10-403 Recitation (2/21/20): Policy & Value Iteration

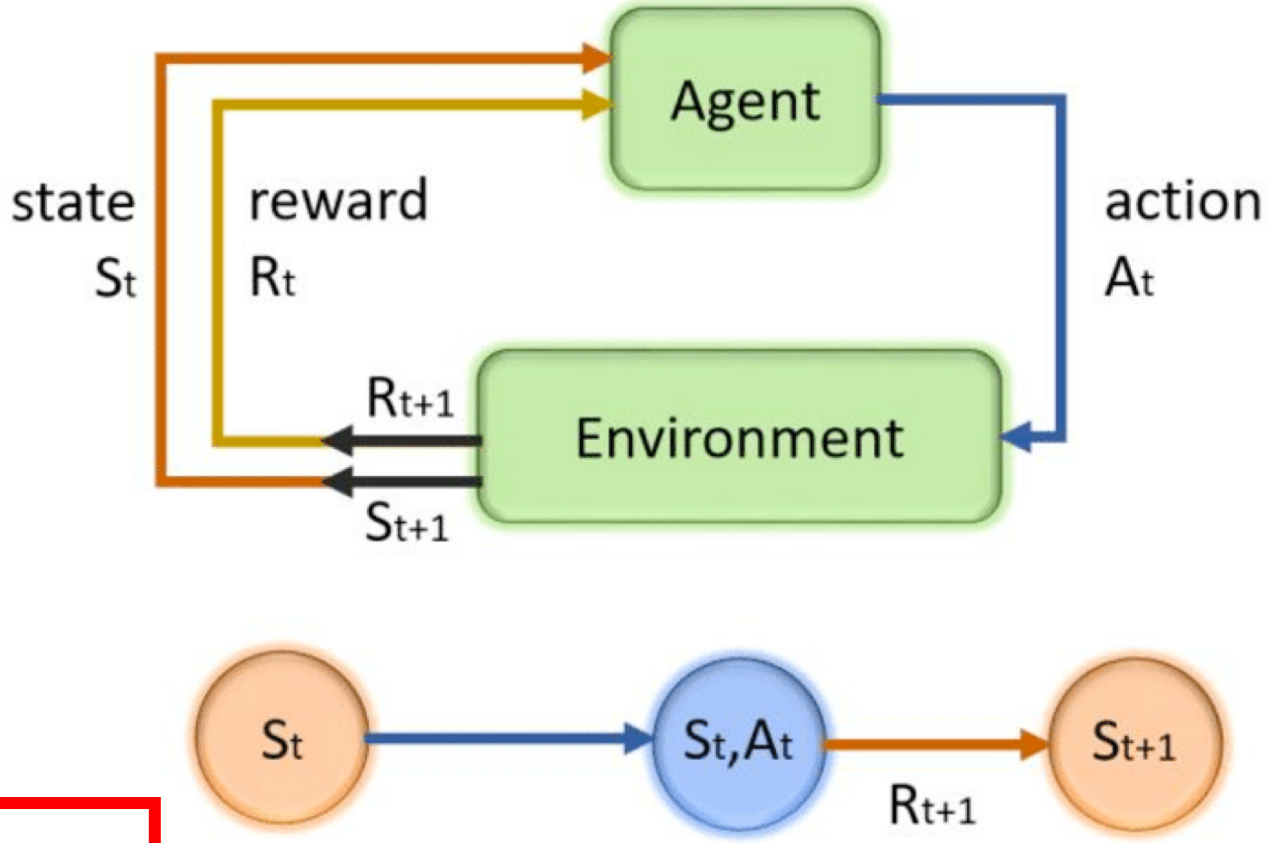
General Recommendation

- Start HW2 EARLY! (Generating plots can take up to a day)
- Strongly recommend to read a textbook to do HW (Chap 3, 4)

Markov Decision Process (MDP)

MDP (S, A, T, R, γ)

