



Reinforcement Learning

Monte Carlo Methods

U Kang
Seoul National University



In This Lecture

- Overview of Monte Carlo (MC) methods for RL
- MC prediction
- MC control
- Off-policy methods for MC



Overview

- Monte Carlo (MC) Methods
 - Do not assume complete knowledge of the environment
 - MC methods require only experience (sample sequences of states, actions, and rewards) from actual or simulated interaction with an environment
 - Learning from actual experience is useful because it requires no prior knowledge of the environment
 - Learning from simulated experience is also powerful; Although a model is required, the model needs only to generate sample transitions, not the complete probability distributions of all possible transitions as in DP



Overview

- Monte Carlo (MC) Methods
 - MC methods are ways of solving RL problem based on averaging sample returns
 - MC methods are defined only for episodic tasks; only on the completion of an episode are value estimates and policies changed
 - MC methods can thus be incremental in an episode-by-episode sense, but not in a step-by-step (online) sense
 - “Monte Carlo” is often used more broadly for any estimation method whose operation involves a significant random component



Overview

- Monte Carlo (MC) Methods
 - MC methods sample and average returns for each state–action pair much like the bandit methods sample and average rewards for each action
 - In MC, there are multiple states, acting like different bandit problems which are interrelated
 - The return after taking an action in one state depends on the actions taken in later states in the same episode
 - Because all the action selections are undergoing learning, the problem becomes nonstationary from the point of view of the earlier state



Overview

- Monte Carlo (MC) Methods
 - To handle the nonstationarity, we adapt the idea of general policy iteration (GPI)
 - In DP, we *computed* value functions from knowledge of the MDP; in MC, we *learn* value functions from sample returns with the MDP
 - Outline
 - Prediction problem (computation of v_π and q_π for a fixed policy π)
 - Policy improvement
 - Control problem and its solution by GPI



Outline

- ➔ **Monte Carlo (MC) Prediction**
- MC Estimation of Action Values
- MC Control
- MC Control without Exploring Starts
- Off-policy Prediction via Importance Sampling
- Incremental Implementation
- Off-policy MC Control
- Conclusion



MC Prediction

- Learn the state-value function for a given policy by MC methods
- Value of a state: expected return (expected cumulative future discounted reward) starting from the state
- MC methods estimate it from experience; average the returns observed after visits to that state
- As more returns are observed, the average should converge to the expected value



MC Prediction

- Suppose we wish to estimate $v_{\pi}(s)$, the value of a state s under policy π , given a set of episodes obtained by following π and passing through s
- Each occurrence of state s in an episode is called a visit to s ; s may be visited multiple times in the same episode
- First-visit MC method: estimates $v_{\pi}(s)$ as the average of the returns following first visits to s
- Every-visit MC method: averages the returns following all visits to s



MC Prediction

First-visit MC prediction, for estimating $V \approx v_\pi$

Input: a policy π to be evaluated

Initialize:

$V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$

$Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

Loop forever (for each episode):

Generate an episode following π : $S_0, A_0, R_1, S_1, A_1, R_2, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \dots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless S_t appears in S_0, S_1, \dots, S_{t-1} :

Append G to $Returns(S_t)$

$V(S_t) \leftarrow \text{average}(Returns(S_t))$



MC Prediction

- Both first-visit MC and every-visit MC converge to $v_\pi(s)$ as the number of visits to s goes to infinity
- First-visit MC
 - Each return is an independent, identically distributed estimate of $v_\pi(s)$ with finite variance
 - By the law of large numbers the sequence of averages of these estimates converges to their expected value
 - Each average is itself an unbiased estimate, and the standard deviation of its error falls as $1/\sqrt{n}$, where n is the number of returns averaged
- Every-visit MC: also converges to $v_\pi(s)$



Example: Blackjack





Example: Blackjack

- Object of blackjack: to obtain cards whose sum is as great as possible without exceeding 21
- All face cards count as 10, and an ace can count either as 1 or as 11
- The game begins with two cards dealt to both dealer and player; one of the dealer's cards is face up and the other is face down
- If the player has 21 immediately (an ace and a 10-card), it is called a natural; he then wins unless the dealer also has a natural, in which case the game is a draw
- If the player does not have a natural, then he can request additional cards, one by one (hits), until he either stops (sticks) or exceeds 21 (goes bust)
- If he goes bust, he loses; if he sticks, then it becomes the dealer's turn. The dealer hits or sticks according to a fixed strategy without choice: he sticks on any sum of 17 or greater, and hits otherwise
- If the dealer goes bust, then the player wins; otherwise, the outcome is determined by whose final sum is closer to 21.



