Autonomous Agents

5 October 2012

Assignment 2: Single Agent Learning

By:
Paris Mavromoustakos
Georgios Methenitis
Patrick de Kok
Marios Tzakris

Introduction

In this assignment, we implement the learning scenario: the transition function is unknown to the agents, and so is the reward structure. We introduce model-based algorithms, which give agents the ability to learn high-reward policies while ignoring the model.
Our code is based on the previous assignment, on which we added the classes necsessary to implement learning algorithms. Moreover, we have explicitly used the 21 statespace environment representation which reduced our algorithms' runtime on the previous assignment.

Exercise 1

In this first exercise, we have implemented the Q-Learning algorithm with ϵ-greedy action selection, as described in chapter 6.5 of the Sutton and Barto book.
On each step of this algorithm, we chose an action a of state s, and observed the next state and reward we got from it. Then, we updated the Q-learning table according to the following update rule:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma max_{a'} Q(s',a') - Q(s,a)\right]$$

In this update rule, $Q(s,a)$ represents the value of the Q-learning table based on the previous step, while $Q(s',a')$ represents the observed state and possible actions, after action a is chosen. $Q(s,a)$ could be written as $Q(s_t,a_t)$ and $Q(s',a')$ as $Q(s_{t+1},a_{t+1})$.
α represents the algorithm's learning rate, whereas $\alpha = 0$ means that the Q-learning table will not be updated (the agent is not learning anything at all), while $\alpha = 1$ means that that the agent learns based only on its immediate past.
γ represents the discount factor, in the same way as described in DP algorithms, and $max_{a'}$ represents the action which is most likely to return the maximum reward among all the possible actions in state s'.

ϵ was given the value of 0.1 and the Q-learning table was initialized with 15.0 being assigned to all values.

The figures below indicate the performance of the predator over time, given different values for α , and different values for γ.

The graphs 1-5 clearly indicate how the predator learns faster while we increment the value of α. Figure 1 shows us that for $\alpha = 0.1$ the predator needs about 150 episodes to converge to the optimal policy. However, while we increase alpha, the predator will need less episodes for its Q-learning table to converge.

Increasing the value of α means that we consider the most recent information we observe, more important. A value of $\alpha = 0$ would mean that the predator is not learning at all, while $\alpha = 1$ means that the predator only obtains information from its immediately previous actions.

Exercise 2

In this exercise we have implemented Q-learning with ϵ -greedy action decision for different values for ϵ , using different initial values for the Q-learning table each time. We have used 3 different values for the initialization of the Q-table, a pessimistic one (5), a realistic one (10) and an optimistic one (15), to see how the agent behaves for different values of ϵ in each case.

We have chosen particular values for ϵ to test how, and what the predator learn to do. For $\epsilon = 0$, the predator will always choose the action for which the Q-learning table has maximum value, thus, the agent will converge to the optimal path really fast if the initial values of the Q-table are lower than the immediate reward of the absorbing state, but it will become explorative if the Q-table values are optimistically initialized.

If $\epsilon = 0.1$, the agent will try to exploit the action for which the Q-table has the maximum value with a probability of $1 - \epsilon = 0.9$. That means that our predator is trying to exploit more than it is trying to explore, and there is a small probability that it will not choose the action it considers optimal.

We have used the value of 0.8 for ϵ aswell, because in this case, each possible action for the predator might have the same probability to be chosen. For example, if the predator has 5 possible actions, there will be $1 - \epsilon = 0.2$ probability for the "optimal" action to be chosen, and each of the other possible states will also have $\epsilon/4 = 0.2$ probability to be chosen. As a consequence, in the best case scenario our agent will be moving randomly. If there are less than 5 possible moves, he will be most likely be choosing one of the "non-optimal" moves.

Finally, we have also included $\epsilon = 0.99$ in our experimentation, for the reason that this value leaves almost zero probability for the agent to choose the optimal action. So, what happens in this case is that the predator learns not to use the optimal action each time he has to take a decision, so as we see in the graphs, the number of steps it takes him to find the prey is increased dramatically as time goes by.

