CS 747, Autumn 2020: Week 8, Lecture 1

Shivaram Kalyanakrishnan

Department of Computer Science and Engineering
Indian Institute of Technology Bombay

Autumn 2020

Reinforcement Learning

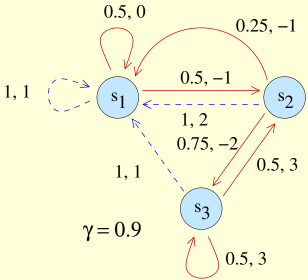
1. Reinforcement Learning problem
   - Prediction, control
   - Assumptions
2. Basic algorithm for control
3. Prediction with a Monte Carlo method

Reinforcement Learning

1. Reinforcement Learning problem
   - Prediction, control
   - Assumptions
2. Basic algorithm for control
3. Prediction with a Monte Carlo method

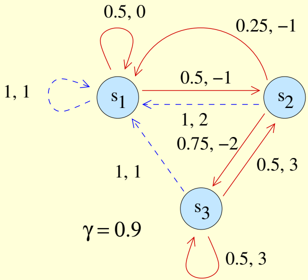
Agent-Environment Interaction

Underlying MDP:

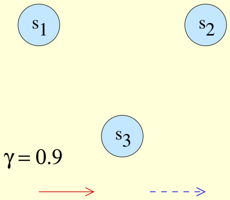


Agent-Environment Interaction

Underlying MDP:

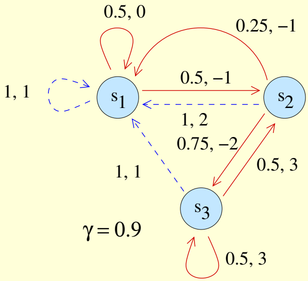


Agent's view:

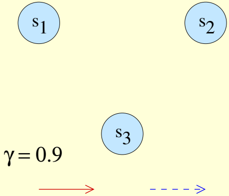


Agent-Environment Interaction

Underlying MDP:



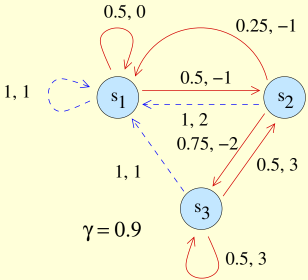
Agent's view:



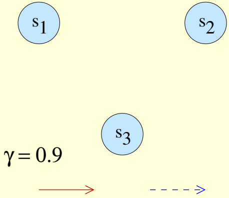
- From current state, agent takes action.

Agent-Environment Interaction

Underlying MDP:



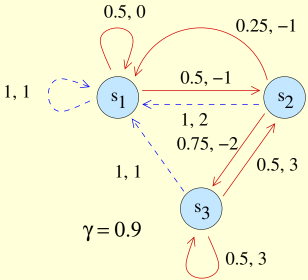
Agent's view:



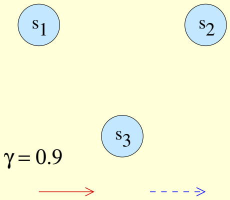
- From current state, agent takes action.
- Environment (MDP) decides next state and reward.

Agent-Environment Interaction

Underlying MDP:



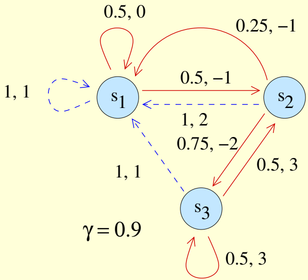
Agent's view:



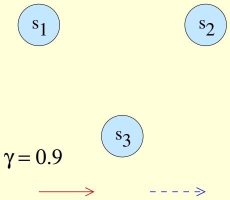
- From current state, agent takes action.
- Environment (MDP) decides next state and reward.
- Possible history: s_2, RED, -2, s_3, BLUE, 1, s_1, RED, 0, s_1, ...

