6.867 Machine Learning Fall 2017

Lecture 23. Reinforcement Learning

http://stellar.mit.edu/S/course/6/fa17/6.867/

Announcements

- Exercise 12 posted
    - Covers Non-negative Matrix Factorization, Reinforcement learning

- Projects
    - Final project report due on December 12
    - Please reach out to TAs / Instructors if you need any help

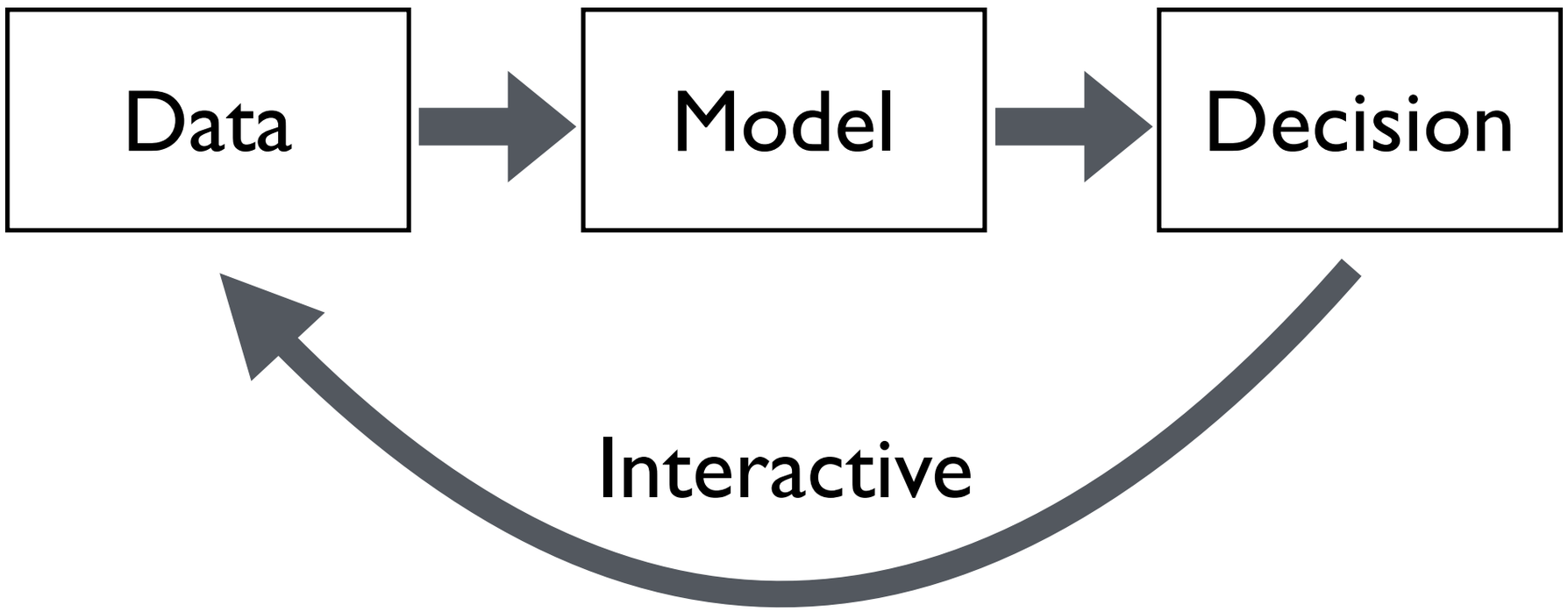
- Class evaluation: let us use *few* minutes *now*
- OH Monday 10-11am G464
- Today is my last lecture (Prof Kaelbling will deliver final lecture on 12/12)

Outline

- Reinforcement Learning (RL)
    - Examples, Applications
- Markov Decision Process
    - Formalism for generic RL
- Model-based policies
    - Value Iteration
    - Policy Iteration
    - Linear programing formulation
- Model-free policies
    - Multi-arm bandit: an instance of Reinforcement learning

