REINFORCEMENT LEARNING

CP8319/CPS824
Lecture 4

Instructor: Nariman Farsad

* Some of the slides in this deck are adopted from courses offered David Silver, Emma Brunskill, and Sergey Levine.

Today’s Agenda

1. **Last Lecture Review**
2. Finish Markov Reward Process
3. Markov Decision Process
4. Policy Evaluation
5. Optimal Policy

Markov Process

A Markov process is a memoryless random process, i.e. a sequence of random states $S_1, S_2, \ldots$ with the Markov property.

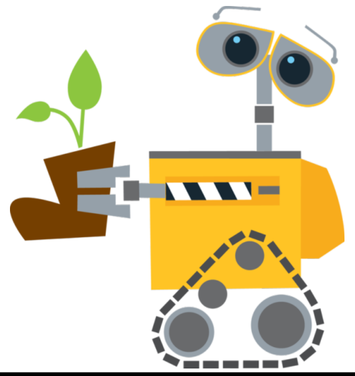
Definition

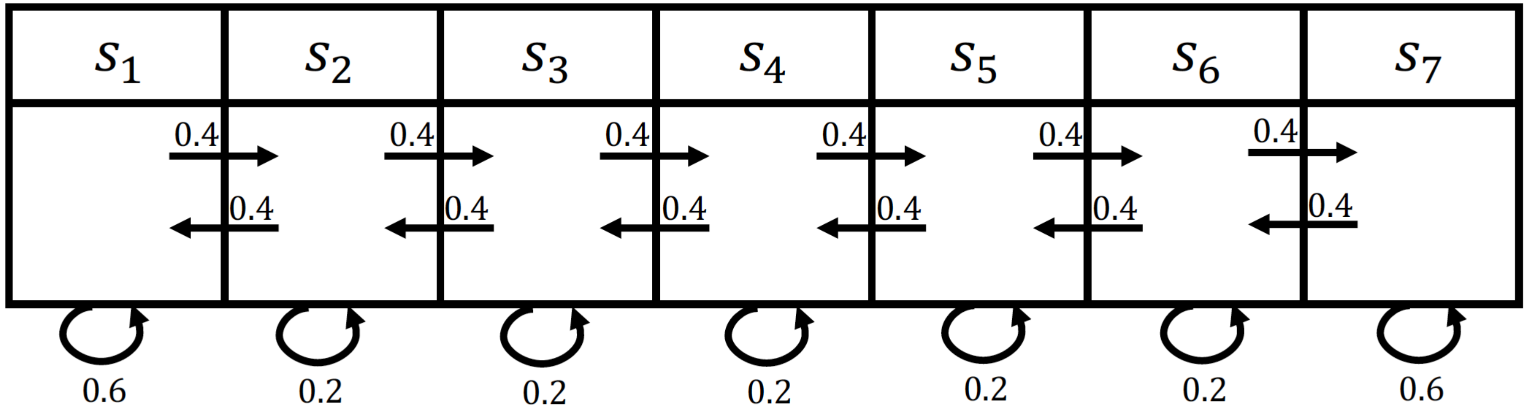
A *Markov Process* (or *Markov Chain*) is a tuple $\langle \mathcal{S}, \mathbf{P} \rangle$

- $\mathcal{S}$ is a (finite) set of states
- $\mathbf{P}$ is a state transition probability matrix,

$$P_{s,s'} = P\left(S_{t+1} = s' \mid S_t = s\right)$$

Example: A Robot





$$
\mathbf{P} = \begin{pmatrix}
0.6 & 0.4 & 0 & 0 & 0 & 0 & 0 \\
0.4 & 0.2 & 0.4 & 0 & 0 & 0 & 0 \\
0 & 0.4 & 0.2 & 0.4 & 0 & 0 & 0 \\
0 & 0 & 0.4 & 0.2 & 0.4 & 0 & 0 \\
0 & 0 & 0 & 0.4 & 0.2 & 0.4 & 0 \\
0 & 0 & 0 & 0 & 0.4 & 0.2 & 0.4 \\
0 & 0 & 0 & 0 & 0 & 0.4 & 0.6
\end{pmatrix}
$$

Markov Reward Process

A Markov reward process is a Markov chain with values.