Agent-Environment Interaction

Underlying MDP:



Agent's view:



- From current state, agent takes action.
- Environment (MDP) decides next state and reward.
- Possible history: s_2, RED, -2, s_3, BLUE, 1, s_1, RED, 0, s_1, ...
- History conveys information about the MDP to the agent.

The Control Problem

- For $t \geq 0$, let $h^t = (s^0, a^0, r^0, s^1, a^1, r^1, s^2, \ldots, s^t)$ denote a t-length history.

The Control Problem

- For $t \geq 0$, let $h^t = (s^0, a^0, r^0, s^1, a^1, r^1, s^2, \ldots, s^t)$ denote a t-length history.

- A learning algorithm L is a mapping from the set of all histories to the set of all (probability distributions over) arms.

The Control Problem

- For $t \geq 0$, let $h^t = (s^0, a^0, r^0, s^1, a^1, r^1, s^2, \ldots, s^t)$ denote a t-length history.
- A learning algorithm L is a mapping from the set of all histories to the set of all (probability distributions over) arms.
- Actions are selected by the learning algorithm (agent); next states and rewards by the MDP (environment).

The Control Problem

- For $t \geq 0$, let $h^t = (s^0, a^0, r^0, s^1, a^1, r^1, s^2, \ldots, s^t)$ denote a t-length history.

- A learning algorithm L is a mapping from the set of all histories to the set of all (probability distributions over) arms.

- Actions are selected by the learning algorithm (agent); next states and rewards by the MDP (environment).

- **Control problem**: Can we construct L such that

$$\lim_{T \to \infty} \frac{1}{T} \left(\sum_{t=0}^{T-1} \mathbb{P}\{a^t \sim L(h^t) \text{ is an optimal action for } s^t\} \right) = 1?$$

The Prediction Problem

- We are given a policy π that the agent follows.
  The aim is to estimate V^{π}.

The Prediction Problem

- We are given a policy π that the agent follows.
  The aim is to estimate V^{π}.

- For $t \geq 0$, let $h^t = (s^0, a^0, r^0, s^1, a^1, r^1, s^2, \ldots, s^t)$ denote a t-length history (note that $a^t \sim \pi(s^t)$).

The Prediction Problem

- We are given a policy π that the agent follows.
  The aim is to estimate V^{π}.

- For $t \geq 0$, let $h^t = (s^0, a^0, r^0, s^1, a^1, r^1, s^2, \ldots, s^t)$ denote a t-length history (note that $a^t \sim \pi(s^t)$).

- A learning algorithm L is a mapping from the set of all histories to the set of all mappings of the form $S \rightarrow \mathbb{R}$.

The Prediction Problem

- We are given a policy π that the agent follows.
  The aim is to estimate V^π.
- For $t \geq 0$, let $h^t = (s^0, a^0, r^0, s^1, a^1, r^1, s^2, \ldots, s^t)$ denote a t-length history (note that $a^t \sim \pi(s^t)$).
- A learning algorithm L is a mapping from the set of all histories to the set of all mappings of the form $S \to \mathbb{R}$.
- In other words, at each step t the learning algorithm provides an estimate $\hat{V}^t$.

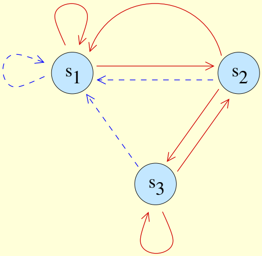
The Prediction Problem

- We are given a policy π that the agent follows. The aim is to estimate V^π.
- For $t \geq 0$, let $h^t = (s^0, a^0, r^0, s^1, a^1, r^1, s^2, \dots, s^t)$ denote a t-length history (note that $a^t \sim \pi(s^t)$).
- A learning algorithm L is a mapping from the set of all histories to the set of all mappings of the form $S \to \mathbb{R}$.
- In other words, at each step t the learning algorithm provides an estimate $\hat{V}^t$.
- **Prediction problem**: Can we construct L such that

