MP1- part2 yangsu4

1. (a)

   I observed that for the first step, the robot took several random steps. But he would get slipped and a negative reward. And then, the robot will keep staying in the left of the slippery area, send no package and get no positive rewards.

   The reason is that Epsilon is 0 and the robot would choose his action completely on the QValue and reward he get. There will be no random action unless there is a tie between two action's values. But when robot tried slipper area, he might get some negative reward, so the robot keeps staying left without trying to go cross the slippery area, in case of the probable penalty.

   (b)

   I observed that at first, the robot get a number of negative reward, but time goes on, the reward became positive. Although the value of reward goes up and down, but the value keeps positive most time.

   The reason is that Epsilon is 0.1, and there is a probability for the robot to take random steps without consideration of the qValue matrix. Therefore, the robot choose some different steps from (a), even though there are areas which might cause slippery. And when he send two package successfully, he will learned that there is a big reward to do this. So the robot could deliver package successfully several times.

   (c)

   I observed that most of the rewards are negative, and most time the robot took steps without consideration of the qValue matrix, some time he even chose to take steps that may cause a great negative reward.

   The reason is that the Epsilon is too big to allow the robot take steps with his own knowledge—qValue Matrix. Most time the robot take steps randomly, so he get negative rewards most time.

2. (a)

   Most time the rewards are negative.

   The reason is that the thief is moving but the robot didn't know it. Since the thief is moving, the qValue of every state could me great different when there was a thief or not. And qValue matrix is not that useful in this situation, so the robot get many penalty.

   (b)

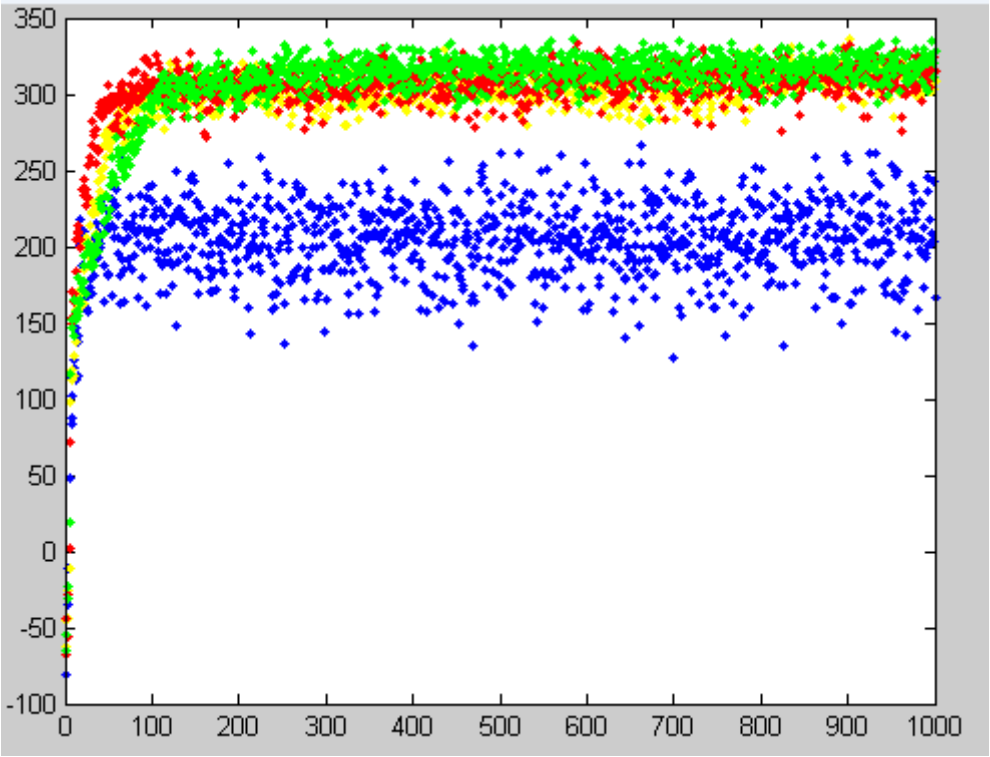
   Although at first there is some negative rewards, but the robot get positive rewards almost all the time. That is the most difference between (a) and (b). And the positive rewards are pretty stable. The rewards are much higher than that of world without thief.

   The reason is that the robot knew the thief. Since the robot knew thief, it could avoid the thief and take other actions, so he got positive reward. Because the slippery areas are less than that of world without thief, the rewards are higher than that of world without thief.