Definition

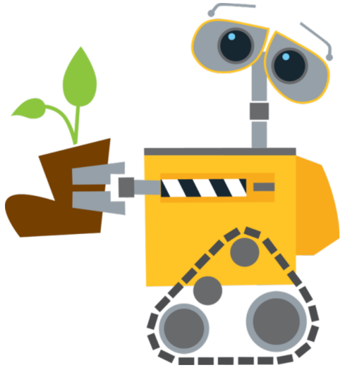
A *Markov Reward Process* is a tuple $\langle \mathcal{S}, \mathbf{P}, R, \gamma \rangle$

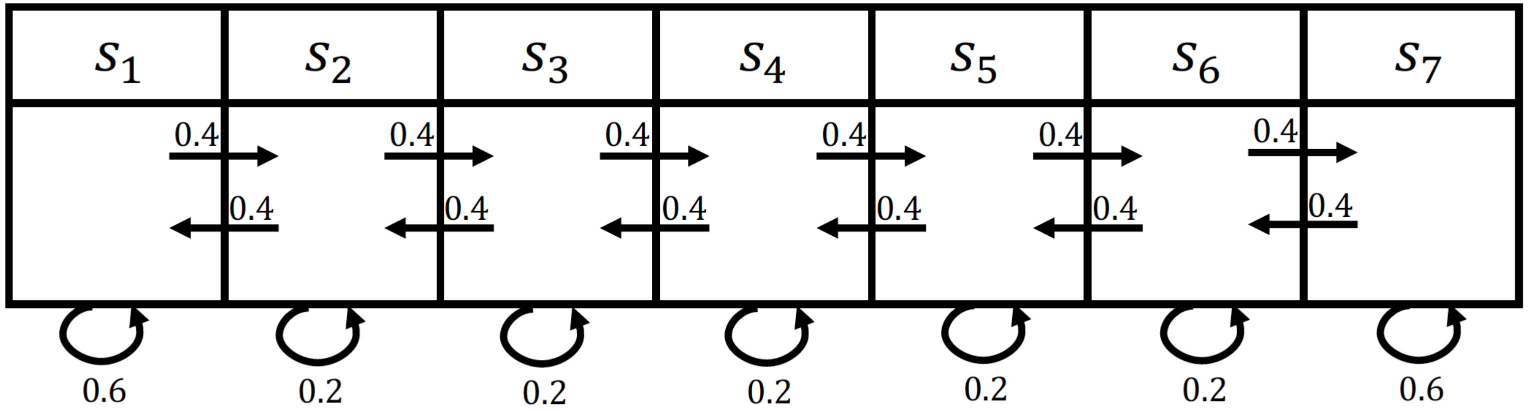
- $\mathcal{S}$ is a (finite) set of states
- $\mathbf{P}$ is a state transition probability matrix,

$$P_{s,s'} = P\left(S_{t+1} = s' \mid S_t = s\right)$$

- R is the reward function, $R(\mathrm{s}) = \mathbb{E}[r_t | S_t = s]$
- γ is a discount factor, $\gamma \in [0, 1]$

Example: A Robot





Reward: +1 in s_1, +10 in s_7, 0 in all other states

Horizon and Return

Definition of *horizon*, H

- Number of time steps in each episode
- Can be infinite
- Otherwise called finite Markov reward process

Definition

The *return* G_t is the total discounted reward from time-step t.

$$G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots = \sum_{k=0}^{H} \gamma^k r_{t+k}$$

- The *discount* $\gamma \in [0, 1]$ is the present value of future rewards
- The value of receiving reward r after k time-steps is $\gamma^k r$
- This values immediate reward above delayed reward.
  - γ close to 0 leads to "myopic" evaluation
  - γ close to 1 leads to "far-sighted" evaluation

Value Function

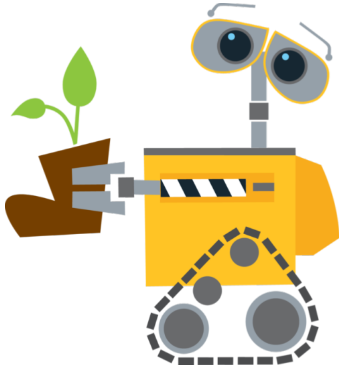
The value function $v(s)$ gives the long-term value of state s

