Margaret Stutz
9/27/14
CS440
MP1 Part 2

1. WorldWithoutThief
   a. If you run the simulation multiple times with epsilon set to 0.0 you will see that almost every time all the episodes have a reward of 0.0, besides the first episode, which has a reward of -0.8. Occasionally the first episode will have a reward of -0.9 and the second -0.5. The reason for this low reward is because there is no randomness. In the first episode the robot moves around and learns where the slippery spots are that can give it a negative reward . Then in the following episodes the robot doesn't want to move anywhere that will give it a bad reward, so it stays in the same area and never get's any reward (never delivers any packages).
   b. With epsilon set to 0.1 the first 20 or so episodes have negative rewards, slowly getting closer to 0. Since epsilon is 0.1 this allows for some randomness so the robot is randomly forced to move in a direction even if it may receive a bad reward, unlike the previous example. As the robot learns where the slippery spots are the rewards increase rapidly after around episode 20. However because of the randomness there is not a steady linear increase in rewards. The rewards fluctuate, with the occasional negative or low reward among rewards of 100-200, because the robot may randomly move to a slippery spot, drop the package, and cause the reward to be less.
   c. With epsilon set at 0.5 almost every single reward is negative (fluctuating between large and small negative numbers); there are very few rewards ever above 0.0. The reason for such low rewards is the high epsilon value, which cause a lot of randomness. Half of the actions the robot takes will be random, so the robot will randomly go into slipperly spots, drop packages, and get low rewards.
2. WorldWithThief
   a. In the world with a thief where epsilon set to 0.05, and the robot is not aware of the position of the thief the majority of the rewards are negative, once again. These negative values also fluctuate, like the previous simulation, however the rewards are typically not as low (not as large of negative values). The negative rewards in this case are largely due to the robot running into the thief, and less due to randomness because epsilon is only 0.05 (though because there is some randomness there are times the reward of an episode is lower because of the robot randomly moving into a slippery spot and dropping a package, as well as running into a thief and getting it's packages stolen).
   b. When epsilon is set to 0.05 and the robot is aware of the position of the thief, the first 6 or so episodes have negative rewards, as the robot learns where the slippery spots are. Then the rewards drastically increase to the 100s, and after about 20 episodes all the rewards are in the 200s. The reason for this large increase and consistent high rewards is because the robot has learned the best actions to take to avoid slippery spots and it knows how to avoid the thief,

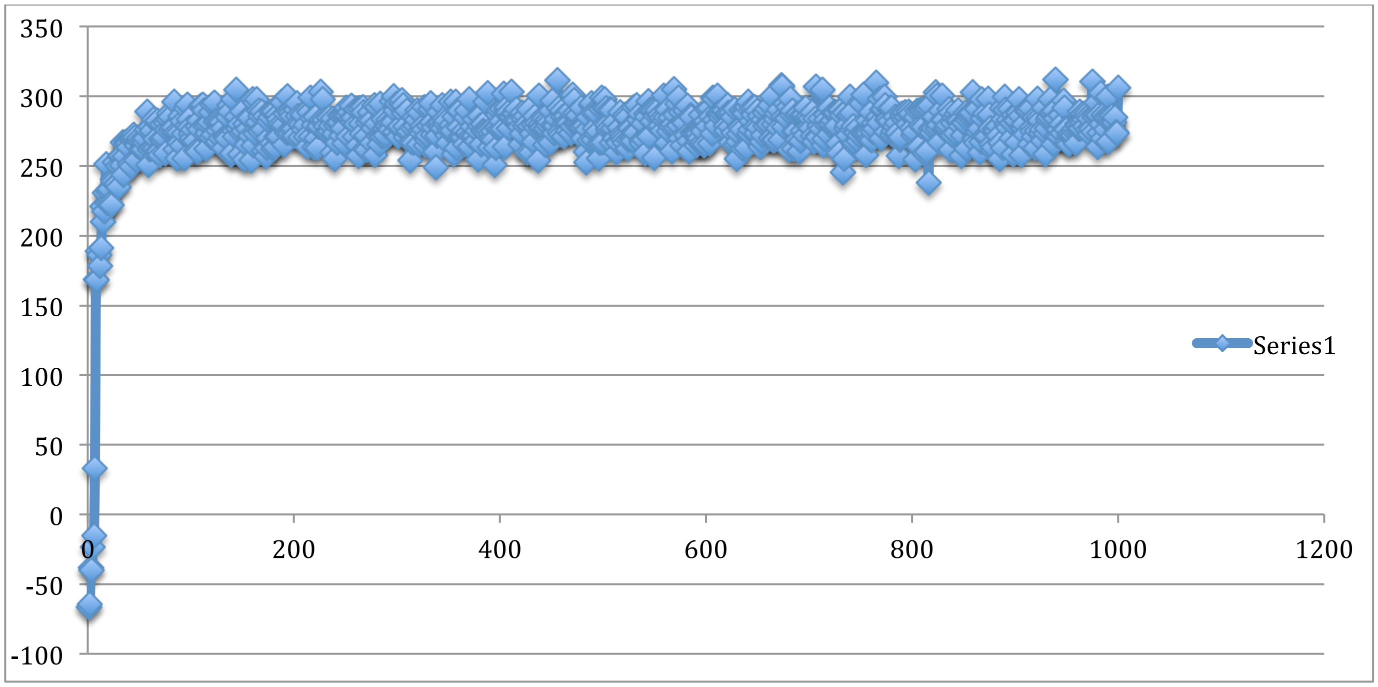
unlike the last simulation. So the only bad rewards would be due to randomness (the robot randomly moves to a slippery spot and drops it's packages), but since the randomness is only 0.05, this won't happen that often and therefore won't have a drastic negative affect on the total reward of each episode.

