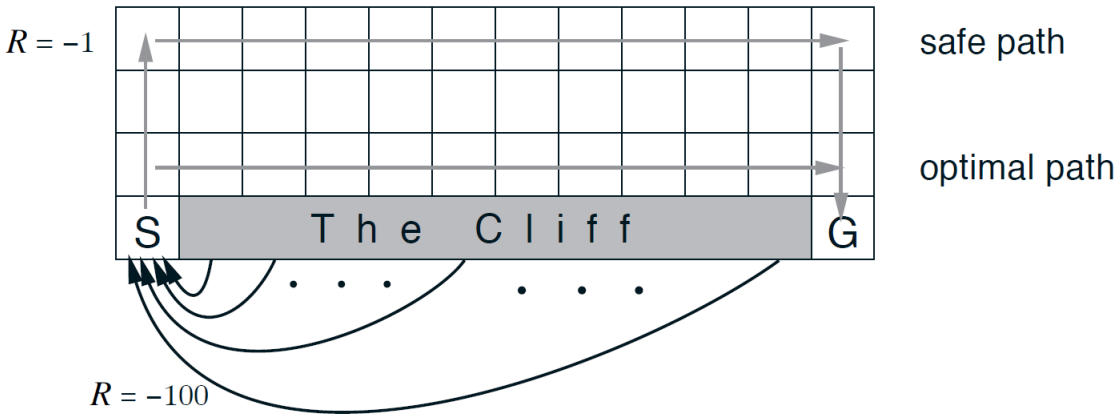
CS47100 : Introduction to Arificial Intelligence

Project : Temporal Difference Learning in Gridworlds

October 13, 2021

In this project, you will implement two RL algorithms, Q-learning and SARSA. You will study their behavior with different hyper-parameters (discussed below) on the cliff walk environment.

Cliff walking



Consider the above environment, known as "Cliff World". At the beginning of the agent-environment interaction, i.e., time step 0, the agent is initialized at the start position "S". The agent interacts with the environment until one the following two events take place;

- The agent reaches the goal position "G"
- Time runs out, i.e. the agent lives for 250 steps ($t = 0, \ldots, 249$)

All transitions are rewarded a -1 (known as the living cost) except those to the region marked as "The Cliff", where a reward of -100 is received, and the agent is pushed back to the start position. The episode terminates when the agent reaches the goal position or it has completed 250 steps. Recall that an MDP is a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$. For the MDP above:

1. $\mathcal{S}$ is the set of individual positions possible in the grid along with number of steps taken by the agent. Each state is represented as a vector (x, y, t) and $\mathcal{S} = \{(x, y, t) | x \in \{0, 1, 2, 3\}, y \in \{0, 1, 2, .., 11\}, t \in \{0, 1, 2, .., 249\}\}$, and $|\mathcal{S}| : 4 \times 12 \times 250$. Furthermore, let $S_c \subset \mathcal{S}$ denote the set of cliff states, $\mathcal{S}_c = \{(x, y, t) | (x, y)$ *marked as* "The Cliff",$t \in \{0, 1, 2, .., 249\}\}$. The agent state is in $\mathcal{S}_c$ whenever the agent is in the "The Cliff" position.
2. $\mathcal{A}$ (action space) is the set $\{up : (0, 1), down : (0, -1), left : (-1, 0)), right : (0, 1)\}$ corresponding to movements in the 4 cardinal directions.

3. P (Transition Probabilities): For any $s \in \mathcal{S}/\mathcal{S}_c$ we have,

$$P_a(s, s') = \mathbb{P}(s_{t+1} = s' | s_t = s, a_t = a) = \begin{cases} 1 & s' = s + (a(0), a(1), 0) + (0, 0, 1) \text{ and } s' \in \mathcal{S} \\ 1 & s' = s + (0, 0, 1) \text{ and } s + (a(0), a(1), 0) + (0, 0, 1) \notin \mathcal{S} \\ 0 & otherwise \end{cases}$$

The above description of the transition function simply says that the environment is deterministic and that if you take an action that moves you out of the grid, the agent does not move.

For $s \in S_c$ we have,

$$P_a(s, s') = Pr(s_{t+1} = s' | s_t = s, a_t = a) = \begin{cases} 1 & s' = (0, 0, 1 + s(2)) \\ 0 & otherwise \end{cases}$$

The above definition suggests that the agent is moved to the start state when it goes to the cliff state and the time continues to tick.

$\forall s \in \mathcal{S},\ s.t.\ s(2) = 249$ *or* $(s(0), s(1)) = (11, 0)$ are terminal states, i.e., the episode stops.

4. $R(s', s) =$

$$= \begin{cases} -1 & s' \notin S_c \\ -100 & s' \in S_c \end{cases}$$

5. The cliff walk environment in this project is an undiscounted MDP, i.e., $\gamma = 1$

Recall that an *episode* is the simulation of an agent from start-state to terminal-state.

Your task is to run the SARSA and Q-learning algorithms to learn a policy for the above environment. Define the Q function and the policies as functions of positions $p := (s(0), s(1))$ and actions instead of state s and action to save memory (why it saves memory?)