   (c)

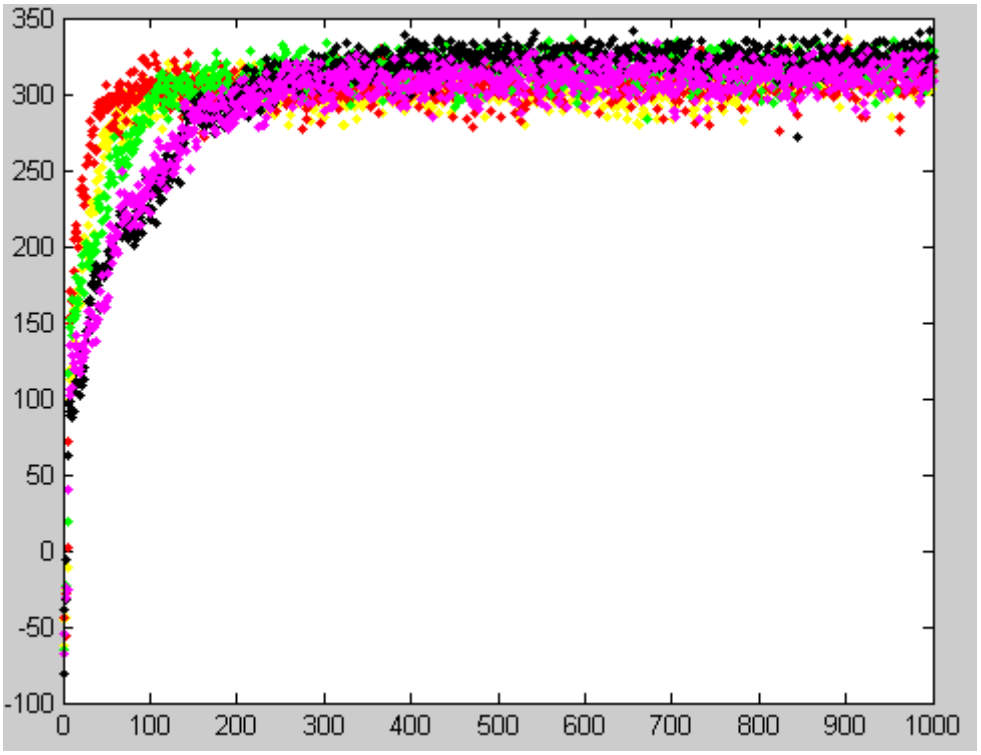
   The best learning rate and Epsilon is 0.15 and 0.01.

Firstly, I make the Epsilon stable(Epsilon=0.025), and change learning rate, as we can see in the picture below(2-1), when rate is 0.1, 0.15 and 0.9, the plot represent them are yellow, red and blue (by matlab). So rewards are highest when learning rate is around 0.15. I also take experiment about rate=0.05, rate=0.2, rate=0.5, but the output are all smaller than that of 0.15.

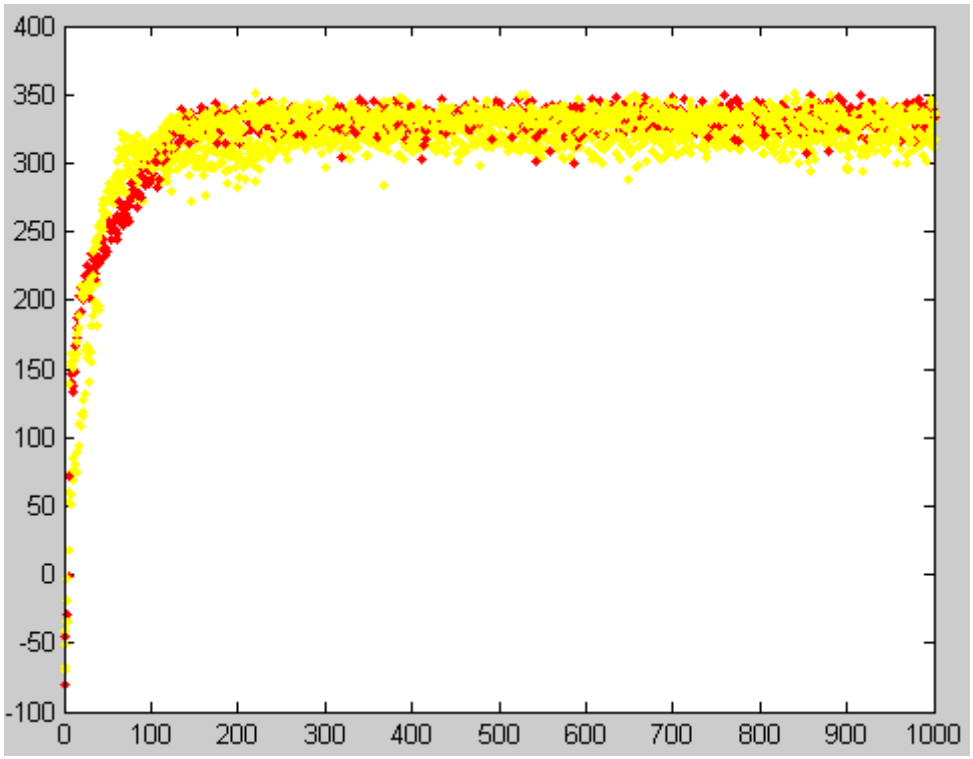


2-1

Secondly, I make rate stable at 0.15, and changed the Epsilon. When Epsilon=0.025, the red plot is the picture(2-1). And when Epsilon=0.01, 0.015， 0.02, the pictures of plot are below.(0.01=black;0.02=green in 2-2) (0.01=red; 0.015=yellow in 2-3)