Exercise 3

Exercise 4

Exercise 5

Conclusion

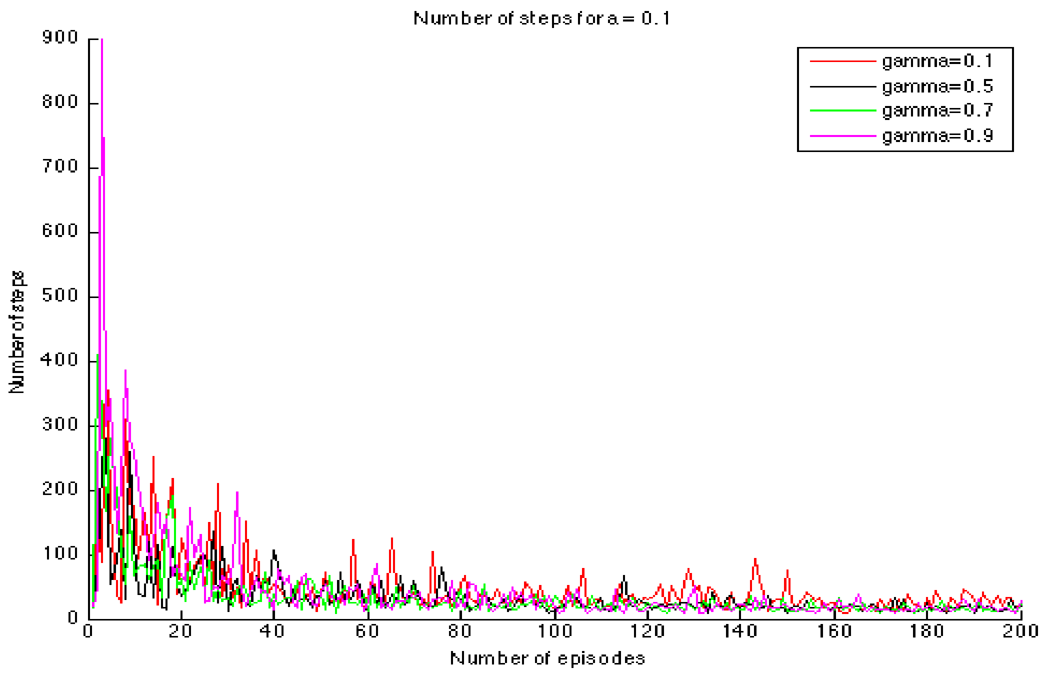

Figure 1: Performance of Q-Learning with ϵ-greedy for $\alpha = 0.1$.