- S: a set of states
- A: a set of actions
- T: a state transition probability
- R: reward function
- γ: a discount factor



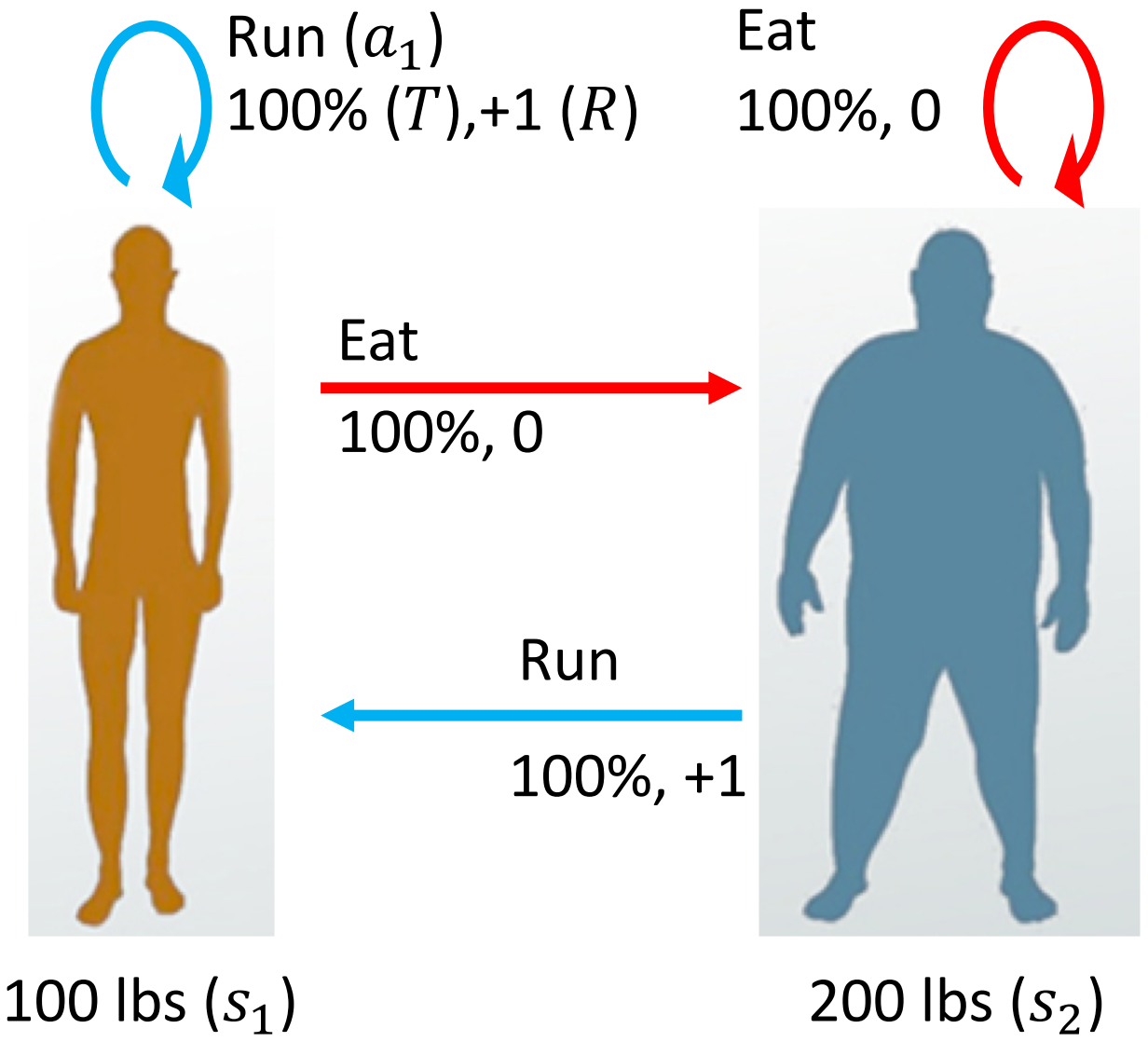
GOAL: FIND OPTIMAL POLICY π_*

Simple Example of MDP

MDP (S, A, T, R, γ)

- S = {100, 200}
- A = {Eat, Run}
- T(100, Eat, 200) = 1, …
- R(200, Run, 100) = 1, …
- γ = 1

OPTIMAL POLICY π_* ?



