CS7641 Machine Learning Assignment 4

Markov Decision Processes

Tao Peng, November 2016

1. Introduction

This assignment is to study the Markov Decision Processes (MDP) and use the reinforcement learning algorithms to solve those MDP problems. We will use two MDP with different difficulty and solve them using Value Iteration, Policy Iteration, and Q-Learning, and analyse the different behaviors of these algorithms under the two different problems. The tool used for this assigment was BURLAP [1], and the code from Ref. [2] was also used to create and analyse the MDP problems.

2. Problems

2.1 Description

The MDP problems chosen for this assignment are two variants of the classic Grid World problem. In this problem, there is a grid structure in which there is an agent who traverses the Grid World and try to reach the terminal state. There are walls and obstacles which can block the agent's path. If the agent hits a wall or an obstacle, it would stay at the same state.

The agents can go any of the four directions (north, south, east and west), but the actual direct is not always the same as planned. In this assignment, the probability for the agent to go in the same direction as intended was set to be 70%, and the probability to go in each of the other three directions was 10%.

Each state has a reward for the agents. The terminal state has the greatest reward, which was set to 50 in this assignment. The other states where there is no obstacle, has a reward of -1 each, which is acutally a punishment. These rewards urge the agent to get out of the ordinary states and move into the terminal states as much as it could. The purpose of the agent is to get to the terminal state with the greatest rewards.

In order to compare the behaviors of different algorithms under different circumstances, I created two MDPs using Grid Worlds with different difficulties, as shown in Figure 1. The easy one is a 4 × 4 Grid World with 12 states, while the hard one is a 10 × 10 Grid World with 82 states. In both the MDPs, the agent initially starts at the lower left corner denoted by a gray circle. The terminal state is the blue grid on the top right corner. The black grids are the obstacles.

2.2 Why interesting

Although the Grid World problem looks simple and theoretical, it is actually the abstract of some real life applications. The first one we can think of is that the Grid World is like a simplified map. Actually,

every ingredient of the Grid World problem has a corresponding thing in the real map.

In a real map, there are some places we can go, like the roads and public plaza, and some places we can not go, like the private houses, factories and water ponds. The places we can not go are the obstacles in the Grid World problem.

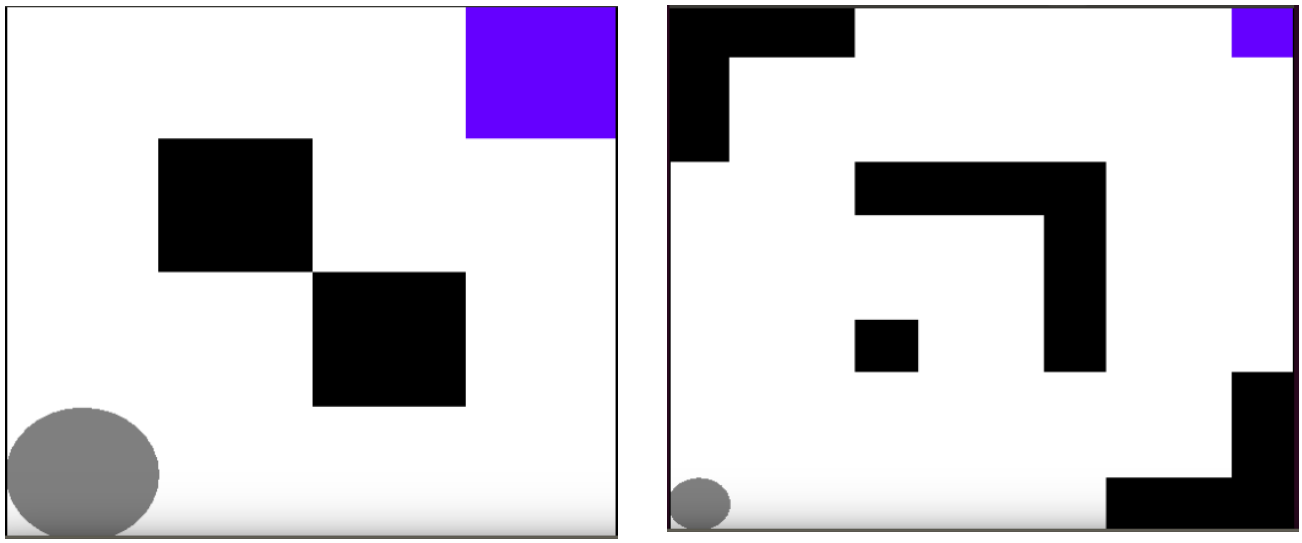


Figure 1: The two Grid World MDPs. The left one is the easy 4 × 4 Grid World with 12 states, and the right one is the hard 10 × 10 Grid World with 82 states. The gray circle on the lower left corner is the starting point, and the blue grid on the upper right corner is the destination.

