

05124265: Reinforcement Learning  
Exercise 3

Tal Grossman, 201512282, Moshe Yelisevitch, 207423104

10/07/2024

## 1 Theory

for theory sections please see handwritten solution.

RL-Ex 3.

Q1. Yes. Proof:

$$\pi^*(s) \stackrel{\text{by definition}}{=} \operatorname{argmax}_a Q_M^*(s, a) = \operatorname{argmax}_a Q_M^*(s, a) + \underset{\substack{\uparrow \\ \text{const.}}}{f(s)} = \operatorname{argmax}_a Q_{M'}^*(s, a)$$

$\pi^* = \operatorname{argmax}_a Q_{M'}^*(s, a)$  is defined as the <sup>w.r.t. a</sup> optimal policy

for  $Q_M^*(s, a)$ , so we proved that  $\pi^*(s)$  is the optimal

Policy for  $Q_{M'}^*(s, a)$  also!

Q3. 1. Each iteration we run  $\pi_t$ , the opt. policy for iter.  $t$ .

Q 2 :

- Bellman eq for  $m$ :

$$Q_m^* = \sum_{s'} P(s'|s, a) [R(s, a, s') + \gamma \max_{a'} Q_m^*(s', a')]$$

- Bellman eq for  $m'$ :

$$Q_{m'}^* = \sum_{s'} P(s'|s, a) [R(s, a, s') + \gamma \max_{a'} Q_{m'}^*(s', a')]$$

Plug in  
 $a'$  as  
 $a$

$$= \sum_{s'} P(s'|s, a) [R(s, a, s') + \phi(s) - \gamma \phi(s') + \gamma \max_{a'} Q_m^*(s', a')]$$

$$= \sum_{s'} P(s'|s, a) R(s, a, s') + \sum_{s'} P(s'|s, a) \phi(s) - \gamma \sum_{s'} P(s'|s, a) \phi(s') \\ + \gamma \sum_{s'} P(s'|s, a) \max_{a'} Q_m^*(s', a')$$

$\phi(s)$   
is  
independent  
of  $s'$

$$= \sum_{s'} P(s'|s, a) R(s, a, s') + \phi(s) \sum_{s'} P(s'|s, a) - \gamma \sum_{s'} P(s'|s, a) \phi(s') \\ + \gamma \sum_{s'} P(s'|s, a) \max_{a'} Q_m^*(s', a')$$

$$\sum_{s'} P(s'|s, a) = 1 \Rightarrow \sum_{s'} P(s'|s, a) R(s, a, s') + \phi(s) \cdot 1 - \gamma \sum_{s'} P(s'|s, a) \phi(s') \\ + \gamma \sum_{s'} P(s'|s, a) \max_{a'} Q_m^*(s', a')$$

$$\Rightarrow Q_{m'}^*(s, a) = Q_m^*(s, a) + \underbrace{\phi(s) - \gamma \sum_{s'} P(s'|s, a) \phi(s')}_{=0}$$

Since this term is a state dependent and independent of  $a$ , it does not affect on the selection of the optimal policy.

$\Rightarrow$  The optimal policy for  $m$  and  $m'$  are identical //

Q3. 1. Each iteration we run  $\pi_t$ , the opt. policy for iter.  $t$ ,

until a new state-action is found. Let's assume by

contradiction that exists a pair  $(s', a')$  such that

$R'_t(s', a') = 1$  we didn't discover at iteration  $t$ . In other words, the

optimal policy  $\pi_t$  stuck on  $(s, a) : R'_t(s, a) = 0$ , so  $V_t^\pi = 0$ .

Given a strongly connected MDP, we know  $(s', a')$  can be visited

So a policy  $\pi'$  that does so will get a reward of 1 at  $(s', a')$ ,

thus  $\pi$  is not optimal, in contradiction!

Each iteration must end!



2.  $T_{\text{iteration}} = O(|S|)$ . For example, in the case that we should go over all the states that we have already visited in order to go from  $S_0$  to  $(s', a')$

3. every state  $s$  in the set  $S$  has a set of possible action  $A(s)$ , So in order to go over the entire

$S$ , a space we need  $\sum_{s \in S} |A(s)| = |S| \cdot |A|$  iterations

$$\begin{aligned} 4. T_{\text{exploitation}} &= T_{\text{iteration}} \cdot \text{num\_iterations} = O(|S|) \cdot |S| \cdot |A| = \\ &= O(|S|^2 \cdot |A|) \end{aligned}$$

Q 4:

