Paula Bezares
20393015
Eshaan Mittal
20620983

# LAB 3: REINFORCEMENT LEARNING

**1. We will start by tackling the simple problem seen in class of finding a path from start to goal in the following scenario.**

**a. Implement the Q-learning algorithm to find the optimal path considering a reward of -1 everywhere except for the goal, with reward 100.**

**i. Print the first, two intermediate and the final Q-table. What sequence of actions do you obtain?**

```
Episode: 0
Total reward: 56.0
Q-Learning Matrix:
                Up          Down        Left        Right
(0, 0): [-0.2   -0.394 -0.2    0.   ]
(0, 1): [0. 0. 0. 0.]
(0, 2): [-0.2 -0.2  0.   0. ]
(0, 3): [0. 0. 0. 0.]
(1, 0): [-0.36  -0.394 -0.2   -0.2  ]
(1, 1): [0. 0. 0. 0.]
(1, 2): [-0.2 -0.2 -0.2 -0.2]
(1, 3): [20.  -0.2  0.  -0.2]
(2, 0): [-0.36  -0.394 -0.394 -0.36 ]
(2, 1): [-0.36  -0.2   -0.234 -0.394]
(2, 2): [-0.36  -0.2   -0.234 -0.488]
(2, 3): [-0.2    -0.2    -0.5492  0.    ]
```

```
Episode: 50
Total reward: 96.0
Q-Learning Matrix:
                Up          Down        Left        Right
(0, 0): [-0.732724   -0.7647146  -0.732724   17.29831817]
(0, 1): [ 4.25804697 -0.394      -0.4484     55.26636709]
(0, 2): [40.801132   32.65890638 -0.234      99.99319435]
(0, 3): [0. 0. 0. 0.]
(1, 0): [ 2.11690645 -1.20504064 -1.25091741 -1.19399268]
(1, 1): [0. 0. 0. 0.]
(1, 2): [83.91685945 33.94569494 13.29944546 19.5836416 ]
(1, 3): [83.222784  1.05984   0.         5.76     ]
(2, 0): [-1.32816072 19.52273598  9.63802632 47.1972004 ]
(2, 1): [ 9.7222342  16.37190511 13.41863733 57.99820354]
(2, 2): [70.09945951 27.38121608 25.30627488  8.79291597]
(2, 3): [35.1887872 -0.394      -0.67336    2.067712 ]
```

```
Episode: 100
Total reward: 94.0
Q-Learning Matrix:
                Up          Down        Left        Right
(0, 0): [-0.732724   -0.7647146  -0.732724   47.29865202
(0, 1): [16.66688452 30.17723664 11.66306084 80.91462364
(0, 2): [56.35272437 40.20595109 51.13268615 99.99999985
(0, 3): [0. 0. 0. 0.]
(1, 0): [11.32044383 -1.20504064 -1.25091741 -1.19399268
(1, 1): [0. 0. 0. 0.]
(1, 2): [83.99999556 50.6552158  47.00167211 38.21012787
(1, 3): [89.26258176  1.05984    14.07063961  5.76
(2, 0): [ 0.20346566 32.00202426 30.42770334 49.01369049
(2, 1): [28.89511476 45.50123217 37.71052664 58.83988378
(2, 2): [70.39997453 45.93361445 42.77593461 12.8164266
(2, 3): [42.09890304 -0.394      -0.67336    2.067712
```

```
Converged after 130 episodes
Total reward: 96.0
Q-Learning Matrix:
                Up          Down        Left        Right
(0, 0): [ 7.9508701   6.16751069 20.4552456  61.88485021]
(0, 1): [16.66688452 37.69727533 17.17121952 83.35294967]
(0, 2): [ 61.88217947  51.07580868  59.79339701 100.        ]
(0, 3): [0. 0. 0. 0.]
(1, 0): [37.43172753 -1.20504064  3.42349111  4.67509906]
(1, 1): [0. 0. 0. 0.]
(1, 2): [83.99999998 53.60173785 51.68133762 38.21012787]
(1, 3): [95.60195349  1.05984    14.07063961  5.76      ]
(2, 0): [12.96744823 33.73399804 34.11198659 49.01399686]
(2, 1): [38.71311512 47.8629308  39.45298174 58.839999  ]
(2, 2): [70.39999976 53.55354172 45.82010022 31.03388669]
(2, 3): [63.09897115 -0.394      -0.67336    2.067712  ]
```

(2,0)→(2,1)→(2,2)→(1,2)→(0,2)→(0,3)

**ii. After trying for a bit, what is your parameter choice for alpha, gamma and epsilon? Why?**

Alpha = 0.2, Gamma = 0.85, epsilon = 0.3
Alpha and gamma values chosen by criteria of least episodes taken to converge.
Best epsilon value that allowed for a most cells to be fairly evaluated, without over exploring. We avoid over exploring as the environment is small, meaning there will not be many paths that are better.