In the real maps, different actions can have different rewards. For example, different roads may have different speed limits. A person driving from one location to the other location may like to drive in the quicker roads in order to save time. In this case, roads with higher speed limit has higher rewards. Another example of rewards is better road condition means higher rewards, because many people may like to drive in not crowded roads. The purpose is to get to the destination with greatest reward, in terms of a real map, within shortest time or travel on the most pleasant roads. These are some of the functions Google Map. I am not sure what algorithms are used by Google Map. But the study of Grid World may provide us with some inspirations.

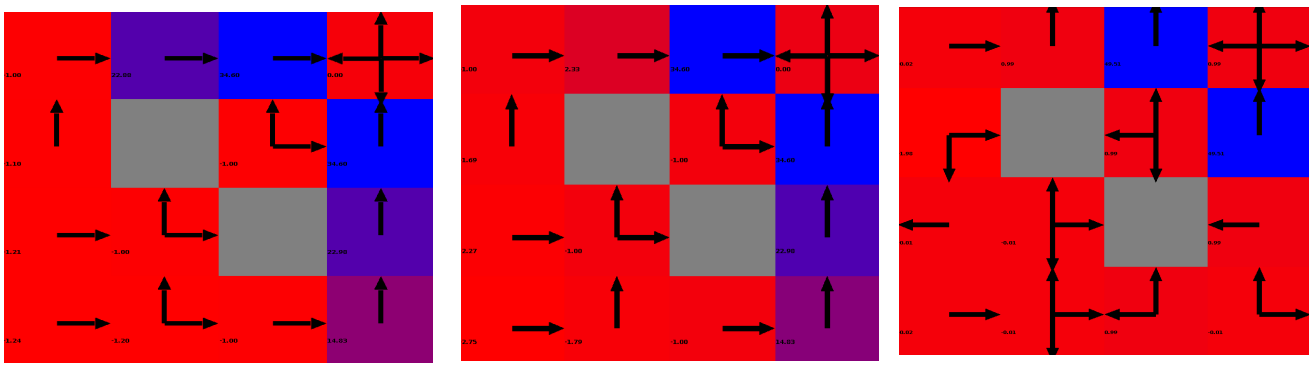
At a cross where different roads meet, the person may have to choose which way to go. Sometimes we know for sure, but sometimes we do not. When we hesitate, there is a non-zero probability for us to go in every way. This corresponds to the movement probability in the Grid World problem. This means that the Grid World problem can also be used to simulate the non-deterministic feature of a person driving in a complicated map. This may be used in the field of Artificial Intelligence where robots can be developed to simulate the hesitation of a human when making a decision.

Finally, the Grid World problem itself is a classic MDP. It is very valuable in theoretical studies of reinforcement learning and MDP. This problem is simple yet still have every element of an MDP: the states, actions, rewards, movement probailities, and policies. Studying this problem can help developing reinforcement learning and MDP algorithms.

3. Solving MDPs and analysis

3.1 The easy Grid World after 1 iteration

We first ran the Value Iteration, Policy Iteration and Q-Learning algorithms for one iteration. The results of the policies are shown in Figure 2. Arrows are the policies, showing which direction the agent should go. Blue grids reprent high utility, while red grids represent low utility. The utility values are also in every grid but too small to see.



Value Iteration Policty Iteration Q-Learning

Figure 2: Policies of running Value Iteration, Policty Iteration, and Q-Learning after 1 iteration for the easy MDP. Arrows are the policies. Blue grids reprent high utility, while red grids represent low utility. Gray grids are obstacles.

The first thing we can see from Figure 2 is that the Value Iteration and Policy Iteration gave almost the same policies, except one grid on the bottom. This expected because of the similarity between these two algorithms. For example, both used the Bellman Equation.