From The Lens of Decision Making

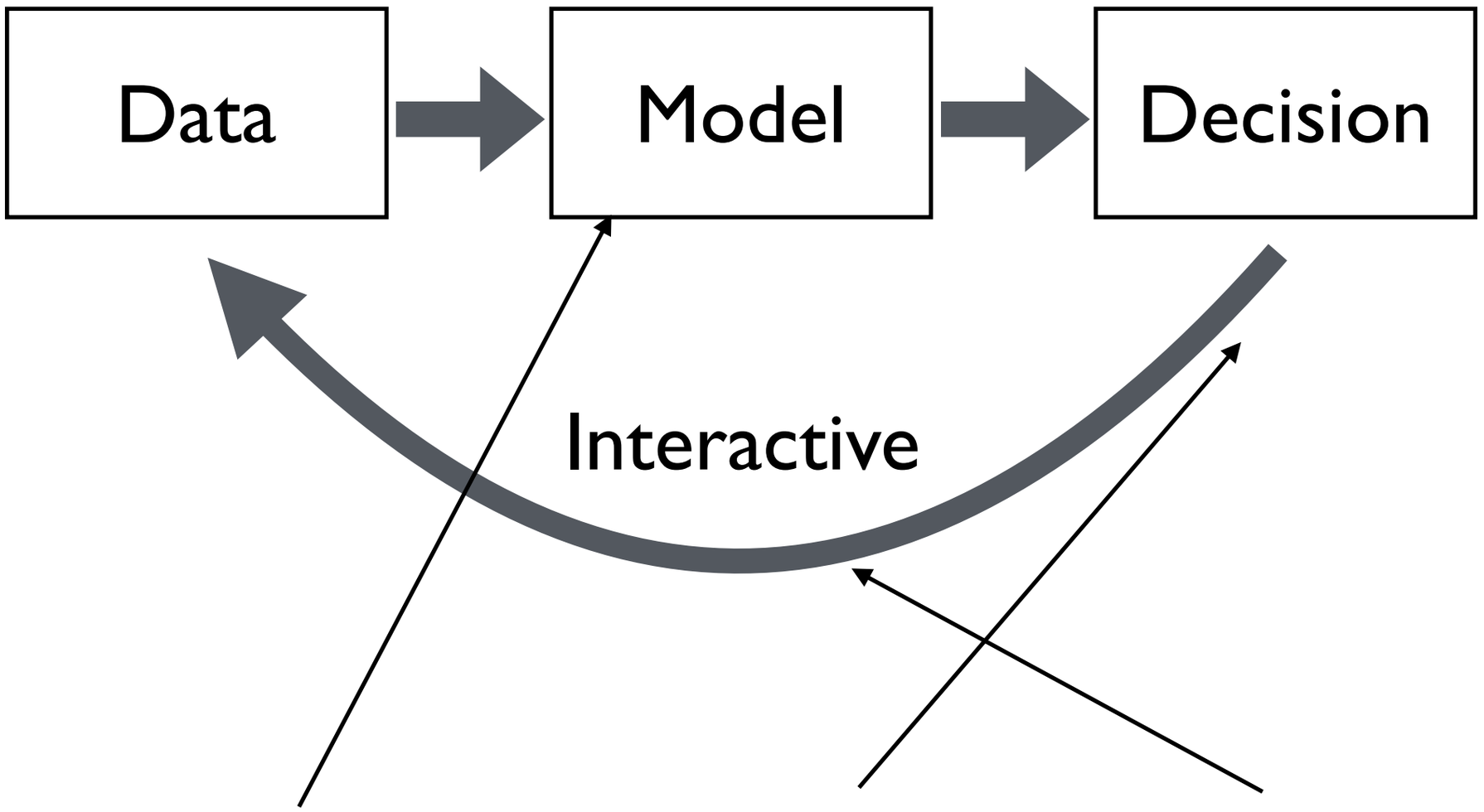
supervised learning



reinforcement learning

explore vs exploit