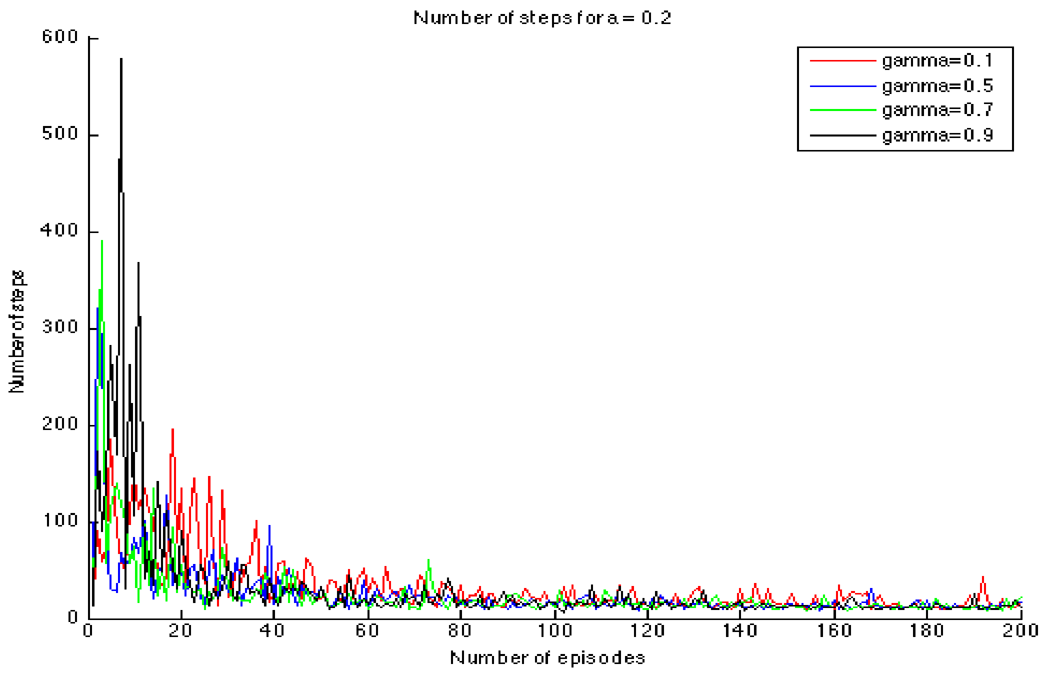

Figure 2: Performance of Q-Learning with ϵ-greedy for $\alpha = 0.2$.

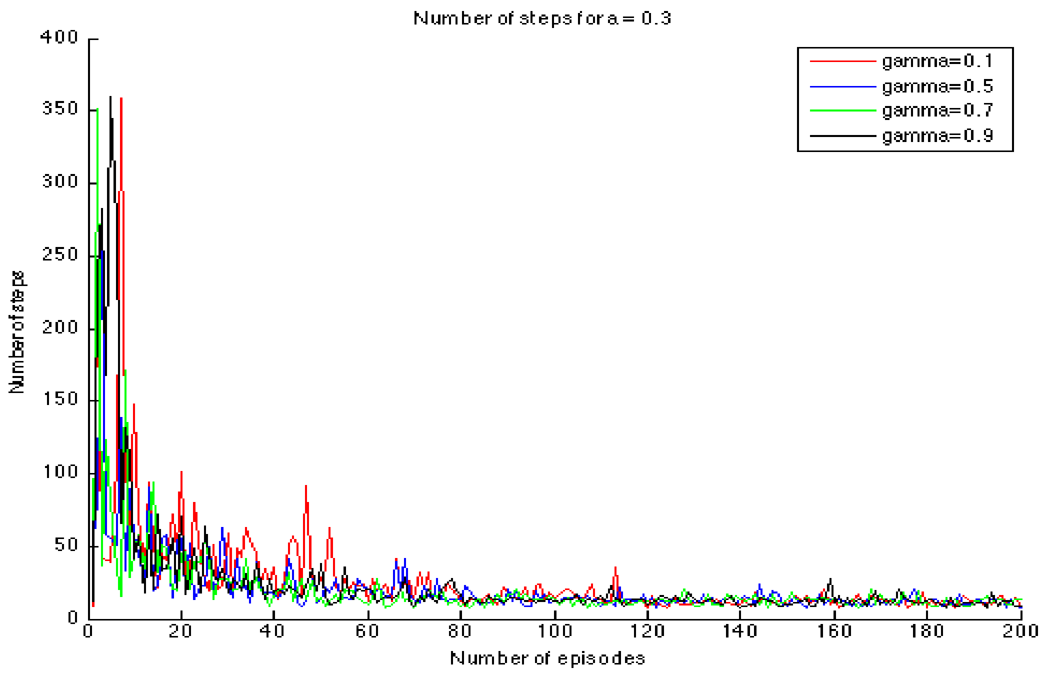

Figure 3: Performance of Q-Learning with ϵ-greedy for $\alpha = 0.3$.

