# Homework 2

ELEN E6885: Introduction to Reinforcement Learning

Due: October 15, 2019

**Problem 1** (**True or False, 10 Points**)

Please indicate *True* or *False* for the following statements. No explanation is needed.

1. [2 pts] In non-stationary reinforcement learning problems, we usually use a constant step-size in the incremental update of a reinforcement learning algorithm, e.g., $Q_{k+1} = Q_k + \alpha[r_{k+1} - Q_k]$. Solution: True

2. [2 pts] To find the optimal policy for a given MDP, one can choose to compute and store state-value function $v(s)$ or action-value function $q(s, a)$. One argument in favor of $q(s, a)$ is that it needs to store fewer values. Solution: False

3. [2 pts] In the Monte Carlo estimation of action values, if a deterministic policy $\pi$ is used, in most cases one agent (with the same starting state) could not learn the values of all actions. Solution: True

4. [2 pts] For a stationary problem, Sarsa converges with probability 1 to an optimal policy and action-value function as long as all state-action pairs are visited an infinite number of times and the policy converges in the limit to the greedy policy. Solution: True

5. [2 pts] Both SARSA and Q-learning are on-policy learning algorithms. Solution: False

**Problem 2** (**Policy Iteration vs. Value Iteration, 10 Points**)

Regarding the policy iteration and value iteration algorithms, answer the following questions.

1. [5 pts] Summarize the differences between policy iteration and value iteration algorithms.

2. [5 pts] Prove that given the same starting value function, one iteration of the policy iteration algorithm, including greedy policy improvement followed by **one** step of policy evaluation, will generate the same value function as one iteration of the value iteration algorithm.

*Solution.*

1. Policy iteration method has two major iterations. It will first iteratively backup the values of all states based on the current policy until the state values converge. Then, it runs 0policy improvement to generate a better policy. These two steps are repeated until both the state values and policy converge. The backup in the policy evaluation step uses Bellman expectation equation.

   Value iteration method merges these two step together. It is written as a particularly simple backup operation on state values that combines the policy improvement and truncated policy evaluation (one sweep over the state space) steps. The backup in value iteration method uses Bellman optimality equation.

2. Let $v(s)$ denote the starting value function of state $s$. The greedy policy $\pi$ with respect to $v(s)$ can be represented as

$$\pi(a|s) = \begin{cases} 1, & \text{if } a = a^*(s), \\ 0, & \text{otherwise,} \end{cases}$$

   where $a^*(s) = \arg\max_a \left(R_s^a + \gamma \sum_{s'} P_{ss'}^a v(s')\right)$. Then, one step of policy evaluation of policy $\pi$ can be written as

$$\begin{aligned} v'(s) &= \sum_a \pi(a|s) \left( R_s^a + \gamma \sum_{s'} P_{ss'}^a v(s') \right) \\ &= R_s^{a^*(s)} + \gamma \sum_{s'} P_{ss'}^{a^*(s)} v(s') \\ &= \max_a \left( R_s^a + \gamma \sum_{s'} P_{ss'}^a v(s') \right). \end{aligned}$$

   It is exactly the same as one backup of value iteration algorithm starting from value function $v(s)$.

**Problem 3** (**Model-Based MDP, 25 Points**)

Consider the model-based MDP problem as shown in Fig. 1. The number above the line refers to the probability of taking that action and $r$ is the reward for the corresponding action.

1. [10 pts] Use matrix-form Bellman equation to find the values of all states ($\gamma = 0.5$).

2. [10 pts] Suppose that the initial values of all states are 0. The agent follows the policy given in the graph and assumes the discount factor $\gamma = 1$. Find the values of all states in the first two steps $k = 1$ and $k = 2$ in the algorithm of iterative policy evaluation.

3. [5 pts] Following the results of the previous question, update the values of all states for $k = 3$ using the value iteration method.