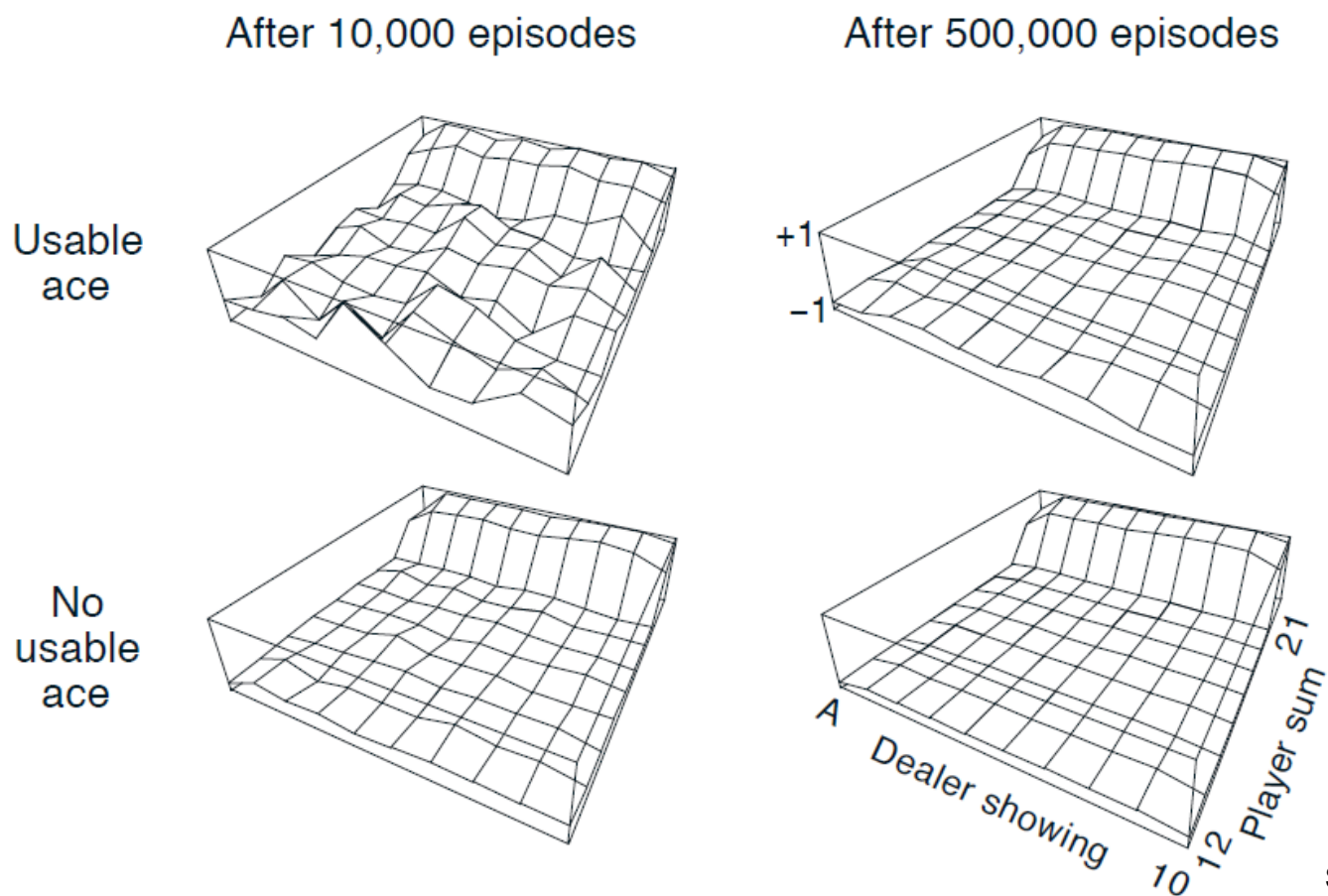
Example: Blackjack

- Playing blackjack is naturally formulated as an episodic finite MDP
- Rewards: +1, -1, and 0 for winning, losing, and drawing
 - All rewards within a game are zero, and we do not discount; thus the terminal rewards are also the returns
- Actions: hit or stick
- States: depend on the player's cards and the dealer's showing card
 - If the player holds an ace that he could count as 11 without going bust, then the ace is said to be usable.
 - The player makes decisions on the basis of three variables: his current sum (12–21), the dealer's one showing card (ace–10), and whether or not he holds a usable ace
 - This makes for a total of 200 states



Example: Blackjack

- Example policy: sticks if the player's sum is 20 or 21, and otherwise hits

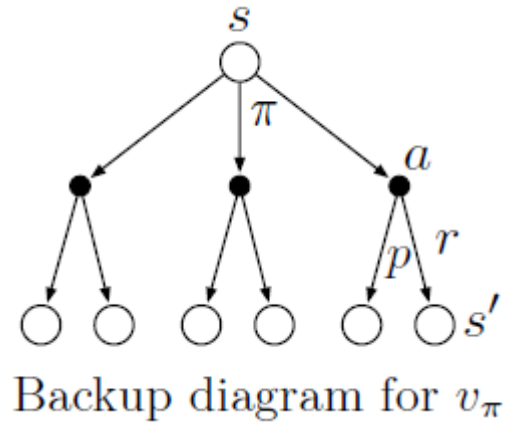




Backup Diagram for MC



MC



DP

Sutton and Barto,
Reinforcement
Learning, 2018



MC's Advantages over DP

- Advantage 1
 - MC does not require complete knowledge of the environment
- MC for blackjack is easy: one simulates many blackjack games using a policy and averages the returns following each state
- DP for blackjack
 - Although we have complete knowledge of the environment in the blackjack task, it would not be easy to apply DP methods to compute the value function
 - DP methods require the distribution of next events, which is non-trivial to compute
 - For example, suppose the player's sum is 14 and he chooses to stick. What is his probability of terminating with a reward of +1 as a function of the dealer's showing card? Computing the probabilities is complex and error-prone



MC's Advantages over DP

- Advantage 2
 - MC allows to learn from actual experience and from simulated experience
 - DP does not learn from experiences



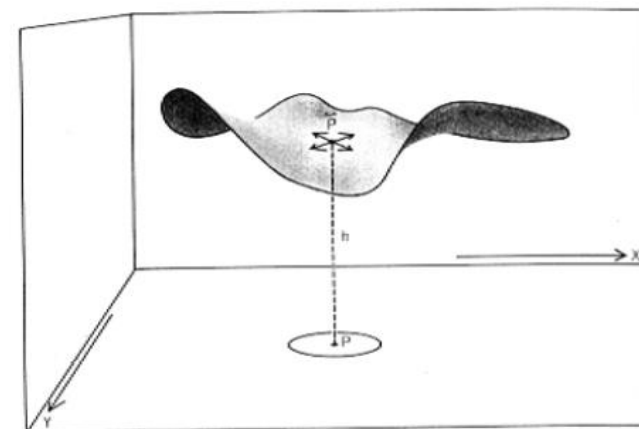
MC's Advantages over DP

- MC vs. DP
 - DP: bootstrap (an estimate for each state builds on the estimate of any other state)
 - MC: does not bootstrap (an estimate for each state does not build on the estimate of any other state)
- Advantage 3
 - The computational expense of estimating the value of a single state in MC is independent of the number of states
 - This is useful especially when one requires the value of only a subset of states
 - In MC, one can generate many sample episodes starting from the states of interest, ignoring all others



Example: Soap Bubble

- Suppose a wire frame forming a closed loop is dunked in soapy water to form a soap bubble conforming at its edges to the wire frame
- If the geometry of the wire frame is known, how to compute the shape of the surface?
- The total force on each point exerted by neighboring points is zero (or else the shape would change); the surface's height at any point is the average of its heights at points in a small circle around that point
- The surface must also meet at its boundaries with the wire frame



A bubble on a wire loop.




