Homework 4 - Group 03

Planning, Learning and Intelligent Decision Making

Duarte Almeida
95565

Martim Santos
95638

Exercise 1.

(a) Given a sample transition (x_t, a_t, c_t, x_{t+1}), we get the Q-learning update

$$\hat{Q}_{t+1}(x_t, a_t) = (1-\alpha)\hat{Q}_t(x_t, a_t) + \alpha(c_t + \gamma \min_{a' \in \mathcal{A}} \hat{Q}_t(x_{t+1}, a'))$$

The Q-learning update can also be written as

$$\hat{Q}_{t+1}(x_t, a_t) = \hat{Q}_t(x_t, a_t) + \alpha \left[c_t + \gamma \min_{a' \in \mathcal{A}} \hat{Q}_t(x_{t+1}, a') - \hat{Q}_t(x_t, a_t) \right]$$

where the quantity

$$\delta_t = c_t + \gamma \min_{a' \in \mathcal{A}} \hat{Q}_t(x_{t+1}, a') - \hat{Q}_t(x_t, a_t)$$

is a temporal difference at time step t.

At each step, Q-learning updates the Q-value for a single state-action pair, (x_t, a_t). For a sample transition $(x_t, a_t, c_t, x_{t+1}) = ((E, 1, 0, 1),\ R,\ 0.2,\ (F, 1, 0, 1))$ and, consequently, the state-pair action $(x_t, a_t) = ((E, 1, 0, 1), R)$, an update with step-size $\alpha = 0.1$, resulting at time step t with $\gamma = 0.9$ is given by

$$\begin{aligned}
\hat{Q}_{t+1}((E,1,0,1), R) &= \hat{Q}_t((E,1,0,1), R) + \alpha \left[c_t + \gamma \min_{a' \in \mathcal{A}} \hat{Q}_t((F,1,0,1), a') - \hat{Q}_t((E,1,0,1), R) \right] \\
&= 2.0 + 0.1 \cdot (0.2 + 0.9 \cdot 2.0 - 2.0) \\
&= 2.0
\end{aligned}$$

The resulting Q-values estimated by the agent for state $(E, 1, 0, 1)$ after this update are:

$$\mathbf{Q}^{(t+1)}_{(E,1,0,1),:} = \begin{bmatrix} 2.8 & 2.8 & 2.8 & 2.8 & 2.54 & \mathbf{2.0} \end{bmatrix}$$

and therefore

$$\mathbf{Q}^{(t+1)}_{x,a} = \begin{cases} \mathbf{2.0} & \text{if } x = (E, 1, 0, 1) \wedge a = R \\ \mathbf{Q}^{(t)}_{x,a} & \text{otherwise} \end{cases}$$

(b) Similarly, given a sample transition $(x_t, a_t, c_t, x_{t+1}, a_{t+1})$, we get the SARSA update

$$\hat{Q}_{t+1}(x_t, a_t) = \hat{Q}_t(x_t, a_t) + \alpha \left[c_t + \gamma \hat{Q}_t(x_{t+1}, a_{t+1}) - \hat{Q}_t(x_t, a_t) \right]$$

where the quantity

$$\delta_t = c_t + \gamma \hat{Q}_t(x_{t+1}, a_{t+1}) - \hat{Q}_t(x_t, a_t)$$

is also a temporal difference at time step t.

For a sample $(x_t, a_t, c_t, x_{t+1}, a_{t+1}) = ((E, 1, 0, 1),\ R,\ 0.2,\ (F, 1, 0, 1),\ R)$ and consequently, the state-pair action $(x_t, a_t) = ((E, 1, 0, 1), R)$, an update with step-size $\alpha = 0.1$, resulting at time step t with $\gamma = 0.9$ is given by

$$\begin{aligned}
\hat{Q}_{t+1}((E,1,0,1),R) &= \hat{Q}_t((E,1,0,1),R) + \alpha \left[c_t + \gamma \hat{Q}_t((F,1,0,1),R) - \hat{Q}_t((E,1,0,1),R)\right] \\
&= 2.0 + 0.1 \cdot (0.2 + 0.9 \cdot 2.8 - 2.0) \\
&= 2.072
\end{aligned}$$

The resulting Q-values estimated by the agent for state $(E, 1, 0, 1)$ after this update are:

$$\mathbf{Q}^{(t+1)}_{(E,1,0,1),:} = \begin{bmatrix} 2.8 & 2.8 & 2.8 & 2.8 & 2.54 & \mathbf{2.072} \end{bmatrix}$$

and therefore

$$\mathbf{Q}^{(t+1)}_{x,a} = \begin{cases} \mathbf{2.072} & \text{if } x = (E,1,0,1) \ \wedge \ a = R \\ \mathbf{Q}^{(t)}_{x,a} & \text{otherwise} \end{cases}$$

