SMU: Lecture 3

Monday, February 28, 2022

(Heavily inspired by the Stanford RL Course of Prof. Emma Brunskill, but all potential errors are mine.)

Plan for Today

- Recap of important concepts from lectures 1 and 2.
- Model-free control:
    - **Monte-Carlo Online Control**
    - **SARSA**
    - **Q-Learning**

Part 1: Where are we? (Recap from the previous lectures)

Markov Decision Process

- **Markov decision process = Markov reward process + Actions**
- **An MDP is given by:**
  - A set of states S.
  - A set of actions A.
  - A transition model $P[X_{t+1} = s' | X_t = s, A_t = a] \underbrace{= P(s' | s, a)}_{\text{notation}}$
  - A reward $R(s, a) = \mathbb{E}[R_t | X_t = s, A_t = a]$, i.e. the expected reward that the agent receives when performing action a in state s.
  - Discount factor γ.

Policy

- Policy determines which action to take in each state s.
- It can be either deterministic or random — that is also why policy will not simply be a function from states to actions.
- **We define policy:** $\pi(a \mid s) = P(A_t = a \mid X_t = s)$.
- **Example** (policy for our ant 🐜):
  - $A = \{\text{left}, \text{right}\}$
  - $\pi(\text{left} \mid 1) = 0$, $\pi(\text{right} \mid 1) = 1$, $\pi(\text{left} \mid 2) = 0.5$, $\pi(\text{right} \mid 1) = 0.5, \ldots$

(Bellman equation for MDP)

State Value Function of MDP

General case:

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

Version for deterministic policy:

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

MDP Control Problem

How to find $\pi^*(s) = \arg\max_{\pi} V^{\pi}(s)$???

State-Action Value Q

- **Definition:**

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

- **Intuition:**
  - The value of the return that we obtain if we first take the action a in the state s and then follow the policy π (including when we visit s again).
  - *Think of it as perturbing the policy π — we deviate from following the policy π only in the first step in s.*

Policy Improvement Step

- **Given:** An MDP and a **policy π_i that we want to improve** (if possible).

- **DO:**

    - For all $s \in S$, compute $Q^{\pi_i}(s, a)$ as defined on the previous slide, i.e. $Q^{\pi_i}(s, a) = R(s, a) + \gamma \cdot \sum_{s' \in S} P(s' \mid s, a) \cdot V^{\pi_i}(s')$.

    - **Compute new policy for all $s \in S$:**

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

Here, we use the fact that our policy is deterministic for simpler notation (treating policy as a function). Using our previous notation we could write:

$$\pi(a \mid s) = \begin{cases} 1 & \textit{if } a = \arg\max_{a \in A} Q^{\pi_i}(s, a) \\ 0 & \textit{otherwise} \end{cases}$$

Policy Iteration

$i = 0$

Initialize π_0 randomly.

DO

$V^{\pi_i} =$ Compute the state-value function, evaluating π_i.

$\pi_{i+1} =$ Policy improvement of π_i.

$i = i + 1$

WHILE $||\pi_i - \pi_{i-1}||_1 > 0$ /* if policy changed */

Policy iteration finds the globally optimal policy!

Value Iteration

Set $k = 1$

Initialize $V_0(s) = 0$ for all $s \in S$

DO:

$$V_k(s) = \max_{a \in A} \left[R(s, a) + \gamma \cdot \sum_{s' \in S} P(s' \mid s, a) \cdot V_{k-1}(s') \right]$$

Bellman backup B[V]

WHILE $\|V_k - V_{k-1}\|_\infty \geq \varepsilon$

- To extract an optimal policy, we can extract a deterministic (not necessarily unique) policy:

$$\pi(s) = \arg\max_{a \in A} \left[R(s, a) + \sum_{s' \in S} P(s' \mid s, a) \cdot V(s') \right].$$