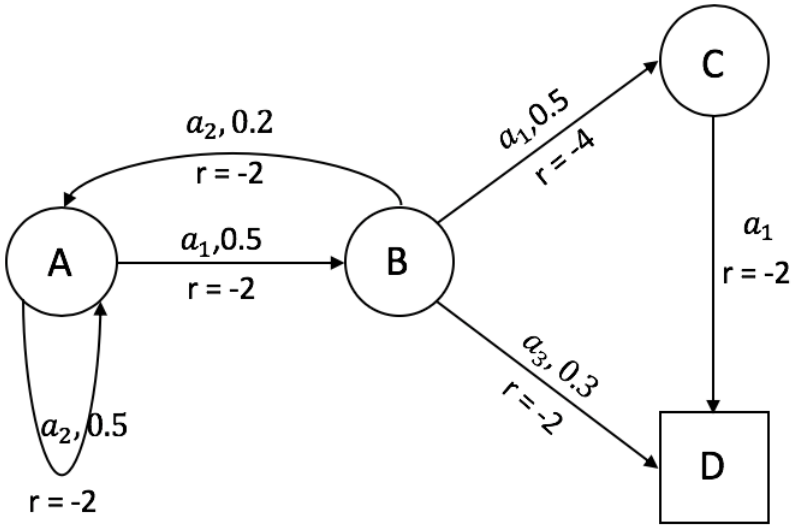


Figure 1: Model-based MDP.

*Solution.*

1. Since $D$ is the terminal state in the MDP, $v(D) = 0$. According to the matrix form of Bellman equation, we have the following:

$$\begin{bmatrix} v(A) \\ v(B) \\ v(C) \\ v(D) \end{bmatrix} = \begin{bmatrix} R_A \\ R_B \\ R_C \\ R_D \end{bmatrix} + \gamma \begin{bmatrix} 0.5 & 0.5 & 0 & 0 \\ 0.2 & 0 & 0.5 & 0.3 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} v(A) \\ v(B) \\ v(C) \\ v(D) \end{bmatrix},$$

   where

$$\begin{aligned} R_A &= 0.5 \times (-2) + 0.5 \times (-2) = -2, \\ R_B &= 0.2 \times (-2) + 0.5 \times (-4) + 0.3 \times (-2) = -3, \\ R_C &= 1 \times (-2) = -2, \\ R_D &= 0. \end{aligned}$$

   Solve the linear equations, we have

$$\begin{bmatrix} v(A) \\ v(B) \\ v(C) \\ v(D) \end{bmatrix} = \begin{bmatrix} -3.966 \\ -3.897 \\ -2 \\ 0 \end{bmatrix}.$$

2. The initial state value is $v(A) = v(B) = v(C) = 0$. When $k = 1$, we have

$$\begin{aligned} v(A) &= 0.5 \times (-2 + 0) + 0.5 \times (-2 + 0) = -2, \\ v(B) &= 0.2 \times (-2 + 0) + 0.5 \times (-4 + 0) + 0.3 \times (-2 + 0) = -3, \\ v(C) &= 1 \times (-2 + 0) = -2. \end{aligned}$$

When $k = 2$, we have

$$v(A) = 0.5 \times (-2 + -2) + 0.5 \times (-2 + -3) = -4.5,$$
$$v(B) = 0.2 \times (-2 + -2) + 0.5 \times (-4 + -2) + 0.3 \times (-2 + 0) = -4.4,$$
$$v(C) = 1 \times (-2 + 0) = -2.$$

3. According to Bellman optimality backup in value iteration method:

   For state $A$: two possible actions

   Take $a_1$, $q(A, a_1) = -2 + 1 \times (-4.4) = -6.4$

   Take $a_2$, $q(A, a_2) = -2 + 1 \times (-4.5) = -6.5$

   $v(A) = \max_a q(A, a) = -6.4$

   For state $B$: three possible actions

   Take $a_1$, $q(B, a_1) = -4 + 1 \times (-2) = -6$

   Take $a_2$, $q(B, a_2) = -2 + 1 \times (-4.5) = -6.5$

   Take $a_3$, $q(B, a_3) = -2 + 1 \times 0 = -2$

   $v(B) = \max_a q(B, a) = -2$

   For state $C$: one possible action

   Take $a_1$, $q(C, a_1) = -2 + 1 \times 0 = -2$

   $v(C) = \max_a q(C, a) = -2$

**Problem 4 (Convergence of Value Iteration, 15 Points)**

For the value iteration algorithm, the best policy at step $k$ can be extracted as a result of the "max" operation (i.e., greedy action selection) based on the value function at step $k$.