[20 points] Fixed Epsilon (Exploration parameter)

- Choose epsilon = 0.1 in both Q-learning and SARSA - Initialize the Q function in both Q-learning and SARSA with zero.

1. [**8 points**] Run each of the algorithms for the total of 500 episodes, store the cumulative reward of each episode. Repeat this process 10 times for each of the algorithms. Now you should have 10 arrays of size 500. Draw the reward curve by averaging out the cumulative reward across run (a plot with y axis as cumulative reward and x axis as episode count).

2. [**8 points**] For each of these 10 runs, look at the learned policy. Visualize the learned policies by SARSA and Q-learning at the last run by plotting the action of the policy for each position on the grid.

3. [**4 points**] Are the policies different? Explain.

[Bonus 5 points] Initialization of the Q function

- Choose epsilon = 0.1 in both Q-learning and SARSA - Initialize the Q function in both Q-learning and SARSA with -1000.

1. **[3 points]** Run each of the algorithms for the total of 500 episodes, store the cumulative reward of each episode. Repeat this process 10 times for each of the algorithms. Now you should have 10 arrays of size 500. Draw the reward curve by averaging out the cumulative reward across run (a plot with y axis as cumulative reward and x axis as episode count).

2. **[2 points]** For each of these 10 runs, look at the learned policy. Visualize the learned policies by SARSA and Q-learning at the last run by plotting the action of the policy for each position on the grid.

[20 points] Varying Epsilon

In this section you gradually reduce the ε at end of each episode, starting from $\varepsilon = 0.1$ to $\varepsilon = 0$. After each episode reduce the ε by $0.1/500$.

- Initialize the Q function in both Q-learning and SARSA with zero.

1. **[7 points]** Run each of the SARSA and Q-learning algorithms with the varying ε for the total of 500 episodes, store the cumulative reward of each episode. Repeat this process 10 times for each of the algorithms. Now you should have 10 arrays of size 500. Draw the reward curve by averaging out the cumulative reward across run (a plot with y axis as cumulative reward and x axis as episode count).

2. **[7 points]** For each of these 10 runs, look at the learned policy. Visualize the learned policies by SARSA and Q-learning at the last run by plotting the action of the policy for each position on the grid.

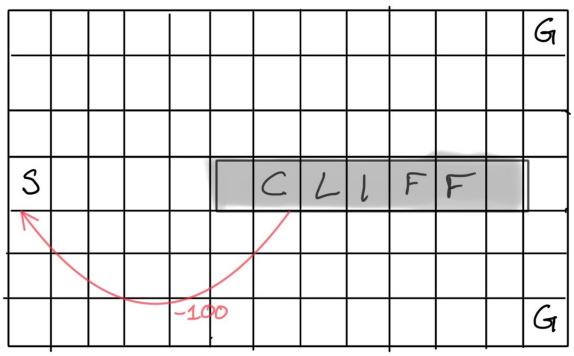
3. **[2 points]** Are the policies learn by Q-learning with varying ε and fixed $\varepsilon = 0.1$ different? explain.

4. **[2 points]** Are the policies learn by SARSA with varying ε and fixed $\varepsilon = 0.1$ different? explain.

5. **[2 points]** Are the policies learn by Q-learning with varying ε and SARSA with varying ε different? explain.

In the following you study another grid world with multiple goal positions

[10 points] Multiple goal states



In the above gridworld with multiple goal states, all transitions are rewarded a -1 except those to the region marked "The Cliff", where the reward of -100 is received and the agent is pushed back to the start. The episode terminates when the agent reaches one of the goal state or it has completed 250 steps. Everything is similar to the cliffwalk you studied so far except that we have two goal positions instead of one. We leave out the description of the MDP in this cases (it trivially follows from the previous description).

- Choose epsilon = 0.1 in both Q-learning and SARSA - Initialize the Q function in both Q-learning and SARSA with zero.

1. [**4 points**] Run each of the algorithms for the total of 500 episodes, store the cumulative reward of each episode. Repeat this process 10 times for each of the algorithms. Now you should have 10 arrays of size 500. Draw the reward curve by averaging out the cumulative reward across run (a plot with y axis as cumulative reward and x axis as episode count).

2. [**4 points**] For each of these 10 runs, look at the learned policy. Visualize the learned policies by SARSA and Q-learning at the last run by plotting the action of the policy for each position on the grid.

3. [**2 points**] Are the policies different? Explain.

Report

The report should be concise and clear. You need to answer the questions in their orders. Plots needs to be drawn using Python, C, or Matlab (not by hand). For example, the report for the first section should be formatted as follows:

Section 1 (corresponding to "Fixed epsilon") should contain the following :

1. The plots of the average reward curve for Q-learning and SARSA along with the explanation of your observations.

2. Figures of grids with the action denoted at each cell for the policies learnt by Q-learning and SARSA along with the explanation of your observations.

3. Explanation of the difference/similarities between the policies learned by the two algorithms.