1) - from the starting position  $(b, 1)$  the legal moves are:

$(b, 1) \rightarrow (a, 3)$

$(c, 3)$

$(d, 2)$

- to return to  $(b, 1)$  after 2 steps the knight must move to a position which it can return to  $(b, 1)$ , possible moves are:

$(a, 3) \rightarrow (b, 1)$

$(c, 3) \rightarrow (b, 1)$

$(d, 2) \rightarrow (b, 1)$

- So, initially the knight has 3 legal moves

- from each it can return to  $(b, 1)$

$$\Rightarrow P = \frac{1}{3} \cdot \frac{1}{8} \cdot 3 = \frac{1}{8} //$$

2) - the markov chain is irreducible since it's possible to get from any state to any other,  
for the knight it can eventually get to any square from any starting position.

- the markov chain is aperiodic since for the knight, returning to any position is not confined to multiples of particular number of steps.

3) the mean recurrence time  $T_i$  to a state  $i$  is the reciprocal of the stationary distribution  $\pi_i$  for that state

Since the knight spends an equal amount of time on each square:

$$\pi_i = 1/64 \quad \forall i$$

$\Rightarrow$

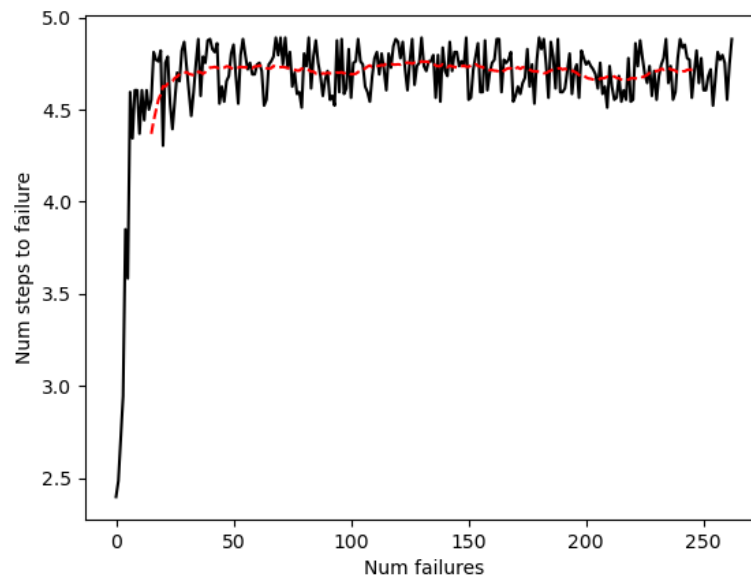
$$T_{\text{corner}} = \frac{1}{\pi_{\text{corner}}} = 64$$

## 2 Programming

### 2.1 Question 1: Off-Policy Model-Based

completed in python in the attached file **control.py**

In that specific run, it took 263 iterations to converge. The plot of the failure rate is as follows:



### 2.2 Question 2: Q-Learning

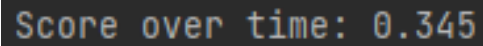
#### 2.2.1 Item 1 - tabular\_Q.py

The percentage of successful episodes is roughly 56.6% . The Q-table is as follows:

```
Score over time: 0.566
Final Q-Table Values
[[1.21427361e-01 1.83254485e-02 1.48874363e-02 1.92531811e-02]
 [1.82363151e-03 1.64267572e-03 5.90394023e-04 1.63101858e-01]
 [0.00000000e+00 3.74026090e-03 2.63652325e-03 2.57552930e-01]
 [5.99671903e-04 2.84269525e-04 2.21128911e-04 1.03078260e-01]
 [2.72628630e-01 2.31807525e-03 6.13800917e-04 1.96965210e-03]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]
 [1.72937261e-04 8.75123858e-08 1.84143980e-01 3.24574625e-05]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]
 [2.86188773e-03 4.22361064e-04 5.35911767e-04 3.60103396e-01]
 [0.00000000e+00 3.54639242e-01 0.00000000e+00 2.76921632e-03]
 [1.05852451e-01 1.79812038e-03 2.10456247e-04 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 2.85918529e-01 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 8.55830849e-01 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]]
```

**2.2.2 Item 2 - network\_Q.py**

The percentage of successful episodes is roughly 34.5%. This result is worse than what we achieved with tabular Q-learning, probably due to the fact that the network is not deep and complicated enough to capture the complexity of the environment. We believe that with a deeper network with some activation functions, we could achieve better results.



```
Score over time: 0.345
```