Example: Soap Bubble

- DP solution for soap bubble problem
 - Put a grid over the area covered by the surface and solve for its height at the grid points by an iterative computation
 - This process then iterates, much like DP's iterative policy evaluation, and ultimately converges to a solution
- MC solution for soap bubble problem
 - Imagine standing on the surface and taking a random walk until you reach the boundary
 - The expected value of the height at the boundary is a close approximation to the height of the desired surface at the starting point
 - Thus, one can closely approximate the height of the surface at a point by simply averaging the boundary heights of many walks started at the point
 - If one is interested in only the value at one point, or any fixed small set of points, then this Monte Carlo method can be far more efficient than the iterative method based on local consistency



Outline

- Monte Carlo (MC) Prediction
-  **MC Estimation of Action Values**
- MC Control
- MC Control without Exploring Starts
- Off-policy Prediction via Importance Sampling
- Incremental Implementation
- Off-policy MC Control
- Conclusion



MC Estimation of Action Values

- If a model is not available, then it is particularly useful to estimate action values (the values of state–action pairs) rather than state values
- With a model, state values alone are sufficient to determine a policy; one simply looks ahead one step and chooses whichever action leads to the best combination of reward and next state
- Without a model, state values alone are not sufficient; one must explicitly estimate the value of each action in order for the values to be useful in suggesting a policy
 - Thus, one of our primary goals for MC methods is to estimate q_* ; to do this, we first consider the policy evaluation problem for action values



Monte Carlo state value $v_{\pi}(s)$



Monte Carlo action value $q_{\pi}(s, a)$



MC Estimation of Action Values

- Policy evaluation problem for action values: estimate $q_{\pi}(s, a)$, the expected return when starting in state s , taking action a , and thereafter following policy π
- MC methods for the problem are essentially the same as those for state values
 - Every-visit MC method: estimates the value of a state–action pair as the average of the returns that have followed all the visits to it
 - First-visit MC method: averages the returns following the first time in each episode that the state was visited and the action was selected
 - These methods converge to the true expected values as the number of visits to each state–action pair approaches infinity



MC Estimation of Action Values

- Complication: many state–action pairs may never be visited; if π is a deterministic policy, then one will observe returns only for one of the actions from each state
- MC estimates of the other actions will not be learned; this is a serious problem because the purpose of learning action values is to help in choosing among the actions available in each state




MC Estimation of Action Values

- Maintaining exploration
 - For policy evaluation to work for action values, we must assure continual exploration
 - A solution: exploring starts
 - Specifying that the episodes start in a state–action pair, and that every pair has a nonzero probability of being selected as the start
 - This guarantees that all state–action pairs will be visited an infinite number of times
 - However, this method does not work well when learning directly from actual interaction with an environment
 - Alternative solution: consider only stochastic policy
 - Stochastic policy: a nonzero probability of selecting all actions in each state



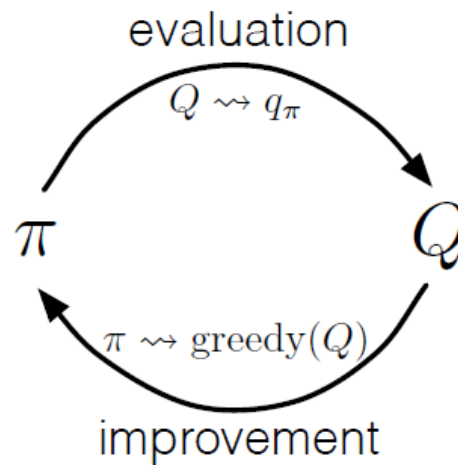
Outline

- Monte Carlo (MC) Prediction
- MC Estimation of Action Values
-  **MC Control**
- MC Control without Exploring Starts
- Off-policy Prediction via Importance Sampling
- Incremental Implementation
- Off-policy MC Control
- Conclusion



MC Control

- Goal: approximate optimal policies using MC estimation
- Main idea: generalized policy iteration (GPI)
 - The value function is repeatedly altered to more closely approximate the value function for the current policy, and the policy is repeatedly improved with respect to the current value function





MC Control

- MC version of policy iteration
 - Alternate complete steps of policy evaluation and policy improvement, beginning with an arbitrary policy π_0 and ending with the optimal policy and optimal action-value function

$$\pi_0 \xrightarrow{E} Q_{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} Q_{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \cdots \xrightarrow{I} \pi_* \xrightarrow{E} Q_*$$



MC Control

- MC version of policy iteration

$$\pi_0 \xrightarrow{E} q_{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} q_{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \cdots \xrightarrow{I} \pi_* \xrightarrow{E} q_*$$

- Policy evaluation: after experiencing many episodes, the approximate action-value function approaches the true function asymptotically



MC Control

- MC version of policy iteration

$$\pi_0 \xrightarrow{E} q_{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} q_{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \cdots \xrightarrow{I} \pi_* \xrightarrow{E} q_*$$

- Policy improvement: making the policy greedy with respect to the current action-value function (no model is required)

- $\pi(s) = \operatorname{argmax}_a q(s, a)$

- π_{k+1} is constructed as the greedy policy wrt q_{π_k}

- $$\begin{aligned} q_{\pi_k}(s, \pi_{k+1}(s)) &= q_{\pi_k}(s, \operatorname{argmax}_a q_{\pi_k}(s, a)) \\ &= \max_a q_{\pi_k}(s, a) \\ &\geq q_{\pi_k}(s, \pi_k(s)) \\ &= v_{\pi_k}(s) \end{aligned}$$