1. [5 pts] Prove that if the value function at step $k$ is the same as that of step $k + 1$, the best policy will never change with further iterations of the value function.

2. [10 pts] Provide an example to show that if the best policy at step $k$ is the same as that of step $k + 1$, the best policy can still change with further iterations of the value function.

*Proof.*

1. Define the Bellman optimality backup operator $T^*$ as

$$T^*(\boldsymbol{v}) = \max_{a \in \mathcal{A}} \boldsymbol{R}^a + \gamma \boldsymbol{P}^a \boldsymbol{v}.$$

Thus, we have $\boldsymbol{v}_{n+1} = T^*(\boldsymbol{v}_n)$ for any $n \in \mathbb{N}$. In the following, we show that $\boldsymbol{v}_{n+1} = \boldsymbol{v}_n$ for all $n \geq k$. It is trivial for the base case of $n = k$ by the assumption. Assume $\boldsymbol{v}_{n+1} = \boldsymbol{v}_n$ is true, then we have

$$\boldsymbol{v}_{n+2} = T^*(\boldsymbol{v}_{n+1}) = T^*(\boldsymbol{v}_n) = \boldsymbol{v}_{n+1},$$

which gives the desired result for induction. Now since $\boldsymbol{v}_{n+1} = \boldsymbol{v}_n$ for all $n \geq k$, the value function as well as the best policy will not change after step $k$.

2. Consider an undiscounted MDP with 5 states shown in Fig. 2 below, where state $E$ is a terminal state. The value function of all non-terminal states using value iteration method is summarized in the following table.

| Iteration | $v(A)$ | v(B) | v(C) | v(D) |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | -1 | -2 | -2 | 10 |
| 2 | -2 | -4 | 8 | 10 |
| 3 | -3 | 6 | 8 | 10 |
| 4 | 4 | 6 | 8 | 10 |

We can see that after the first and second iteration, the optimal action from state $A$ is the one going back to state $A$. But after 3 iterations, the optimal action from state $A$ changes to the one going to state $B$.

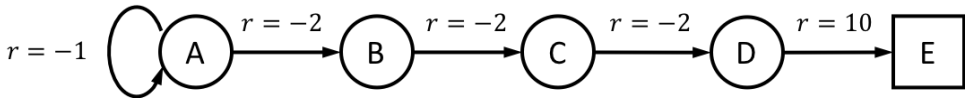


Figure 2: Five-state MDP

**Problem 5** (**Last-Visit Monte Carlo, 15 Points**)

Similarly as in first-visit and every-visit Monte Carlo methods, we can estimate $v_\pi(s)$ by averaging over the discounted return starting from the *last* occurrence of $s$ in each episode following policy $\pi$. We call it the *last-visit* Monte Carlo method. Prove by providing an example that the last-visit Monte Carlo method for estimating $v_\pi$ is not guaranteed to converge to $v_\pi$ for a finite MDP with bounded rewards and $\gamma \in [0, 1)$.

*Proof.* Consider a simple undiscounted MDP with 2 states shown in Fig. 3 below, where state $B$ is a terminal state. The return from last visit of state $A$ in any episode will be 1. Thus, the estimated value function of state $A$ is always 1. However, according to Bellman expectation equation, we have

$$v(A) = 0.5 \times (-1 + v(A)) + 0.5 \times (1 + 0).$$

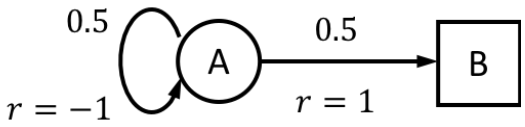


Figure 3: Simple two-state MDP

It follows that $v(A) = 0$. Therefore, it shows that the estimation of $v(A)$ using last-visit Monte Carlo does not converge to its true value.