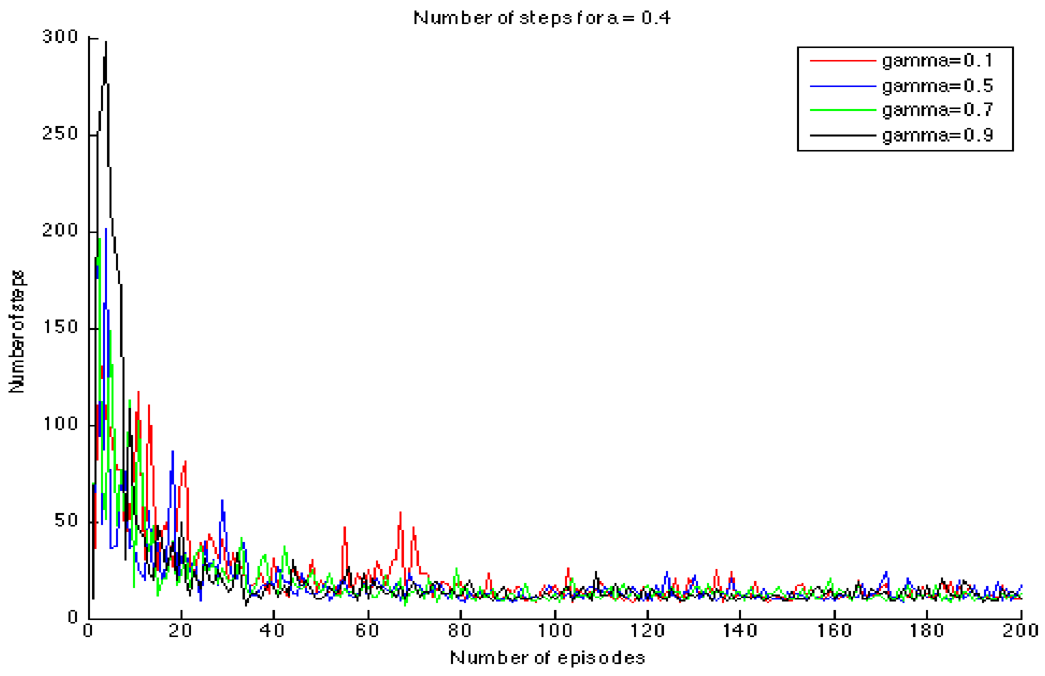

Figure 4: Performance of Q-Learning with ϵ-greedy for $\alpha = 0.4$.

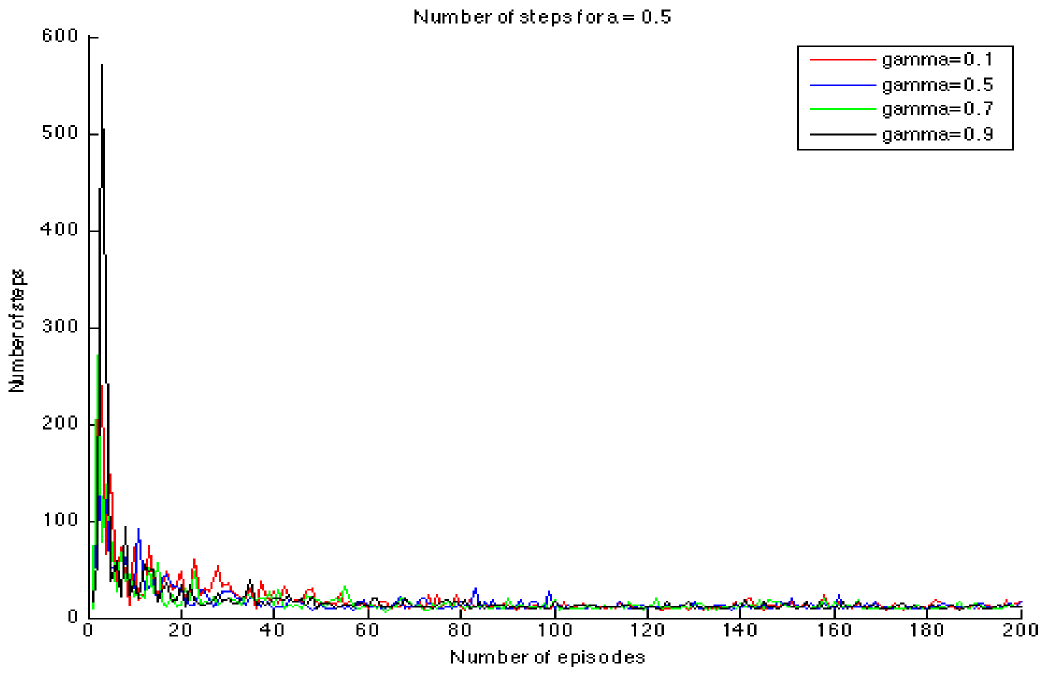

Figure 5: Performance of Q-Learning with ϵ-greedy for $\alpha = 0.5$.