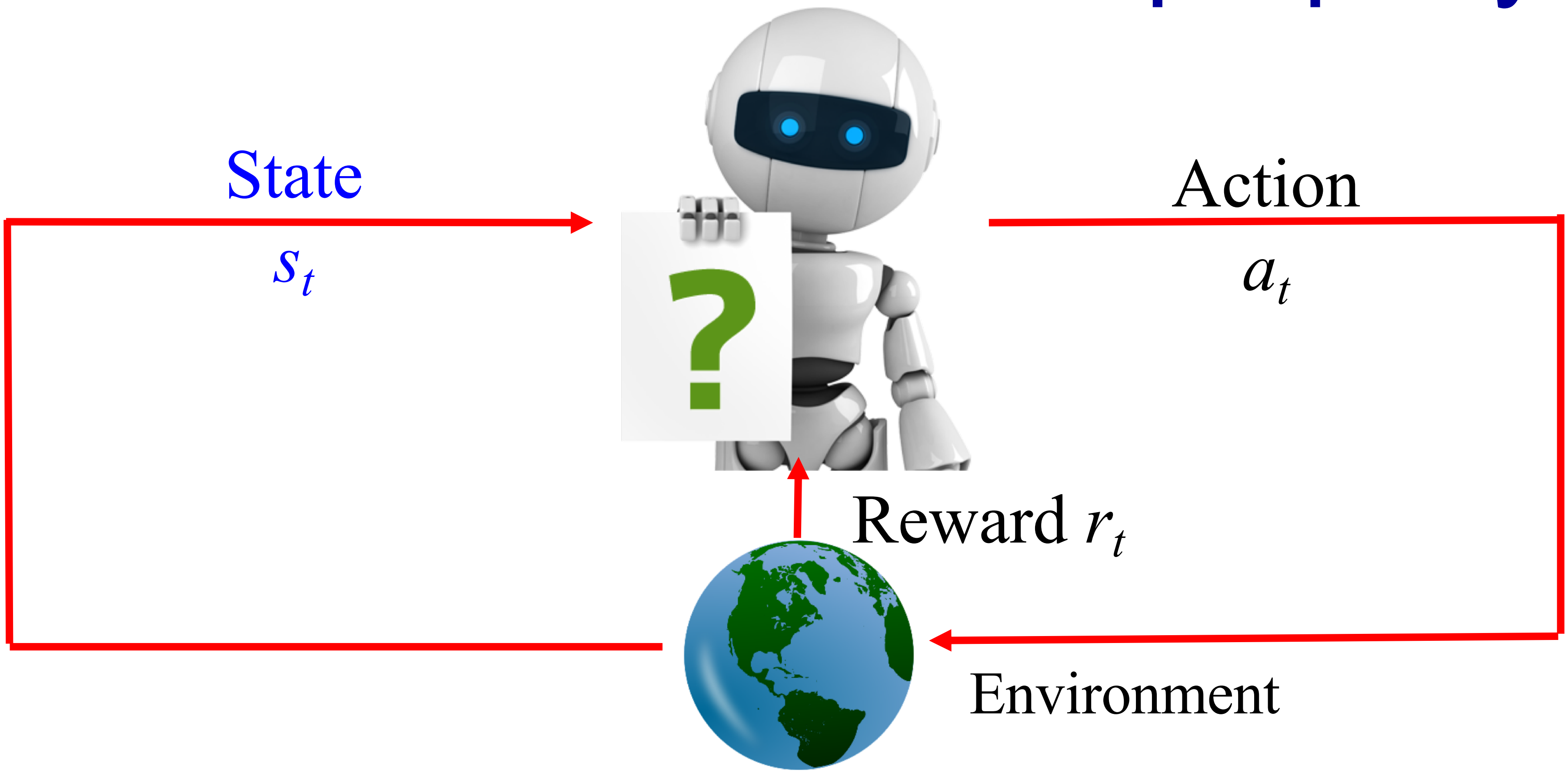
History $h_t = (a_1, s_1, r_1, \ldots, a_t, s_t, r_t)$

Agent chooses action a_{t+1} based on history h_t

State: s_t

In many cases, for simplicity, s_t is *observation at t.*

State transition & Markov property



Transition Probability $p(s_{t+1}|s_t, a_t)$

State s_t is Markov if and only if:

$$p(s_{t+1}|s_t, a_t) = p(s_{t+1}|h_t, a_t)$$

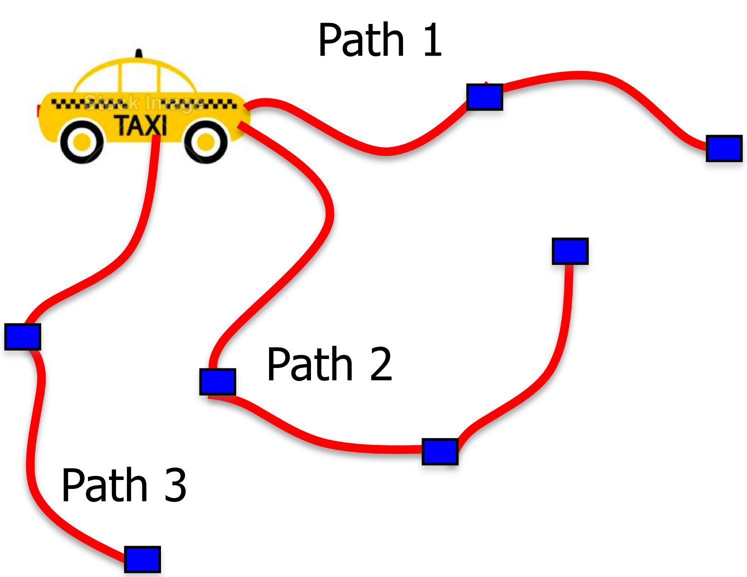
Future is independent of past, given present.

Questions: Markov or not?

A taxi driver seeks for Passengers:

State (observation):
(Current location,
with or without passenger)

Action: A direction to go



Hypertension control

State:
(current blood pressure)

Action: take medication or not

More on Markov Property

?

1. **Does Markov Property always hold?**
    1. No
2. **What if Markov Property does not hold?**

More on Markov Property

?

1. **Does Markov Property always hold?**
   1. **No**
2. **What if Markov Property does not hold?**
   1. Make it Markov by setting state as the history: $s_t = h_t$

Again, in practice, we often assume the most recent observation as s_t is sufficient statistic of history.