**Problem 6** (**Small Grid World, 25 Points**)

Consider the small grid world MDP as shown in Fig. 4. The states are grid squares, identified by their row and column numbers (row first). The agent always starts in state $(1, 1)$. There are two terminal states, $(2, 3)$ with reward $+5$ and $(1, 3)$ with reward $-5$. Rewards are 0 in non-terminal states. (The reward for a state is received as the agent moves into the state.) The agent can take one action each time from up, down, left and right. However, the intended movement only happens with probability 0.8. And with probability 0.1 each, the agent ends up in one of the states perpendicular to the intended direction. For example, if the agent at $(1, 2)$ chooses to move up to $(2, 2)$, it can reach $(2, 2)$ only with probability 0.8. The agent may hit $(1, 1)$ or the terminal state (with reward $-5$) with equal probability 0.1. If a collision with a wall happens, the agent stays in the same state.

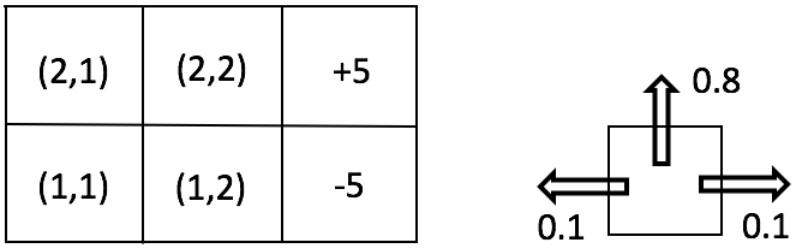


Figure 4: Grid world MDP and one transition example.

1. [5 pts] What is the optimal policy for this small grid world problem? Assume the discount factor $\gamma = 1$.

2. [8 pts] Suppose the agent knows the transition probabilities. Write down the first two rounds of (synchronous) *value iteration* updates for state $(1, 2)$ and $(2, 1)$, with a discount factor of 0.9. (Assume $V_0$ is 0 everywhere. Also, we assume the values of terminal states $V(1, 3) = V(2, 3) = 0$ for all iterations).

3. [4 pts] The agent starts with the policy that always chooses to go right, and executes the following three trials: 1) $(1, 1) - (1, 2) - (1, 3)$, 2) $(1, 1) - (1, 2) - (2, 2) - (2, 3)$, and

3) $(1,1) - (2,1) - (2,2) - (2,3)$. What are the Monte Carlo estimates for states $(1,1)$ and $(2,2)$, given these traces? Assume the discount factor $\gamma = 1$.

4. [8 pts] Using a learning rate of $\alpha = 0.1$, a discount factor of $\gamma = 0.9$, and assuming initial $V$ values of 0, what updates does the TD(0)-learning agent make after trials 1) and 2) above?

*Solution.*

1. One optimal policy is $\pi(1,1) =$ up, $\pi(1,2) =$ left, $\pi(2,1) =$ right, $\pi(2,2) =$ right. In fact, it can be verified that with $\gamma = 1$, the optimal value function of all non-terminal states are $+5$. So the optimal policy only requires that $\pi(1,2) =$ left.

2. After the first round, we have $V_1(1,2) = V_1(2,1) = 0$. Also, $V_1(1,1) = 0$ and $V_1(2,2) = 0.8 \times 5 = 4$. Then after the second round,

$$V_2(1,2) = 0.8 \times 0.9 \times 4 + 0.1 \times (-5) = 2.38,$$
$$V_2(2,1) = 0.8 \times 0.9 \times 4 = 2.88.$$

3. $V(1,1) = (-5+5+5)/3 = 5/3$ and $V(2,2) = (5+5)/2 = 5$.

4. For trial 1,

$$V(1,2) = 0 + 0.1 \times (-5 + 0.9 \times 0 - 0) = -0.5.$$

For trial 2,

$$V(1,1) = 0 + 0.1 \times (0 + 0.9 \times (-0.5) - 0) = -0.045,$$
$$V(1,2) = -0.5 + 0.1 \times (0 + 0.9 \times 0 + 0.5) = -0.45,$$
$$V(2,2) = 0 + 0.1 \times (5 + 0.9 \times 0 - 0) = 0.5.$$