MDP Dynamics

Start in some state $s_0 \in S$

Choose action $a_0 \in A$ in state s_0

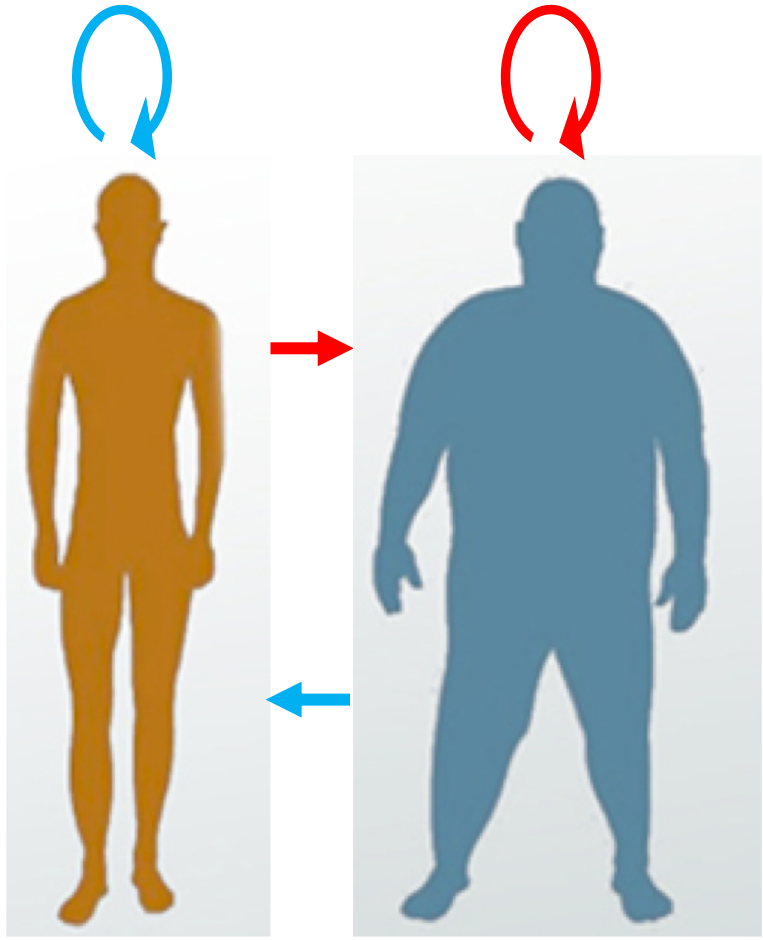
New MDP state $s_1 \in S$ chosen according to $P_{s_0 a_0}$: $s_1 \sim P_{s_0 a_0}$

Choose action $a_1 \in A$ in state s_1

New MDP state $s_2 \in S$ chosen according to $P_{s_1 a_1}$: $s_2 \sim P_{s_1 a_1}$

Choose action $a_2 \in A$ in state s_2, and so on..

$$s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} s_3 \xrightarrow{a_3} \dots$$



Preliminary: Return

- The sum of the rewards (in a episode)

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$$

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1},$$

$$\begin{aligned} G_t &\doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots \\ &= R_{t+1} + \gamma\big(R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \cdots\big) \\ &= R_{t+1} + \gamma G_{t+1} \end{aligned}$$

Preliminary: Value Function and Policy

- Value Function: functions of states that estimate how good it is for the agent to be in a given state