State representation has big implications for:
1. Computational complexity
2. Data required
3. Resulting performance

Today's topics

- Reinforcement Learning Components
  - State, History, Markov Property
  - Stochastic vs deterministic model and policy
  - 3 key components: Model, Value function, Policy
- Model-based Planning
  - Policy Evaluation, Policy Iteration, Value Iteration
- Project 1 demo and description.

Deterministic vs Stochastic Environment Model

Deterministic: Given history & action, single *state & reward*

Common assumption in robotics and controls

$p(s_{t+1}| s_t, a_t) = 1, s_{t+1} = s$
$p(s_{t+1}| s_t, a_t) = 0, s_{t+1} \neq s$

$r(s_t, a_t) = 3, s_t = s, a_t = a$

Stochastic: Given history & action, many potential *states & rewards*

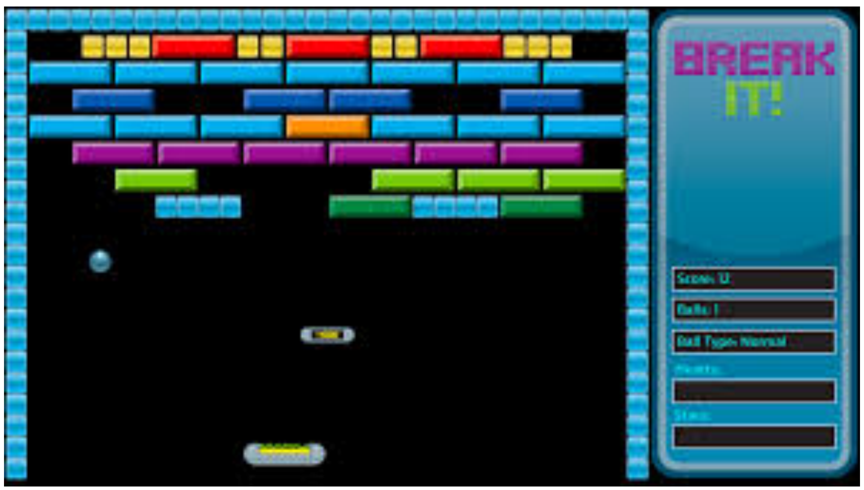
Common assumption for customers, patients, hard to model domains

$0 \leq p(s_{t+1}| s_t, a_t) < 1$

$P[r(s_t, a_t) = 3] = 50\%,$
$P[r(s_t, a_t) = 5] = 50\%,$
$s_t = s, a_t = a$

Questions:
Deterministic vs Stochastic?

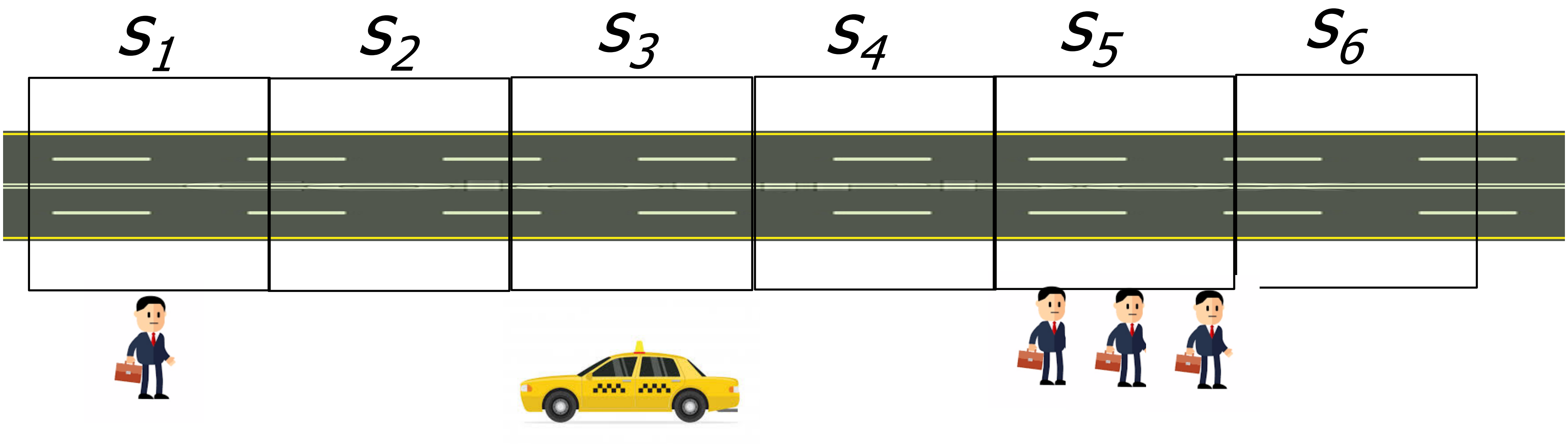
Breakout game



Hypertension control

For both transition and reward

Example: Taxi passenger-seeking task as a decision-making process



States: Locations of taxi ($s_1, \ldots, s_6$) on the road
Actions: Left or Right
Rewards:
+1 in state s_1
+3 in state s_5
0 in all other states

RL components

- **Model:** Representation of how the world changes in response to agent's action
- **Policy:** function mapping agent's <u>states</u> to <u>action</u>
- **Value function:** <u>Future rewards</u> from being in a state and/or action when following a particular policy

RL components

- **Model:** Representation of how the world changes in response to agent's action
- **Policy:** function mapping agent's states to action
- **Value function:** Future rewards from being in a state and/or action when following a particular policy

RL components: Model

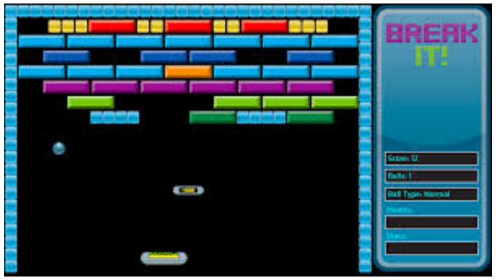
- Agent's representation of how the world changes in response to agent's action, with two parts:

