Delft University of Technology

Artificial Intelligence Techniques
IN4010-12

Assignment 3: Reinforcement Learning

Authors:

Francesca Drummer (5413990)
Justin de Haan (4623584)
Marios Marinos (5353106)
Tim Polderdijk (4689313)

January 12, 2021

1 Solutions

1.1 Question 1

The agents learns that a pothole must be avoided, as there is a penalty attached to falling in the hole. In the "walkInThePark" environment there are ways to avoid potholes while still reaching the goal. "theAlley" is more difficult to learn in, as it wants to avoid moving right when near a pothole. However, the agent has to move right and take a chance if it ever wants to reach the goal.

1.2 Question 2

Figure 1 shows the policy which the agent finds after 1000 episodes. Firstly, a policy is considered as optimal if the agent is able to choose the best action for each state, for maximum rewards over time. Therefore, the policy is not optimal, because some states remain unexplored (indicated by `N`) and also, it is obvious that from the starting point (top left corner) going up is not the best action as it doesn't lead us to the goal (reward).In addition to that, in other states the actions of the agent leads into a hole, e.g. last row position 3 which has a right arrow and therefore, leads the agent into the hole on the right.

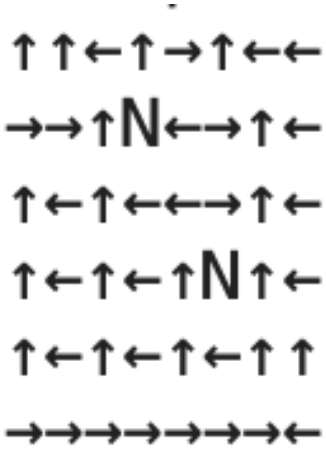

Figure 1: Optimal Policy with $\epsilon = 0.05$, $\gamma = 0.9$, $\alpha = 0.1$

1.3 Question 3

Figure 2 shows the optimal Q* table for "theAlley" map:

```
[[ 2.40017964  2.2924347   2.0390918   2.46727822]
 [ 1.77619548  1.72388051  2.42706248  1.74494444]
 [ 1.96155427  1.96120566  2.04985677  2.518936  ]
 [ 1.61163821  1.8354506   0.92292536  2.12872135]
 [-2.75692382 -2.79061693 -2.24588374 -3.48516348]
 [ 1.64608308  1.66044418  0.58065287  2.29970907]
 [ 2.04558421  0.61889138  0.49206373  0.75636608]
 [ 0.7590087   0.76097088  0.6483935   0.41575807]
 [-2.50679629 -3.03219126 -3.588031   -1.01888664]
 [ 0.          0.27        0.          0.        ]
 [ 0.          0.          0.          0.        ]
 [ 0.          0.          1.          0.        ]
 [ 0.          0.          0.          0.        ]]
```

Figure 2: Optimal Q* table for "theAlley" map with parameters $\epsilon = 0.05$, $\gamma = 0.9$, $\alpha = 0.1$, BROKEN_LEG_PENALTY = -10

1.4 Question 4

Figure 3 shows the optimal policy for "theAlley" map with parameters $\epsilon = 0.05$, $\gamma = 0.9$, $\alpha = 0.1$, BROKEN_LEG_PENALTY = -10. The policy found by the agent is quite good for the first states where it tries to avoid the holes by going up and not choosing to directly step into the holes. However, the actions chosen according to the optimal policy gets worse later on in the map as the agent haven't been able to explore further in 1000 episodes with such a small exploration rate ($\epsilon = 0.05$). For example, it would only choose random actions for two states in the end (indicated by `N`) while it should just learn to go to the right towards the goal.

Figure 3: Optimal Policy for "theAlley" map with parameters $\epsilon = 0.05$, $\gamma = 0.9$, $\alpha = 0.1$, BROKEN_LEG_PENALTY = -10

1.5 Question 5

Figure 4 shows the optimal Q* table with BROKEN_LEG_PENALTY = -5 for "theAlley" map:

```
[[ 2.4016272   2.54830783  2.60319993  2.7       ]
 [ 2.31285246  2.33245585  1.99930982  2.39452066]
 [ 1.99620876  1.72004095  2.60080414  1.69483613]
 [ 1.75970766  1.86507601  0.82668923  2.2309656 ]
 [ 0.44692647 -0.77882953 -1.00607517 -0.54683991]
 [ 1.43157348  0.          0.          0.        ]
 [ 0.          0.          0.          0.        ]
 [ 0.          0.          0.          0.        ]
 [-0.5         0.          0.          0.        ]
 [ 0.          0.          0.          0.        ]
 [ 0.          0.          0.          0.        ]
 [ 0.          0.          0.          0.        ]
 [ 0.          0.          0.          0.        ]]
```