- π_{k+1} is uniformly better than or equal to π_k , which assures that the overall process converges to the optimal policy

- MC policy iteration requires only sample episodes, without knowledge of the environment



MC Control

- Convergence of MC policy iteration
 - Two assumptions
 - 1) Episodes have exploring starts
 - 2) Policy evaluation could be done with an infinite number of episodes
 - Removing assumption 2
 - Idea 1: approximate q_{π_k} as much as possible; the error will be related to the degree of approximation
 - Idea 2: give up trying to complete policy evaluation before returning to policy improvement. On each evaluation step we move the value function toward q_{π_k} , but we do not expect to actually get close to it
 - Example of idea 2: value iteration (only one iteration of iterative policy evaluation is performed between each step of policy improvement)



MC Control

- Monte Carlo with Exploring Starts
 - Alternate between evaluation and improvement on an episode-by-episode basis; after each episode, the observed returns are used for policy evaluation, and then the policy is improved at all the states visited in the episode

Monte Carlo ES (Exploring Starts), for estimating $\pi \approx \pi_*$

Initialize:

$\pi(s) \in \mathcal{A}(s)$ (arbitrarily), for all $s \in \mathcal{S}$

$Q(s, a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$

$Returns(s, a) \leftarrow$ empty list, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$

Loop forever (for each episode):

Choose $S_0 \in \mathcal{S}, A_0 \in \mathcal{A}(S_0)$ randomly such that all pairs have probability > 0

Generate an episode from S_0, A_0 , following π : $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \dots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1, \dots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$

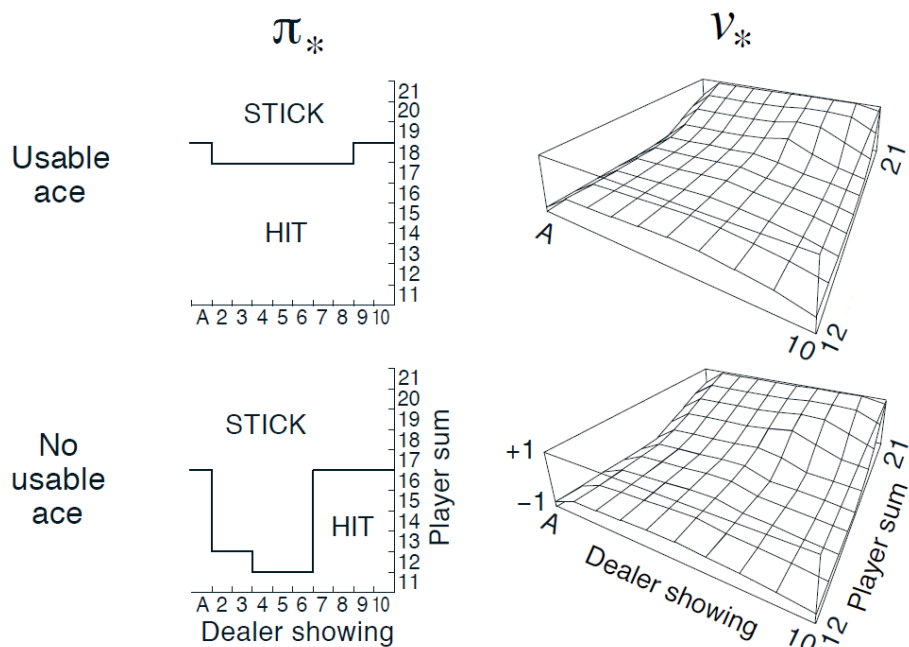
$\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$

Sutton and Barto,
Reinforcement
Learning, 2018




Example: Blackjack

- It is easy to arrange for exploring starts that include all possibilities since the episodes are all simulated games
- One simply picks the dealer's cards, the player's sum, and whether or not the player has a usable ace, all at random with equal probability
- As the initial policy we use the policy which sticks only on 20 or 21
- The initial action-value function can be zero for all state-action pairs





Outline

- Monte Carlo (MC) Prediction
- MC Estimation of Action Values
- MC Control
-  **MC Control without Exploring Starts**
- Off-policy Prediction via Importance Sampling
- Incremental Implementation
- Off-policy MC Control
- Conclusion



MC Control w.o. Exploring Starts

- Goal: ensure that all actions are selected infinitely often
- Question: How to achieve it without the unlikely assumption of exploring starts?
- Solutions:
 - On-policy methods: attempt to evaluate or improve the policy that is used to make decisions
 - Monte Carlo ES method (previous section)
 - Monte Carlo w.o. ES (focus of this section)
 - Off-policy methods: evaluate or improve a policy different from that used to generate the data



MC Control w.o. Exploring Starts

- ϵ -greedy policy: most of the time choose an action that has maximal estimated action value, but with probability ϵ select an action at random
 - All nongreedy actions: given the minimal probability of selection $\frac{\epsilon}{|A(s)|}$
 - Greedy action: $1 - \epsilon + \frac{\epsilon}{|A(s)|}$
- ϵ -soft policy: $\pi(a|s) \geq \frac{\epsilon}{|A(s)|}$ for all states and actions, for some $\epsilon > 0$
 - ϵ -greedy policy is a ϵ -soft policy



MC Control w.o. Exploring Starts

- GPI does not require that the policy be taken all the way to a greedy policy, only that it be moved toward a greedy policy
- MC control w.o. exploring starts
 - Move the policy to an ε -greedy policy
 - For any ε -soft policy π , any ε -greedy policy wrt q_π is guaranteed to be better than or equal to π



MC Control w.o. Exploring Starts

Sutton and Barto,
Reinforcement
Learning, 2018

On-policy first-visit MC control (for ε -soft policies), estimates $\pi \approx \pi_*$

Algorithm parameter: small $\varepsilon > 0$

Initialize:

$\pi \leftarrow$ an arbitrary ε -soft policy

$Q(s, a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

$Returns(s, a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Repeat forever (for each episode):

Generate an episode following π : $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \dots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1, \dots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow$ average($Returns(S_t, A_t)$)