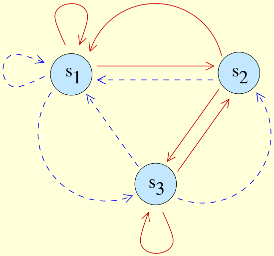
$$\lim_{t \to \infty} \hat{V}^t = V^\pi?$$

Assumption 1: Irreducibility

- Fix an MDP $M = (S, A, T, R, \gamma)$ and a policy π.
- Draw a graph with states as vertices and every non-zero-probability transition under π as a directed edge.
- Is there a directed path from s to s' for every $s, s' \in S$?
- If yes, M is irreducible under π.
- If M is irreducible under all $\pi \in \Pi$, then M is irreducible.



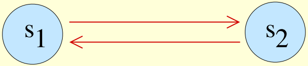
Reducible



Irreducible

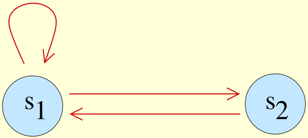
Assumption 2: Aperiodicity

- Fix an MDP $M = (S, A, T, R, \gamma)$ and a policy π.
- For $s \in S$, $t \geq 1$, let $X(s, t)$ be the set of all states s' such that there is a non-zero probability of reaching s' in exactly t steps by starting at s and following π.
- For $s \in S$, let $Y(s)$ be the set of all $t \geq 1$ such that $s \in X(s, t)$; let $p(s) = \gcd(Y(s))$.
- M is aperiodic under π if for all $s \in S$: $p(s) = 1$.
- If M is aperiodic under all $\pi \in \Pi$, then M is aperiodic.



$Y(s_1) = \{2, 4, 6, \ldots\}$.

Periodic.



$Y(s_1) = \{1, 2, 3, \ldots,\}$.

$Y(s_2) = \{2, 3, 4, \ldots,\}$.

Aperiodic.

Ergodicity

- An MDP that is irreducible and aperiodic is called an ergodic MDP.

Ergodicity

- An MDP that is irreducible and aperiodic is called an ergodic MDP.
- In an ergodic MDP, every policy π induces a unique steady state distribution $\mu^\pi : S \to (0, 1)$, subject to $\sum_{s \in S} \mu^\pi(s) = 1$, which is independent of the start state.

Ergodicity

- An MDP that is irreducible and aperiodic is called an ergodic MDP.
- In an ergodic MDP, every policy π induces a unique steady state distribution $\mu^\pi : S \to (0, 1)$, subject to $\sum_{s \in S} \mu^\pi(s) = 1$, which is independent of the start state.
- For $s \in S$, $t \geq 0$, let $p(s, t)$ be the probability of being in state s at step t, after starting at some (arbitrarily) fixed state and following π.

Ergodicity

- An MDP that is irreducible and aperiodic is called an ergodic MDP.
- In an ergodic MDP, every policy π induces a unique steady state distribution $\mu^\pi : S \to (0, 1)$, subject to $\sum_{s \in S} \mu^\pi(s) = 1$, which is independent of the start state.
- For $s \in S$, $t \geq 0$, let $p(s, t)$ be the probability of being in state s at step t, after starting at some (arbitrarily) fixed state and following π. Then

$$\mu^\pi(s) = \lim_{t \to \infty} p(s, t).$$

Ergodicity

- An MDP that is irreducible and aperiodic is called an ergodic MDP.
- In an ergodic MDP, every policy π induces a unique steady state distribution $\mu^\pi : S \to (0, 1)$, subject to $\sum_{s \in S} \mu^\pi(s) = 1$, which is independent of the start state.
- For $s \in S$, $t \geq 0$, let $p(s, t)$ be the probability of being in state s at step t, after starting at some (arbitrarily) fixed state and following π. Then

$$\mu^\pi(s) = \lim_{t \to \infty} p(s, t).$$

- We'll use ergodicity in some of the later lectures.