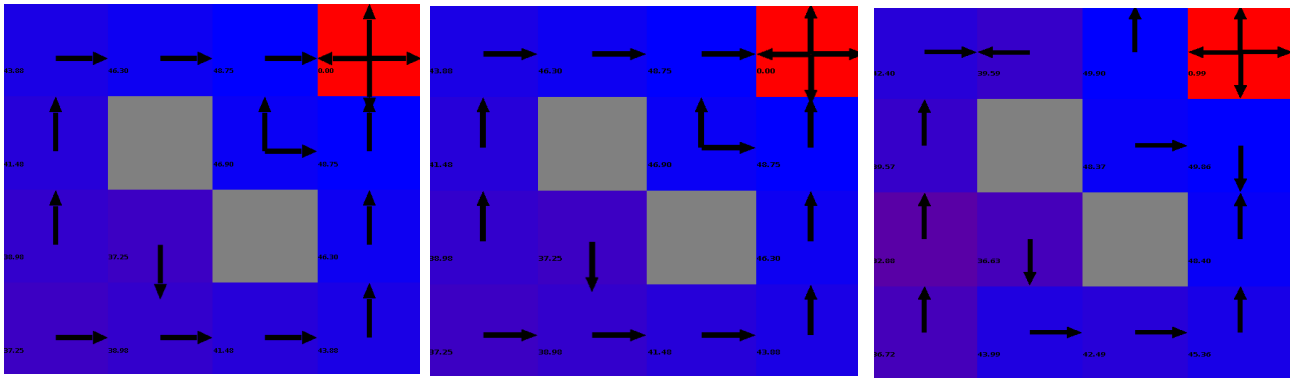
Even though Value Iteration and Policy Iteration agree well with each other, they are not the optimal policies. First, most of the grids are red, meaning low utility. Second, some of the arrows are not the optimal directions. For example, the arrows in the grid below the upper obstacle (also on the left of the lower obstacle) lead the agent to hit the obstacles, which are not wanted. The optimal arrows should lead the agent get away from the obstacles. The Q-Learning, however, gives very different policies and many of the arrows are not in good directions, leading the agents to hit the wall or the obstacles. All these not optimal policies of the algorithms are not surprising, because there was only one iteration.

3.2 The easy Grid World after 150 iterations

We then ran the three algorithms for 150 iterations. The results of the policies are shown in Figure 3. Now the Value Iteration and Policy Iteration completely agree, and every arrow points to the optimal direction. And all the grids have high utility values. This means that 150 iterations are enough for the Value Iteration and Policy Iteration to converge to the optimal solution.

However, the Q-Learning still does not find the optimal direction for all the grids. For example, the arrow in the grid on the left of the destination suggests the agent to go north and hit the wall, which is

apparently not good because the agent can go east and reach the destination in one step. The arrow in the grid on the left of that also makes the agent to go farther away from the destination, which is also not optimal. This means that 150 iterations is still not enough for the Q-Learning to get the optimal solution.



Value Iteration Policty Iteration Q-Learning

Figure 3: Policies of running Value Iteration, Policty Iteration, and Q-Learning after 150 iteration for the easy MDP. Arrows are the policies. Blue grids reprent high utility, while red grids represent low utility. Gray grids are obstacles.

To better study the how the algorithms converge, we show in Figure 4 how they behave after every number of iterations. We let each algorithm to run for some certain number of iterations, and then use the resultant policy to move the agents and count the number of steps and time needed for the agents to reach the destination. From the number of steps, we can see how the algorithms converge by looking at how the curves fall down to their stable values. We see that Value Iteration and Policy Iteration can converge with less than 10 iterations, but Q-Learning still seems not to converge even after 150 iterations. This is consistent with our conclusions from the previous figures. Value Iteration and Policy Iteration converge much faster than Q-Learning, I think this is because Value Iteration and Policy Iteration have the reward function and transition probabilities upfront so they can use them directly, while Q-Learning has to take more iterations to learn these information.