Transition model
predicts next agent state

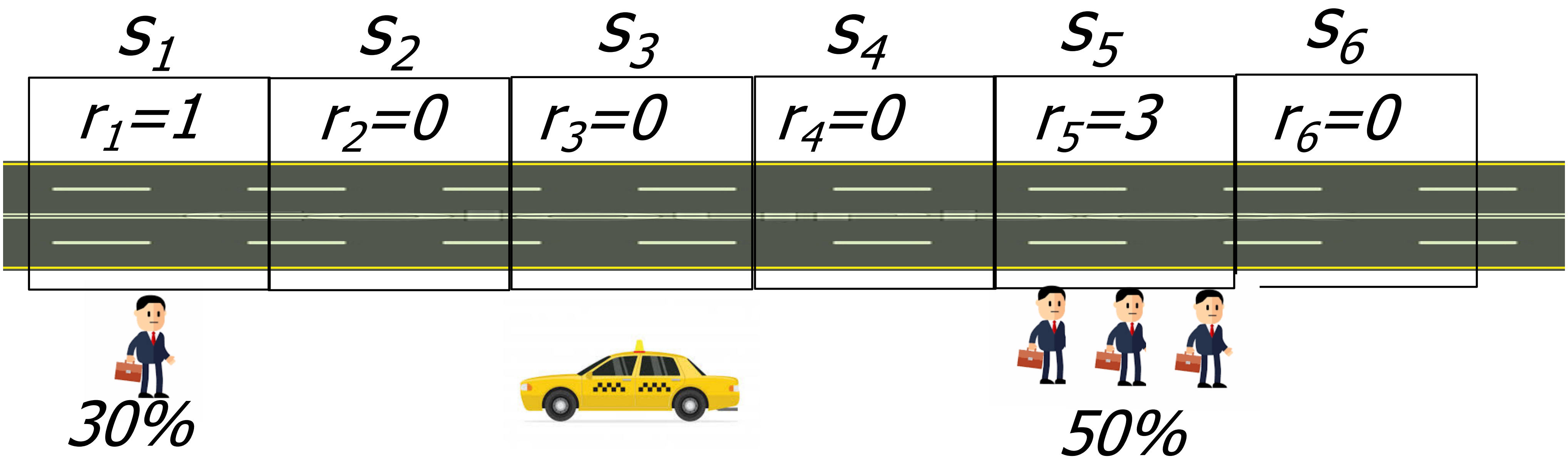
$$p(s_{t+1} = s' \mid s_t = s, a_t = a)$$

Reward model
predicts immediate reward

$$r(s_t = s, a_t = a) = \mathbb{E}[r_t \mid s_t = s, a_t = a]$$



Taxi passenger-seeking task
Stochastic Markov Model



Taxi agent's transition model:

0.5 = p(s_3|s_3, right) = p(s_4|s_3, right)

0.5 = p(s_4|s_4, right) = P(s_5|s_4, right)

Numbers above show RL agent's reward model ,

r_1=1 with 30% chance, and 0 with 70% chance

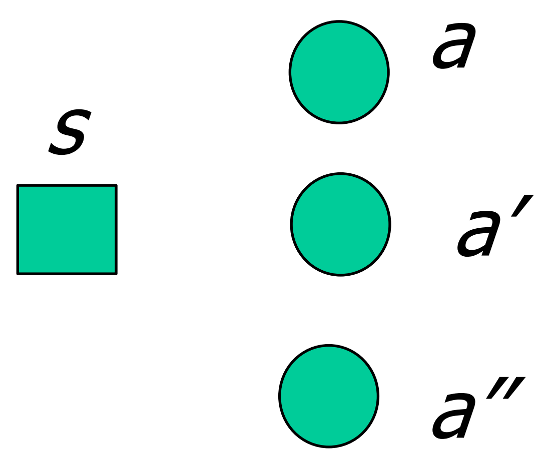
r_5=3 with 50% chance, and 0 with 50% chance

RL components

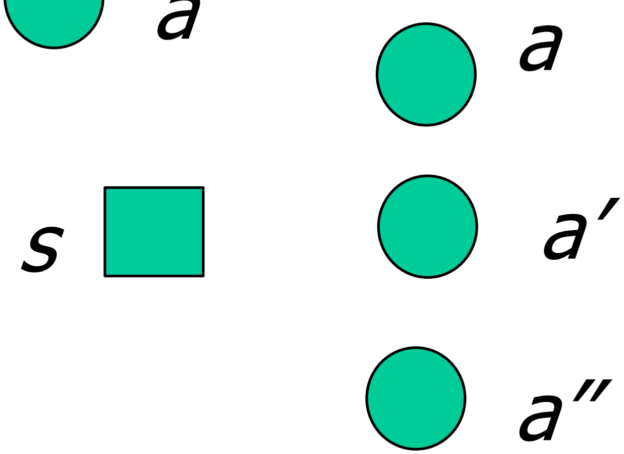
- **Model:** Representation of how the world changes in response to agent's action
- **Policy:** function mapping agent's states to action
- **Value function:** Future rewards from being in a state and/or action when following a particular policy

RL components: Policy

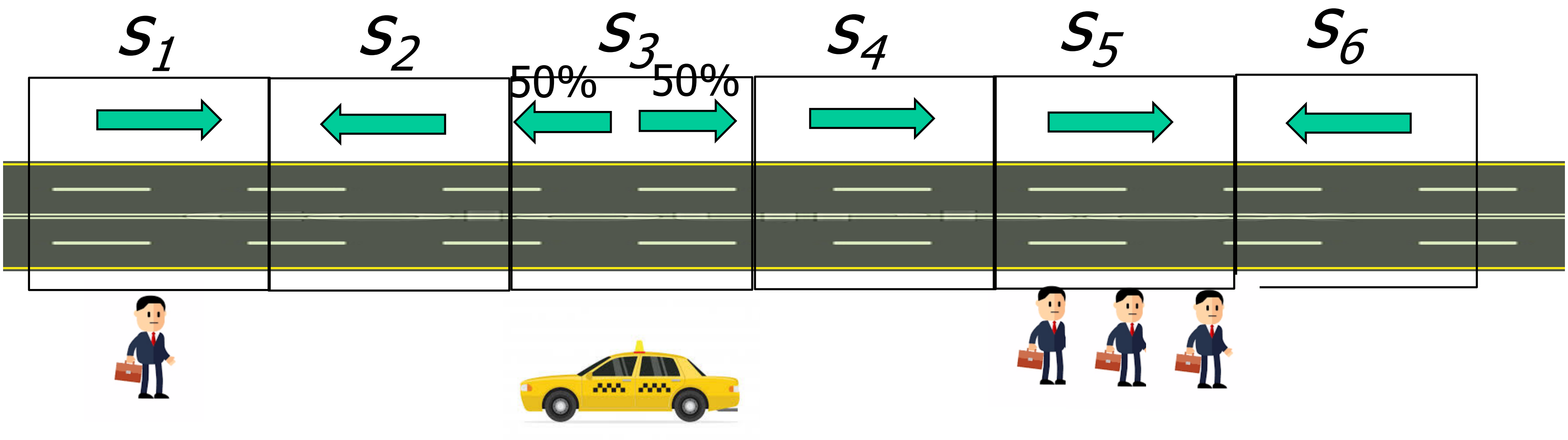
- Policy π determines how the agent chooses actions
  - $\pi : S \rightarrow A$, mapping from states to actions
- Deterministic policy:
  - $\pi(s) = a$
  - *In the other word,*
    - $\pi(a|s) = 0$,
    - $\pi(a'|s) = \pi(a''|s)=0$,



- Stochastic policy:
  - $\pi(a|s) = \Pr(a_t = a|s_t = s)$



Taxi passenger-seeking task
Policy



Action set: {left, right}

Policy presented by arrow.