**iii. How do you judge convergence of algorithm? How long does it take to converge?**

Convergence judged by the norm of the difference between the Q tables of two successive episodes. Must be smaller than delta = 0.0001 10 times. Takes 130 episodes.

Paula Bezares
20393015
Eshaan Mittal
20620983

**b. Try implementing the more accurate reward given by:**

**i. Answer the questions of the previous section for this case.**

a. **Print the first, two intermediate and the final Q-table. What sequence of actions do you obtain?**

```
Episode: 0
Total reward: 30.0
Q-Learning Matrix:
             Up          Down         Left        Right
(0, 0): [-1.53 -1.2  -0.9  -1.02]
(0, 1): [-0.6   -1.074 -1.008 -0.51 ]
(0, 2): [-0.3   -0.6   -0.654 30.   ]
(0, 3): [0. 0. 0. 0.]
(1, 0): [-1.53  -1.5   -2.202 -1.2  ]
(1, 1): [0. 0. 0. 0.]
(1, 2): [-0.3  0.    0.    0. ]
(1, 3): [0. 0. 0. 0.]
(2, 0): [-2.04  0.    -1.5   0.  ]
(2, 1): [0. 0. 0. 0.]
(2, 2): [0. 0. 0. 0.]
(2, 3): [0. 0. 0. 0.]
```

```
Episode: 25
Total reward: 79.0
Q-Learning Matrix:
             Up          Down         Left        Right
(0, 0): [-1.53       -2.43636   -1.692      18.0503858]
(0, 1): [-0.6         -0.49806    -1.008       53.10790079]
(0, 2): [47.28004145 16.90357159  9.6425746  99.88601105]
(0, 3): [0. 0. 0. 0.]
(1, 0): [ 1.71326785 -4.35785256 -3.13776    -3.18528   ]
(1, 1): [0. 0. 0. 0.]
(1, 2): [58.07152886  2.94140568  9.2949313  30.102936  ]
(1, 3): [91.76457    3.2586846 -0.6        22.74294  ]
(2, 0): [-4.82420468 -6.66603032 -6.0474744   1.90782127]
(2, 1): [-4.95735936 -3.9175104  -5.56423485 14.00050726]
(2, 2): [31.25263575  4.61971514  0.42928612  1.542552  ]
(2, 3): [32.221647  -0.6        -1.008       3.2586846]
```

```
Episode: 75
Total reward: 76.0
Q-Learning Matrix:
             Up          Down         Left        Right
(0, 0): [-1.53        -2.43636     1.8026446  32.04607244]
(0, 1): [20.82608321  9.49150375 -1.008       58.88241453]
(0, 2): [57.61126739 27.7412776  24.78071684 99.99999998]
(0, 3): [0. 0. 0. 0.]
(1, 0): [13.85078993 -4.35785256 -3.13776    -1.520246  ]
(1, 1): [0. 0. 0. 0.]
(1, 2): [58.99999839 10.13141094 25.13193209 47.34503123]
(1, 3): [99.32177693  3.2586846  -0.6        22.74294   ]
(2, 0): [ 1.86730523 -2.10533745 -2.30868712  6.22387205]
(2, 1): [ 5.55060417  4.96151815 -2.83962859 17.03994782]
(2, 2): [33.39998474 13.99595425  3.34616199 20.29804981]
(2, 3): [51.69139797  7.76549096  4.4058071  15.51209009]
```

```
Converged after 101 episodes
Total reward: 90.0
Q-Learning Matrix:
             Up          Down         Left        Right
(0, 0): [-1.53        -2.43636     6.20058951 33.05459476]
(0, 1): [20.82608321  9.49150375  4.31888826 58.98023741]
(0, 2): [ 58.31952102  31.45905818  29.15665367 100.        ]
(0, 3): [0. 0. 0. 0.]
(1, 0): [15.58225135 -4.35785256 -3.13776    -1.520246  ]
(1, 1): [0. 0. 0. 0.]
(1, 2): [59.         15.38124619 27.61235237 47.34503123]
(1, 3): [99.66767069  3.2586846  -0.6        22.74294   ]
(2, 0): [ 3.79318745 -1.55363414 -1.6233794   6.2239999 ]
(2, 1): [ 5.89403592  5.790968   -1.80549512 17.03999997]
(2, 2): [33.4        15.99589182  4.8138592  25.11630227]
(2, 3): [55.24787279  7.76549096  4.4058071  15.51209009]
```

