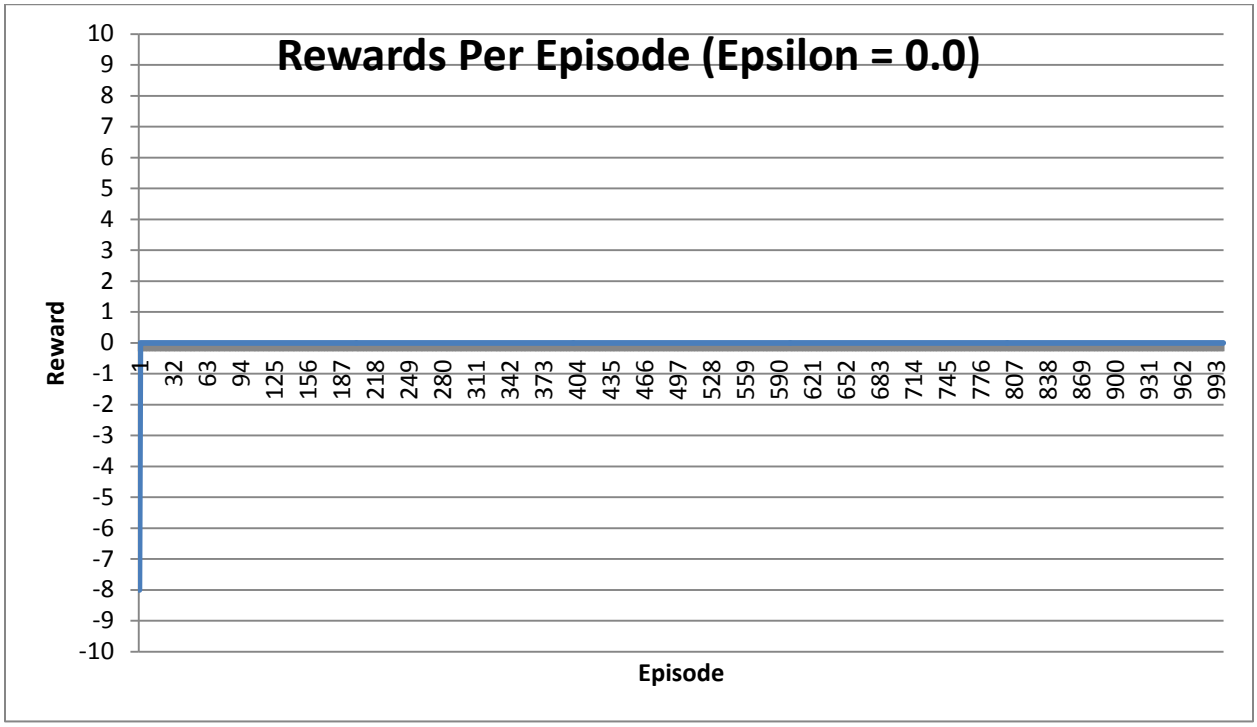
Robert Gazdziak

CS 440 FALL 2014

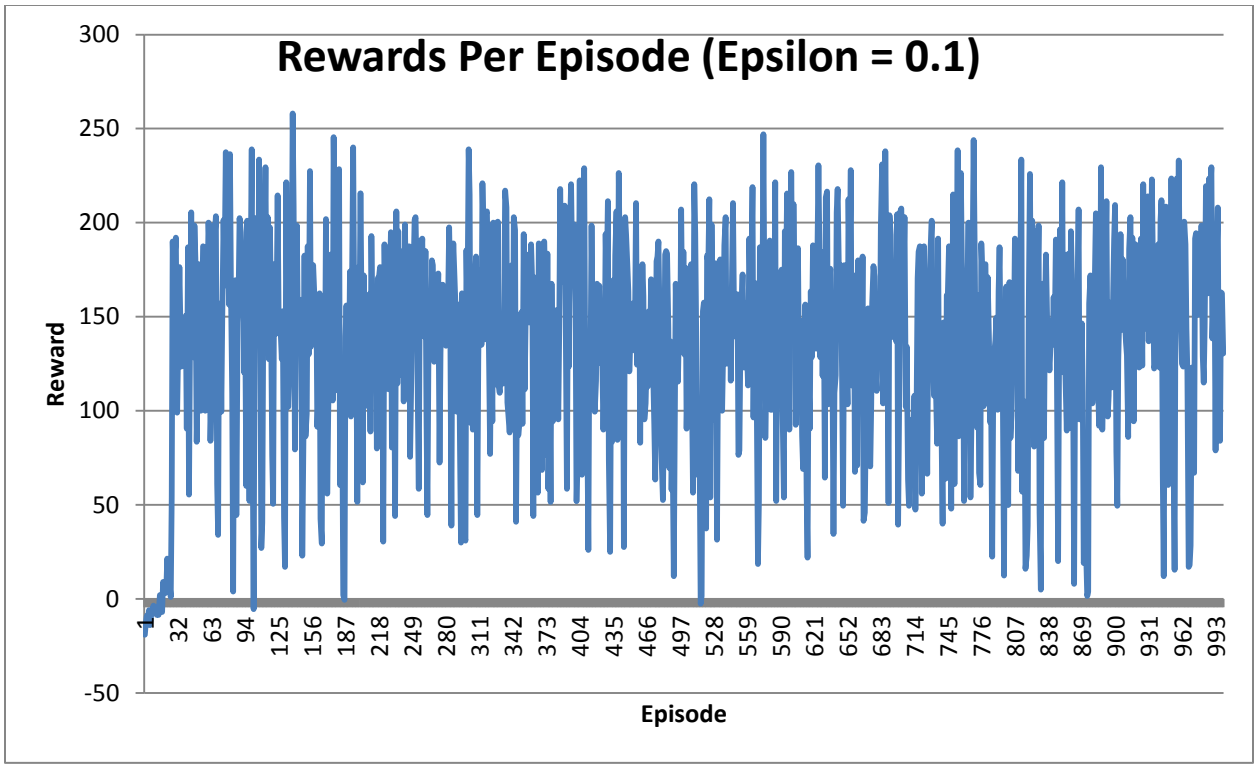
MP 1.2

1. World Without Thief
    a. After setting epsilon to 0.0, I observed this graph:



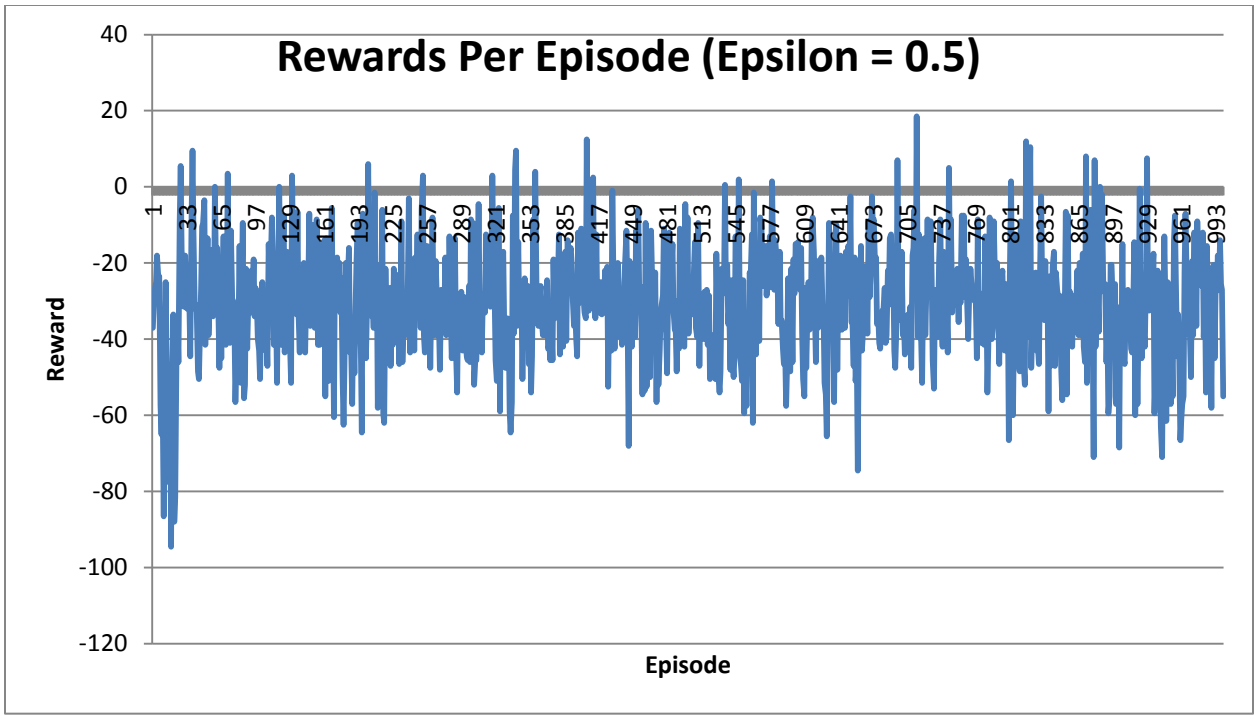
As we can see from this graph, the only reward that is non-zero is the very first episode. Every other episode has a reward of 0. This makes sense because we made the Learning Agent run only on the best path it knows; it will never take a path it doesn't know if there is one where it knows the reward to. The first episode the Agent doesn't know any of the paths so it chooses randomly where to go. But after that it realizes that its safest known route is just to sit where it starts and never move.