Q1: Is this a deterministic or stochastic policy?

Q2: Give an example of another policy type?

RL components

- **Model:** Representation of how the world changes in response to agent's action
- **Policy:** function mapping agent's states to action
- **Value function:** Future rewards from being in a state and/or action when following a particular policy

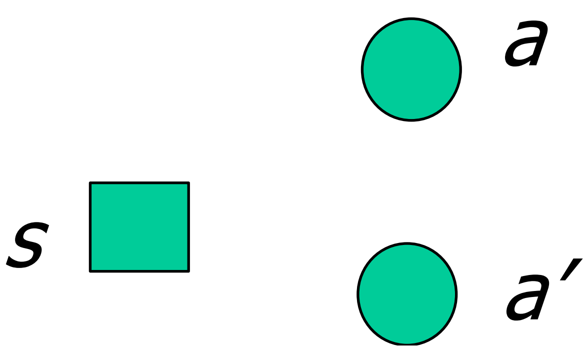
?

RL components: Value Function

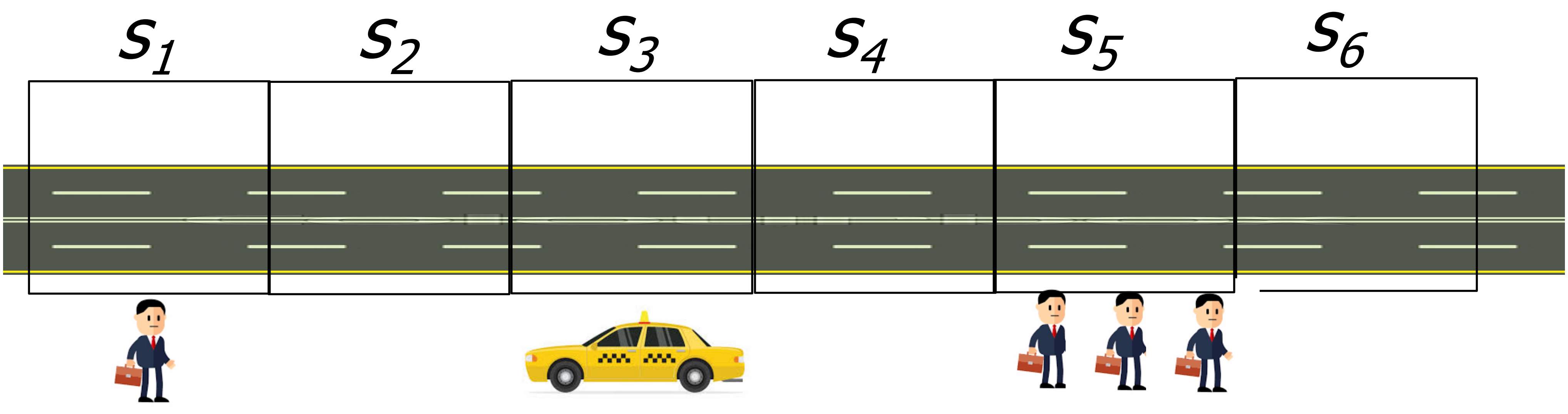
- Value function V^{π}: expected discounted sum of future rewards under a particular policy π

$$V^{\pi}(s_t = s) = \mathbb{E}_{\pi}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots | s_t = s]$$

- Discount factor γ weighs immediate vs future rewards, with γ in [0,1].
- It can be used to quantify goodness/badness of states and actions
- And decide how to act by comparing policies



Taxi passenger-seeking task: Value function



$$V^{\pi}(s_t = s) = \mathbb{E}_{\pi}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots | s_t = s]$$

Discount factor, $\gamma = 0$

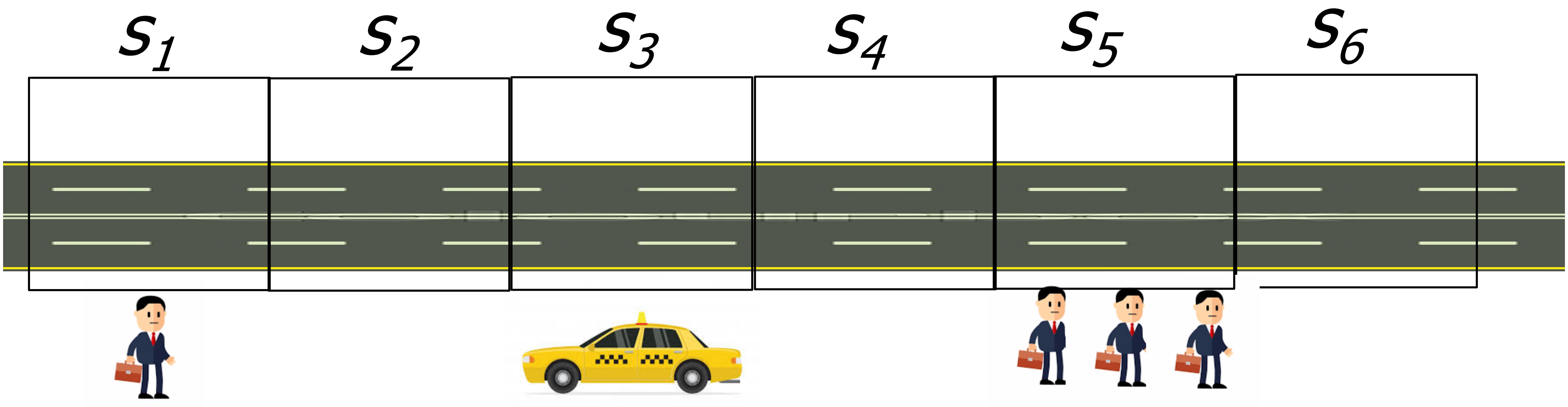
Policy #1: $\pi(s_1) = \pi(s_2) = \cdots = \pi(s_6) = \text{right}$

Q: V^π?