Goal of Reinforcement Learning



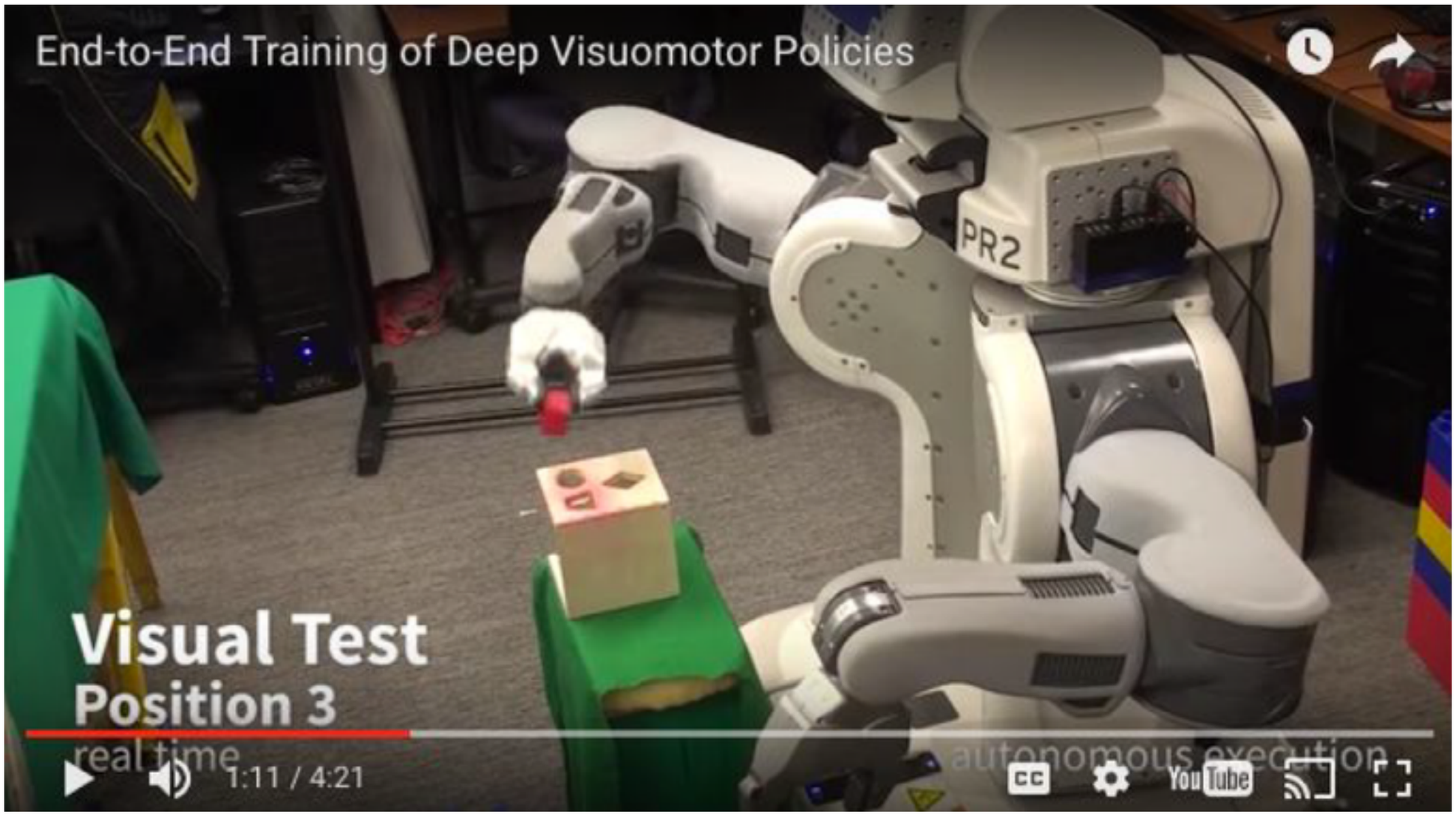
In Short: Learn to make good sequence of decisions