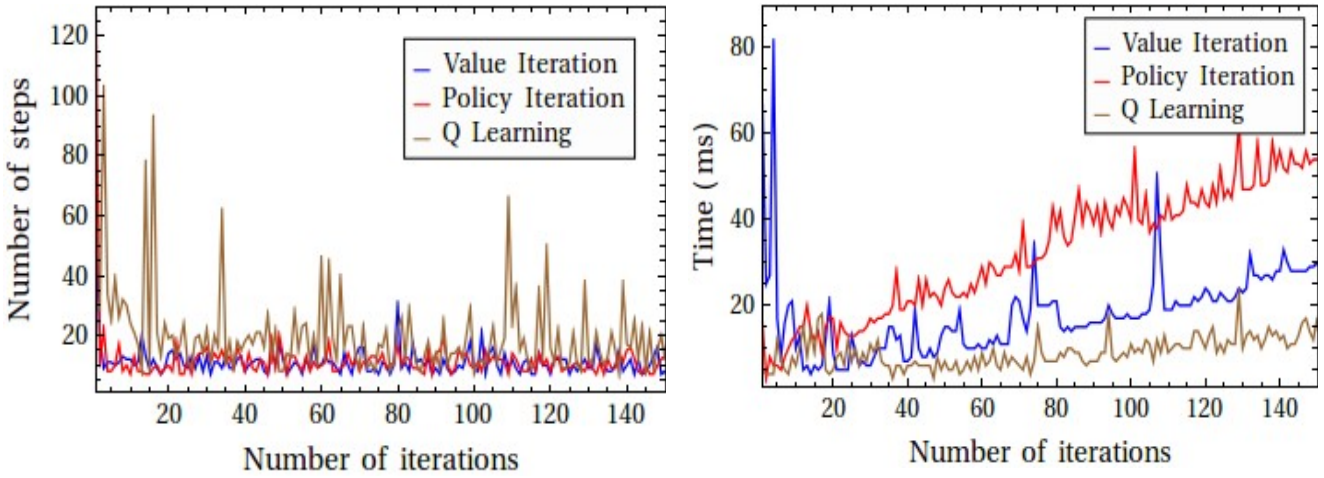


Figure 4: Number of steps of the optimal policy vs the number of iterations it took to obtain that policy (left) and time vs number of iterations (right) of the Value Iteration, Policty Iteration, and Q-Learning, for the easy MDP.

It is also interesting to look at the runing time of the algorithms. The first thing we notice is that the time of all the algorithms grows linearly with number of iterations. This is because every iteration essentially does the same amount of work. We also see that there is a clear seperation of the three curves. The Policy Iteration takes most of the time, while Value Iteration takes less and Q-Learning the least. I think this is because in each iteration, Policy Iteration has one more thing to do than Value Iteration, which is evaluating the utility given then current policy. Q-Learning takes the least time, because every iteration has less work to do, in every iteration there is no Bellman Equation, no transition probablilities, no need to evaluate the utility, etc.

The relation between rewards and number of iterations shown in Figure 5 can also show how the algorithms converge and can verify what we concluded earlier: Value Iteration and Policy Iteration converge in less than 10 iterations, while Q-Learning still tend not to converge after 150 iterations. All three algorithms starts from negative reward, this is also consistent with the redness of the grids in Figure 2, where there are only one iteration.

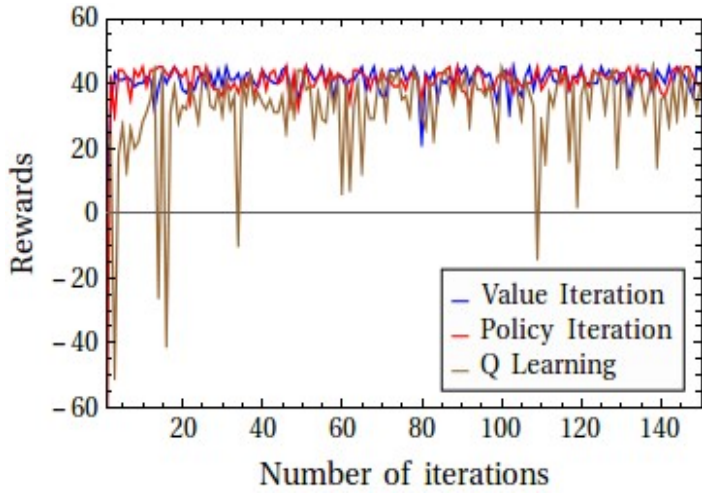


Figure 5: Total rewards gained for the optimal policy vs the number of iterations took to obtain that policy, of Value Iteration, Policty Iteration, and Q-Learning, for the easy MDP.