Policy #2: $\pi(\text{left}|s_i) = \pi(\text{right}|s_i) = 50\%$, for i=1,…,6

Q: V^π?

Taxi passenger-seeking task: Value function



$$V^\pi(s_t = s) = \mathbb{E}_\pi[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots | s_t = s]$$

Discount factor, $\gamma = 0$

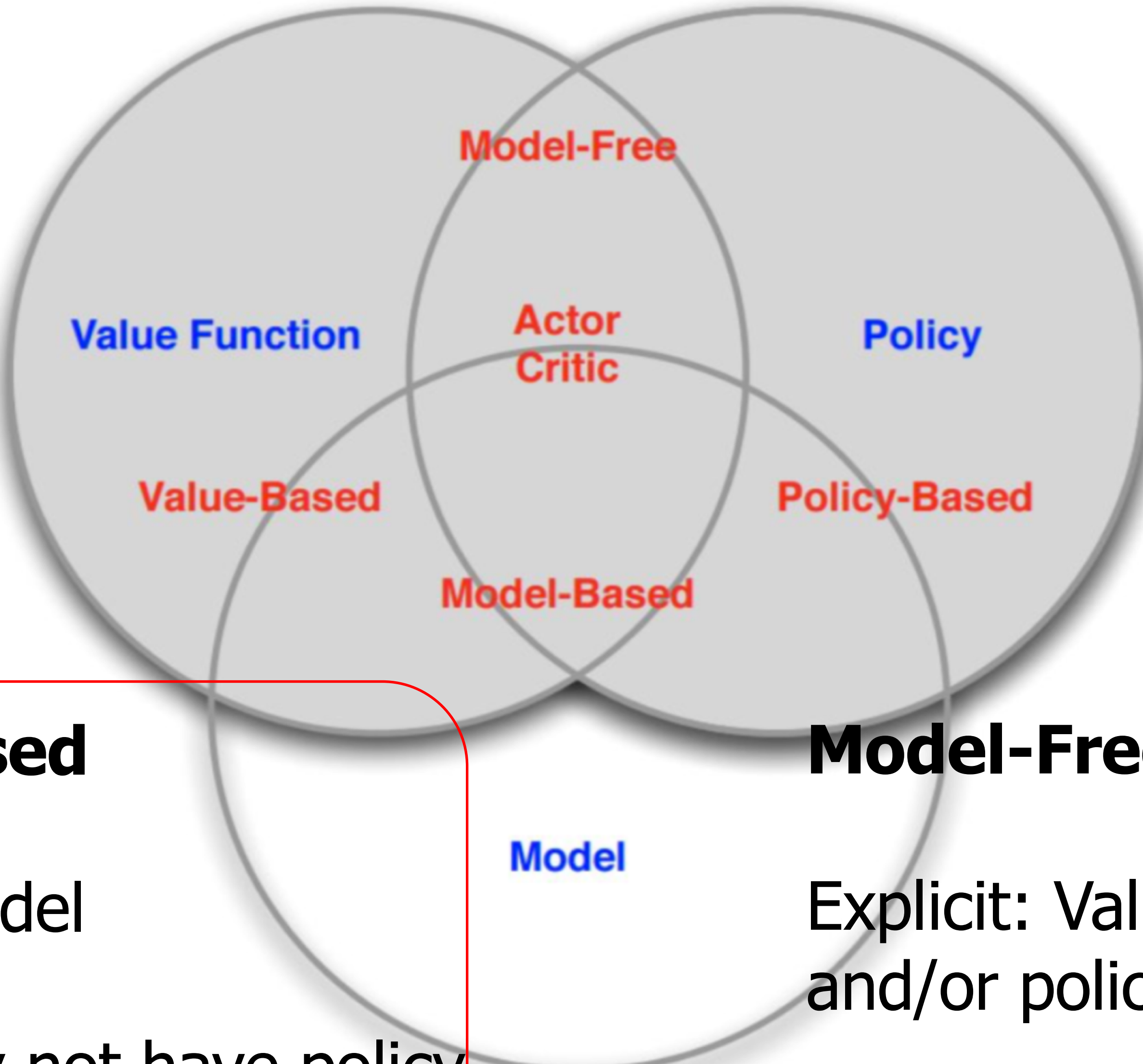
Policy #1: $\pi(s_1) = \pi(s_2) = \cdots = \pi(s_6) =$ right

Q: V^π? $[V^\pi(s_1), \ldots, V^\pi(s_6)]=[1,0,0,0,3,0]$

Policy #2: $\pi(\text{left}|s_1) = \pi(\text{right}|s_1) = 50\%$, for i=1,…,6

Q: V^π? $[V^\pi(s_1), \ldots, V^\pi(s_6)]=[1,0,0,0,3,0]$

Types of RL agents/algorithms



Model-based

Explicit: Model

May or may not have policy and/or value function

Model-Free: (Next Week)

Explicit: Value function and/or policy function

No model

Today’s topics

- Reinforcement Learning Components
  - Model, Value function, Policy
- Model-based Planning
  - MDP model
  - Policy Evaluation, Policy Iteration, Value Iteration
- Project 1 demo and description.

MDP

- Markov Decision Process

Markov Decision Process (MDP)

- Markov Decision Process is Markov Reward Process + actions
- Definition of MDP
  - S is a (finite) set of Markov states $s \in S$
  - A is a (finite) set of actions $a \in A$
  - P is dynamics/transition model for **each action**, that specifies $P(s_{t+1} = s' | s_t = s, a_t = a)$
  - R is a reward function[1]