Reinforcement Learning

1. Reinforcement Learning problem
   - Prediction, control
   - Assumptions
2. Basic algorithm for control
3. Prediction with a Monte Carlo method

A Model-based Approach

- A model is an estimate of the MDP, which is usually updated based on experience.

  We keep estimates $\hat{T}$ and $\hat{R}$, and hope to get them to converge to T and R, respectively.

A Model-based Approach

- A model is an estimate of the MDP, which is usually updated based on experience.
  We keep estimates $\hat{T}$ and $\hat{R}$, and hope to get them to converge to T and R, respectively.

- At convergence, acting optimally for MDP $(S, A, \hat{T}, \hat{R}, \gamma)$ must be optimal for the original MDP (S, A, T, R, γ), too.

A Model-based Approach

- A model is an estimate of the MDP, which is usually updated based on experience.

  We keep estimates $\hat{T}$ and $\hat{R}$, and hope to get them to converge to T and R, respectively.

- At convergence, acting optimally for MDP $(S, A, \hat{T}, \hat{R}, \gamma)$ must be optimal for the original MDP (S, A, T, R, γ), too.

- We must visit every state-action pair infinitely often.

A Model-based Approach

- A model is an estimate of the MDP, which is usually updated based on experience.

  We keep estimates $\hat{T}$ and $\hat{R}$, and hope to get them to converge to T and R, respectively.

- At convergence, acting optimally for MDP $(S, A, \hat{T}, \hat{R}, \gamma)$ must be optimal for the original MDP (S, A, T, R, γ), too.

- We must visit every state-action pair infinitely often.

- Remember GLIE?

Algorithm

Model-based RL

//Initialisation

For $s, s' \in S, a \in A$:

$\hat{T}[s][a][s'] \leftarrow 0$; $\hat{R}[s][a][s'] \leftarrow 0$.

$modelValid \leftarrow False$.

For $s, s' \in S, a \in A$:

$totalTransitions[s][a][s'] \leftarrow 0$;

$totalReward[s][a][s'] \leftarrow 0$.

For $s \in S, a \in A$:

$totalVisits[s][a] \leftarrow 0$.

Assume that the agent is born in state s^0.

(Continued on next slide.)

Algorithm

(Continued from previous slide.)

Assume that the agent is born in state s^0.

//For ever
For $t = 0, 1, 2, \ldots$:
    If *modelValid*:
        $\pi^{opt} \leftarrow MDPPlan(S, A, \hat{T}, \hat{R}, \gamma)$.
        $$a^t \leftarrow \begin{cases} \pi^{opt}(s^t) & \text{w. p. } 1 - \epsilon_t, \\ UniformRandom(A) & \text{w. p. } \epsilon_t. \end{cases}$$
    Else:
        $a^t \leftarrow UniformRandom(A)$.

    Take action a^t; obtain reward r^t, next state s^{t+1}.
    $UpdateModel(s^t, a^t, r^t, s^{t+1})$.

Algorithm

UpdateModel$(\mathbf{s}, \mathbf{a}, \mathbf{r}, \mathbf{s}')$

$totalTransitions[s][a][s'] \leftarrow totalTransitions[s][a][s'] + 1.$
$totalVisits[s][a] \leftarrow totalVisits[s][a] + 1.$
$totalReward[s][a][s'] \leftarrow totalReward[s][a][s'] + r.$

For $s'' \in S$:

$$\hat{T}[s][a][s''] \leftarrow \frac{totalTransitions[s][a][s'']}{totalVisits[s][a]}.$$

$$\hat{R}[s][a][s'] \leftarrow \frac{totalReward[s][a][s']}{totalTransitions[s][a][s']}.$$

If $\neg modelValid$:

If $\forall s'' \in S, \forall a'' \in A : totalVisits[s''][a''] \geq 1$:

$modelValid \leftarrow True.$

Discussion

- Algorithm takes a sub-linear number of sub-optimal actions. Can still be optimised in many ways (computational complexity, exploration, etc.).