$A^* \leftarrow \operatorname{argmax}_a Q(S_t, a)$ (with ties broken arbitrarily)

For all $a \in \mathcal{A}(S_t)$:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(S_t)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(S_t)| & \text{if } a \neq A^* \end{cases}$$



MC Control w.o. Exploring Starts

- For any ε -soft policy π , any ε -greedy policy π' wrt q_π is guaranteed to be better than or equal to π

- (proof)

$$q_\pi(s, \pi'(s)) = \sum_a \pi'(a|s) q_\pi(s, a)$$

$$= \frac{\varepsilon}{|\mathcal{A}(s)|} \sum_a q_\pi(s, a) + (1 - \varepsilon) \max_a q_\pi(s, a)$$

$$\geq \frac{\varepsilon}{|\mathcal{A}(s)|} \sum_a q_\pi(s, a) + (1 - \varepsilon) \sum_a \frac{\pi(a|s) - \frac{\varepsilon}{|\mathcal{A}(s)|}}{1 - \varepsilon} q_\pi(s, a)$$

$$= \frac{\varepsilon}{|\mathcal{A}(s)|} \sum_a q_\pi(s, a) - \frac{\varepsilon}{|\mathcal{A}(s)|} \sum_a q_\pi(s, a) + \sum_a \pi(a|s) q_\pi(s, a) = v_\pi(s)$$

- By the policy improvement theorem, $\pi' \geq \pi$ (i.e., $v_{\pi'}(s) \geq v_\pi(s)$)



MC Control w.o. Exploring Starts

- Policy iteration works for ε -soft policies
- Using the natural notion of greedy policy for ε -soft policies, one is assured of improvement on every step, until convergence
- Although achieving the best policy only among the ε -soft policies, we have eliminated the assumption of exploring starts

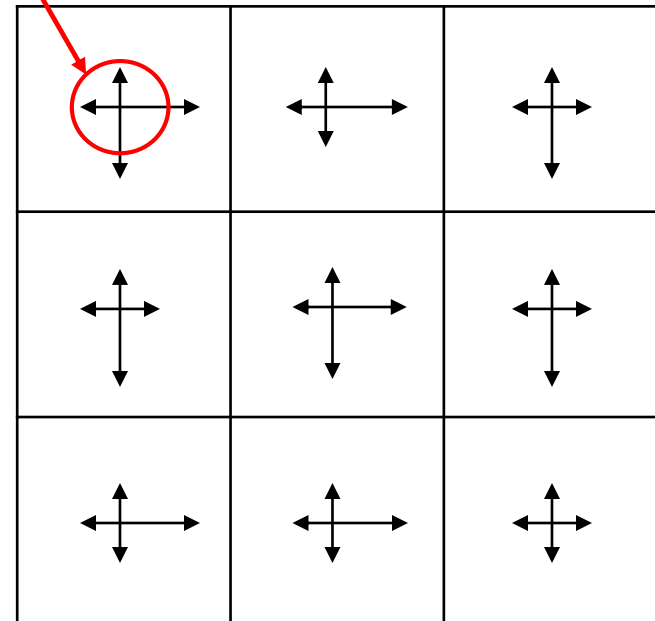


Random start for each episode

S_1	S_2	S_3
S_4	S_5	S_6
S_7	S_8	T

MC with exploring starts

Fixed start point



MC without exploring starts



Outline

- Monte Carlo (MC) Prediction
- MC Estimation of Action Values
- MC Control
- MC Control without Exploring Starts
- Off-policy Prediction via Importance Sampling**
- Incremental Implementation
- Off-policy MC Control
- Conclusion



Off-Policy Prediction via Importance Sampling

- All learning control methods face a dilemma: they seek to learn action values conditional on subsequent optimal behavior, but they need to behave non-optimally in order to explore all actions (to find the optimal actions)
- How to learn about the optimal policy while behaving according to an exploratory policy?
- A straightforward approach is to use two policies
 - Target policy: one that is learned about and that becomes the optimal policy
 - Behavior policy: one that is more exploratory and is used to generate behavior
- In this case we say that learning is from data *off* the target policy, and the overall process is termed off-policy learning



Off-Policy Prediction via Importance Sampling

- On-policy methods are generally simpler and are considered first
- Off-policy methods
 - (Pros) more powerful and general. They include on-policy methods as the special case in which the target and behavior policies are the same
 - (Pros) have a variety of additional uses in applications; e.g., they can be applied to learn from data generated by a conventional non-learning controller, or from a human expert
 - (Cons) often of greater variance and are slower to converge, since the data is due to a different policy



Problem Setting

- Off-policy prediction problem
 - Both target and behavior policies are fixed
 - Assume we wish to estimate v_π or q_π , but all we have are episodes following another policy $b \neq \pi$
 - π : target policy, b : behavior policy



Problem Setting

- Assumption of coverage
 - In order to use episodes from b to estimate values for π , every action taken under π should be also taken under b ; $\pi(a | s) > 0$ implies $b(a | s) > 0$; b must be stochastic in states where their actions with non-zero probabilities are not identical to those of π
 - The target policy π , on the other hand, may be deterministic, and, in fact, this is a case of particular interest in control applications
 - In control, the target policy is typically the deterministic greedy policy with respect to the current estimate of the action-value function. This policy becomes a deterministic optimal policy while the behavior policy remains stochastic and more exploratory, for example, an ε -greedy policy
 - In this section, we consider the prediction problem, in which π is unchanging and given



Off-Policy Prediction via Importance Sampling

- Importance sampling
 - A general technique for estimating expected values under one distribution given samples from another
 - Apply importance sampling to off-policy learning by weighting returns according to the relative probability of their trajectories occurring under the target and behavior policies, called the importance-sampling ratio



