

Reinforcement Learning Assignment Report

Li Kaiyu (S220048)

AI6101
School of Computer Science and Engineering
Nanyang Technological University
2023

Background

In this assignment, we are required to choose one of RL algorithms to solve a specific CliffBoxPushing grid-world game. The game map is shown in figure 1. In this game, the agent 'A' tries to 'push' the box 'B' to the goal 'G' as quick as possible while avoiding the cliff which is denoted as 'x' on the map.

	0	_1_	_2_	_3_	_4_	_5_	_6_	_7_	_8_	_9_	_10_	_11_	_12_	_13_
0							x	x						
1							x	x						
2				x			x	x					x	
3				x			x					x	x	
4		B		x								x	x	G
5	A			x								x	x	

Figure 1: Game Map

Algorithm Implementation

Q-Learning

We implement the Q-Learning algorithm to solve the game. Q-learning is one of the fundamental RL algorithms that allows agents to learn optimal policies without the need of transition probability distribution. It works based on the concept of a Q-table, keeping updating $Q_\pi(s, a)$ (which is the expected reward if action a is taken at state s under policy π for every states and corresponding action sets).

Now we briefly describe the Q-Learning method. Before training, we initialize an all-zero Q-table and a ϵ -soft policy as follow:

$$\pi(s, a) = \begin{cases} 1 - \epsilon + \epsilon/|A(s)|, & \text{if } a = a^*. \\ \epsilon/|A(s)|, & \text{if } a \neq a^*. \end{cases} \quad (1)$$

where $|A(s)|$ is the size of action set at state s . Then at each iteration, an episode is generated using π , and at each step in each episode, we update the Q-table using the following updating rule:

$$Q_{new}(S_t, A_t) = Q_{old}(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_a Q_{old}(S_{t+1}, a) - Q_{old}(S_t, A_t)) \quad (2)$$

where α is the learning rate, and γ is the discount factor to reduce future rewards to help reaching convergence.

Epsilon Decay

During the experiments we find out that the algorithm finds out a policy that leads to the goal at around 6000-7000 episodes, so we set the total episode number to be 8000.

step: 35, state: (1, 13, 2, 13), actions: 2, reward: -4

Action: 2

```
[b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'_'
[b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'A']
[b'_' b'_' b'_' b'x' b'_' b'_' b'x' b'x' b'_' b'_' b'_' b'_' b'_' b'x' b'_' b'B']
[b'_' b'_' b'_' b'x' b'_' b'_' b'x' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'_]
[b'_' b'_' b'_' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'G']
[b'_' b'_' b'_' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'_] ]]
```

step: 36, state: (2, 13, 3, 13), actions: 2, reward: -3

Action: 2

```
[b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'_'
[b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'_]
[b'_' b'_' b'_' b'x' b'_' b'_' b'x' b'x' b'_' b'_' b'_' b'_' b'_' b'x' b'_' b'A']
[b'_' b'_' b'_' b'x' b'_' b'_' b'x' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'B']
[b'_' b'_' b'_' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'G']
[b'_' b'_' b'_' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'_] ]]
```

step: 37, state: (3, 13, 4, 13), actions: 2, reward: 998

Action: 2

```
[b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'_'
[b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'_]
[b'_' b'_' b'_' b'x' b'_' b'_' b'x' b'x' b'_' b'_' b'_' b'_' b'_' b'x' b'_' b'_]
[b'_' b'_' b'_' b'x' b'_' b'_' b'x' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'A']
[b'_' b'_' b'_' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'B']
[b'_' b'_' b'_' b'x' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'_' b'x' b'x' b'_' b'_] ]]
```