    b. After setting epsilon to 0.1, I observed this graph:



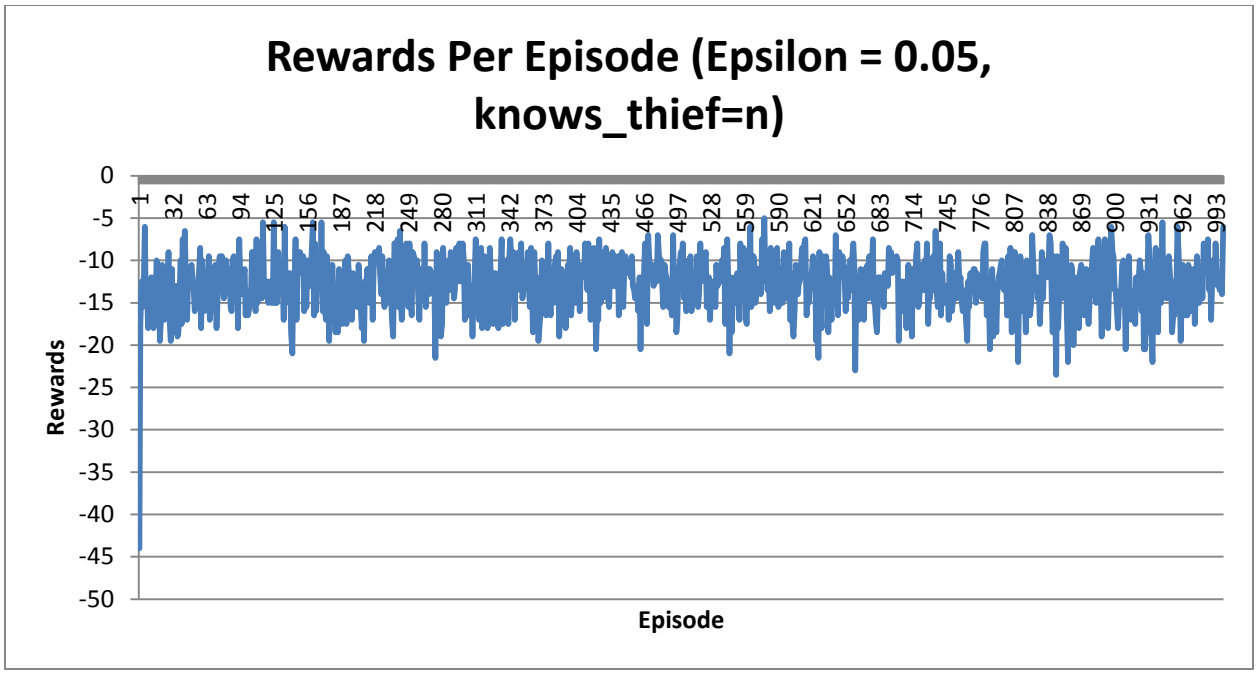
As we can see the first reward is nearly the same due to the Learning Agent not knowing where to go. Once the learning agent runs through a few episodes it learns the best path possible. This is why the average reward after about the first 30 episodes is 150 compared to the 0 from part a. This is because the Learning agent has some form of random movement (10% chance of random movement) so it learns all of the best possible paths and doesn't just stick to what it knows. The reason why the rewards aren't constant is because of the random factor, although it helps find the best rewards, it also can make the Learning Agent randomly move to a spot of negative rewards.