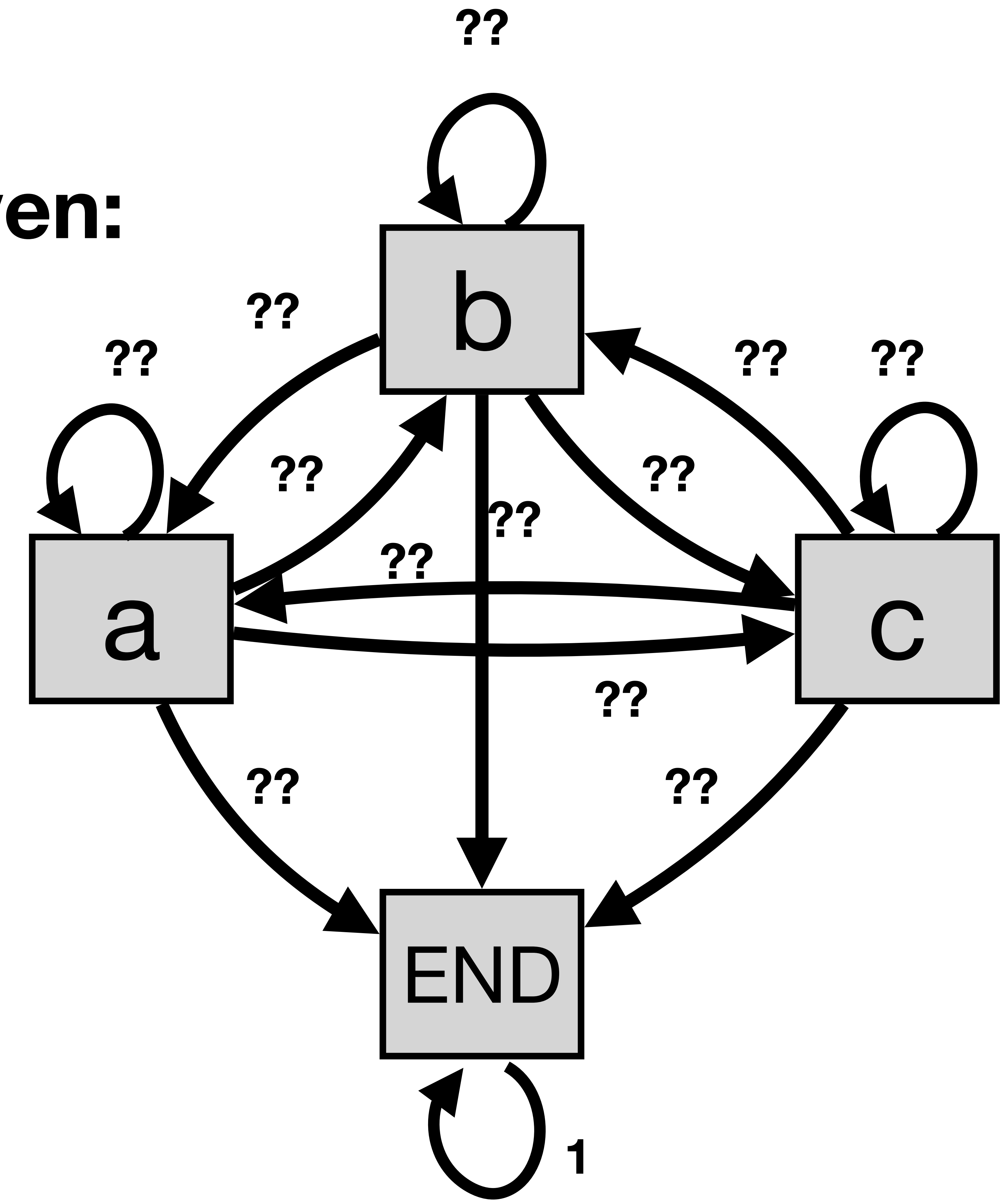
Problem: Model-Free Policy Evaluation

- Given a policy and an MDP with unknown parameters (or generally an environment with which we can interact), **estimate the value function.**

Example

Agent: 

States are given:



Rewards??

Actions are given:

$$A = \{l, r\}$$

Policy is given, e.g.:

$$\pi(l \mid a) = 0.2,\ \pi(r \mid a) = 0.8,$$
$$\pi(l \mid b) = 0.3,\ \pi(r \mid b) = 0.7,$$

...

First/Every-Visit Monte-Carlo Evaluation

Initialize: $G(s) = 0$, $N(s) = 0$ **for all** $s \in S$**.**

For $i = 1, \ldots, N$**:**

Sample episode $e_i := s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$.