2-2

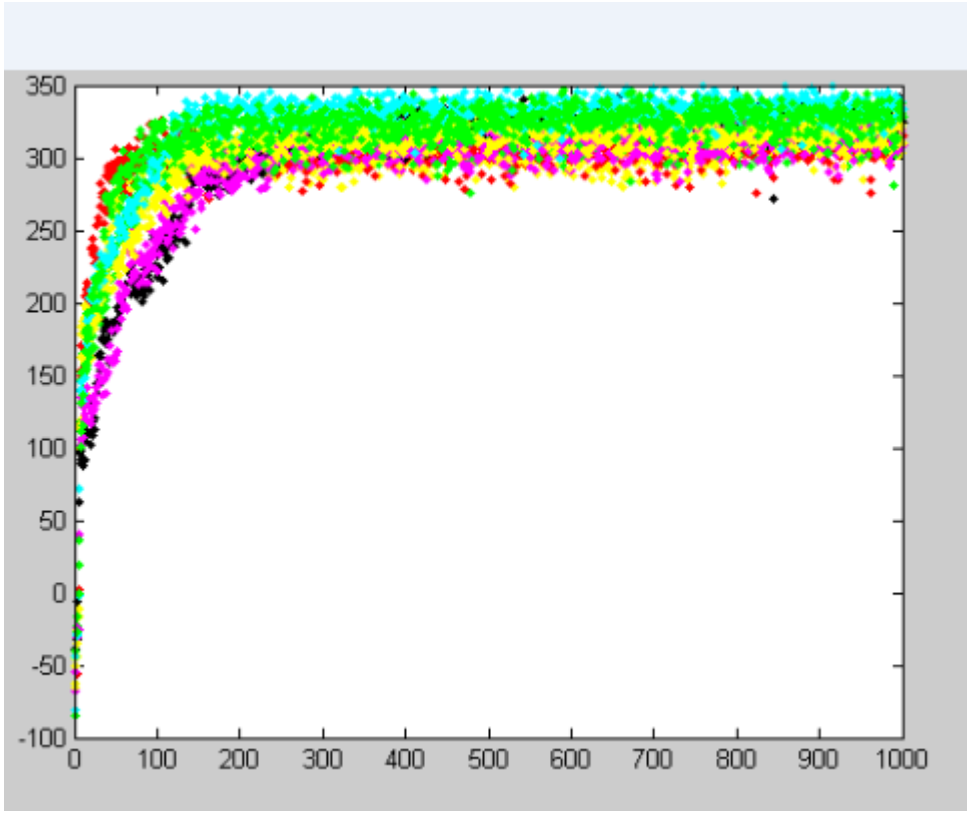


2-3

I also changed Epsilon to 0.2, 0.5 and some bigger value, but the reward are smaller than that of 0.01. And I found that when learning rate is 0.15, Epsilon is 0.01, the reward is highest.

Thirdly, to make sure that there is no bigger reward around rate=0.15 and epsilon=0.01, I changed in a smaller range to observe data. For picture 2-4, I have tried rate=0.18, rate=0.16,