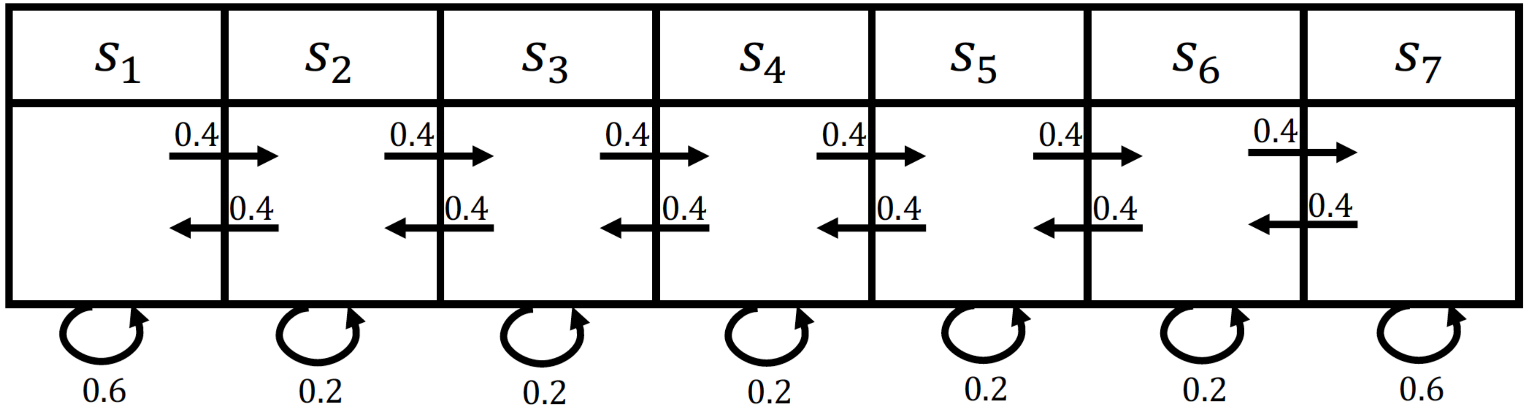
Definition

The *state value function* $v(s)$ of an MRP is the expected return starting from state s

$$v(s) = \mathbb{E}[G_t \mid S_t = s]$$

Example: A Robot





Reward: +1 in s_1, +10 in s_7, 0 in all other states

What about the value function?

$$V = [1.53\ 0.37\ 0.13\ 0.22\ 0.85\ 3.59\ 15.31]$$

Today’s Agenda

1. Last Lecture Review
2. **Finish Markov Reward Process**
3. Markov Decision Process
4. Policy Evaluation
5. Optimal Policy

Computing the Value of MRP: Simulation

Could estimate by simulation (Monte Carlo)

- Generate a large number of episodes
- Average returns
- Concentration inequalities bound how quickly average concentrates to expected value
- Requires no assumption of Markov structure

Computing the Value of MRP: Bellman Equation

$$v(s) = \mathbb{E}[r_t | S_t = s] + \gamma \mathbb{E}[v(S_{t+1}) | S_t = s]$$

$$= \underbrace{R(s)}_{\text{immediate reward}} + \underbrace{\gamma \sum_{s' \in \mathcal{S}} P_{s,s'} v(s')}_{\text{Discounted sum of future rewards}}$$

The value function can be decomposed into two parts:

- immediate reward R_t
- discounted value of successor state $\gamma v(S_{t+1})$

Computing the Value of MRP: Bellman Equation

For finite state MRP, we can express $V(s)$ using a matrix equation

$$\begin{pmatrix} V(s_1) \\ \vdots \\ V(s_N) \end{pmatrix} = \begin{pmatrix} R(s_1) \\ \vdots \\ R(s_N) \end{pmatrix} + \gamma \begin{pmatrix} P(s_1|s_1) & \cdots & P(s_N|s_1) \\ P(s_1|s_2) & \cdots & P(s_N|s_2) \\ \vdots & \ddots & \vdots \\ P(s_1|s_N) & \cdots & P(s_N|s_N) \end{pmatrix} \begin{pmatrix} V(s_1) \\ \vdots \\ V(s_N) \end{pmatrix}$$