For each time step $1 \leq t \leq T_i$:

If t is the first occurrence of state s in the episode e_i /* This is for first-visit MC */

s is the state visited at time t in the episode e_i

$g_{i,t} := r_{i,t} + \gamma \cdot r_{i,t+1} + \gamma^2 \cdot r_{i,t+2} + \ldots + \gamma^{T_i - t} \cdot r_{i,T_i}$

$N(s) := N(s) + 1$ /* *Increment total visits counter* */

$G(s) := G(s) + g_{i,1}$ /* *Increment total return counter* */

$V^\pi(s) := G(s)/N(s)$ /* *Update current estimate* */

Temporal Difference Learning

- **TD learning** combines Monte-Carlo estimation and dynamic programming ideas.
- **TD learning** can be used both in episodic and infinite-horizon non-episodic settings,
- **TD learning** updates estimates of V^{π} continually, after every consecutive tuple *state-action-reward-state* (therefore we do not need to wait till the end of an episode).

….

TD-Learning: Pseudocode

Initialize: $V^{\pi}(s) = 0$ for all $s \in S$

Loop:

Sample tuple (s_t, a_t, r_t, s_{t+1}).

Update $V^{\pi}(s_t) := V^{\pi}(s_t) + \alpha \cdot (\underbrace{r_{i,t} + \gamma \cdot V^{\pi}(s_{t+1})}_{\text{TD target}} - V^{\pi}(s_t))$

Part 2: Model-Free Control
(Problem Statement)

Model-Free Control

- Given a policy and an MDP with unknown parameters (or generally an environment with which we can interact), **find the optimal policy π.**

Part 3: Model-Free Policy Iteration

On-Policy and Off-Policy Methods

- **On-policy methods**: samples must be from the policy that we are learning.

- **Off-policy methods**: samples do not have to be from the policy that we are learning.

- We will see examples of these methods and then it will become clearer.

MC Estimation of $Q^{\pi}(s, a)$

- Last time we talked about MC Estimation of the value function. We can now try to use the same idea for the estimation of the state-action value function $Q^{\pi}(s, a)$.

Exploration vs Exploitation

- **A simple idea** (that will not work yet… and will illustrate why we need to think about exploration)**:**
  - **THIS WILL NOT WORK (YET):**

Initialize: $G(s, a) = 0$, $N(s, a) = 0$ **for all** $s \in S$**.**

For $i = 1,\ldots, N$**:**

Sample episode $e_i := s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$ using π.

For each time step $1 \leq t \leq T_i$:

(**If** t is the first occurrence of state s in the episode e_i - Use this if you want first-visit MC)

s_t is the state visited at time t in the episode e_i

a_t is the action taken at time t in the episode e_i

$g_{i,t} := r_{i,t} + \gamma \cdot r_{i,t+1} + \gamma^2 \cdot r_{i,t+2} + \ldots + \gamma^{T_i - t} \cdot r_{i,T_i}$

$N(s) := N(s) + 1$ /* *Increment total visits counter* */

$G(s_t, a_t) := G(s_t, a_t) + g_{i,1}$ /* *Increment total return counter* */

$Q^\pi(s_t, a_t) := G(s_t, a_t)/N(s_t, a_t)$ /* *Update current estimate* */

Exploration vs Exploitation

- **A simple idea** (that will not work yet… and will illustrate why we need to think about exploration)**:**
  - **THIS WILL NOT WORK (YET):**

Initialize: $G(s, a) = 0$, $N(s, a) = 0$ **for all** $s \in S$**.**

For $i = 1, \ldots, N$**:**

Sample episode $e_i := s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$ using π.

For each time step $1 \le t \le T_i$:

(**If** t is the first occurrence of state s in the episode e_i - Use this if you want first-visit MC)

s_t is the state visited at time t in the episode e_i

a_t is the action taken at time t in the episode e_i

$g_{i,t} := r_{i,t} + \gamma \cdot r_{i,t+1} + \gamma^2 \cdot r_{i,t+2} + \ldots + \gamma^{T_i - t} \cdot r_{i,T_i}$

$N(s) := N(s) + 1$ /* *Increment total visits counter* */

$G(s_t, a_t) := G(s_t, a_t) + g_{i,1}$ /* *Increment total return counter* */

$Q^\pi(s_t, a_t) := G(s_t, a_t)/N(s_t, a_t)$ /* *Update current estimate* */

Exploration vs Exploitation

- **A simple idea** (that will not work yet… and will illustrate why we need to think about exploration)**:**
  - **Why this does not work? Suppose that the policy π is deterministic. Then we will only see actions (s, a) where $a = \pi(s)$. So, essentially, we will only be able to have Q^π for actions taken by π, which is useless for what we want to use Q for.**

Initialize: $G(s, a) = 0$, $N(s, a) = 0$ **for all** $s \in S$.