c. After setting epsilon to 0.5, I observed this graph:



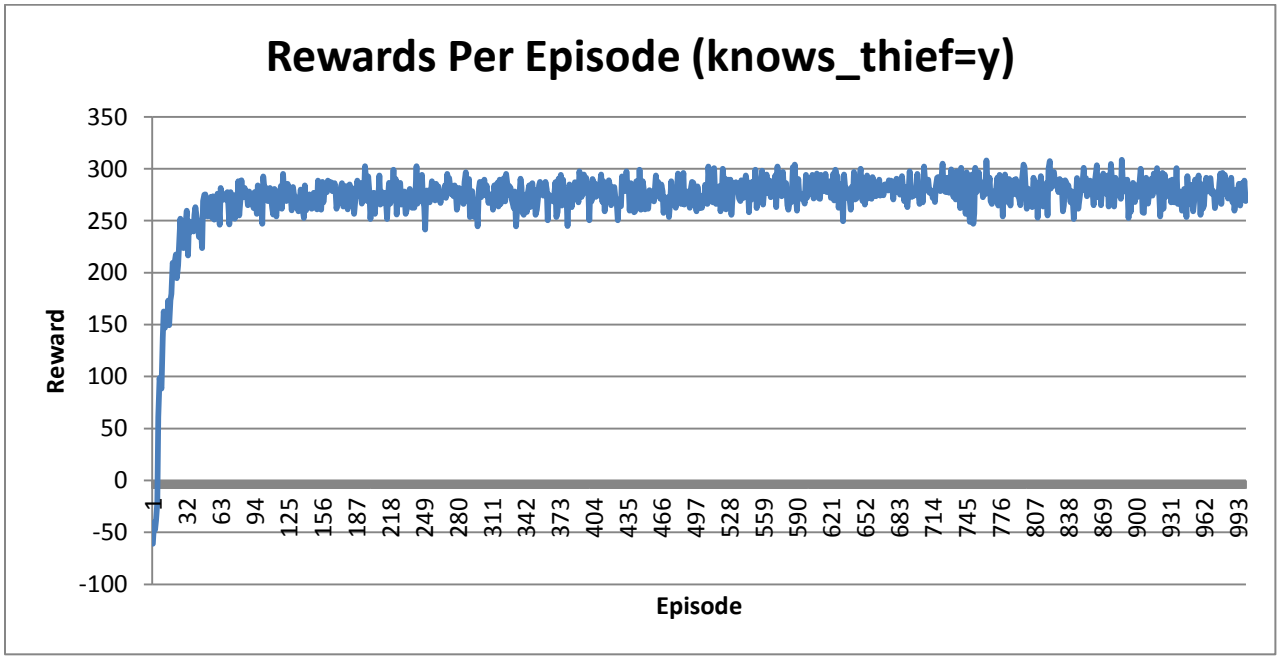
As we can see the average reward per episode is below 0. This is because the epsilon is set to 0.5 and therefore about half of the movements the agent makes are random movements. Since there are so many random movements, the agent can very easily walk through the slippery squares and drop the packages. This in turn leads to total negative reward for almost each episode even though the agent is trying to learn the best route because it is making so many random decisions.