Examples

Atari using Deep Q-learning by DeepMind

https://www.youtube.com/watch?v=V1eYniJ0Rnk

Examples



Robotics, Finn et al JMLR 2017

https://youtu.be/CE6fBDHPbP8?t=71

Examples

Educational Game. Refraction 1 by Mandel et al 2014

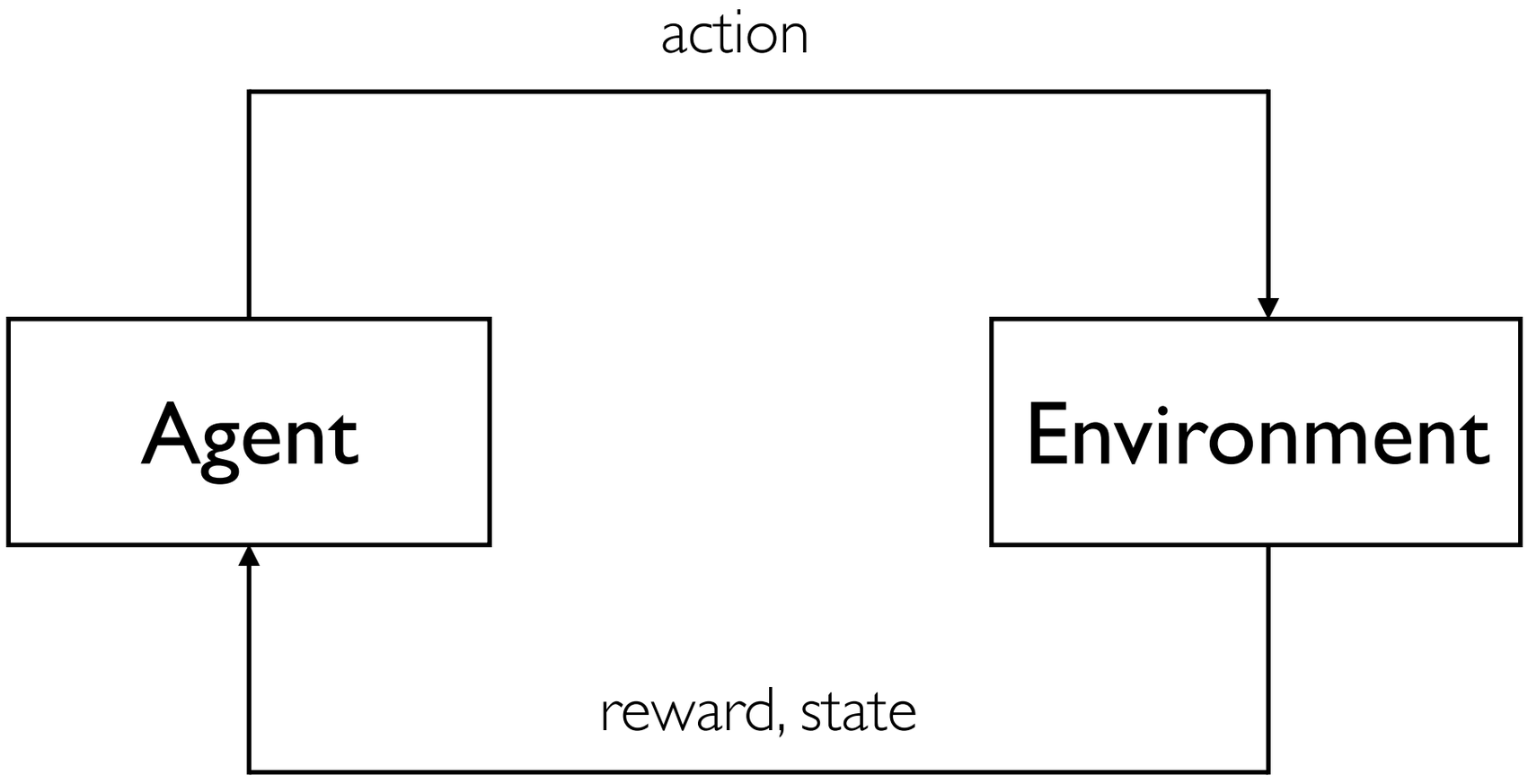
https://www.youtube.com/watch?v=XSWKOuLoitQ