For $i = 1, \ldots, N$**:**

Sample episode $e_i := s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$ using π.

For each time step $1 \leq t \leq T_i$:

(**If** t is the first occurrence of state s in the episode e_i - Use this if you want first-visit MC)

s_t is the state visited at time t in the episode e_i

a_t is the action taken at time t in the episode e_i

$g_{i,t} := r_{i,t} + \gamma \cdot r_{i,t+1} + \gamma^2 \cdot r_{i,t+2} + \ldots + \gamma^{T_i - t} \cdot r_{i,T_i}$

$N(s) := N(s) + 1$ */* Increment total visits counter */*

$G(s_t, a_t) := G(s_t, a_t) + g_{i,1}$ */* Increment total return counter */*

$Q^\pi(s_t, a_t) := G(s_t, a_t)/N(s_t, a_t)$ */* Update current estimate */*

Let’s see why it will not work!

$S = \{a, b, c, \text{END}\}, A = \{l, r\}$

$\pi_1(a) = l, \pi_1(b) = l, \pi_1(c) = l$

$e_1 = a, l, 1, b, l, 1, a, l, 1, c, l, 2, \text{END}$

$e_2 = \ldots$

But how can we ever estimate, e.g., $Q^\pi(a, r)$??

- **A simple idea** (that will not work yet… and will illustrate why we need to think about exploration)**:**
  - **THIS WILL NOT WORK (YET):**

Initialize: $G(s, a) = 0$, $N(s, a) = 0$ **for all** $s \in S$.

For $i = 1, \ldots, N$**:**

Sample episode $e_i := s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$ using π.

For each time step $1 \le t \le T_i$:

(**If** t is the first occurrence of state s in the episode e_i
- Use this if you want first-visit MC)

s_t is the state visited at time t in the episode e_i

a_t is the action taken at time t in the episode e_i

$g_{i,t} := r_{i,t} + \gamma \cdot r_{i,t+1} + \gamma^2 \cdot r_{i,t+2} + \ldots + \gamma^{T_i - t} \cdot r_{i,T_i}$

$N(s) := N(s) + 1$ /* *Increment total visits counter* */

$G(s_t, a_t) := G(s_t, a_t) + g_{i,1}$ /* *Increment total return counter* */

$Q^\pi(s_t, a_t) := G(s_t, a_t)/N(s_t, a_t)$ /* *Update current estimate* */

ε-Modified* Policy (Deterministic Case)

- We will now modify a given policy to "sometimes take a random action".

- **Definition:** Given a **deterministic** policy π the ε-greedy of π, denoted π_ε, is the policy which is given as follows:

Number of actions

$$\pi_\varepsilon(a \mid s) = \begin{cases} 1 - \varepsilon \cdot \left(1 + \frac{1}{|A|}\right) & \text{for } a = \pi(s), \\ \varepsilon \cdot \left(1 + \frac{1}{|A|}\right) & \text{otherwise} . \end{cases}$$

**This is not a standard terminology.*

ε-Modified Policy (General Case)

- We will now modify a given policy to "sometimes take a random action".

- **Definition:** Given a policy π the ε-modified version of π, denoted π_ε, is the policy which is given as follows:

$$\pi_\varepsilon(a \mid s) = (1 - \varepsilon) \cdot \pi(a \mid s) + \frac{\varepsilon}{|A|}.$$

Example: Q-Value Estimation with ε-Modified Policy

$S = \{a, b, c, \text{END}\}, A = \{l, r\}$

$\pi_1(a) = l, \pi_1(b) = l, \pi_1(c) = l$

Suppose that π_ε is an

$e_1 = a, l, 1, b, l, 1, a, l, 1, c, l, 2, \text{END}$

$e_2 = \ldots$

But how can we ever estimate, e.g., $Q^\pi(b, r)$?? **This time we are guaranteed to see the pair (b, r) infinitely many times (in the limit and with probability 1) as long as b has non-zero probability of being visited.**

- **A simple idea** (that will not work yet… and will illustrate why we need to think about exploration)**:**
  - **THIS WILL NOT WORK (YET):**

    Initialize: $G(s, a) = 0, N(s, a) = 0$ **for all** $s \in S$**.**

    For $i = 1, \ldots, N$**:**

    Sample episode $e_i := s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$ using π.