$$\mathbf{v} = \mathbf{r} + \gamma \mathbf{P} \mathbf{v}$$

Computing the Value of MRP: Bellman Equation

For finite state MRP, we can express $V(s)$ using a matrix equation

$$\begin{pmatrix} V(s_1) \\ \vdots \\ V(s_N) \end{pmatrix} = \begin{pmatrix} R(s_1) \\ \vdots \\ R(s_N) \end{pmatrix} + \gamma \begin{pmatrix} P(s_1|s_1) & \cdots & P(s_N|s_1) \\ P(s_1|s_2) & \cdots & P(s_N|s_2) \\ \vdots & \ddots & \vdots \\ P(s_1|s_N) & \cdots & P(s_N|s_N) \end{pmatrix} \begin{pmatrix} V(s_1) \\ \vdots \\ V(s_N) \end{pmatrix}$$

$$\mathbf{v} = \mathbf{r} + \gamma \mathbf{P} \mathbf{v}$$

$$\mathbf{v} - \gamma \mathbf{P} \mathbf{v} = \mathbf{r}$$

$$(\mathbf{I} - \gamma \mathbf{P})\mathbf{v} = \mathbf{r}$$

$$\mathbf{v} = (\mathbf{I} - \gamma \mathbf{P})^{-1}\mathbf{r}$$

Solving directly requires taking a matrix inverse $\sim O(N^3)$

Computing the Value of MRP: Iterative Algorithm

Use *dynamic programming Value Iteration*

- Initialize $v_0(s) = 0 \; \forall s \in \mathcal{S}$
- For $k = 1$ until convergence (i.e., iterations):
  - For all $s \in \mathcal{S}$:
    - $v_k(s) = R(s) + \gamma \sum_{s' \in \mathcal{S}} P_{s,s'} v_{k-1}(s')$

Computational complexity: for each iteration $O(N^2)$, where $N = |\mathcal{S}|$

Today's Agenda

1. Last Lecture Review
2. Finish Markov Reward Process
3. **Markov Decision Process**
4. Policy Evaluation
5. Optimal Policy

Markov Decision Process (MDP)

A Markov decision process (MDP) is a Markov reward process with decisions/actions

Definition

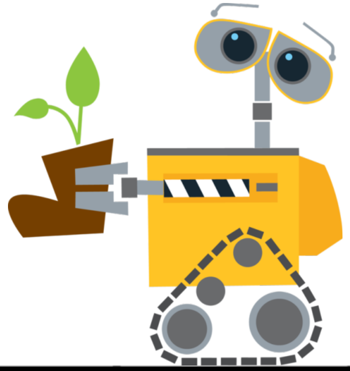
A *Markov Decision Process* is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathbf{P}, R, \gamma \rangle$

- $\mathcal{S}$ is a (finite) set of states
- $\mathcal{A}$ is a finite set of actions
- $\mathbf{P}$ is dynamics/transition model for each action,

$$P_{s,s'}^{a} = P\left(S_{t+1} = s' | S_t = s, A_t = a\right)$$

- R is the reward function, $R(s, a) = \mathbb{E}[r_t | S_t = s, A_t = a]$
- γ is a discount factor, $\gamma \in [0, 1]$

Example: A Robot



s_1	s_2	s_3	s_4	s_5	s_6	s_7

What are these two deterministic actions?

$$P(s'|s,a_1) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad P(s'|s,a_2) = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

MDP: Policy

Definition

A *policy* π is a distribution over actions given states,

$$\pi(a|s) = P\left(A_t = a|S_t = s\right)$$

- Policy specifies what action to take in each state
  - Can be deterministic or stochastic
- A policy fully defines the behaviour of an agent
- MDP policies depend on the current state (not the history)
- i.e. Policies are *stationary* (time-independent),

$$A_t \sim \pi(\cdot|S_t), \forall t > 0$$

MDP Given a Policy

- Given an MDP $M = \langle \mathcal{S}, \mathcal{A}, \mathbf{P}, R, \gamma \rangle$ and a policy π
- The state sequence $S_1, S_2, \ldots$ is a Markov process $\langle \mathcal{S}, \mathbf{P}^{\pi} \rangle$
- The state and reward sequence $S_1, R_1, S_2, R_2 \ldots$ is a Markov reward process $\langle \mathcal{S}, \mathbf{P}^{\pi}, R^{\pi}, \gamma \rangle$

