Artificial Intelligence: Programming 3 (P3)
Reinforcement Learning

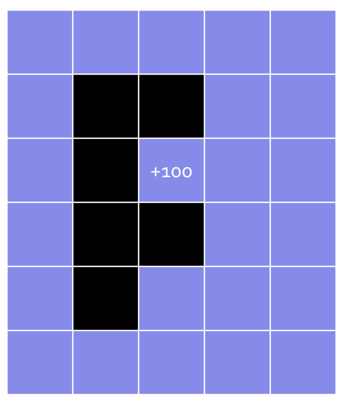
Instructor: Dr. Shengquan Wang

Due Time: 10PM, 12/3/2020

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

1 Instructions

We extend the windy maze defined in P1 with probabilistic outcome after an action and one terminal state (top left) with a reward +100. It becomes a MDP problem. The maze map is shown in the following figure. 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 reward 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 0.15. 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.

Reinforcement 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 $\epsilon = 0.05$: the agent chooses a latest optimal action at each state with 95% and a random action with 5%. 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.

$$
\begin{aligned}
N(s,a) &\leftarrow N(s,a)+1 &\quad (1)\\
Q(s,a) &\leftarrow Q(s,a)+\frac{1}{N_{s,a}}\Big(R(s,a)+\gamma \max_{a'} Q(s',a')-Q(s,a)\Big) &\quad (2)
\end{aligned}
$$

We choose $\gamma = 0.9$.

Testing and Outputs In your testing, generate 10,000 trials starting from a random open square. We initialize the Q-values at any state-action as 0 except for one terminal state with +100 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 10,000 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** $N(s,a)$**:**

28	72	689	85	38
36 2071	63 3596	128 8347	93 86	35 45
285	406	175	6643	2711
1584			309	139
150 53	####	####	335 328	1015 155
1235			26916	11212
38			428	167
44 37	####	+100	35796 474	13041 185
2929			484	503
68			25141	1836
70 69	####	####	312 342	16690 256
4526			312	245
107		83	10266	13045
132 110	####	70 5767	871 163	162 179
7730		115	154	223
260	149	172	8794	5131
243 5352	157 9061	187 12647	128 147	88 78
12459	1934	934	539	578

- **Table of $Q(s,a)$:**

```
    -1.1              6.4               8.2               22.5              20.3

-3.3     1.7     1.7     12.4     9.7     22.7     22.2    26.0     24.8    24.2

    -4.7              -2.3              8.8               33.7              30.4

    -4.5                                                  31.2              27.9

-6.0    -6.5          ####              ####         40.6    39.5     33.5    33.9

    -6.2                                                  51.7              41.7

    -7.8                                                  48.8              36.9

-6.9    -7.1          ####              +100         67.7    40.5     47.3    35.3

    -6.7                                                  54.4              39.8

    -8.4                                                  46.7              33.6

-7.5    -7.1          ####              ####         43.3    31.7     37.1    28.9

    -6.8                                                  30.0              28.2

    -8.5                                10.8              28.1              25.5

-7.4    -7.2          ####          5.5     13.4     15.2    19.7     22.3    18.6

    -6.3                                4.5               9.1               13.9

    -7.6              -4.3              4.9               13.9              14.1

-7.1    -4.9     -5.5    -0.5     -0.0     6.4     7.2      8.9     9.0      8.3

    -5.9              -5.4              -3.9              -2.1              -2.3
```

- **Table of the optimal policy:**

```
>>>>     >>>>     >>>>     vvvv     vvvv

^^^^     ####     ####     vvvv     vvvv

vvvv     ####     +100     <<<<     <<<<

vvvv     ####     ####     ^^^^     <<<<

vvvv     ####     ^^^^     ^^^^     ^^^^

>>>>     >>>>     >>>>     ^^^^     ^^^^
```

where `<<<<`: moving westward; `^^^^`: moving northward; `>>>>`: moving eastward; `vvvv`: moving southward; +100: the terminal reward.

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

Form a group on Canvas if you want to work with another student. 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.