$$R(s_t = s, a_t = a) = \mathbb{E}[r_t | s_t = s, a_t = a]$$

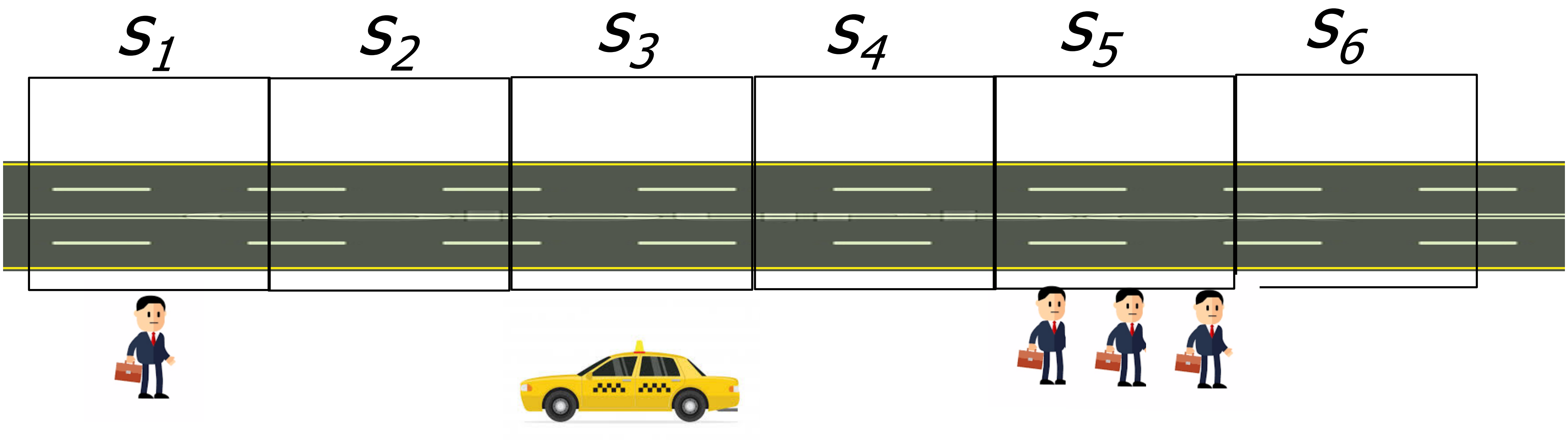
  - Discount factor $\gamma \in [0, 1]$
- MDP is a tuple: (S, A, P, R, γ)

Transition Model
Reward Model
Policy function
Value function

[1] Reward is sometimes defined as a function of the current state, or as a function of the (state, action, next state) tuple. Most frequently in this class, we will assume reward is a function of state and action

Taxi passenger-seeking task: MDP

Transition Model
Reward Model
Policy function
Value function



$\xleftarrow{\text{a1}}$

$$P(s'|s,a_1) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$\xrightarrow{\text{a2}}$

$$P(s'|s,a_2) = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

deterministic transition model

MDP Policies

- Policy specifies what action to take in each state
  - Can be deterministic or stochastic
- For generality, consider as a conditional distribution
  - Given a state, specifies a distribution over actions
- Policy: $\pi(a|s) = P(a_t = a|s_t = s)$

MDP Policy Evaluation, Iterative Algorithm

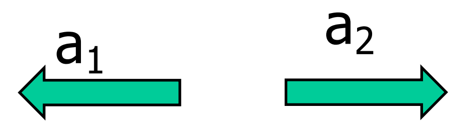
Transition Model
Reward Model
Policy function
Value function

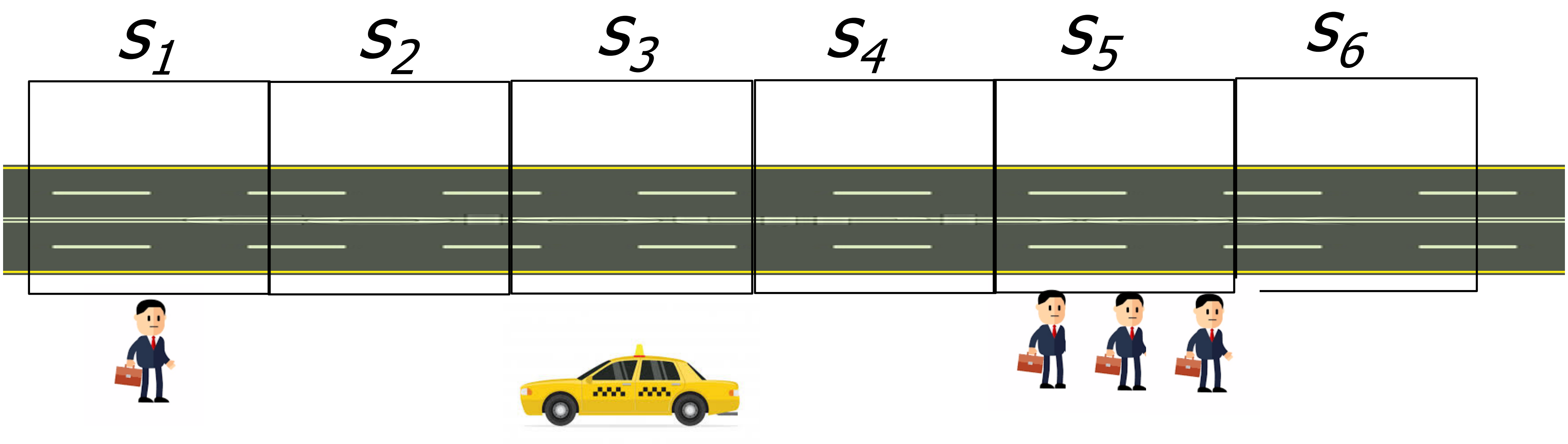
- Initialize $V_0(s) = 0$ for all s
- For $k = 1$ until convergence
  - For all s in S

$$V_k^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} p(s'|s, \pi(s)) V_{k-1}^\pi(s')$$

