CS6700: Reinforcement Learning (Jan-May 2024) Programming Assignment 1

Deadline: 11:59 PM, February 28th, 2024

Release Date: February 14, 2024

1 Environment Description

This exercise aims to familiarize you with two popular Temporal Difference Learning algorithms: **SARSA and Q-Learning**. You will solve several variants of the Grid World problem (a sample world is shown in Figure 2).

This is a grid world with 4 deterministic actions ('up', 'down', 'left', 'right'). The agent transitions to the next state determined by the direction of the action chosen with a probability of $p \in [0, 1]$. We also define a parameter called $b \in [0, 1]$. Consider the direction of the action chosen as the agent's "North". For example, if the action is 'left', it is the agent's North, and the agent's East would be the direction of the action 'up'. Figure 1 provides an illustration of the same. The agent transitions to the state West of the chosen action with probability $(1 - p) \times b$, and to the East of the chosen action with probability $(1 - p) \times (1 - b)$.

The environment may also have a wind blowing that can push the agent one **additional** cell to the right **after transitioning to the new state** with a probability of 0.4. An episode is terminated either when a goal is reached or when the timesteps exceed 100. Transitions that take you off the grid will not result in any change in state.

The dimensions of the grid are 10×10. The following types of states exist:

- **Start state**: The agent starts from this state.

- **Goal state**: The goal is to reach one of these states. There are 3 goal states in total.

- **Obstructed state**: These are walls that prevent entry to the respective cells. Transition to these states will not result in any change.

- **Bad state**: Entry into these states will incur a higher penalty than a normal state.

- **Restart state**: Entry into these states will incur a very high penalty and will cause agent to teleport to the start state without the episode ending. Once the restart state is reached, no matter what action is chosen, it goes to the start state at the next step.

- **Normal state**: None of the above. Entry into these states will incur a small penalty.

Rewards: -1 for normal states, -100 for restart states, -6 for bad states, +10 for goal states.
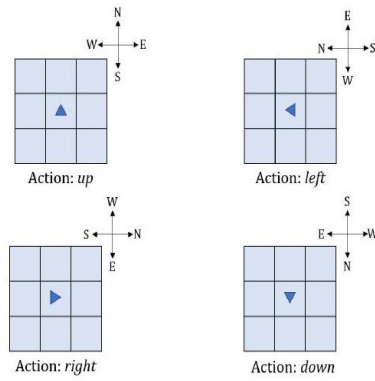
Figure 1: The intended direction of the action chosen is considered as 'North'

2 Additional Information

The code for the environment can be found here. The `env.step()` function takes as arguments the current state and action, and returns the reward and next state. The appropriate termination conditions have to be specified by the student in the code (as explained in section 1, `env.reset ()` resets the environment. In the notebook containing the code, cell 1 contains the environment class, cell 2 contains the environment instantiation, and cell 3 lists some environment variables. For each experiment, the start state is fixed and does not change. Different experiments may have different start states.
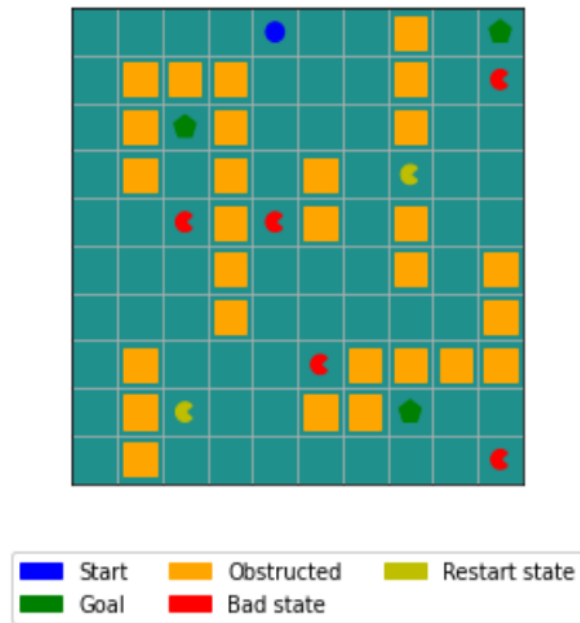

Figure 2: An example grid with start point at (0, 4)

3 Tasks

- Implement SARSA and Q-Learning algorithms.