Discussion

- Algorithm takes a sub-linear number of sub-optimal actions. Can still be optimised in many ways (computational complexity, exploration, etc.).

- For convergence to optimal behaviour, does the algorithm need irreducibility and aperiodicity?

Discussion

- Algorithm takes a sub-linear number of sub-optimal actions. Can still be optimised in many ways (computational complexity, exploration, etc.).
- For convergence to optimal behaviour, does the algorithm need irreducibility and aperiodicity?

  Needs irreducibility, not aperiodicity.

Discussion

- Algorithm takes a sub-linear number of sub-optimal actions. Can still be optimised in many ways (computational complexity, exploration, etc.).

- For convergence to optimal behaviour, does the algorithm need irreducibility and aperiodicity?

  Needs irreducibility, not aperiodicity.

- Why is this a "model-based" algorithm?

Discussion

- Algorithm takes a sub-linear number of sub-optimal actions. Can still be optimised in many ways (computational complexity, exploration, etc.).
- For convergence to optimal behaviour, does the algorithm need irreducibility and aperiodicity?

  Needs irreducibility, not aperiodicity.
- Why is this a “model-based” algorithm?

  Uses $\theta(|S|^2|A|)$ memory. Will soon see a “model-free” method that needs $\theta(|S||A|)$ memory.

Reinforcement Learning

1. Reinforcement Learning problem
   - Prediction, control
   - Assumptions

2. Basic algorithm for control

3. Prediction with a Monte Carlo method

Prediction

- Assume we have an episodic task. $S = \{s_1, s_2, s_3\}$, $\gamma = 1$. On each episode, start state picked uniformly at random.

Prediction

- Assume we have an episodic task. $S = \{s_1, s_2, s_3\}$, $\gamma = 1$. On each episode, start state picked uniformly at random.

- Here are the first 5 episodes.

Episode 1: $s_1, 5, s_1, 2, s_2, 3, s_2, 1, s_\top$.
Episode 2: $s_2, 2, s_3, 1, s_3, 1, s_3, 2, s_2, 1, s_\top$.
Episode 3: $s_1, 2, s_2, 2, s_1, 5, s_1, 1, s_\top$.
Episode 4: $s_3, 1, s_\top$.
Episode 5: $s_2, 3, s_2, 3, s_1, 1, s_\top$

Prediction

- Assume we have an episodic task. $S = \{s_1, s_2, s_3\}$, $\gamma = 1$. On each episode, start state picked uniformly at random.

- Here are the first 5 episodes.

Episode 1: $s_1, 5, s_1, 2, s_2, 3, s_2, 1, s_\top$.
Episode 2: $s_2, 2, s_3, 1, s_3, 1, s_3, 2, s_2, 1, s_\top$.
Episode 3: $s_1, 2, s_2, 2, s_1, 5, s_1, 1, s_\top$.
Episode 4: $s_3, 1, s_\top$.
Episode 5: $s_2, 3, s_2, 3, s_1, 1, s_\top$

- What is your estimate of V^π (call it $\hat{V}^5$)?

Prediction

- Assume we have an episodic task. $S = \{s_1, s_2, s_3\}$, $\gamma = 1$. On each episode, start state picked uniformly at random.

- Here are the first 5 episodes.

Episode 1: $s_1, 5, s_1, 2, s_2, 3, s_2, 1, s_\top$.
Episode 2: $s_2, 2, s_3, 1, s_3, 1, s_3, 2, s_2, 1, s_\top$.
Episode 3: $s_1, 2, s_2, 2, s_1, 5, s_1, 1, s_\top$.
Episode 4: $s_3, 1, s_\top$.
Episode 5: $s_2, 3, s_2, 3, s_1, 1, s_\top$

- What is your estimate of V^π (call it $\hat{V}^5$)? Monte Carlo methods estimate based on sample averages.