With

$$P_{s,s'}^{\pi} = \sum_{a \in \mathcal{A}} \pi(a|s) P_{s,s'}^{a}$$

$$R^{\pi}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) R(s,a)$$

State Value Function

Definition

The *state-value function* $v^{\pi}(s)$ of an MDP is the expected return starting from state s, and then following policy π

$$v^{\pi}(s) = \mathbb{E}_{\pi}[G_t \mid S_t = s]$$

Today’s Agenda

1. Last Lecture Review
2. Finish Markov Reward Process
3. Markov Decision Process
4. **Policy Evaluation**
5. Optimal Policy

Bellman Expectation Equation

The state-value function can again be decomposed into immediate reward plus discounted value of successor state,

$$v^{\pi}(s) = \mathbb{E}_{\pi}\left[r_t + \gamma v^{\pi}(S_{t+1}) \mid S_t = s\right]$$

$$= \sum_{a \in \mathcal{A}} \pi(a|s) \left(R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P_{s,s'}^{a} v^{\pi}(s') \right)$$

MDP Policy Evaluation: Bellman Expectation Equation

The Bellman expectation equation can be expressed concisely using the induced MRP,

$$\mathbf{v}^{\pi} = \mathbf{r}^{\pi} + \gamma \mathbf{P}^{\pi} \mathbf{v}^{\pi}$$

with direct solution

$$\mathbf{v}^{\pi} = (\mathbf{I} + \gamma \mathbf{P}^{\pi})^{-1} \mathbf{r}^{\pi}$$

Not computationally efficient!

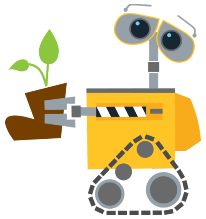
MDP Policy Evaluation: Iterative Algorithm

Initialize $v_0(s) = 0$ for all s
For $k = 1$ until convergence:
    For all $s \in \mathcal{S}$:

$$v_{k+1}^{\pi}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left(R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P_{s,s'}^{a} v_k^{\pi}(s') \right)$$

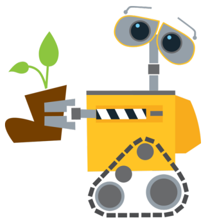
This is known as Bellman expectation backup

Example: A Robot

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

Dynamics: $P(s_6|s_6, a_2) \;=\; 0.5, P(s_7|s_6, a_2) \;=\; 0.5, \ldots$
Reward: for all actions, +1 in state s_1, +10 in state s_7, 0 otherwise
Policy: Let $\pi(s) = a_2 \;\forall s$
What is v_1^π?

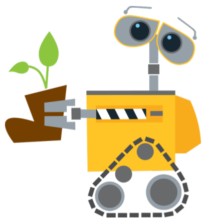
Example: A Robot

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

Dynamics: $P(s_6|s_6, a_2) \ = \ 0.5, P(s_7|s_6, a_2) \ = \ 0.5, \ldots$
Reward: for all actions, +1 in state s_1, +10 in state s_7, 0 otherwise
Policy: Let $\pi(s) = a_2 \ \forall s$
What is v_1^π?

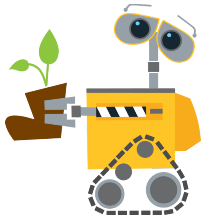
$$v_1^\pi = [1,0,0,0,0,0,10]$$

Example: A Robot

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

<u>Dynamics</u>: $P(s_6|s_6, a_2) = 0.5, P(s_7|s_6, a_2) = 0.5, \ldots$
<u>Reward</u>: for all actions, +1 in state s_1, +10 in state s_7, 0 otherwise
<u>Policy</u>: Let $\pi(s) = a_2\ \forall s$
Assume $v_1^\pi = [1,0,0,0,0,0,10]$ and $k = 1, \gamma = 0.5$
What is $v_2^\pi(s_6)$?