c. To search for the best learning rate I first changed the learning rate and kept epsilon constant at 0.05. I graphed the rewards earned from setting learning rate to 0.1, 0.5, and 0.8. I can compare the rate of convergence between these three by looking at when (which episode) each different learning rate reached their highest reward. The highest rewards for these learning rates were:
    - 311.5 at episode 456 (rate = 0.1)
    - 289.5 at episode 569, (rate = 0.5)
    - 245 at episode 842 (rate = 0.8)

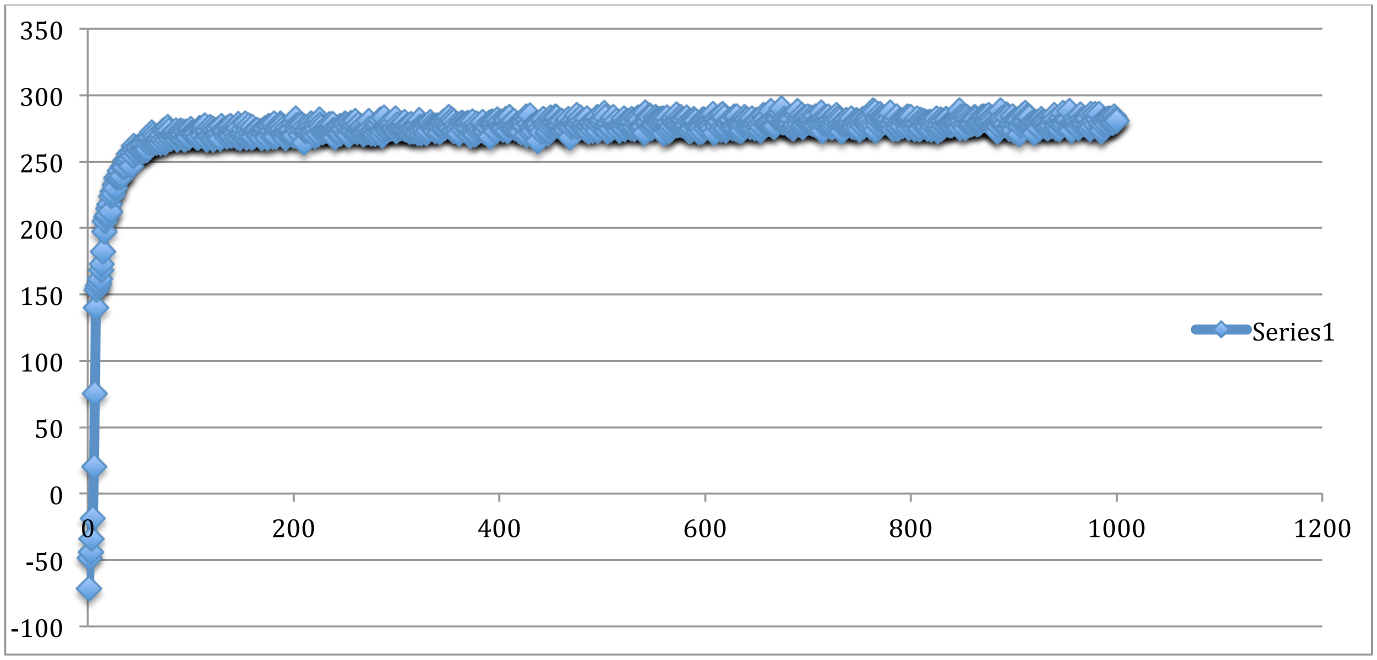
      It's clear from these values that the higher the learning rate the lower the value of the highest reward, and the highest reward will be reached later. The higher learning rates also have lower rewards per episode on average. So I concluded since we want higher rewards, it is best to keep the learning rate at 0.1. Then I changed the epsilon values to 0.05, 0.2, 0.6 (keeping learning rate at 0.1) and compared the rates of convergence again. The highest rewards for these learning rates were:
    - 311.5 at episode 456, (epsilon = 0.05)
    - 149.5 at episode 179 (epsilon = 0.2)
    - and -73.5 at episode 6. (epsilon = 0.6)

      Even though when epsilon is at 0.2 it reaches it's highest reward faster than the rest (at episode 179), that highest reward (149.5) is much smaller than the 311.5 reached when epsilon was 0.05. The smaller epsilons also have higher rewards per episode on average. So I can conclude that, since we want high rewards, a smaller epsilon, around 0.05 and a smaller learning rate, around 0.1 gives the highest rewards and optimal policy.

3. Using epsilon = 0.05 and learning rate = 0.1, I simulated my agent once and notice that the first 8 episodes have negative rewards, which makes sense because the robot is learning where the slippery spots are that will give it a bad reward. The rewards drastically increase (now the robot has a good knowledge of the slippery spots and how to avoid them) into the 100s, 200s, and early 300s. There is some small fluctuation in these rewards because of randomness, or bad luck (robot randomly moving in a slipper spot). You can see this fluctuation on the graph. Here is a graph of this first simulation:



Then I repeated this simulation 9 more times and plotted the rewards for each episode averaged over the ten total simulations. The graphs look pretty similar for the first few episodes, when the robot is learning where the slippery spots are. However, there is significantly less fluctuation in reward value for the majority of the episodes in the averaged graph. You can see this in the graph; there is almost a straight line in the averaged graph. This is because when you take the average of the rewards for each episode there is less bad luck and the rewards become closer to the same value. Here is the graph of the average rewards per episode:


Series1