- For each algorithm, run experiments for the start states $(0, 4)$ and $(3, 6)$. Consider the following three variants of stochasticity for each start state.

  1. `wind=False`(clear), $p = 1.0$(deterministic step).
  2. `wind=False`(clear), $p = 0.7$(stochastic step).
  3. `wind=True`(windy), $p = 1.0$(deterministic step).

  In total, you have to conduct **12 different experiments** - (3 grid world variants $\times$ 2 start states $\times$ 2 algos).

- For each of the 12 experiments, determine the best set of hyperparameters($\{\tau$ in softmax or ϵ in ϵ-greedy$\}$, learning rate α, and discount factor γ), along with the best action selection policy(ϵ-greedy or softmax) with justification. You are also required to plot the following (4 plots per exp) for each experiment(with the best hyperparameters and action selection policy):

  1. Reward curves and the number of steps to reach the goal in each episode.
  2. Heatmap of the grid with state visit counts, i.e., the number of times each state was visited throughout the training phase.
  3. Heatmap of the grid with Q values after training is complete, and optimal actions for the best policy.

- <u>**Note:**</u> Each plot should be obtained by training the agent for at least 5000 episodes and the plots reported should be averaged over 5 runs. For the reward and steps plot, mean and standard deviation should be shown as in Figure 3. The mean values are denoted by the line graph and the shaded region above and below the line denotes the range of values within one standard deviation of the mean. For the heatmaps, it suffices to report mean state-visits and Q values.
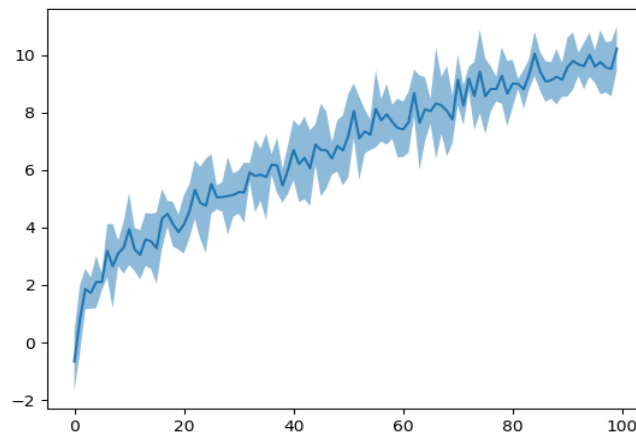

Figure 3: Example of a reward plot

- For each of the experiment, provide a written description of the similarities and differences between best policy learnt by SARSA and Q-learning along with the hyperparameters, justifications and plots requested above. Your description should include the explanation for the behaviour of the policy learnt by each algorithm.

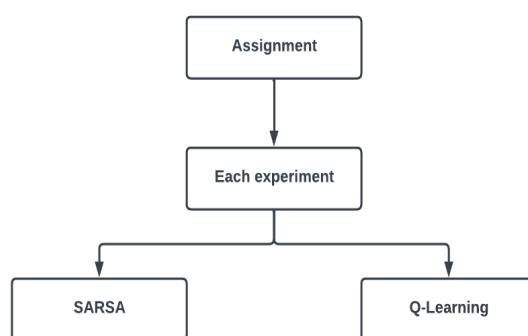Figure 4 summarises the structure of your report.

Figure 4: Report Flowchart

4 Submission Instructions

You are required to submit both your report (as pdf) and your code. All files (code and report included) must be submitted compressed into a single ".zip" file. Please submit in **teams of two**. One submission per team will suffice. The due date for this programming assignment is **11:59 pm on Wednesday, Feb 28**.