Examples

Learning from Senses, Yeung et al 2016

https://tinyurl.com/ybfvwfs3

Markov Decision Process



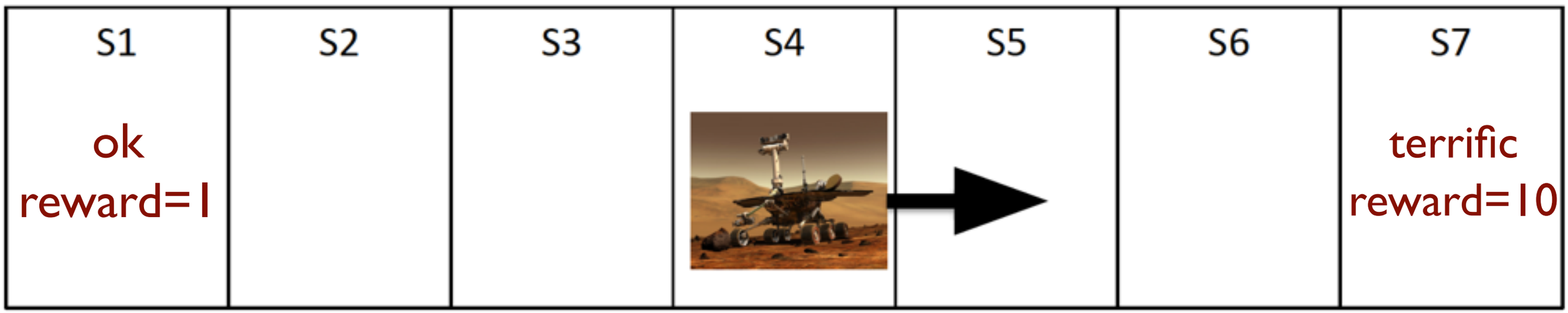
Policy: past (actions, states, rewards) → next action

Markov Decision Process (MDP)

- MDP tuple (S, A, p, R, γ)
  - State space S
  - Action space A
  - Stochastic transition kernel $p(s'|s, a)$
    - probability of reaching s' given in state s with action a
  - Reward $R : S \times A \to \mathbb{R}$
  - Discount factor $\gamma \in [0, 1]$
- Policy $\pi : S \to A$
- Objective: find π that maximizes reward for every starting state $s \in S$

$$\mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s\Big]$$

Example of Navigation



- States: *seven locations*
- Actions: *left* or *right*
- Transition Kernel:
    - go left / right with probability 1 if action is *left / right*
- Reward:
    - 10 if state is 7, 1 if state is 1 and 0 otherwise

Value Function

- MDP tuple (S, A, p, R, γ)
- Policy $\pi : S \to A$
- Value function $V^\pi : S \to \mathbb{R}$

$$V^\pi(s) = \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^t R(s_t, \pi(s_t)) | s_0 = s\Big]$$

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

immediate reward

discounted future reward

Value Function

- MDP tuple (S, A, p, R, γ)
- Policy $\pi : S \to A$
- Optimal policy $\pi^\star : S \to A$ such that

$$V^{\pi^\star}(s) \geq V^{\pi}(s), \quad \forall s \in S,\ \pi : S \to A$$

- It satisfies the so called *Bellman* equation

$$V^{\pi^\star}(s) = \max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^{\pi^\star}(s') \right]$$

- Given $V^{\pi^\star}$
  - How to find $\pi^\star$

Q Function

- MDP tuple (S, A, p, R, γ)
- Define $Q^\pi : S \times A \to \mathbb{R}$ for policy $\pi : S \to A$

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

- Optimal Q function

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

- Hence, optimal policy $\pi^\star : S \to A$

$$\pi^\star(s) = \arg\max_{a \in A} Q^\star(s,a)$$

Optimality Through Linear Programming

- We wish to find $V^{\pi^\star}$
- We know that

$$V^{\pi^\star}(s) = \max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^{\pi^\star}(s') \right]$$

$$\geq \left[R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^{\pi^\star}(s') \right], \quad \forall a \in A$$

- This suggests linear program:

$$\text{minimize} \sum_s V(s)$$

$$\text{such that } V(s) \geq \left[R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V(s') \right],$$

$$\forall a \in A, \ \forall s \in S.$$

Optimality Through Fixed Point Equations

- MDP tuple (S, A, p, R, γ)
- Optimal value function

$$V^{\pi^\star}(s) = \max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^{\pi^\star}(s') \right]$$

- Optimal Q function

$$Q^\star(s,a) = R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) \max_{a' \in A} Q^\star(s',a')$$

- Optimal policy $\pi^\star : S \to A$

$$\pi^\star(s) = \arg\max_{a \in A} Q^\star(s,a)$$

Optimal Policy Via Solving Fixed Point

- We wish to find $x^\star$ such that

$$f(x^\star) = x^\star$$

- A simple iterative algorithm
  - initialize x^0
  - iteratively update

$$x^{k+1} = f(x^k)$$

  - stop upon convergence

Value Iteration

- Initialize:

$$V^0(s) = 0, \quad \forall\ s \in S$$

- Iteratively update: for each $s \in S$

$$V^{k+1}(s) = \max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^k(s') \right]$$

- Question:
  - Does it converge?