    For each time step $1 \leq t \leq T_i$:

    (**If** t is the first occurrence of state s in the episode e_i
    - Use this if you want first-visit MC)

    s_t is the state visited at time t in the episode e_i

    a_t is the action taken at time t in the episode e_i

    $g_{i,t} := r_{i,t} + \gamma \cdot r_{i,t+1} + \gamma^2 \cdot r_{i,t+2} + \ldots + \gamma^{T_i - t} \cdot r_{i,T_i}$

    $N(s) := N(s) + 1$ /* *Increment total visits counter* */

    $G(s_t, a_t) := G(s_t, a_t) + g_{i,1}$ /* *Increment total return counter* */

    $Q^\pi(s_t, a_t) := G(s_t, a_t)/N(s_t, a_t)$ /* *Update current estimate* */

ε-Greedy Policy

- Given a Q-function $Q(s, a)$, we define the ε-greedy policy w.r.t. Q as

We assume ties are decided consistently

$$\pi(a \mid s) = \begin{cases} 1 - \varepsilon \cdot \left(1 - \frac{1}{|A|}\right) & \text{when } a = \arg\max_{a \in A} Q(s, a) \\ \frac{\varepsilon}{|A|} & \text{when } a \neq \arg\max_{a \in A} Q(s, a) \end{cases}$$

Monotonic ε-Greedy Policy Improvement

- **Theorem:** Assume that we can compute Q^π and V^π exactly *(which is not always the case where we will use ε-greedy policy improvements)*.
  1. Let π_i be some ε-greedy policy.
  2. Let Q^{π_i} be the Q-function w.r.t. π_i.
  3. Let π_{i+1} be the ε-greedy policy w.r.t. Q^{π_i} as defined on the previous slide.

  Then $V^{\pi_{i+1}}(s) \geq V^{\pi_i}(s)$ for all $s \in S$.

- ***Proof (****Not this time but see the lectures of Emma Brunskill if you are interested.)*

MC On Policy Improvement

Initialize: $G(s,a) = 0,\ N(s,a) = 0,\ Q(s,a) = 0$ **for all** $s \in S, a \in A$.

Initialize: $\varepsilon = 1,\ k = 1$

For $i = 1,\ldots,N$:

Sample episode $e_i := s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$ **given** π_k.

For each time step $1 \leq t \leq T_i$:

~~(~~**If** t is the first occurrence of state s in the episode e_i - Use this if you want first-visit MC)

s_t is the state visited at time t in the episode e_i

a_t is the action taken at time t in the episode e_i

$g_{i,t} := r_{i,t} + \gamma \cdot r_{i,t+1} + \gamma^2 \cdot r_{i,t+2} + \ldots + \gamma^{T_i - t} \cdot r_{i,T_i}$

$N(s) := N(s) + 1$ /* *Increment total visits counter* */

$G(s_t, a_t) := G(s_t, a_t) + g_{i,1}$ /* *Increment total return counter* */

$Q(s_t, a_t) := G(s_t, a_t)/N(s_t, a_t)$ /* *Update current estimate* *

EndFor

$k = k + 1,\ \varepsilon = 1/k$

$\pi_k = \varepsilon-$**greedy policy w.r.t.** Q

GLIE

- GLIE = “greedy in the limit of infinite exploration”.