(2,0)→(2,1)→(2,2)→(1,2)→(0,2)→(0,3)

b. **After trying for a bit, what is your parameter choice for alpha, gamma and epsilon? Why?**

Alpha = 0.3, Gamma = 0.6, epsilon = 0.3
Alpha increased as rewards are different, so we care more about the current episode's values. Gamma decreased as we want faster routes now that most rewards are lower. Epsilon untouched for the above reason.

c. **How do you judge convergence of the algorithm? How long does it take to converge?**

Same as above. Now takes 101 episodes.

**ii. What is the effect of the new reward function on performance?**

The effect of the new reward function is that the algorithm now converges faster given that the rewards of each state are more representative of their respective position. Due to this, we could further refine the alpha and gamma values to continue improving the algorithm.

**iii. How does this relate to the search algorithms studied in P1? Could you apply one of those in this case?**

This change to the reward structure can be compared to the heuristic functions in search algorithms, as they help guide the q-table to better movements faster. So we can use this to construct the heuristic functions to use in the greedy or A* search algorithms.

**c. The main novelty in RL algorithms with respect to the search algorithms in P1 is that they can be applied in stochastic environments, where the agent doesn't fully determinate the outcome of its actions.**

**i. Drunken sailor. Your agent is now a drunken sailor trying to get to bed after a good share of whiskey and shanties: their legs don't seem to obey them all the time. Introduce stochasticity (=randomness) by enforcing that only 99% of the steps intended by the sailor are actually taken, the rest leading randomly in any other possible direction.**

**ii. Use at least one of the two rewards proposed:**

**a. What is your parameter choice? Why?**

Alpha = 0.3, gamma = 0.6, epsilon = 0.3
We decided to keep them the same as the randomness is only 1%, there for is no improvement changing them. Furthermore, as we use the second reward table, it provides a more accurate Q-table in the end.

**b. Assuming the sailor is in a state that allows learning: how many drunken nights are necessary for them to master the perilous path to bed? Compare to the previous, deterministic scenario.**

122 episodes. In comparison to deterministic scenarios there is not a noticeable difference as the drunken state has a small effect being only 1%.

**c. What is the optimal path found? If we watched the sailor try to follow it, would they always follow the same path?**

(2,0)→(2,1)→(2,2)→(2,3)→(1,3)→(0,3)
No, he would not, as the agent is in a drunken state, even if we tell him to follow the correct movement, there is a small chance that another action is performed.

**d. Could we apply one of the algorithms in P1 here? Why?**

No, we cannot as we are now in a non-deterministic scenario. And as the algorithms for P1 are only useful in deterministic scenarios, they would not be useful here.

Paula Bezares
20393015
Eshaan Mittal
20620983

**2. Now, let's move back to the chess scenario, namely the first board configuration of P1. Remember that we have the black king, the white king and one white rook, and that only whites move. Remember to provide the first, two intermediate and the final Q-table in every case.**

**a. Adapt your Q-learning implementation to find the optimal path to a check mate considering a reward of -1 everywhere except for the goal (check mate for the whites), with reward 100.**

**i. After trying for a bit, what is your parameter choice for alpha, gamma and epsilon? Why?**

Alpha = 0.3, gamma = 0.95, epsilon = 0.05
Gamma is raised to a much higher value as the solution will be a longer path, and by applying a higher value, we do not discount the previous values. Epsilon is decreased to a very small value as we do not want to select too many random actions given that the action set is very large, and many of them are also invalid moves.

**ii. How do you judge convergence of the algorithm? How long does it take to converge?**

We judge convergence of the algorithm by comparing the last and current Q-table with only a delta of 0.01. The algorithm takes 47,307 episodes to converge.

**b. Try now with a more sensible reward function adapted from the heuristic used for the A* search:**

**i. Answer the questions of the previous section for this case.**

The parameters are maintained as they were before for them same reasons. And convergence is measured the same way. However, the algorithm now takes 8,926 episodes to converge.

**ii. What is the effect of the new reward function on performance?**

The new reward function helps decrease the execution time of the algorithm by a significant amount.