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right], \text{ for all } s \in \mathcal{S},$$

$$q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s, A_t = a\right]$$

- Policy: a mapping from states to probabilities of selecting each possible action

a policy $\boldsymbol{\pi}$(a|s) means the probability that $\boldsymbol{A_t = a}$ if $\boldsymbol{S_t = s}$

Bellman Equation

State / Action	S_1	S_2
A_1	0.2	0.8
A_2	0.6	0.4

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right], \text{ for all } s \in \mathcal{S},$$

$$\begin{aligned} v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a)\Big[r + \gamma \mathbb{E}_\pi[G_{t+1}|S_{t+1} = s']\Big] \\ &= \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a)\Big[r + \gamma v_\pi(s')\Big], \quad \text{for all } s \in \mathcal{S}, \end{aligned}$$

Bellman Optimality Equation

$$
\begin{aligned}
v_*(s) &= \max_{a \in \mathcal{A}(s)} q_{\pi_*}(s,a) \\
&= \max_a \mathbb{E}_{\pi_*}[G_t \mid S_t = s, A_t = a] \\
&= \max_a \mathbb{E}_{\pi_*}[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a] \\
&= \max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \max_a \sum_{s',r} p(s', r \mid s, a) \left[r + \gamma v_*(s') \right].
\end{aligned}
$$

- Assumption 0: Finite (or discrete) set of states and actions
- Assumption 1: The dynamics of the environment is given.
- Assumption 2: Markov property
- Assumption 3: Enough computational resources to solve these equations

Problem: Solving linear equations is expensive, $O(|S|^3)$

Dynamic Programming (Iterative Method)

- Definition: Simplifying a complicated problem by breaking it down into simpler sub-problems in a recursive manner

$$
\begin{aligned}
v_*(s) &= \max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \max_a \sum_{s',r} p(s', r \mid s, a) \Big[r + \gamma v_*(s') \Big]
\end{aligned}
$$

Is this still an exact solution method?

Policy Evaluation

- Does this converge? YES! (Contraction Mapping Theorem)

Iterative policy evaluation

Input π, the policy to be evaluated
Initialize an array $V(s) = 0$, for all $s \in \mathcal{S}^+$
Repeat
  $\Delta \leftarrow 0$
  For each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)
Output $V \approx v_\pi$

Policy Improvement

- Is the new policy generated by this algorithm always better? YES! (Policy Improvement Theorem)

policy-stable $\leftarrow$ *true*
For each $s \in \mathcal{S}$:
    old-action $\leftarrow \pi(s)$
    $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r \mid s, a)\left[r + \gamma V(s')\right]$
    If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

$$q_\pi(s, \pi'(s)) \geq v_\pi(s)$$

Policy Iteration (Policy Evaluation + Improvement)

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Repeat
      $\Delta \leftarrow 0$
      For each $s \in \mathcal{S}$:
         $v \leftarrow V(s)$
         $V(s) \leftarrow \sum_{s',r} p(s', r \mid s, \pi(s)) \left[r + \gamma V(s') \right]$
         $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number)

Prediction

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
      old-action $\leftarrow \pi(s)$
      $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r \mid s, a) \left[r + \gamma V(s') \right]$
      If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Control

Value Iteration

Value Iteration, for estimating $\pi \approx \pi_*$

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
| $\Delta \leftarrow 0$
| Loop for each $s \in \mathcal{S}$:
| $v \leftarrow V(s)$
| $V(s) \leftarrow \max_a \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$
| $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Output a deterministic policy, $\pi \approx \pi_*$, such that
$\pi(s) = \text{argmax}_a \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$

Value Iteration vs. Policy Iteration

Small state spaces: Policy Iteration typically very fast and converges quickly

Large state spaces: Policy Iteration may be slow

- Reason: Policy Iteration needs to solve a large system of linear equations
- Value iteration is preferred in such cases

Very large state spaces: Value function can be *approximated* using some regression algorithm

- Optimality guarantee is lost however