- **Definition** (GLIE conditions**):**

  1. If a state $s \in S$ is visited infinitely often, then each action in that state is chosen infinitely often (with probability 1)

  2. In the limit (as t → ∞), the learning policy is greedy with respect to the learned Q-function (with probability 1). By *greedy* we mean (ignoring the possibility of ties in the $\arg\max$ for simplicity) that

$$\pi_{k+1}(a \mid s) = \begin{cases} 1 & \text{for } a = \arg\max_{a \in A} Q_k(s, a), \\ 0 & \text{otherwise.} \end{cases}$$

$\varepsilon_i = 1/i$ is GLIE

- For a proof, see, e.g. *Singh, S., Jaakkola, T., Littman, M. L., & Szepesvári, C. (2000). Convergence results for single-step on-policy reinforcement-learning algorithms. Machine learning, 38(3), 287-308.*

- The formal proof is a bit tricky…

A Theorem (Why GLIE Matters)

- **Theorem**: GLIE Monte-Carlo Control converges to the optimal state-action value function, i.e. $Q_k(s, a) \to Q^*(s, a)$ as $k \to \infty$.

A Theorem (Why GLIE Matters)

- **Theorem**: GLIE Monte-Carlo Control converges to the optimal state-action value function, i.e. $Q_k(s, a) \to Q^*(s, a)$ as $k \to \infty$.

- *Partially this follows from the theorem about monotonic ε-greedy policy improvement (think of what happens when the estimates of Q-function w.r.t. some policy converge, but the real proof is more difficult than that and we will not show it.*

Part 4: SARSA and Q-Learning

General Form of TD-Based Methods

- **Basic idea:**
  - Replace Monte Carlo Policy Evaluation by a temporal-difference method.
  - Still use ε-greedy policies to guarantee that exploration will take place.

Bellman Equations for Q-Function

(Something we skipped when we talked about Q-functions for MDPs but something that will be useful now.)

We have:

$$V^\pi(s) = \sum_{a \in A} \pi(a \,|\, s) \cdot Q^\pi(s, a)$$

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

Combining the above:

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

TD-Target

Bellman for Q-function:

$$Q^\pi(s_t, a_t) = R(s_t, a_t) + \gamma \cdot \boxed{\sum_{s_{t+1} \in S} P(s_{t+1} \mid s_t, a_t) \cdot \sum_{a_{t+1} \in A} \pi(a_{t+1} \mid s_{t+1}) \cdot Q^\pi(s_{t+1}, a_{t+1})}$$

$$\mathbb{E}[Q^\pi(X_{t+1}, A_{t+1}) \mid X_t = s_t, A_t = a_t]$$

Temporal difference update (SARSA)…

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha \left(\boxed{r_t + \gamma Q(s_{t+1}, a_{t+1})} - Q(s_t, a_t) \right)$$

SARSA

- SARSA is an on-policy algorithm.

1. **Initialize:** set π to be some ε-greedy policy, set $t = 0$
2. **Sample** a using the distribution given by π_0 in the state s_0 *(for sampling, we will use the notation $a \sim \pi(s)$)*. **Take** the action a and **observe** r_0, s_1**.**
3. **While** s_t is not a terminal state:
    1. **Take** action $a \sim \pi(s_t)$ and observe r_{t+1}, s_{t+a}.
    2. $Q(s_t, a_t) := Q(s_t, a_t) + \alpha \left(r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \right)$
    3. $\pi := \varepsilon$-greedy(Q)
    4. Set $t := t + 1$. Update ε, α /* *see next slides* */

Convergence (SARSA)

- SARSA converges to the optimal state-value function Q^* if the following conditions are satisfied:

    1. The sequence of policies π_t satisfies the GLIE conditions (enough to have $\varepsilon_t = 1/t$).

    2. Step-sizes satisfy the Robbins-Monro conditions:

$$\sum_{t=1}^{\infty} \alpha_t = \infty,$$

$$\sum_{t=1}^{\infty} \alpha_t^2 < \infty.$$

Q-Learning (1/2)

- The **Optimal Bellman Equation** (we have not talked about it yet but it is similar to what we already saw)**:**

$$Q^*(s,a) = R(s,a) + \gamma \boxed{\sum_{s_{t+1} \in S} P(s_{t+1} \mid s_1, a_t) \cdot \max_{a_{t+1} \in A} Q^*(s_{t+1}, a_{t+1}).}$$

$$\mathbb{E}\left[\max_{a_{t+1} \in A} Q^*(X_{t+1}, a_{t+1}) \,\middle|\, X_t = s_t, A_t = a_t \right]$$

- Q-Learning update rule:

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha \left(r_t + \gamma \max_{a \in A} Q(s_{t+1}, a) - Q(s_t, a_t) \right)$$

Q-Learning (2/2)

- Q-Learning is an off-policy algorithm.