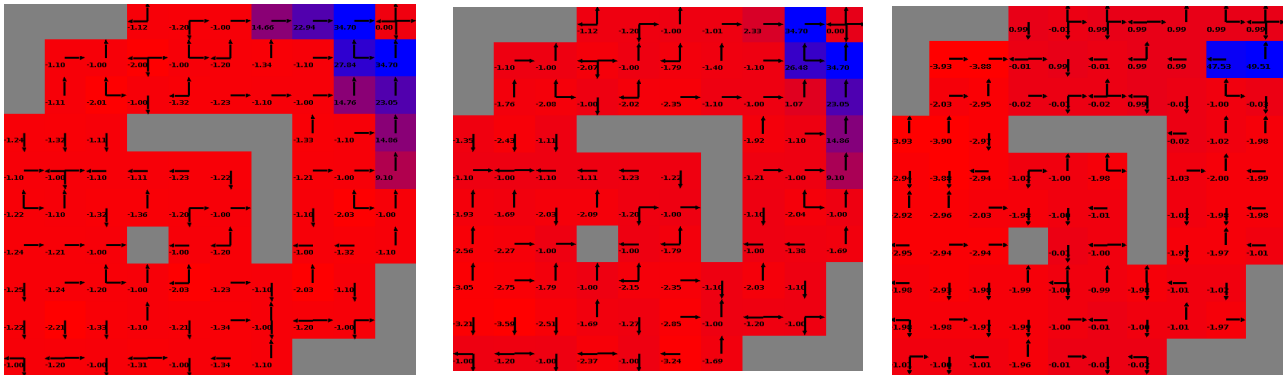
3.3 The hard Grid World after 1 iteration

We then study the hard Grid World problem and compare it to the easy one. Similar as in the case of the easy problem, we first run all the algorithms for one iteration. The resultant policies are shown in Figure 6. We can see that similar to the easy Grid World, Value Iteration and Policy Iteration agree on most of the grids, while Q-Learning is more different.

Altough Value Iteration and Policy Iteration agree on most grids, there are still some grids they disagree. For example, in the grid in the middle of the top row, Value Iteration wants the agent to go east which is the optimal direction, while Policty Iteration wants north towards the wall which is not good. Also, for the Value Iteration and Policy Iteration, there are many grids with not optimal directions. For example, around the single-grid obstacle in the lower left of the map, both Value Iteration and Policy Iteration give three arrows pointing towards the obstacle, which is not good because we normally want to get away from the obstacle. Another example is on the bottom of the map,

both the two algorithms give three arrows pointing down towards the wall, which is not good either because we want to move away from the way. Q-Learning works much worse, many arrows hit the wall or the obstacle, or points away from the destination, and there are less blue grids than the other two algorithms. All these behaviors mean that the three algorithms did not do a good job. This is expected because there is only one iteration. Q-Learning does much worse than the other two, because of lacking the rewards and transition probabilities and it needs more iterations to learn them.

To compare with the easy Grid World in Figure 2. We noticed that in the easy case, although there are a couple arrows pointing to the non-optimal directions, most other arrows are in the optimal directions. In the hard Grid World in Figure 6, there are more arrows in the non-optimal directions. This means that as the difficulty level increases, it is harder for all the algorithms to get the optimal policy.



