CS440: Artificial Intelligence **Fall 2014**

Machine Problem 1 - Part 2

Gaston Gerchkovich *Handed In: 9/28/2014*

I Will be using the Agent I made in part 1, because I know my implementation, making it easier to understand the results and modify the parameters as needed.

1. a. The obtained reward is 0 and the policy seems to always be going up and staying in the top cell of the first row. This is because there is no randomness (unless there is more than one optimal action). Consequently, given there exist a slippery barrier (a column full of negative rewards) and the agent will not try a random path just for the sake of learning, he will get stuck in the first half of the map (Reward 0 is safer than a possible negative reward).

   b. The existence of a minimum randomness allows the existence of some exploration and therefore unbiased learning. This leads to sooner or later finding a better policy than before. In average, the agent gets rewards between 300 and 50. The big variance is because the rewards are non-deterministic, and so is the result.

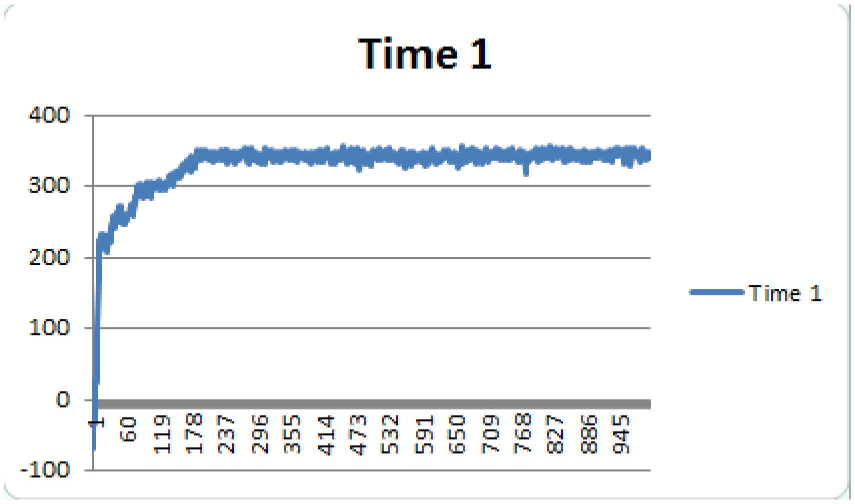
   c. Using a big ϵ, like 0.5 is good for exploring but makes it less optimal. By increasing epsilon we increase the randomness of our agent. Since the board opportunities for positive reward are really little compared to the negative rewards, the more random the agent, the more probable he will get negative rewards. Therefore as expected, the agent's rewards seems to usually be in the range of -50 to -10.

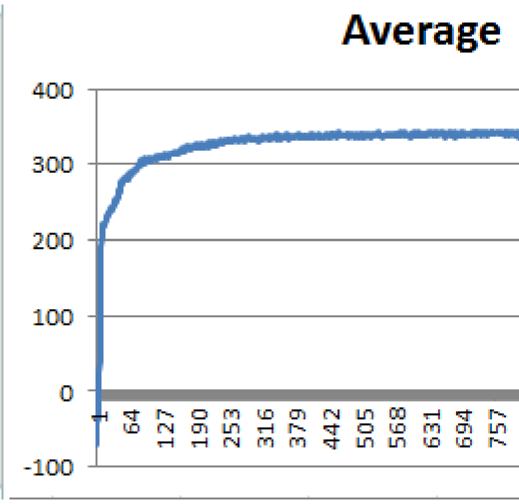
2. a.The agent gets negative rewards on average in the range of -20 to -10. This is because crucial information such as the existence of an opponent agent is unknown. With this unknown information, the agent sees it as a random high negative rewards in some unpredictable cells.

   b. Now with this known information on average the reward ranges between 250 and 300. By knowing the thief current location, after enough learning the agent can figure out when it's safe to go across "the thief column" making the path mostly completely safe.

   c. I did binary search for the best reward average as a function of ϵ and seems to be that 0.005 gives a really good reward average range (320 to 340).

3. The expected reward seems to be a logarithmic function of the quantity of previous episodes. When averaging the results over 10 independent runs, this function seems to have less error margin and be even more a logarithmic function. Based on the graphs and the fact that is a logarithmic function, we can say that after N episodes the improvement in the reward is not really meaningful (in this case $N \approx 600$).





(For more specific values, please look at attached excel)