(c) Broadly speaking, **off-policy** learning is a technique present in reinforcement learning algorithms that learn the **optimal policy** from data generated by following a **different policy** in the interaction with the environment. Q-learning, which was used in Question 1a, is an example of such class of algortihms. To see why, note that, given a sample transition (x_t, a_t, c_t, x_{t+1}), Q-learning performs the update

$$\hat{Q}_{t+1}(x_t, a_t) = \hat{Q}_t(x_t, a_t) + \alpha \left[c_t + \gamma \min_{a' \in \mathcal{A}} \hat{Q}_t(x_{t+1}, a') - \hat{Q}_t(x_t, a_t)\right]$$

which in turn is a **stochastic approximation update with *bootstrapping*** used to find the value $\hat{Q}_{t+1}(x_t, a_t)$ that satisfies:

$$\begin{aligned}
&\hat{Q}_{t+1}(x_t, a_t) = \mathbb{E}_{\pi,\ y \sim P(\cdot \mid x_t,\ a_t)} \left[c_t + \gamma \min_{a' \in \mathcal{A}} Q^*(y, a') \mid \mathrm{x}_t = x_t,\ \mathrm{a}_t = a_t\right] \\
\Leftrightarrow\ &\mathbb{E}_{\pi,\ y \sim P(\cdot \mid x_t,\ a_t)} \left[c_t + \gamma \min_{a' \in \mathcal{A}} Q^*(y, a') - \hat{Q}_{t+1}(x_t, a_t) \mid \mathrm{x}_t = x_t,\ \mathrm{a}_t = a_t\right] = 0
\end{aligned}$$

Since the solution to the equation above is $Q^*(x_t, a_t)$ (as $\boldsymbol{Q^*}$ is the only fixed point of the underlying operator $\mathbf{H}$) and that $c_t + \gamma \min_{a' \in \mathcal{A}} \hat{Q}_t(x_{t+1}, a') - \hat{Q}_t(x_t, a_t)$ is considered to be a (noisy) sample of the term that appears inside the expectation in the last expression, we have that **Q-learning learns the optimal Q-function (and, consequently, the optimal policy) irrespective of the policy used to generate data**.

In turn, **on-policy** learning is a reinforcement learning approach that **learns the policy followed by the agent** in the course of its execution. This type of learning was employed in Question 1b when we used SARSA. Given a transition $(x_t, a_t, c_t, x_{t+1}, a_{t+1})$ attained by following the **current policy** π, SARSA performs the following **stochastic approximation update with *bootstraping***:

$$\hat{Q}_{t+1}(x_t, a_t) = \hat{Q}_t(x_t, a_t) + \alpha \left[c_t + \gamma \hat{Q}_t(x_{t+1}, a_{t+1}) - \hat{Q}_t(x_t, a_t)\right]$$

used to find the value of $\hat{Q}_{t+1}(x_t, a_t)$ that satisfies:

$$\begin{aligned}
&\hat{Q}_{t+1}(x_t, a_t) = \mathbb{E}_{\pi,\ y \sim P(\cdot \mid x_t,\ a_t),\ a \sim \pi(y)} \left[c_t + Q^\pi(y, a_t) \mid \mathrm{x}_t = x_t,\ \mathrm{a}_t = a_t\right] \\
\Leftrightarrow\ &\mathbb{E}_{\pi,\ y \sim P(\cdot \mid x_t,\ a_t),\ a \sim \pi(y)} \left[c_t + Q^\pi(y, a_t) - \hat{Q}_{t+1}(x_t, a_t) \mid \mathrm{x}_t = x_t,\ \mathrm{a}_t = a_t\right] = 0
\end{aligned}$$

where $c_t + \gamma \hat{Q}_t(x_{t+1}, a_{t+1}) - \hat{Q}_t(x_t, a_t)$ is considered to be a sample of the term inside the expectation in the last expression. In a similar fashion as before, we note that the only solution to the aforementioned equation is $Q^{\pi}(x_t, a_t)$ (since the only fixed point of $\mathbf{H}_{\boldsymbol{\pi}}$ is $\boldsymbol{Q}^{\boldsymbol{\pi}}$), and thus we conclude that the algorithm learns the **value of the currently followed policy**. The value of the optimal policy is then found through *generalized policy iteration*, which is achieved by successively interleaving the previously mentioned updates with computations of the greedy policy with respect to the currently estimated Q-function.