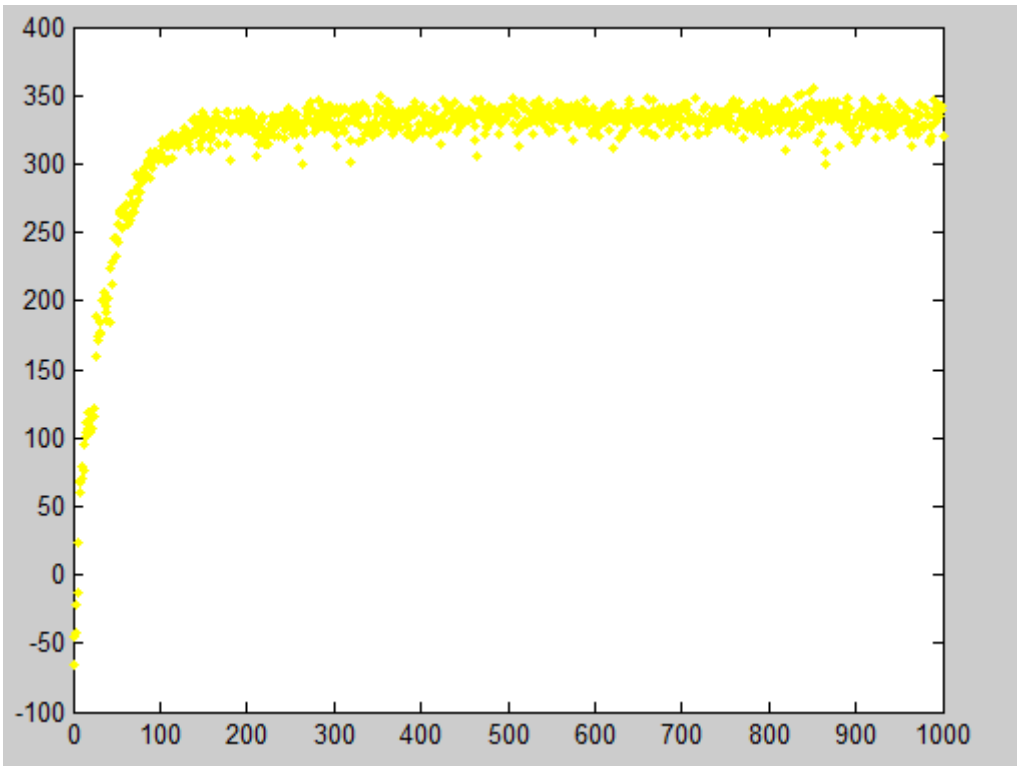
rate=0.13. (0.15=bule 0.18=green 0.16=yellow and 0.13=pink) From this picture, I found that when rate=0.15, we can get the highest value.



2-4

Therefore, the conclusion is that we could get the highest value when rate=0.15 and epsilon=0.01.

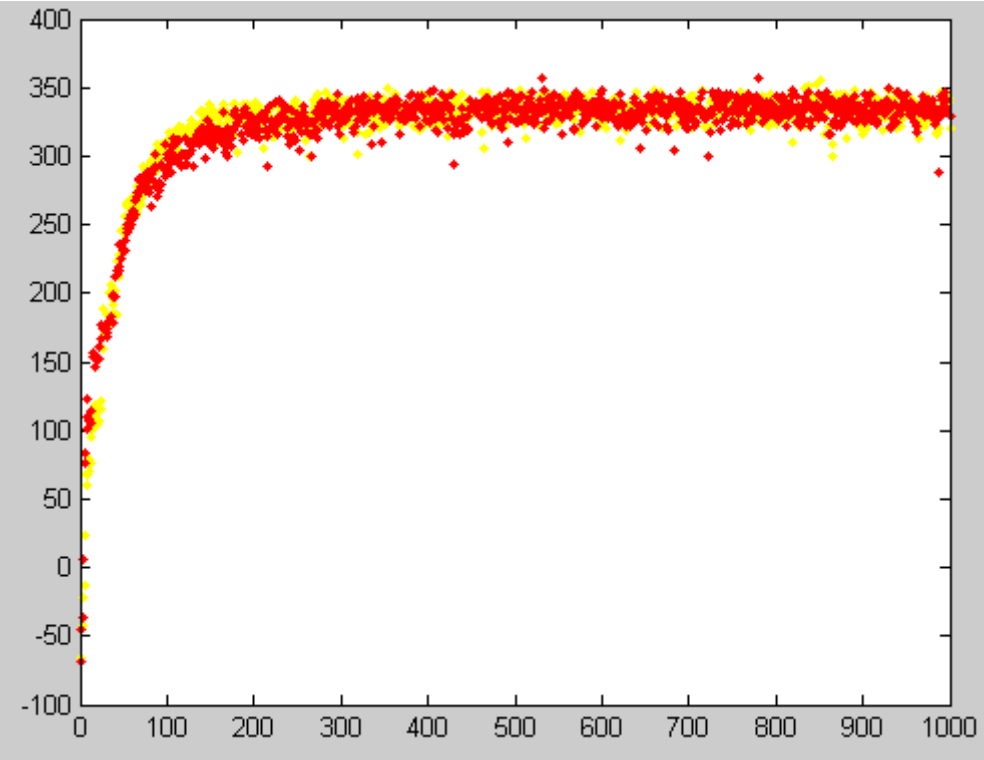
3. The picture of expected discounted reward at the end of each episode are in picture 3-1:



3-1

I noticed that at first the reward increased exponentially, and when it get a higher value, it keeps stable without much change, and the high value will keep until the simulator end.

The average of reward after each episode is in the picture 3-2. I observed that the rewards are more compact than the first time to running. And the average value has less very high or very small points than every time we run the simulation, instead, it is more stable.



3-2

The reason that it become more compact and stable is because the average reward mitigated the possibility and randomness that we got a very different reward. It represent the situations that appear most possibly. We could use average to find the best action that the robot should take in every state to get the highest value.