Defining Relevant Quantities

- For $s \in S$, $i \geq 1, j \geq 1$, let
  - $\mathbf{1}(s, i, j)$ be 1 if s is visited at least j times on episode i is s (else $\mathbf{1}(s, i, j) = 0$), and
  - $G(s, i, j)$ be the discounted long-term reward starting from the j-th visit of s on episode i,
  - Taking $G(s, i, j) = 0$ if $\mathbf{1}(s, i, j) = 0$; also $0/0 = 0$.

Episode 1: $s_1, 5, s_1, 2, s_2, 3, s_2, 1, s_\top$.
Episode 2: $s_2, 2, s_3, 1, s_3, 1, s_3, 2, s_2, 1, s_\top$.
Episode 3: $s_1, 2, s_2, 2, s_1, 5, s_1, 1, s_\top$.
Episode 4: $s_3, 1, s_\top$.
Episode 5: $s_2, 3, s_2, 3, s_1, 1, s_\top$

Defining Relevant Quantities

- For $s \in S$, $i \geq 1, j \geq 1$, let
  - $\mathbf{1}(s, i, j)$ be 1 if s is visited at least j times on episode i is s (else $\mathbf{1}(s, i, j) = 0$), and
  - $G(s, i, j)$ be the discounted long-term reward starting from the j-th visit of s on episode i,
  - Taking $G(s, i, j) = 0$ if $\mathbf{1}(s, i, j) = 0$; also $0/0 = 0$.

Episode 1: $s_1, 5, s_1, 2, s_2, 3, s_2, 1, s_\top$.
Episode 2: $s_2, 2, s_3, 1, s_3, 1, s_3, 2, s_2, 1, s_\top$.
Episode 3: $s_1, 2, s_2, 2, s_1, 5, s_1, 1, s_\top$.
Episode 4: $s_3, 1, s_\top$.
Episode 5: $s_2, 3, s_2, 3, s_1, 1, s_\top$

- $\mathbf{1}(s_1, 1, 1) = 1$, $G(s_1, 1, 1) = 5 + \gamma \cdot 2 + \gamma^2 \cdot 3 + \gamma^3 \cdot 1 = 11$.
- $\mathbf{1}(s_1, 1, 3) = 0$.
- $\mathbf{1}(s_2, 5, 1) = 1$, $G(s_2, 5, 1) = 3 + \gamma \cdot 3 + \gamma^2 \cdot 1 = 7$.
- $\mathbf{1}(s_2, 5, 2) = 1$, $G(s_2, 5, 2) = 3 + \gamma \cdot 1 = 4$.

Some Standard Estimates of $V^\pi(s)$

Episode 1: $s_1, 5, s_1, 2, s_2, 3, s_2, 1, s_\top$.
Episode 2: $s_2, 2, s_3, 1, s_3, 1, s_3, 2, s_2, 1, s_\top$.
Episode 3: $s_1, 2, s_2, 2, s_1, 5, s_1, 1, s_\top$.
Episode 4: $s_3, 1, s_\top$.
Episode 5: $s_2, 3, s_2, 3, s_1, 1, s_\top$

Let $\hat{V}^T$ denote estimate after T *episodes*.

First-visit Monte Carlo: Average the G's of every first occurrence of s in an episode.

$$\hat{V}^T_{\text{First-visit}}(s) = \frac{\sum_{i=1}^{T} G(s, i, 1)}{\sum_{i=1}^{T} \mathbf{1}(s, i, 1)}.$$

Hence

$$\hat{V}^5_{\text{First-visit}}(s_2) = \frac{4 + 7 + 8 + 7}{4} = 6.5.$$

Some Standard Estimates of $V^{\pi}(s)$

Episode 1: $s_1, 5, s_1, 2, s_2, 3, s_2, 1, s_\top$.
Episode 2: $s_2, 2, s_3, 1, s_3, 1, s_3, 2, s_2, 1, s_\top$.
Episode 3: $s_1, 2, s_2, 2, s_1, 5, s_1, 1, s_\top$.
Episode 4: $s_3, 1, s_\top$.
Episode 5: $s_2, 3, s_2, 3, s_1, 1, s_\top$

Let $\hat{V}^T$ denote estimate after T *episodes*.