Value Iteration Policy Iteration Q-Learning

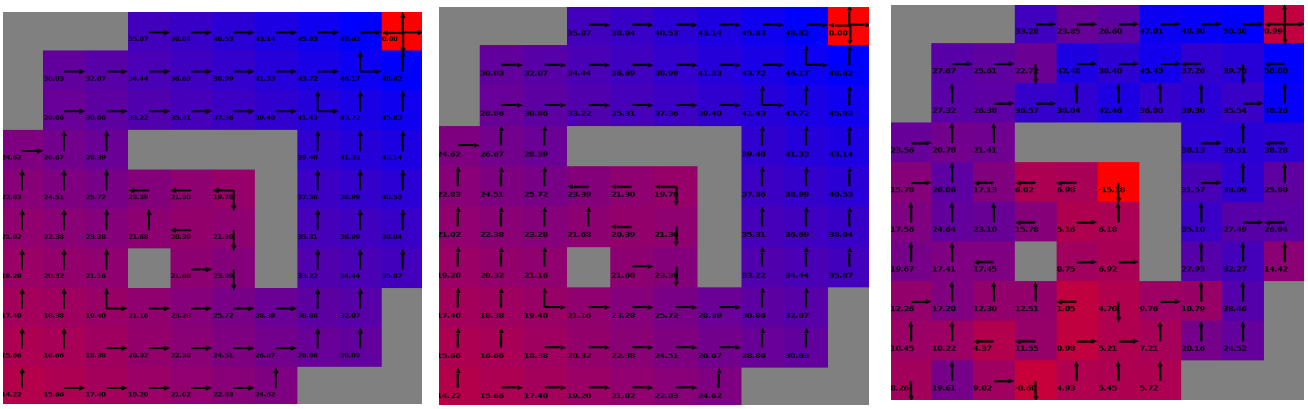
Figure 6: Policies of running Value Iteration, Policty Iteration, and Q-Learning after 1 iteration for the hard MDP. Arrows are the policies. Blue grids reprent high utility, while red grids represent low utility. Gray grids are obstacles.

3.4 The hard Grid World after 150 iteration

Since the more states the harder to converge, let's see what happens if we do more iterations. The results of running 150 iterations are shown in Figure 7. The first thing we notice is that now the Value Iteration and Policy Iteration almost have completely the same answer. And what is better is that they also give the optimal policy for every grid. So this means 150 iterations is already enough for both Value Iteration and Policy Iteration to converge on the optimal solution. If we compare to the easy problem in Figure 3, we see they are consistent, both problem have the optimal solution. One may notice some purple grids in the hard Grid World in Figure 7, while all grids in the easy Grid World in Figure 3 are blue. This is just because there is a longer way for the agent to travel to the destination give more states in the hard world, so it receives more punishment on the way.

The Q-Learning, however, still have not found the optimal policies. The most apparent to notice is the red grid embraced by the large obstacle in the center of the map. It has very low utility value. That is because of the wrong policy below this red grid (let's call the grid below the red one grid A): the arrows in the red grid and grid A point toward each other, making the agent bouncing back and forth in these two grids for a long time, util it escapes from this situation due to the 10% proability of moving in the un-intented direction. This is not good. There are many other grids with non-optimal directions, like those right above the big obstacle in the center of the map. All these behaviors of Q-Learning mean that

150 iterations is not enough for it to converge on the optimal solution. And the reason for this is again lacking of upfront information and more iterations needed to learn them. If we compare to the easy Grid World in Figure 3, we see that the hard Grid World in Figure 7 is much worse, due larger number of states.



Value Iteration Policy Iteration Q-Learning

Figure 7: Policies of running Value Iteration, Policty Iteration, and Q-Learning after 150 iteration for the hard MDP. Arrows are the policies. Blue grids reprent high utility, while red grids represent low utility. Gray grids are obstacles.

We then study more about how the three algorithms converge by looking at their behaviors in more different number of iterations. Similar as the easy Grid World case, Figure 8 shows the number of steps and time taken by the policy obtained by running the algorithms after the corresponding number of iterations. From the number of steps, we can see that Value Iteration and Policy Iteration converge in less than 10 iterations, while Q-Learning still has not converged even after 150 iterations. This is conclusion is same as the easy Grild World. So this means the difficult or the number of states in the MDP does not quite affect how the algorithms converge. What affects more is the algorithms themsevles. Q-Learning is hard to converge no matter what the difficult level the MDP is.

If we look at the time in Figure 8 and compare it to the easy Grid World in Figure 4, we see that the three algorithms take longer time than the easy Grid World. What is interesting is that the time taken by all three algorithms in the hard Grid World is roughly 8 times longer than those in the easy Grid World, which is almost equal to the ratio of the number of states of the two MDPs (82 states for the hard one and 12 states for the easy one). This is interesting because the time is proprotinal to the number of states, and is sort of independent from the shapes and locations of the obstacles. I think the reason for this is that in each iteration of the algorithms, it needs to look at all the states (as in the Bellman Equation) and the number of states it looks at is independent from the shape and locations of obstacles.