Off-Policy Prediction via Importance Sampling

- Importance sampling

- Given a starting state S_t , the probability of the subsequent state-action trajectory, $A_t, S_{t+1}, A_{t+1}, \dots, S_T$, occurring under policy π is

$$\begin{aligned} & \Pr\{A_t, S_{t+1}, A_{t+1}, \dots, S_T \mid S_t, A_{t:T-1} \sim \pi\} \\ &= \pi(A_t | S_t) p(S_{t+1} | S_t, A_t) \pi(A_{t+1} | S_{t+1}) \cdots p(S_T | S_{T-1}, A_{T-1}) \\ &= \prod_{k=t}^{T-1} \pi(A_k | S_k) p(S_{k+1} | S_k, A_k) \end{aligned}$$

- Importance-sampling ratio: relative probability of the trajectory under the target and behavior policies

$$\rho_{t:T-1} = \frac{\prod_{k=t}^{T-1} \pi(A_k | S_k) p(S_{k+1} | S_k, A_k)}{\prod_{k=t}^{T-1} b(A_k | S_k) p(S_{k+1} | S_k, A_k)} = \prod_{k=t}^{T-1} \frac{\pi(A_k | S_k)}{b(A_k | S_k)}$$

- Depends only on the two policies and the sequence, not on the MDP



Off-Policy Prediction via Importance Sampling

- Importance sampling
 - Goal: estimate the expected returns (values) under the target policy
 - We have G_t due to the behavior policy; these returns have the wrong expectation $E[G_t | S_t = s] = v_b(s)$ and so cannot be averaged to obtain v_π
 - Importance sampling ratio $\rho_{t:T-1}$ transforms the returns to have the right expected value: $E[\rho_{t:T-1} G_t | S_t = s] = v_\pi(s)$



Off-Policy Prediction via Importance Sampling

■ Importance sampling

- Importance sampling ratio $\rho_{t:T-1}$ transforms the returns to have the right expected value: $E[\rho_{t:T-1}G_t|S_t = s] = v_\pi(s)$
- (Proof)

Let $P(e^{(i)} \sim \pi | S_t)$ be

$$\begin{aligned} & \Pr\{A_t, S_{t+1}, A_{t+1}, \dots, S_T \mid S_t, A_{t:T-1} \sim \pi\} \\ &= \pi(A_t | S_t) p(S_{t+1} | S_t, A_t) \pi(A_{t+1} | S_{t+1}) \cdots p(S_T | S_{T-1}, A_{T-1}) \\ &= \prod_{k=t}^{T-1} \pi(A_k | S_k) p(S_{k+1} | S_k, A_k) \end{aligned}$$

Let $G^{(i)}$ be the return from the i th episode.

$$\text{Then, } v_b(s) = E[G_t | S_t = s] = \sum_i G^{(i)} P(e^{(i)} \sim b | S_t)$$

$$\text{Thus, } v_\pi(s) = \sum_i G^{(i)} P(e^{(i)} \sim \pi | S_t) = \sum_i G^{(i)} P(e^{(i)} \sim b | S_t) \rho_{t:T-1} = E[\rho_{t:T-1} G_t | S_t = s]$$

$$\rho_{t:T-1} = \frac{\prod_{k=t}^{T-1} \pi(A_k | S_k) p(S_{k+1} | S_k, A_k)}{\prod_{k=t}^{T-1} b(A_k | S_k) p(S_{k+1} | S_k, A_k)} = \prod_{k=t}^{T-1} \frac{\pi(A_k | S_k)}{b(A_k | S_k)}$$



MC Off-policy Algorithm

- Averages returns from a batch of observed episodes following policy b to estimate $v_{\pi}(s)$
- Assume time steps increase across episode boundaries; if the first episode of the batch ends in a terminal state at time 100, then the next episode begins at time $t = 101$
- $A(s)$ = the set of all time steps in which state s is visited (every-visit method)
 - For a first-visit method, $A(s)$ would only include time steps that were first visits to s within their episodes
- $T(t)$ = the first time of termination following time t
- G_t = the return after t up through $T(t)$
- $\{G_t\}_{t \in A(s)}$ are the returns that pertain to state s , and $\{\rho_{t:T(t)-1}\}_{t \in A(s)}$ are the corresponding importance-sampling ratios
- Then, $v_{\pi}(s) = \frac{\sum_{t \in A(s)} \rho_{t:T-1} G_t}{|A(s)|}$
- This is called ordinary importance sampling



MC Off-policy Algorithm

- Ordinary importance sampling

- $v_{\pi}(s) = \frac{\sum_{t \in A(s)} \rho_{t:T-1} G_t}{|A(s)|}$

- Alternative: weighted importance sampling

- $v_{\pi}(s) = \frac{\sum_{t \in A(s)} \rho_{t:T-1} G_t}{\sum_{t \in A(s)} \rho_{t:T-1}}$, or 0 if the denominator is 0



MC Off-policy Algorithm

- Consider the estimates of their first-visit methods after observing a single return from state s
- Weighted importance sampling:
$$v_{\pi}(s) = \frac{\sum_{t \in A(s)} \rho_{t:T-1} G_t}{\sum_{t \in A(s)} \rho_{t:T-1}}$$
 - In the weighted-average estimate, the ratio $\rho_{t:T-1}$ for the single return cancels in the numerator and denominator, so that the estimate is equal to the observed return independent of the ratio (assuming the ratio is nonzero)
 - Given that this return was the only one observed, this is a reasonable estimate, but its expectation is $v_b(s)$ rather than $v_{\pi}(s)$, and thus it is biased
- Ordinary importance sampling:
$$v_{\pi}(s) = \frac{\sum_{t \in A(s)} \rho_{t:T-1} G_t}{|A(s)|}$$
 - The first-visit version of the ordinary importance-sampling estimator is always $v_{\pi}(s)$ in expectation (it is unbiased), but it can be extreme
 - Suppose the ratio were ten, indicating that the trajectory observed is ten times as likely under the target policy as under the behavior policy
 - In this case the ordinary importance-sampling estimate would be ten times the observed return. That is, it would be quite far from the observed return even though the episode's trajectory is considered very representative of the target policy