2. World With Thief
    a. After setting epsilon to 0.05, and knows_thief to n, this was the chart I got:



As we can see the rewards are pretty constant, just slightly negative. This is because the agent knows the best route (and there's very little randomization), but the agent has no idea where the thief is. So the agent probably just walks into the thief most of the time while the agent manages to dodge the thief the rest of the time. This causes the thief to steal the packages a little more than the agent manages to deliver the packages successfully causing a total negative reward.

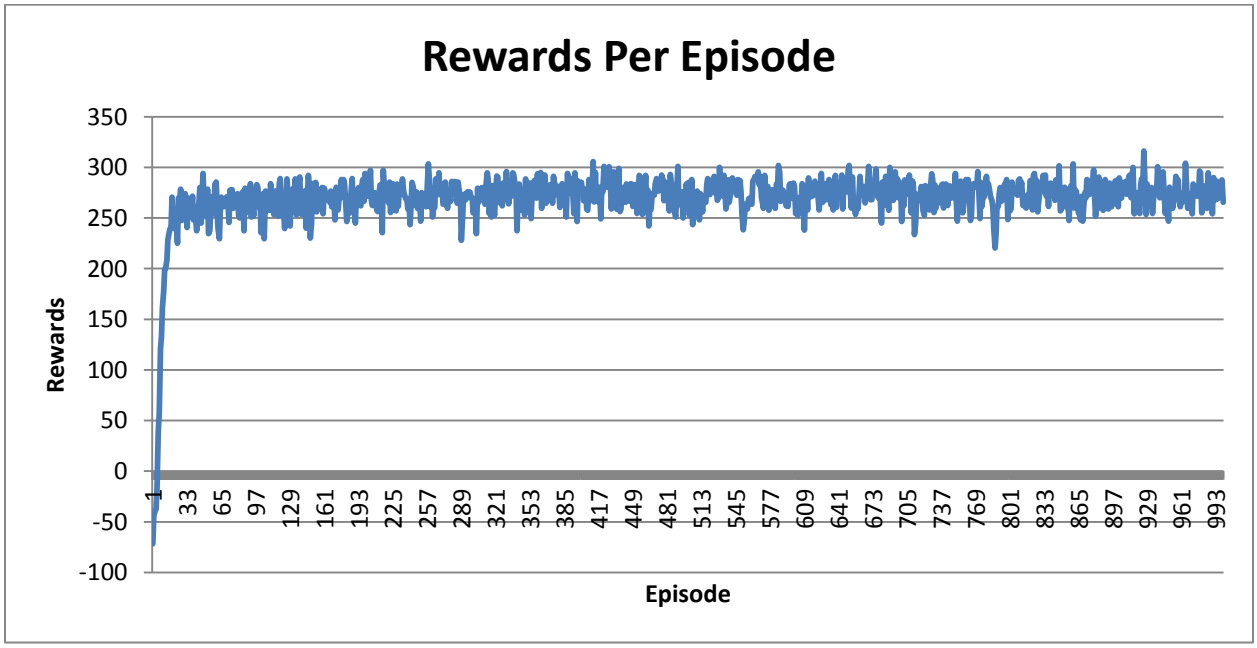
b. After changing knows_thief to y, this is the chart I got:



As we can see, the graph has about the same consistency as part A's graph. But this time, since the agent knows where the thief is, it can avoid the thief and have positive rewards for the episodes (except the first few where it needs to figure out the best possible path) and be a little more consistent since it won't randomly run into the thief. The difference between part A's graph is caused by the fact that the agent (almost) never runs into the thief because it knows where the thief is and can actively avoid the thief.

c. With logic, the best epsilon is going to be very small (smaller than 0.2) because we do not want too much randomization in each episode. Looking at the charts from the earlier problems, it seems to be that epsilon = 0.05 is the best epsilon value. A little further testing of epsilon = 0.05, 0.1, 0.15, and 0.2 with the rate staying the same seems to prove that epsilon = 0.05 is the most consistent option and gives the best rewards.
To find the best learning rate, I decided to keep epsilon at 0.05, and vary the learning rate from 0 to 1 with increments of .1 and then compared all of them together. From what the graphs looked like, it seemed to me that rate of 0.2 was the best option, giving rewards of 250-300, where everything else gave lower values. So overall the best values I found were learning rate = 0.2 and epsilon = 0.05.