- This is a **Bellman backup** for a particular policy

Taxi passenger-seeking task: MDP Policy Evaluation

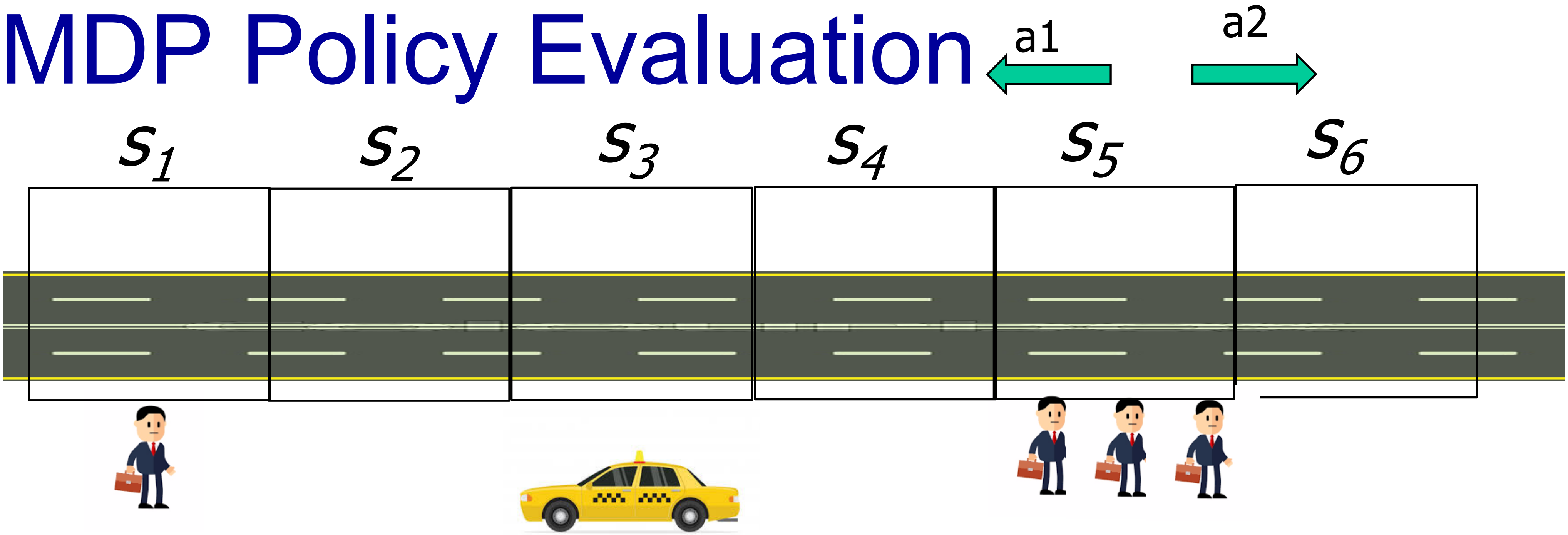




- Let π(s) = a_1 ∀s. γ = 0.
- What is the value of this policy?

$$V_k^{\pi}(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} p(s'|s, \pi(s)) V_{k-1}^{\pi}(s')$$

Taxi passenger-seeking task: MDP Policy Evaluation



- Let π(s) = a_1 ∀s. γ = 0.
- What is the value of this policy?

$$V_k^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} p(s'|s, \pi(s)) V_{k-1}^\pi(s')$$

- $V^\pi = [1\ 0\ 0\ 0\ 3\ \ 0]$

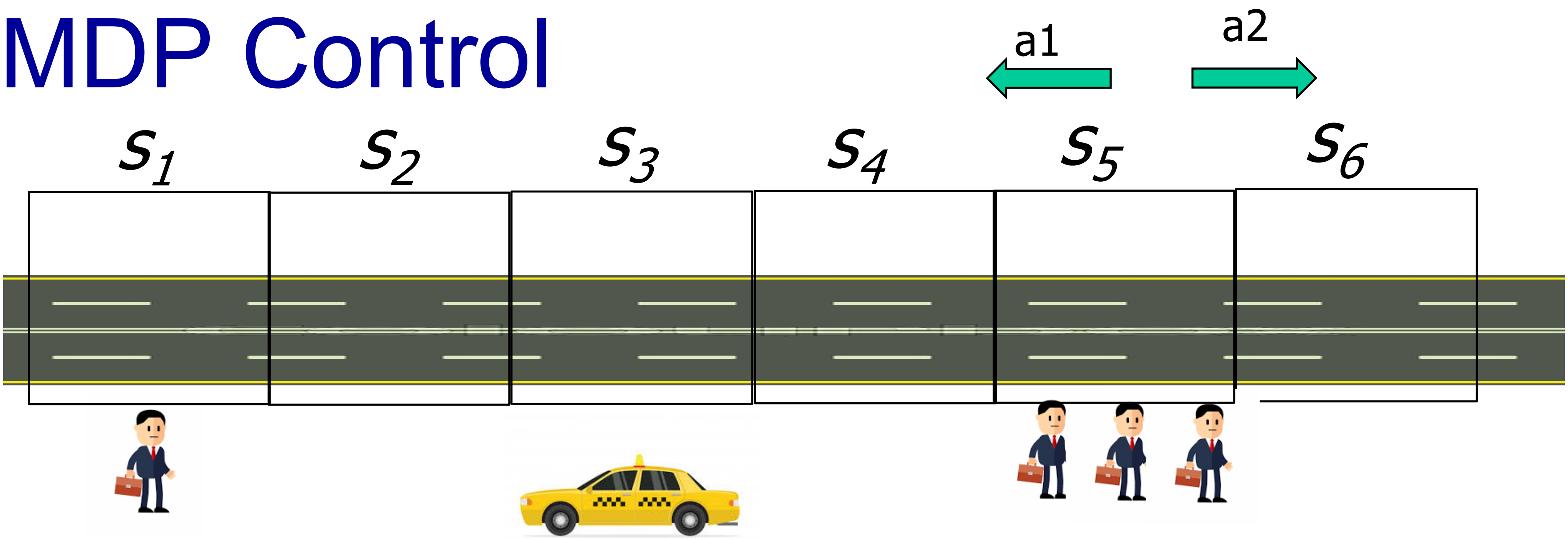
MDP Control

- Compute the optimal policy

$$\pi^*(s) = \arg\max_{\pi} V^{\pi}(s)$$

- There **exists a unique optimal value function**
- Optimal policy for a MDP in an infinite horizon problem is deterministic