Value Iteration

- Let us denote

$$x^k = [V^k(s)]_{s \in S}$$

- And, value iteration as

$$x^{k+1} = f(x^k)$$

- It can be checked that (w.r.t. to infinity norm)

$$|f(x) - f(x')| \leq \gamma |x - x'|, \quad \forall x, x' \in \mathbb{R}^{|S|}$$

- This will imply that Value Iteration will converge to a unique fixed point

Policy Iteration

- Value iteration attempts to solve Bellman fixed point directly
- Alternatively, we can iterative over policies
  - Start with some policy, say π^0
  - Iteratively update policy as follows:
    - *Evaluate* value function V^{π^k} for policy π^k
    - Update policy π^{k+1} : for $s \in S$

$$\pi^{k+1}(s) = \arg\max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^{\pi^k}(s') \right]$$

Policy Evaluation

- How to evaluate value function of a policy, π?
- That is, solve for equation

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

- With notation $x^{\pi} = [V^{\pi}(s)]_{s \in S}$, we wish to solve linear equation

$$x^{\pi} = \mathbf{r}^{\pi} + \gamma P^{\pi} x^{\pi} \;\Leftrightarrow\; x^{\pi} = \left(\mathbf{I} - \gamma P^{\pi}\right)^{-1} \mathbf{r}^{\pi}$$

- In above, we use notation

$$\mathbf{r}^{\pi} = [R(s, \pi(a))]_{s \in S} \qquad \mathbf{P}^{\pi} = [p(s'|s, \pi(a))]_{s', s \in S}$$

Policy Iteration

- Does it converge?
- Each iteration of policy iteration
    - Improves the value function till convergence
    - That is:

$$V^{\pi^{k+1}}(s) \geq V^{\pi^k}(s), \quad \forall s \in S$$

- Number of different policies: $|A|^{|S|}$
- Therefore, policy iteration must converge in those many steps!

Value Iteration vs Policy Iteration

We can solve MDP using two approaches

value iteration	policy iteration
each iteration is *cheap(er)*	each iteration is *expensive*
takes more iterations	takes *fewer* iterations

is that it?
what happened to using "deep RL"?

Decisions under Uncertainty (and interactions)

Learn Unknown Model	Multi-Arm Bandit	Reinforcement Learning
Known Stochastic Model	Supervised Learning	MDP
	Actions Don't Impact State	Actions Change State

Multi-arm Bandit

- Example:
  - You are an e-commerce web-portal
  - You want your customers to *like* (or buy or …)
  - You have N different possible *landing* pages
  - Which *landing* page should you choose for new arriving customer?
  - A priori you have no idea about
    - How customers would react to each of the N options
    - And you do not know anything about every incoming customer

Multi-arm Bandit

- Formally:
  - Actions A (= N *landing* pages) = Arms
  - States S = single state (say null)
    - That is, transition probability kernel is *trivial*

$$p(\text{null}|\text{null}, a) = 1. \quad \forall a \in A$$

  - Reward (= customer (dis-)liking landing pages shown to her/him)
    - That is,

$$\mathbb{E}\big[R(\text{null}, a)\big] = \mu_a, \quad \forall a \in A$$

Multi-arm Bandit

- Formally:
  - Goal: maximize cumulative reward over a given time horizon

$$\mathbb{E}\Big[\sum_{t=1}^{T} R(a_t, \text{null})\Big] \;=\; \mathbb{E}\Big[\sum_{t=1}^{T} \mu_{a_t}\Big]$$

  - The *optimal* policy
    - Always choose $a^\star$ such that

$$a^\star \in \arg\max_{a \in A} \mu_a$$

  - But we do not know the model parameters!

Multi-arm Bandit

- A simple policy:
  - maintain "importance" for each action (arm) w_a^t for $a \in A,\ t \geq 0$
    - initially $w_a^0 = 1,\ \forall a \in A$
  - at time t, choose action at random so that

$$\mathbb{P}(a_t = a) = \frac{w_a^t}{\sum_{a' \in A} w_{a'}^t}$$

  - adjust "importance" as (for small enough $\delta > 0$)

$$w_a^{t+1} = (1+\delta)^{R_t(a)}, \quad \forall a \in A$$

    - where $R_t(a)$ is reward at time t if action a was performed