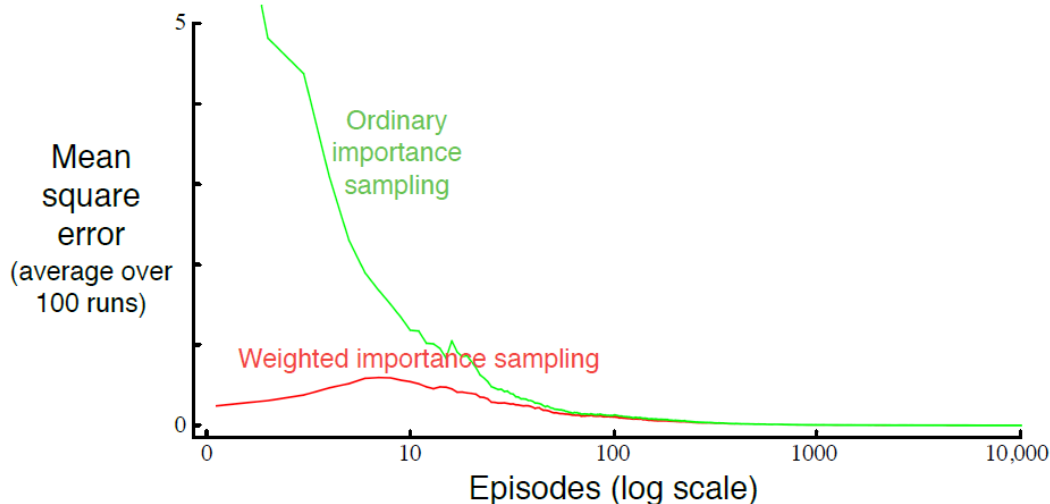
MC Off-policy Algorithm

- Ordinary vs. Weighted Importance Sampling (first-visit methods)
 - Ordinary importance sampling is unbiased whereas weighted importance sampling is biased (though the bias converges asymptotically to zero)
 - The variance of ordinary importance sampling is in general unbounded because the variance of the ratios can be unbounded, whereas in the weighted estimator the largest weight on any single return is one. Assuming bounded returns, the variance of the weighted importance-sampling estimator converges to zero
 - In practice, the weighted estimator usually has dramatically lower variance and is strongly preferred
- First-visit vs. Every-visit methods
 - In practice, every-visit methods are often preferred because they remove the need to keep track of which states have been visited




Example: Off-policy Estimation of a Blackjack State Value

- Goal: estimate the value of a single blackjack state from off-policy data, using both ordinary and weighted importance-sampling methods
- Behavior policy: starting in the state then choosing to hit or stick at random with equal probability
- Target policy: stick only on a sum of 20 or 21
- Both off-policy methods closely approximated this value after 1000 off-policy episodes using the random policy; but the weighted importance-sampling method has much lower error at the beginning





Outline

- Monte Carlo (MC) Prediction
- MC Estimation of Action Values
- MC Control
- MC Control without Exploring Starts
- Off-policy Prediction via Importance Sampling
-  **Incremental Implementation**
- Off-policy MC Control
- Conclusion



Incremental Implementation

- Monte Carlo prediction methods can be implemented incrementally, on an episode-by-episode basis
- Idea: average returns
- On-policy MC method: easy
- Off-policy MC method: need to separately consider those that use ordinary importance sampling and those that use weighted importance sampling



Incremental Implementation

- Ordinary importance sampling
 - The returns are scaled by the importance sampling ratio

$$\rho_{t:T-1} = \frac{\prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k, A_k)}{\prod_{k=t}^{T-1} b(A_k|S_k)p(S_{k+1}|S_k, A_k)} = \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)}$$

- Then, returns are incrementally updated to compute the average

- $$v(s) = \frac{\sum_{t \in A(s)} \rho_{t:T-1} G_t}{|A(s)|}$$



Incremental Implementation

■ Weighted importance sampling

- Suppose we have a sequence of returns G_1, G_2, \dots, G_{n-1} , all starting in the same state and each with a corresponding random weight W_i (e.g., $W_i = \rho_{t_i:T(t_i)-1}$)

- We wish to form the estimate
$$V_n = \frac{\sum_{k=1}^{n-1} W_k G_k}{\sum_{k=1}^{n-1} W_k}, \quad n \geq 2$$

and keep it up-to-date as we obtain a single additional return G_n

- In addition to keeping track of V_n , we must maintain for each state the cumulative sum C_n of the weights given to the first n returns
- Then, the update rules are

$$V_{n+1} = V_n + \frac{W_n}{C_n} [G_n - V_n], \quad n \geq 1$$

$$C_{n+1} = C_n + W_{n+1}$$

where $C_0 = 0$ and V_1 is arbitrary



Incremental Implementation

Sutton and Barto,
Reinforcement
Learning, 2018

- Off-policy MC prediction for estimating $Q \approx q_\pi$

Off-policy MC prediction (policy evaluation) for estimating $Q \approx q_\pi$

Input: an arbitrary target policy π

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:

$Q(s, a) \in \mathbb{R}$ (arbitrarily)

$C(s, a) \leftarrow 0$

$$V_{n+1} = V_n + \frac{W_n}{C_n} [G_n - V_n], \quad n \geq 1$$

$$C_{n+1} = C_n + W_{n+1}$$

Loop forever (for each episode):

$b \leftarrow$ any policy with coverage of π

Generate an episode following b : $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

$W \leftarrow 1$

Loop for each step of episode, $t = T-1, T-2, \dots, 0$, while $W \neq 0$:

$G \leftarrow \gamma G + R_{t+1}$


$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$

$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$

$W \leftarrow W \frac{\pi(A_t|S_t)}{b(A_t|S_t)}$



Outline

- Monte Carlo (MC) Prediction
- MC Estimation of Action Values
- MC Control
- MC Control without Exploring Starts
- Off-policy Prediction via Importance Sampling
- Incremental Implementation
-  **Off-policy MC Control**
- Conclusion



Off-Policy MC Control

- On-policy methods: estimate the value of a policy while using it for control
- Off-policy methods: behavior policy and target policy are separated
 - *Behavior policy* generates behavior
 - *Target policy* is evaluated and improved
 - Advantage: the target policy may be deterministic (e.g., greedy), while the behavior policy can continue to sample all possible actions