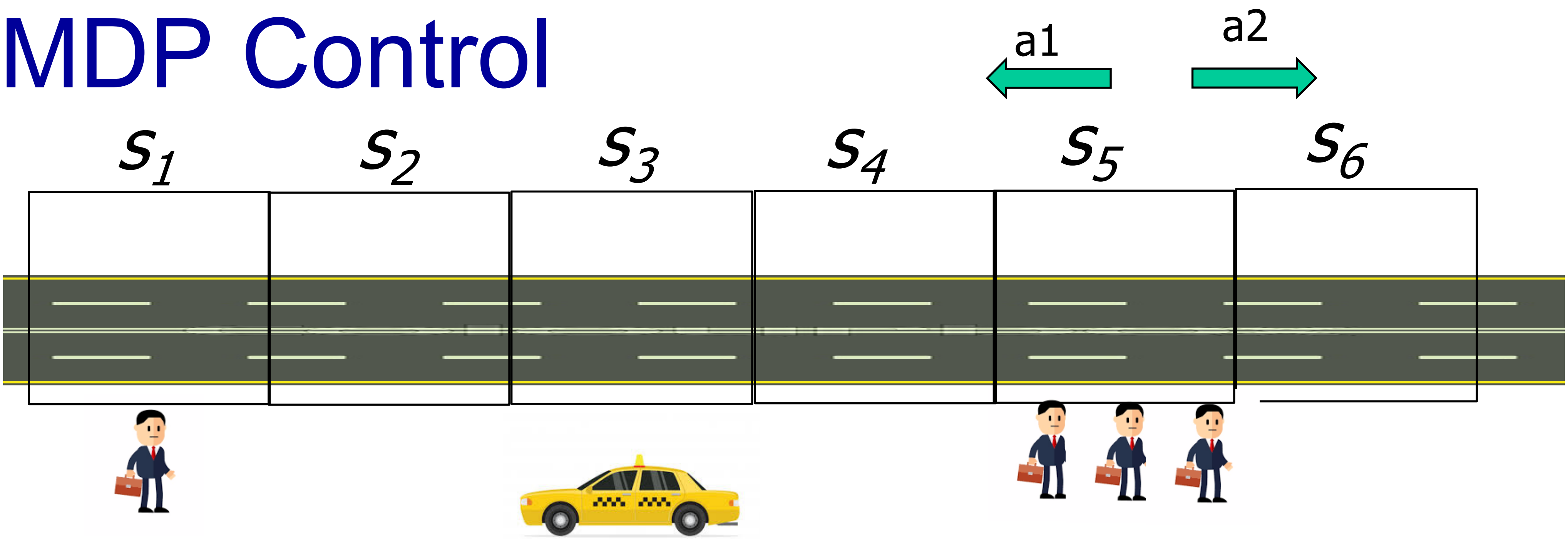
Taxi passenger-seeking task: MDP Control



- 6 discrete states (location of the taxi)
- 2 actions: Left or Right
- How many deterministic policies are there?

- Is the optimal policy for a MDP always unique?

Taxi passenger-seeking task: MDP Control



- 6 discrete states (location of the taxi)
- 2 actions: Left or Right
- How many deterministic policies are there?
- 2^6
- Is the optimal policy for a MDP always unique?
  No, there may be two states that have the same optimal value function

MDP Control

- Compute the optimal policy

$$\pi^*(s) = \arg\max_{\pi} V^{\pi}(s)$$

- There exists a unique optimal value function
- Optimal policy for a MDP in an infinite horizon problem (agent acts forever is
  - Deterministic
  - Stationary (does not depend on time step)
  - Unique? Not necessarily, may have state-actions with identical optimal values

Policy Search

- One option is searching to compute best policy
- Number of deterministic policies is $|A|^{|S|}$
- Policy iteration is generally more efficient than enumeration

MDP Policy Iteration (PI)

- Set $i = 0$
- Initialize $\pi_0(s)$ randomly for all states s
- While $i == 0$ or $\|\pi_i - \pi_{i-1}\|_1 > 0$ (L1-norm, measures if the policy changed for any state):
  - $V^{\pi_i} \leftarrow$ MDP V function policy **evaluation** of π_i
  - $\pi_{i+1} \leftarrow$ Policy **improvement**
  - $i = i + 1$

New Definition: State-Action Value Q

- State-action value of a policy

$$Q^{\pi}(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s'|s, a) V^{\pi}(s')$$

- Take action a, then follow the policy π

Policy Improvement

- Compute state-action value of a policy π_i
  - For s in S and a in A:

$$Q^{\pi_i}(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s'|s, a) V^{\pi_i}(s')$$

- Compute new policy π_{i+1}, for all $s \in S$

$$\pi_{i+1}(s) = \arg\max_a Q^{\pi_i}(s, a) \ \forall s \in S$$

MDP Policy Iteration (PI)

- Set $i = 0$
- Initialize $\pi_0(s)$ randomly for all states s
- While $i == 0$ or $\|\pi_i - \pi_{i-1}\|_1 > 0$ (L1-norm, measures if the policy changed for any state):
  - $V^{\pi_i} \leftarrow$ MDP V function policy **evaluation** of π_i
  - $\pi_{i+1} \leftarrow$ Policy **improvement**
  - $i = i + 1$

MDP Policy Iteration (PI)

- Set $i = 0$
- Initialize $\pi_0(s)$ randomly for all states s
- While $i == 0$ or $\|\pi_i - \pi_{i-1}\|_1 > 0$ (L1-norm, measures if the policy changed for any state):
  - $V^{\pi_i} \leftarrow$ MDP V function policy **evaluation** of π_i
  - $\pi_{i+1} \leftarrow$ Policy **improvement**
  - $i = i + 1$

If policy doesn't change, can it ever change again?

Is there a maximum number of iterations of policy iteration?

MDP Policy Iteration (PI)

- Set $i = 0$
- Initialize $\pi_0(s)$ randomly for all states s
- While $i == 0$ or $\|\pi_i - \pi_{i-1}\|_1 > 0$ (L1-norm, measures if the policy changed for any state):
  - $V^{\pi_i} \leftarrow$ MDP V function policy **evaluation** of π_i
  - $\pi_{i+1} \leftarrow$ Policy **improvement**
  - $i = i + 1$

If policy doesn't change, can it ever change again?
No

Is there a maximum number of iterations of policy iteration?
$|A|^{|S|}$ since that is the maximum number of policies, and as the policy improvement step is monotonically improving, each policy can only appear in one round of policy iteration unless it is an optimal policy.

MDP: Computing Optimal Policy and Optimal Value

- Policy iteration computes optimal value and policy
- Value iteration is another technique
  - Idea: Maintain optimal value of starting in a state s if have a finite number of steps k left in the episode
  - Iterate to consider longer and longer episodes

Bellman Equation and Bellman Backup Operators

- Value function of a policy must satisfy the Bellman equation

$$V_{k+1}(s) = \max_a \left[R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_k(s') \right]$$

- Bellman backup operator
  - Applied to a value function
  - Returns a new value function
  - Improves the value if possible

$$BV(s) = \max_a \left[R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V(s') \right]$$

  - BV yields a value function over all states s

Going Back to Value Iteration (VI)

- Set $k = 1$
- Initialize $V_0(s) = 0$ for all states s
- Loop until [finite horizon, convergence]:
  - For each state s

$$V_{k+1}(s) = \max_a \left[R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_k(s') \right]$$

- To extract optimal policy if can act for $k+1$ more steps,

$$\pi(s) = \arg\max_a \left[R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_{k+1}(s') \right]$$

Project 1 starts today
Due 2/9 mid-night

- https://users.wpi.edu/~yli15/courses/DS595Spring22/Assignments.html

Any Comments & Critiques?