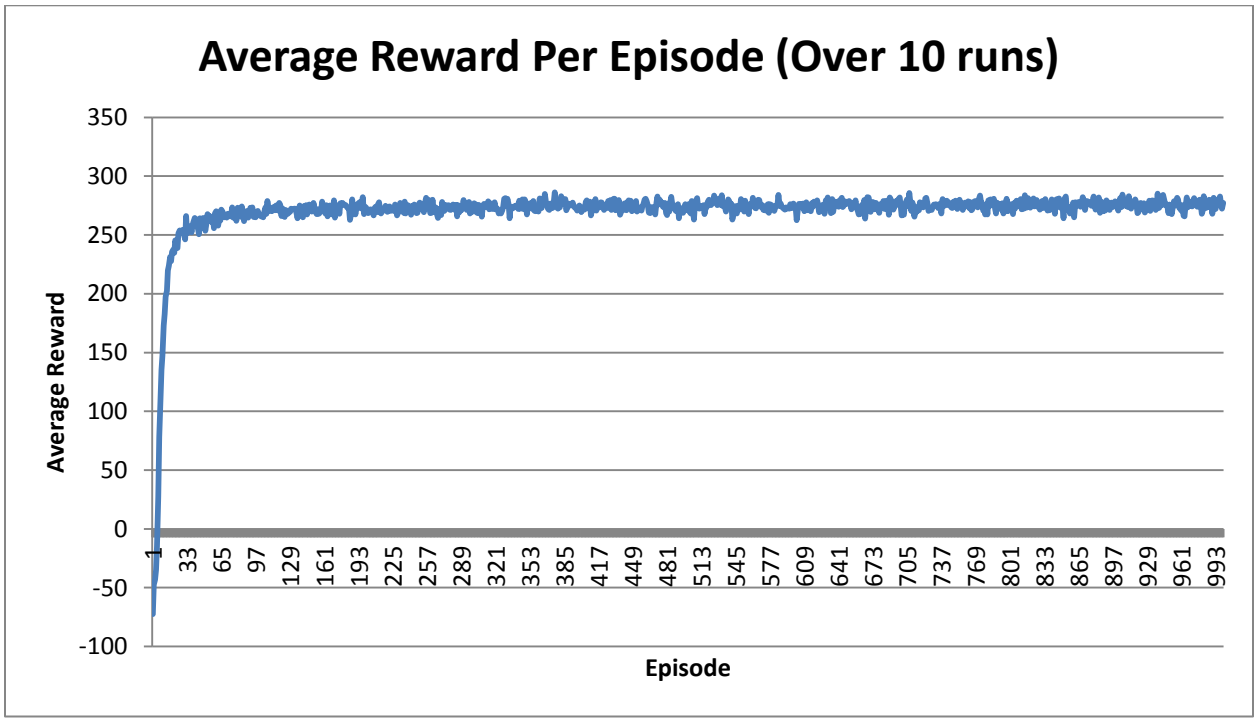
3. Here is the graph of the first run:



As we can see, very quickly the agent learned the best possible route to take. Once it learned that, the random factor of epsilon did not affect the total reward too much,

keeping it in about +/-25 from the average and keeping total rewards pretty consistent. This is the tightest and the highest graph of all of the graphs I have made.

The average graph of the 10 runs looks like this:



Within 50 episodes, the agent gets up to its 275ish average and stays there for the rest of the episodes, varying only slightly. As we can see the small epsilon does not affect the average over each episode by more than +/-10. The average reward per episode is almost level (after about episode 50) with a slight increase. Before that, the agent seems to very quickly find the best path through the world every time. Overall, the agent seems to run through each episode almost the same time every run I averaged because there is little variation between each episode.