Example: A Robot

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

Dynamics: $P(s_6|s_6, a_2) = 0.5, P(s_7|s_6, a_2) = 0.5, \ldots$

Reward: for all actions, +1 in state s_1, +10 in state s_7, 0 otherwise

Policy: Let $\pi(s) = a_2 \; \forall s$

Assume $v_1^\pi = [1,0,0,0,0,0,10]$ and $k = 1, \gamma = 0.5$

What is $v_2^\pi(s_6)$?

$$v_2^\pi(s_6) = R(s_6\,, a_2) + \gamma \times 0.5 \times v_1^\pi(s_6) + \gamma \times 0.5 \times v_1^\pi(s_7)$$

$$= 0+0.5\times0.5\times0+0.5\times0.5\times10 =2.5$$

Today’s Agenda

1. Last Lecture Review

2. Finish Markov Reward Process

3. Markov Decision Process

4. Policy Evaluation

5. **Optimal Policy**

Action Value Function

Definition

The *action-value function* $Q^{\pi}(s,a)$ is the expected return starting from state s, taking action a, and then following policy π

$$Q^{\pi}(s,a) \;=\; \mathbb{E}_{\pi}[G_t | S_t, = s, A_t = a]$$

The action-value function can be decomposed using Bellman equation,

$$\begin{aligned} Q^{\pi}(s,a) &= \mathbb{E}_{\pi}[r_t + \gamma Q^{\pi}(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a] \\ &= R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P^{a}_{s,s'} \left(\sum_{a' \in \mathcal{A}} \pi(a'|s') Q^{\pi}(s',a') \right) \end{aligned}$$

Optimal Value Function

Definition

The *optimal state-value function* $v^*(s)$ is the maximum value function over all policies

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

The *optimal action-value function* $Q^*(s, a)$ is the maximum action-value function over all policies

$$Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a)$$

- The optimal value function specifies the best possible performance in the MDP.
- An MDP is "solved" when we know the optimal value fn.

Optimal Policy

Define a partial ordering over policies

$$\pi \geq \pi' \; if \; v^{\pi}(s) \geq v^{\pi'}(s), \forall s$$

Theorem

For any Markov Decision Process

- *There exists an optimal policy π^* that is better than or equal to all other policies, $\pi^* \geq \pi, \forall \pi$*
- *All optimal policies achieve the optimal value function,* $v^{\pi^*}(s) = v^*(s)$, or $\pi^* = \underset{\pi}{\operatorname{argmax}}\, v^{\pi}(s)$
- *All optimal policies achieve the optimal action-value function,* $Q^{\pi^*}(s, a) = Q^*(s, a)$

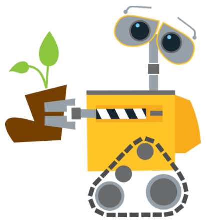
Optimal Policy Using Optimal Action-Value Function

An optimal policy can be found by maximising over $Q^*(s,a)$,

$$\pi^*(a|s) = \begin{cases} 1, & \text{if } a = \max\limits_{a\in\mathcal{A}} Q^*(s,a) \\ 0, & otherwise \end{cases}$$

- There is always a deterministic optimal policy for any MDP
- If we know $Q^*(s,a)$, we immediately have the optimal policy

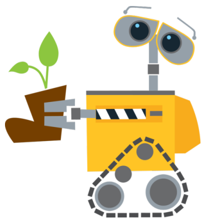
Example: A Robot

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

7 discrete states (location of rover)
2 actions: Left or Right
How many deterministic policies are there?

Is the optimal policy for a MDP always unique?

Example: A Robot

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

7 discrete states (location of rover)
2 actions: Left or Right
How many deterministic policies are there?

2^7

Is the optimal policy for a MDP always unique?
No, there may be two actions that have the same optimal value function

Optimal Policy Search

- One option is searching to compute best policy
- Number of deterministic policies is $|\mathcal{A}|^{|\mathcal{S}|}$
- Better Options (will be covered in future lectures):
    - Value Iteration
    - Policy Iteration
    - Q-learning