Off-Policy MC Control

- Off-policy MC control methods
 - Follow the behavior policy while learning about and improving the target policy
 - The behavior policy should have a nonzero probability of selecting all actions that might be selected by the target policy (coverage)



Off-Policy MC Control

Sutton and Barto,
Reinforcement
Learning, 2018

Off-policy MC control, for estimating $\pi \approx \pi_*$

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:

$Q(s, a) \in \mathbb{R}$ (arbitrarily)

$C(s, a) \leftarrow 0$

$\pi(s) \leftarrow \operatorname{argmax}_a Q(s, a)$ (with ties broken consistently)

Loop forever (for each episode):

$b \leftarrow$ any soft policy

Generate an episode using b : $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

$W \leftarrow 1$

Loop for each step of episode, $t = T-1, T-2, \dots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$

$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$

$\pi(S_t) \leftarrow \operatorname{argmax}_a Q(S_t, a)$ (with ties broken consistently)

If $A_t \neq \pi(S_t)$ then exit inner Loop (proceed to next episode)

$W \leftarrow W \frac{1}{b(A_t|S_t)}$




Off-Policy MC Control

- Off-policy MC control methods
 - Uses GPI and weighted importance sampling, for estimating π_* and q_*
 - The target policy $\pi \approx \pi_*$ is the greedy policy with respect to Q , which is an estimate of q_π
 - The behavior policy b can be anything, but an infinite number of returns must be obtained for each pair of state and action, to assure convergence; this can be assured by choosing b to be ϵ -soft
 - The policy π converges to optimal at all encountered states even though actions are selected according to a different soft policy b , which may change between or even within episodes



Outline

- Agent-Environment Interface
- Goals and Rewards
- Returns and Episodes
- Episodic and Continuing Tasks
- Policies and Value Functions
- Optimal Policies and Value Functions
- Optimality and Approximation
-  **Conclusion**



Conclusion

- Monte Carlo methods learn value functions and optimal policies from experience in the form of sample episodes
- Advantages of MC methods over DP methods
 - They can be used to learn optimal behavior directly from interaction with the environment, with no model of the environment's dynamics
 - They can be used with simulation or sample models; for many applications it is easy to simulate sample episodes even though it is difficult to construct the kind of explicit model of transition probabilities required by DP methods
 - It is easy and efficient to focus MC methods on a small subset of the states; A region of special interest can be accurately evaluated without going to the expense of accurately evaluating the rest of the state set
 - They may be less harmed by violations of the Markov property, since they do not update their value estimates on the basis of the value estimates of successor states; i.e., they do not bootstrap.
 - There are methods that learn from experience, like Monte Carlo methods, but also bootstrap, like DP methods



Conclusion

- MC control methods uses the idea of generalized policy iteration (GPI)
- GPI involves interacting processes of policy evaluation and policy improvement
- MC methods provide an alternative policy evaluation process; rather than using a model to compute the value of each state, they simply average many returns that start in the state
- Because a state's value is the expected return, this average can become a good approximation to the value
- In control methods we are particularly interested in approximating action-value functions which can be used to improve the policy without requiring a model of the environment's transition dynamics
- MC methods intermix policy evaluation and policy improvement steps on an episode-by-episode basis, and can be incrementally implemented on an episode-by-episode basis



Conclusion

- Maintaining sufficient exploration
 - An important issue in Monte Carlo control methods
 - It is not enough just to select the actions currently estimated to be best, because then no returns will be obtained for alternative actions
 - One approach is to assume that episodes begin with state–action pairs randomly selected to cover all possibilities (exploring starts); this may be possible in applications with simulated episodes, but are unlikely in learning from real experience
 - In on-policy methods, the agent commits to always exploring and tries to find the best policy that still explores
 - In off-policy methods, the agent also explores, but learns a deterministic optimal policy that may be unrelated to the policy followed



Conclusion

- Off-policy prediction
 - Learn the value function of a target policy from data generated by a different behavior policy
 - Based on some form of importance sampling: weight returns by the ratio of the probabilities of taking the observed actions under the two policies, thereby transforming their expectations from the behavior policy to the target policy
 - Ordinary importance sampling uses a simple average of the weighted returns, while weighted importance sampling uses a weighted average
 - Ordinary importance sampling produces unbiased estimates, but has larger, possibly infinite, variance, whereas weighted importance sampling always has finite variance and is preferred in practice



Exercise

- (Question 1)
- A) Consider an MDP with a single nonterminal state and a single action that transitions back to the nonterminal state with probability p and transitions to the terminal state with probability $1-p$. Let the reward be $+1$ on all transitions and let $\gamma = 1$. Suppose you observe one episode that lasts 10 steps, with a return of 10. What are the first-visit and every-visit estimators of the value of the nonterminal state?
- B) What would be the answer of A if the discounting factor is changed to 0.9?



Exercise

- (Answer A)

Step	0	1	2	3	4	5	6	7	8	9	10
State	S	S	S	S	S	S	S	S	S	S	T
Reward	1	1	1	1	1	1	1	1	1	1	
Action	p	p	p	p	p	p	p	p	p	1-p	

- First visit = $10 / 1 = 10$
- Every visit = $(1+2+ \dots + 10) / 10 = 55/10 = 5.5$



Exercise

- (Answer B)

Step	0	1	2	3	4	5	6	7	8	9	10
State	S	S	S	S	S	S	S	S	S	S	T
Reward	1	1	1	1	1	1	1	1	1	1	
Action	p	p	p	p	p	p	p	p	p	1-p	

- First visit = $1 + 0.9 + 0.9^2 + \dots + 0.9^9 = \frac{1(1-0.9^{10})}{1-0.9}$
- Every visit = $\sum_{t=1}^{10} (1 + \dots + 0.9^{t-1}) / 10 = \sum_{t=1}^{10} \frac{(1-0.9^t)}{1-0.9} / 10$



Questions?