Figure 4: Optimal Q* table for "theAlley" map with parameters $\epsilon = 0.05$, $\gamma = 0.9$, $\alpha = 0.1$, BROKEN_LEG_PENALTY = -5

1.6 Question 6

The agent does not find the optimal policy for "theAlley" map, because the agent explores even less states than in Question 4. Based on the argumentation to Question 4 we therefore conclude that this agent does also not find the optimal policy.

Figure 5: Optimal Q* table for "theAlley" map with parameters $\epsilon = 0.05$, $\gamma = 0.9$, $\alpha = 0.1$, BROKEN_LEG_PENALTY = -5

1.7 Question 7

The exploration strategy of the agent has been modified in two ways in an attempt to improve the rate it learns the optimal policy.

1. Modified ϵ-greedy exploration phase to allow for an adjustable exploration rate;
2. Penalize actions that would make the agent go out of bounds.

1.7.1 epsilon-greedy exploration phase

The exploration strategy of the agent has been changed to allow for an adjustable exploration rate. The exploration rate is used to balance the agents' behaviour between exploration and exploitation. Having an exploration rate (ϵ) of 0.05 as proposed in the assignment doesn't allow the agent to explore the environment. As the epsilon value is tiny, the agent won't be able to find an optimal policy (fast). For this reason, we start with an exploration rate of 1. This makes the agent explore the environment (by picking arbitrary actions). After each episode, the exploration rate falls off exponentially with a threshold of 0.1. The intuition behind this reasoning is that the agent has no clue about the environment and which actions are good during the first episodes, as the Q-table is full of zeros. Therefore, choosing the action with the highest reward from the Q-table makes little sense. We want the agent to explore the environment for a while before it exploits the knowledge obtained during exploration.

$$
\begin{aligned}
&MIN\_EXPLORATION\_RATE = 0.05 \\
&MAX\_EXPLORATION\_RATE = 1 \\
&EXPLORATION\_DECAY\_RATE = 0.015 \\
&EPSILON(\epsilon) = MIN\_EXPLORATION\_RATE + (MAX\_EXPLORATION\_RATE \\
&- MIN\_EXPLORATION\_RATE) * e^{(-EXPLORATION\_DECAY\_RATE * episode)}
\end{aligned} \tag{1}
$$

So what we aim for based on formula 1 is that the agent starts with a very high exploration rate (ϵ) in order to explore the environment randomly. The ϵ-value decays with each episode, making the agent exploit previous gathered knowledge over time. The EXPLORATION_DECAY_RATE parameter has to be tuned to find the right balance between exploration and exploitation. Therefore the decay rate parameter is dependent on the number of episodes it will run for. If the decay rate is too high, the agent won't explore long enough to learn the environment and tries to do sub-optimal exploitation. Likewise, If the decay rate is too low then the agent keeps exploring for too long and the agent will exploit the gained knowledge too late. The exploration decay rate we opted for allows the agent to have a high exploration rate in the first 200 episodes, or 20% of the total number of episodes, and after that it bounds to the minimum exploration rate of 0.05.

1.7.2 penalize going out of bounds

Whenever the agent performs an action that would make the agent go out of bounds on the map, the Q-table for that [state, action]-pair will be updated with a value of negative infinity. This ensures that the action will never be considered again during the exploitation phase.

1.8 Question 8

The agent seems to throttle randomly and then tries to stabilize. However, it does this in an uncontrolled manner (overshooting/undershooting) and therefore crashes every time. The agent does not learn to land safely between the flags.

1.9 Question 9

The agent seems to stabilize the rocket ship faster and is more willing to hover between the flags. However, it still does not land safely between the flags after 500 episodes.

1.10 Question 10

The agent tries to optimize the usage of the side engines while stabilizing the rocket ship, that is, minimize the usage of these engines. The agent uses the main engine quite a lot, meaning the rocket ship hovers in the air and doesn't try to land fast enough. It does not learn how to land within 500 episodes, and over the last 200 or so episodes no clear improvement can be seen. Therefore it did not improve compared to question 9. This could be due to our code being incorrect though, as we believe it should perform better than question 9. This means that the rocket ship does not reach the landing platform in time