NAME1: Philipp Lintl
STUDENTID1:12152498
MAIL1: philipp.lintl@student.uva.nl

NAME2: Bogdan Floris
STUDENTID2: 12140910
MAIL2 bogdan.floris@student.uva.nl

Homework Assignment 1
Reinforcement Learning, 19/20

2019-09-20

2.2 Dynamic Programming

1.

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

$$q_\pi(s,a) = \sum_{s'} p\left(s'|s,a\right)\left[r\left(s,a,s'\right) + \gamma \sum_{a'} \pi\left(a'|s'\right) q_\pi\left(s',a'\right)\right]$$

stochastic policy $\quad \Rightarrow v_\pi(s) = \sum_a \pi(a|s) \cdot q_\pi(s,a)$

deterministic policy $\quad \Rightarrow v_\pi(s) = q_\pi(s, \pi(s))$

2. Policy evaluation update for sweep k+1:

$$\begin{aligned} q_{k+1}(s,a) &= \sum_{s'} p\left(s'|s,a\right)\left[r(s,a,s') + \gamma v_k\left(s'\right)\right] \\ &= \sum_{s'} p\left(s'|s,a\right)\left[r(s,a,s') + \gamma \sum_a \pi_k(a|s) q_k(s',a)\right] \\ \text{(deterministic policy)} \qquad &= \sum_{s'} p\left(s'|s,a\right)\left[r(s,a,s') + \gamma q_k\left(s', \pi\left(s'\right)\right)\right] \end{aligned}$$

3. Policy improvement:

$$\begin{aligned} \pi'(s) &= \arg\max_a q_\pi(s,a) \\ &= \arg\max_a \sum_{s'} p\left(s'|s,a\right)\left[r(s,a,s') + \gamma v_k\left(s'\right)\right] \\ &= \arg\max_a \sum_{s'} p\left(s'|s,a\right)\left[r(s,a,s') + \gamma q_k\left(s', \pi\left(s'\right)\right)\right] \end{aligned}$$

4.

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

3.1 Monte Carlo

1.
   - First Visit MC (includes 0 rewards for completeness):

$$\begin{aligned} v\left(s_0\right) = \frac{1}{3}(&0 \cdot 1 + 0 \cdot 0.9 + 5 \cdot 0.9^2 + 0 \cdot 1 + 0 \cdot 0.9 + 0 \cdot 0.9^2 + 0 \cdot 0.9^3 + 5 * 0.9^4 \\ &+ 0 \cdot 1 + 0 \cdot 0.9 + 0 \cdot 0.9^2 + 5 * 0.9^3) \approx 3.659 \end{aligned}$$

   - Every Visit MC (does not include 0 rewards for the sake of clarity):

$$v\left(s_0\right) = 5 \cdot \frac{1}{3}\left(\frac{0.9^2 + 0.9 + 1}{3} + \frac{0.9^4 + 0.9^3 + 0.9^2 + 0.9 + 1}{5} + \frac{0.9^3 + 0.9^2 + 0.9 + 1}{4}\right) \approx 4.304$$

2. Ordinary importance sampling in off-policy Monte Carlo is especially disadvantageous, when the ratio between the target and the behaviour policy is different. For both cases, when the target being much more or much less likely than the behavior policy, the estimates are skewed. Suppose, e.g. the observed trajectory is ten times as likely under the target policy as under the behavior policy. Then, as the ordinary importance sample divides by the number of episodes, the estimate would be ten times the observed return. Thus, the obtained estimates have a high variance.

3. The weighted importance sampling strategy on the other hand is preferably low in variance, as the estimates are corrected for the ratios. However supposing only considering one trajectory, the ratio cancels out and therefore, the estimate equals the behaviour return G_t and therefore the estimate is biased towards the expected behaviour v_b instead of the target v_pi.

4.4 Maximization Bias

1.
   - SARSA:
     - $Q(A, \text{left}) = 1$ for a random policy, as half of the times reward is 2 and the other times 0, therefore the expectation is 1. If the policy is greedy, we walk branch 3 and have $Q(A, \text{left}) = 2$. For an $\epsilon-$greedy policy, the value would be close to 2, but as the non optimal actions are chosen with probability ϵ, the expected value is smaller than 2.
     - $Q(A, \text{right}) = 1.5$
     - $Q(B, 1) = 1$
     - $Q(B, 2) = 1$
     - $Q(B, 3) = 2$
     - $Q(B, 4) = 0$
   - Q-Learning
     - $Q(A, \text{left}) = 2$
     - $Q(A, \text{right}) = 1.5$
     - $Q(B, 1) = 1$
     - $Q(B, 2) = 1$
     - $Q(B, 3) = 2$
     - $Q(B, 4) = 0$

2. In this scenario, maximization bias is observed when looking at the Q-values for Q-learning. As Q-Learning entails a maximization operation, the action left in state A is optimistically evaluated. It is optimistic, as the true state action value corresponds to the average of all available state-action values available after choosing left: $Q(A, left) = 1$ In the Q-Learning case, we end up with $Q(A, left) = 2$. Considering this state action value, the optimal policy would be to rather choose left in state A, than right, which actually has a higher Q-value. As mentioned, this happens, as the Q-value at B is chosen to be the maximum (2) of all available state-action values. Considering Sarsa does not draw on an $\epsilon-$greedy policy to generate episodes, it does not suffer from maximization Bias. In fact, it averages over possible state-action values for a non greedy policy. For an ϵ-greedy policy, bias maximization appears, as well, albeit resulting in a smaller Q estimate.

3. În general, double Q-Learning draws on two state-action functions Q_1, Q_2 to overcome overoptimistic values stemming from the fact that one Q function evaluates and selects actions according to a greedy policy. Thus, when updating one of the two, the respective other is used for evaluation, as seen here:

$$Q_2(s, a) \leftarrow Q_2(s, a) + \alpha \left(R + \gamma Q_1 \left(s', \arg\max_{a'} Q_2 \left(s', a' \right) \right) - Q_2(s, a) \right)$$

   In action, we would randomly update the one according to the other or vice versa. This way, the maximum state-action value is not simply chosen but actually checked with regards to another policy. This procedures eliminates the bias, that is otherwise upheld by maximizing over the to be updated state-action function.

4. True state action values: When we keep sampling, we have to consider, that the four actions possibly taken in state B generate different episodes. In fact, as they are drawn from a Uniform distribution [0,2], the expected value of each of those is 1, therefore all of the 4 Q values arising from state B are 1.

   - $Q(A, \text{left}) = 1$
   - $Q(A, \text{right}) = 1.5$
   - $Q(B, 1) = 1$
   - $Q(B, 2) = 1$
   - $Q(B, 3) = 1$
   - $Q(B, 4) = 1$