NAME1: Nikita Tokovenko
STUDENTID1:12185892
MAIL1:
nikita.tokovenko@student.uva.nl

Homework Assignment 1
Reinforcement Learning, 19/20

2019-09-20

1 Dynamic Programming

1.1

Stochastic policy case:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)(r + \gamma v_\pi(s')) = \sum_a \pi(a|s) q_\pi(s,a)$$

Deterministic policy case:

$$v_\pi(s) = q_\pi(s, \pi(s))$$

1.2

$$q_\pi(s,\pi(s)) = \sum_{s',r} p(s',r|s,\pi(s))(r + \gamma q_\pi(s', \pi(s')))$$

1.3

$$\pi(s) = argmax_a \sum_{s',r} p(s',r|s,a)(r + \gamma q_\pi(s', \pi(s')))$$

1.4

$$q_{k+1}(s,a) = \sum_{s',r} p(s',r|s,a)(r + \gamma \max_{a'} q_k(s',a')))$$

2 Monte Carlo

2.1

(a) Using first-visit MC:

$$V(s_0) = \frac{0.9^2 * 5 + 0.9^4 * 5 + 0.9^3 * 5}{3} \approx 3.6585$$

(b) Using every-visit MC:

$$V(s_0) = \frac{(5 + 0.9 * 5 + 0.9^2 * 5) + (5 + 0.9 * 5 + 0.9^2 * 5 + 0.9^3 * 5 + 0.9^4 * 5) + (5 + 0.9 * 5 + 0.9^2 * 5 + 0.9^3 * 5)}{12} \approx 4.2683$$

2.2

We 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. The problem is that this ratio can be very high for different policies, so giving estimations with high variance.

2.3

The weighted importance-sampling estimator is biased with respect to the behavioral policy value, not the target one. It means the episode trajectory represents behavioral policy rather than target.

3 Maximization Bias

3.1

	Q-learning	SARSA
$Q(A, left)$	2	{1 (random), 2 (greedy), [1, 2] (ϵ-greedy)}
$Q(A, right)$	1.5	1.5
$Q(B, 1)$	1	1
$Q(B, 2)$	1	1
$Q(B, 3)$	2	2
$Q(B, 4)$	0	0

3.2

Maximization bias can be observed when we perform a maximization operation for Q-value function estimation. We evaluate the expected future reward of taking action ”left” while being in state A. When being in B and do action ”3” we observed the reward of 2. So we obtain, because MDP is undiscounted, $Q(A, left) = 2 > 1.5 = Q(A, right)$.
Q-learning suffers from maximization bias, because always choosing greedy actions. For SARSA it depends on the value of ϵ - for near-greedy policies problem stays the same.

3.3

Maximization bias can be eliminated using Double Q-learning. It uses different Q-value functions (Q_1, Q_2) - each for it's own subset of observations. Both of them are used to update the estimates of one Q-value function. Suppose we split our data such, that: $Q_1(B, 1) = 0, Q_1(B, 2) = 2, Q_1(B, 3) = 2, Q_1(B, 4) = 0$ and $Q_2(B, 1) = 2, Q_2(B, 2) = 0, Q_2(B, 3) = 2, Q_2(B, 4) = 0$
Double Q-learning updates Q_1 as the Q_2 estimates for choosing greedy action via Q_1: $Q_2(S', argmax_a Q_1(S', a))$, so that the reward will be either 2 or 0 from doing actions 3 and 2 respectively that are being optimal for Q1. The expected future reward can be averaged, in convergence, to 1.
Q-values for state A: $Q_1(A, left) = 1, Q_2(A, left) = 1, Q_1(A, right) = 1.5, Q_2(A, right) = 1.5$.
Now by taking state-action value function as an average of functions Q_1, Q_2 we eliminate maximization bias, so now optimal action is ”right” when being in A.

3.4

	Values
$Q(A, left)$	1
$Q(A, right)$	1.5
$Q(B, 1)$	1
$Q(B, 2)$	1
$Q(B, 3)$	1
$Q(B, 4)$	1