The other behaviors of the time is the same as the easy Grid World: Policy Iteration takes longest time, Value Iteration less, and Q-Learning the least. And they all grows linearly. All those behavior is again independent from the MDP difficulty level, and are more determined by the features of the algorithms themselves: Policy Iteration needs time to evaluate the utility value, and all algorithms do the same amount of work in each iteration so grows linearly.

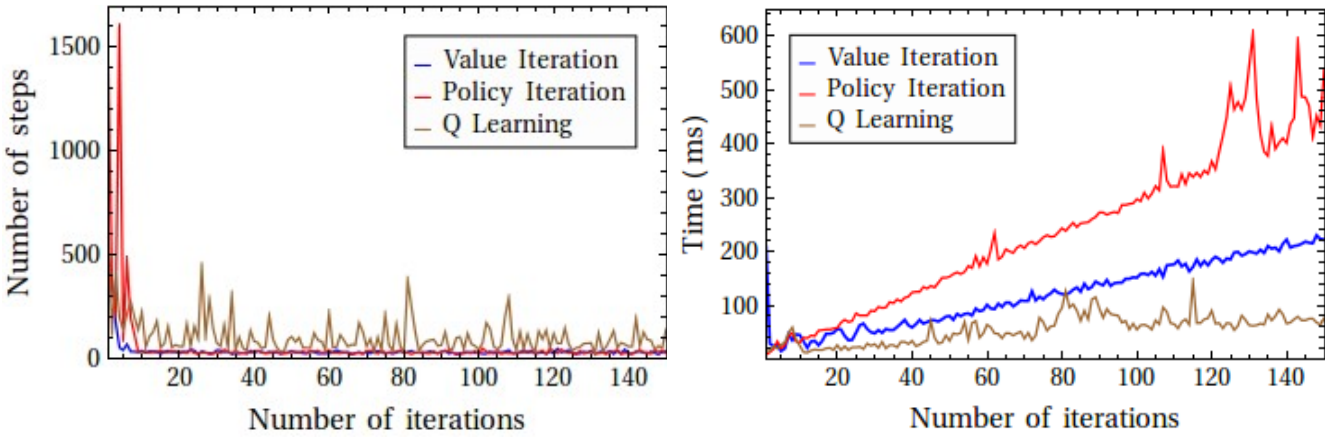


Figure 8: Total rewards gained for the optimal policy vs the number of iterations took to obtain that policy, of Value Iteration, Policty Iteration, and Q-Learning, for the hard MDP.

We then look at the rewards, as shown in Figure 9. The first thing we see is that it verifies our conclusion from the number of steps in Figure 8: Value Iteration and Policy Iteration converges in less than 10 iterations while Q-Learning has not converged even after 150 iterations. And if we compare with the easy Grid World in Figure 5, the first thing we see is that the rewards in the hard Grid World is more negative, that's because there are more states so more punishment to the agent. The other behaviors are the same as the easy case: Q-Learning did not converge.

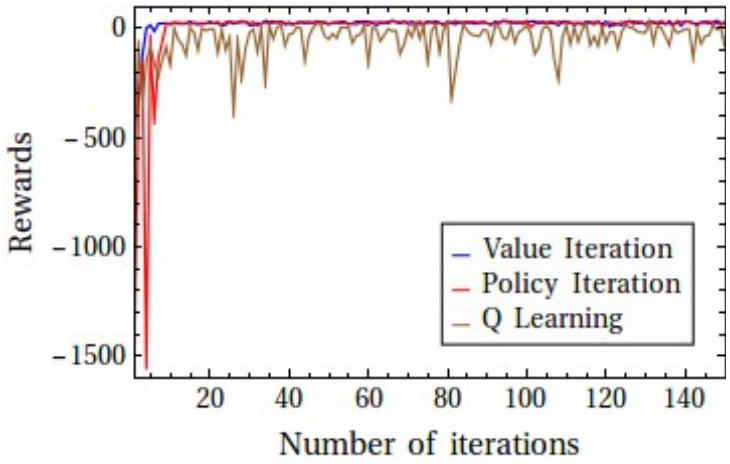


Figure 9: Total rewards gained for the optimal policy vs the number of iterations took to obtain that policy, of Value Iteration, Policty Iteration, and Q-Learning, for the hard MDP.

