Artificial Intelligence: Programming 3 (P3)
Reinforcement Learning

Instructor: Dr. Shengquan Wang

Due Time: 10PM, 11/28/2022

In this programming assignment, we aim to implement one of Reinforcement Learning algorithms: the `Q-Learning` algorithm.

1 Instructions

We extend the windy maze with probabilistic outcome after an action and a few terminal states with rewards $+100$ and -50 respectively. The maze map is shown as follows:

-50	-50	-50	-50	-50	-50	-50
-50						-50
-50						-50
-50			+100			-50
-50						-50
-50	-50	-50	-50	-50	-50	-50

However, we assume that the agent doesn't know either the reward function or the transition model. The agent aims to run many trials in order to obtain Q-value for each (state, action) pair and the optimal action at each state.

Environment In your implementation, you need to simulate the windy maze environment: We assume that the wind comes from the north and the cost of one step for the agent is defined as follows: 1 for moving southward; 2 for moving westward or eastward; 3 for moving northward. The cost will be the negation of the reward. The agent can drift to the left or the right from the perspective of moving direction with probability 0.1. If the drifting direction is an obstacle, it will be bounced back to the original position. If the agent falls into any terminal state, it can't move out.

Q-Learning In your implementation, you will generate many trials, each of which will result in a trajectory of (state, action, reward) tuple. The agent will use the ϵ-`Greedy` algorithm to choose an action at each state along each trajectory, where $\epsilon = 0.1$: the agent chooses a latest optimal action at each state with 90% and a random action with 10%. The initial state for each trial is chosen randomly and each trial will end at the goal state. Along each trajectory, the agent will use `Q-Learning` to update the Q-values. Since the reward function $R(s,a)$ here depends on both the state and the action taken at this state, the Q-value update equations should be revised accordingly (we choose $\gamma = 0.9$).

$$N(s,a) \leftarrow N(s,a) + 1 \tag{1}$$

$$Q(s,a) \leftarrow Q(s,a) + \frac{1}{N_{s,a}}\Big(R(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a)\Big) \tag{2}$$

Testing and Outputs In your testing, generate 50,000 trials starting from a random open square. We initialize the Q-values at any state-action as 0 except for all terminal states respectively. If the number of steps of a trial is more than 100, you can abort this trial and continue with next trial to save time. After 50,000 trials (including the aborted trials), report the following three outcomes for each algorithm:

- the access frequency at each state-action $N_{s,a}$;
- the Q-value function at each state-action $Q(s,a)$;
- the optimal action at each state-action.

The expected outcome should look like as follows:

Table of $N(s,a)$:

-50	-50	-50	-50	-50	-50	-50
	213	307	217	145		
-50	197 7140	312 341	200 210	4999 149	####	-50
	184	11403	8564	175		
		466	607		135	
-50	####	540 15249	571 613	####	542 154	-50
		3281	21621		4582	
		502			281	
-50	####	468 16884	100	####	284 273	-50
		480			10628	
	134	12389	17210	494	278	
-50	146 5318	352 306	446 448	17228 458	10612 296	-50
	139	351	459	423	292	
-50	-50	-50	-50	-50	-50	-50

Table of $Q(s, a)$:

-50	-50	-50	-50	-50	-50	-50
	-32.1	-21.2	-20.0	-26.9		
-50	-40.5 31.4	14.7 45.1	30.3 25.7	30.0 20.9	####	-50
	12.2	52.7	62.3	29.5		
		51.6	56.6		-11.1	
-50	####	55.9 68.3	61.7 73.2	####	-4.3 -32.3	-50
		67.4	81.7		2.5	
		64.3			-8.3	
-50	####	65.2 78.2	100	####	8.9 -27.4	-50
		60.6			13.3	
	24.2	62.3	74.3	39.1	1.8	
-50	-33.1 30.9	23.4 43.7	50.8 28.4	49.3 8.0	26.6 -38.9	-50
	-29.2	-21.7	-15.8	-17.7	-26.5	
-50	-50	-50	-50	-50	-50	-50

Table of the optimal policy:

-50	-50	-50	-50	-50	-50	-50
-50	>>>>	vvvv	vvvv	<<<<	####	-50
-50	####	>>>>	vvvv	####	vvvv	-50
-50	####	>>>>	+100	####	vvvv	-50
-50	>>>>	^^^^	^^^^	<<<<	<<<<	-50
-50	-50	-50	-50	-50	-50	-50

where `<<<<`: moving westward; `^^^^`: moving northward; `>>>>`: moving eastward; `vvvv`: moving southward; $+100, -50$: the terminal rewards.

For the first two tables, it is expected that the trend of your outputs should match the above while the exact values could be very different from the above due to the random operations. For the last table regarding the optimal policy, most actions of your output should match exactly with above.

2 Submission

You are going to report the following things:

(a) Describe in details how you implemented the following modules in the report: `environment simulation`, ϵ-`greedy`, and `Q-learning update`.

(b) Comment your code in details so that the grader can understand it well.

(c) Include the screenshots of all above testing outcomes. Each screenshot should include your username and the current time, which show that you did it by yourself.

(d) Specify the contribution made by each member if you work as a group.

The report should be written in a ".docx", ".doc", or ".pdf" format. Submit both the report and the source code to the assignment folder P3 on Canvas. Any compression file format such as `.zip` is not permitted.