Every-visit Monte Carlo: Average the G's of every occurrence of s in an episode.

$$\hat{V}^T_{\text{Every-visit}}(s) = \frac{\sum_{i=1}^{T} \sum_{j=1}^{\infty} G(s, i, j)}{\sum_{i=1}^{T} \sum_{j=1}^{\infty} \mathbf{1}(s, i, j)}.$$

Hence

$$\hat{V}^5_{\text{Every-visit}}(s_2) = \frac{(4+1) + (7+1) + 8 + (7+4)}{7} \approx 4.57.$$

Some Not-so-standard Estimates of $V^\pi(s)$

Episode 1: $s_1, 5, s_1, 2, s_2, 3, s_2, 1, s_\top$.
Episode 2: $s_2, 2, s_3, 1, s_3, 1, s_3, 2, s_2, 1, s_\top$.
Episode 3: $s_1, 2, s_2, 2, s_1, 5, s_1, 1, s_\top$.
Episode 4: $s_3, 1, s_\top$.
Episode 5: $s_2, 3, s_2, 3, s_1, 1, s_\top$

Let $\hat{V}^T$ denote estimate after T *episodes*.

Second-visit Monte Carlo: Average the G's of every second occurrence of s in an episode.

$$\hat{V}^T_{\text{Second-visit}}(s) = \frac{\sum_{i=1}^T G(s, i, 2)}{\sum_{i=1}^T \mathbf{1}(s, i, 2)}.$$

Hence

$$\hat{V}^5_{\text{Second-visit}}(s_2) = \frac{1+1+4}{3} = 2.$$

Some Not-so-standard Estimates of $V^\pi(s)$

Episode 1: $s_1, 5, s_1, 2, s_2, 3, s_2, 1, s_\top$.
Episode 2: $s_2, 2, s_3, 1, s_3, 1, s_3, 2, s_2, 1, s_\top$.
Episode 3: $s_1, 2, s_2, 2, s_1, 5, s_1, 1, s_\top$.
Episode 4: $s_3, 1, s_\top$.
Episode 5: $s_2, 3, s_2, 3, s_1, 1, s_\top$

Let $\hat{V}^T$ denote estimate after T *episodes*.

Last-visit Monte Carlo: Average the G's of every last occurrence of s in episode i (assume $times(s, i)$ visits).

$$\hat{V}^T_{\text{Last-visit}}(s) = \frac{\sum_{i=1}^{T} G(s, i, times(s, i))}{\sum_{i=1}^{T} \mathbf{1}(s, i, times(s, i))}.$$

Hence

$$\hat{V}^5_{\text{Last-visit}}(s_2) = \frac{1 + 1 + 8 + 4}{4} = 3.5.$$

Question

- Recall that we generate T episodes.
- Which claims below are true?

$$\lim_{T \to \infty} \hat{V}^{T}_{\text{First-visit}} = V^{\pi}.$$

$$\lim_{T \to \infty} \hat{V}^{T}_{\text{Every-visit}} = V^{\pi}.$$

$$\lim_{T \to \infty} \hat{V}^{T}_{\text{Second-visit}} = V^{\pi}.$$

$$\lim_{T \to \infty} \hat{V}^{T}_{\text{Last-visit}} = V^{\pi}.$$

Reinforcement Learning

1. Reinforcement Learning problem
   - Prediction, control
   - Assumptions

2. Basic algorithm for control

3. Prediction with a Monte Carlo method