3.5 More about Q-Learning

Given that Q-Learning did not converge even after 150 iterations in both the easy and hard MDPs, we may be curious about how many iterations it needs to converge. Figure 10 shows the rewards after running the hard Grid World using Q-Learning after 20,000 iterations. We see that there are still large fluctuations and Q-Learning tends not to converge even after 20,000 iterations. My computer took more than 24 hours to run this and I do not have time to run for more iterations. So I do not know how many iterations Q-Learning needs to converge from Figure 10, but at least we know it is very hard to converge.

The difficulty to converge can be an important feature of Q-Learning and this is some thing we have to consider when we are choosing which algorithms to use. Although each iteration of Q-Learning takes less time than Value Iteration and Policy Iteration, as we saw from Figures 4 and 8, the total time Q-Learning needs to find the optimal policy is much longer than the other two algorithms. So if running time is the biggest concern, we may choose Value Iteration or Policy Iteration instead of Q-Learning.

However, since Q-Learning does not require upfront information of rewards and transition probabilities, if we do not have access to these information, we may have to use Q-Learning. I think this is fairly common in real life applications. For instance, when a robot is exploring some unkown area, it does not know the rewards (like the road condition) everywhere util it learns them by itself. Another example is when we are trying to get out of a maze, we will never know what is going to appear next. So in practice, Q-Learning essentially reflects the reality better because its assumption is exactly like what it is when we are exploring the unkown area and need to learn while trying to find out the best policy. Therefore, even though it takes much longer time than the other two algorithms, we have to use it in many situations.

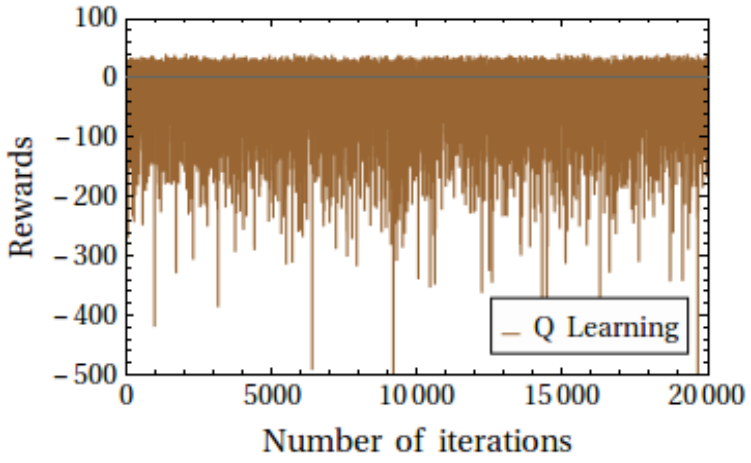


Figure 10: Total rewards gained for the optimal policy vs the number of iterations took to obtain that policy, of Q-Learning, for the hard MDP, with large number of iterations.

4. Summary

In this assignment, we created two MDP problems with different difficulty and solved them using Value Iteration, Policy Iteration, and Q-Learning. We also compared the behaviors between the three algorithms, and we see that Value Iteration and Policy Iteration are much faster to converge, and Q-Learning needs much more iterations to converge. Value Iteration and Policy Iteration can often converge to the same solution quickly. If we compare between the easy and hard MDPs, we see that it does not change the conclusion. Q-Learning still needs many iteration to converge and this is due to the algorithm itself, not the MDP problem. However, Q-Learning is useful in the cases when we do not have the information of the rewards and transition probabilities.

The running time is that Policy Iteration takes the longest time, Value Iteration takes shorter time, and Q-Learning takes shortest time. And all time running time of all three algorithms grow linearly as the number of iteration increases. If we compare between the easy and hard MDPs, we see that the running time of all three algorithms are proportinal to the number of states. This is because in each iteration they have to visit all the states.

From this assignment, I had a better understanding of the MDP problems and the features of different algorithms, and learned how to choose which algorithm to use in different situations.

References

[1] http://burlap.cs.brown.edu/
[2] https://github.com/juanjose49/omscs-cs7641-machine-learning-assignment-4