1. **Initialize:** set π to be some ε-greedy policy, set $t = 0$
2. **Sample** a using the distribution given by π_0 in the state s_0 *(for sampling, we will use the notation $a \sim \pi(s)$)*. **Take** the action a and **observe** r_0, s_1.
3. **While** s_t is not a terminal state:
   1. **Take** action $a \sim \pi(s_t)$ and observe r_{t+1}, s_{t+1}.
   2. $Q(s_t, a_t) := Q(s_t, a_t) + \alpha \left(r_t + \gamma \max_{a \in A} Q(s_{t+1}, a) - Q(s_t, a_t) \right)$
   3. $\pi := \varepsilon$-greedy(Q)
   4. Set $t := t + 1$. Update ε, α /* *see next slides* */

Convergence (Q-Learning)

- For convergence of the state-value Q-function, we need only the Robbins-Monro conditions + every state-action pair needs to be visited infinitely often (with probability 1).

- For convergence of the policy to the optimal policy, we need GLIE (i.e. it needs to also be greedy in the limit…).

Double Q-Learning

Double Q-Learning

Initialize $Q_1(s, a)$ and $Q_2(s, a)$,$\forall s \in S, a \in A$ $t = 0$, initial state $s_t = s_0$
loop
  Select a_t using ϵ-greedy $\pi(s) = \arg\max_a Q_1(s_t, a) + Q_2(s_t, a)$
  Observe (r_t, s_{t+1})
  if (with 0.5 probability) **then**
    $Q_1(s_t, a_t) \leftarrow Q_1(s_t, a_t) + \alpha(r_t + \gamma Q_2(s_{t+1}, \arg\max_a Q_1(s_{t+1}, a)) - Q_1(s_t, a_t))$
  else
    $Q_2(s_t, a_t) \leftarrow Q_2(s_t, a_t) + \alpha(r_t + \gamma Q_1(s_{t+1}, \arg\max_a Q_2(s_{t+1}, a)) - Q_2(s_t, a_t))$
  end if
  $t = t + 1$
end loop

Compared to Q-learning, how does this change the: memory requirements, computation requirements per step, amount of data required?

Why Double Q-Learning?

- To help with maximization bias…
- The following step causes the maximization bias:

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha\left(r_t + \gamma \max_{a \in A} Q(s_{t+1}, a) - Q(s_t, a_t)\right)$$

because, in general:

$\mathbb{E}[\max\{X_1, X_2, \ldots, X_k\}] \neq \max\{\mathbb{E}[X_1], \mathbb{E}[X_2], \ldots, \mathbb{E}[X_k]\}$, and in fact:

$\mathbb{E}[\max\{X_1, X_2, \ldots, X_k\}] \geq \max\{\mathbb{E}[X_1], \mathbb{E}[X_2], \ldots, \mathbb{E}[X_k]\}$.

- So even if the estimates of $Q(s, a)$ were unbiased, $\max_{a \in A} Q(s_{t+1}, a)$ would not have to be unbiased.

Double Q-Learning

Double Q-Learning

Initialize $Q_1(s,a)$ and $Q_2(s,a)$, $\forall s \in S, a \in A$ $t = 0$, initial state $s_t = s_0$
loop
  Select a_t using ϵ-greedy $\pi(s) = \arg\max_a Q_1(s_t, a) + Q_2(s_t, a)$
  Observe (r_t, s_{t+1})
  if (with 0.5 probability) **then**
    $Q_1(s_t, a_t) \leftarrow Q_1(s_t, a_t) + \alpha(r_t + \gamma \underline{Q_2}(s_{t+1}, \arg\max_a \underline{Q_1}(s_{t+1}, a)) - Q_1(s_t, a_t))$
  else
    $Q_2(s_t, a_t) \leftarrow Q_2(s_t, a_t) + \alpha(r_t + \gamma \underline{Q_1}(s_{t+1}, \arg\max_a \underline{Q_2}(s_{t+1}, a)) - Q_2(s_t, a_t))$
  end if
  $t = t + 1$
end loop

Compared to Q-learning, how does this change the: memory requirements, computation requirements per step, amount of data required?