Benjamin Blaisdell
NetID: blaisde2

MP1 – Part 2

1. WorldWithoutThief

   (a) In the first episode the Agent receives a negative total reward. In all subsequent episodes the Agent receives a total reward of 0.0. This is likely because the course of action that it chose in the first episode led to a negative reward, so it thinks that staying in one area and getting no reward is better than getting a negative one.

   (b) The Agent now starts out with several episodes of fairly large negative total rewards, but soon beings to net a fairly large positive total reward for the remainder of the episodes. The change is due to the Agent continuing to explore a little after having some initial negative rewards, which allows it to find a policy which brings a net positive reward.

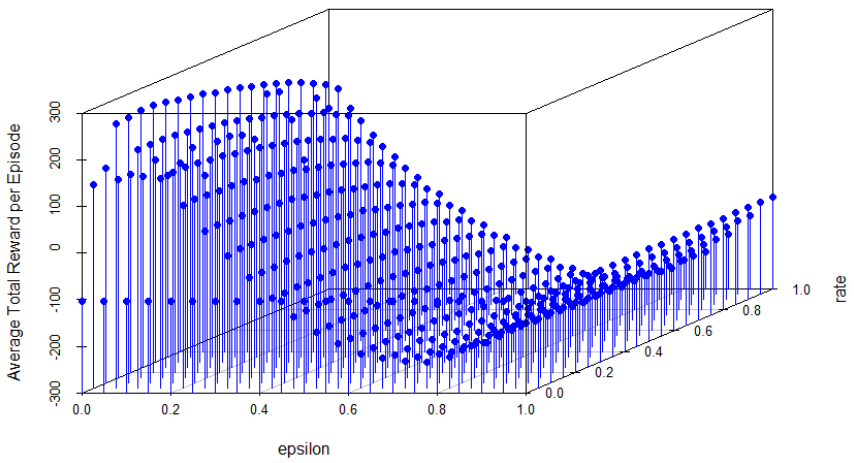
   (c) With epsilon set to a higher value the Agent receives a negative reward fairly consistently throughout the episodes. This is because the Agent acts too randomly, which does not give it the chance to build or follow an adequate policy.

2. WorldWithThief

   (a) The Agent receives a negative reward in nearly all episodes. This is because the Agent can't predict where it will receive a negative reward from moving into a square with the Thief. Sometimes it will avoid squares which are perfectly safe because of past interactions with the Thief, and sometimes it will move into a square that is likely to contain the Thief.

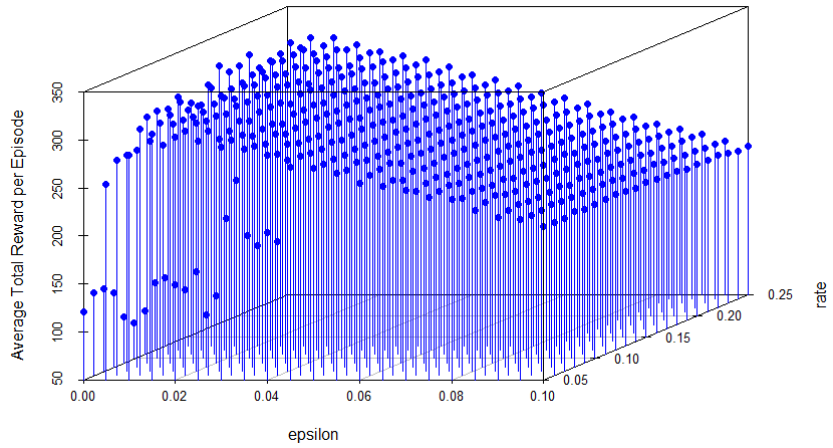
   (b) Now the Agent consistently receives a very high reward, because it is able to change its route based on where it knows the Thief is. This also means that there are many more states to include policies for the Agent to follow based on where the Thief is.

   (c) I modified the Simulator class to iterate over all possible combinations of rate and epsilon values from 0.0 to 1.0 in increments of 0.05 for each. For each combination I ran the Simulator with 1000 episodes of 10000 steps and calculated the average total reward for all the episodes in that run. Below is the plot of the output.

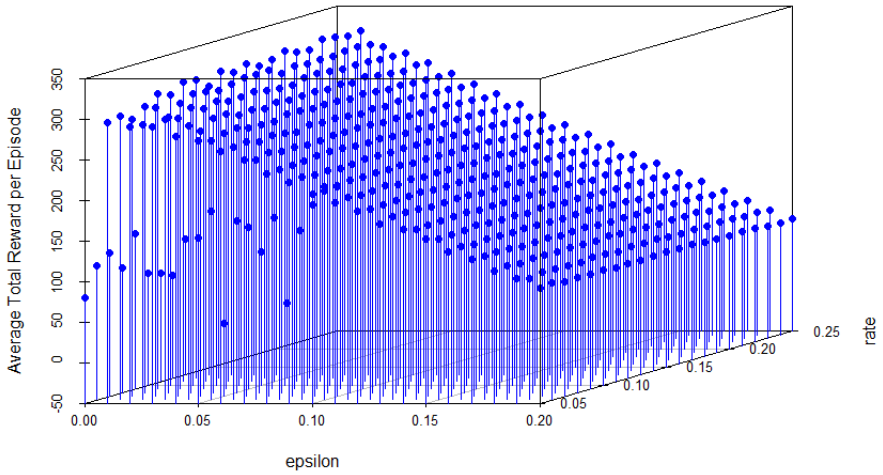


It seems that as long as the learning rate is a value greater than zero (i.e., the Agent is making at least some effort to learn the correct policy), there will be a net positive average reward for small values of epsilon (i.e., the Agent is acting on policy most of the time). I ran this program twice, and found the maximum average reward was obtained at rate=0.1 epsilon=0.05 the first time and rate=0.15 epsilon=0.05 the second time. The average total reward was around 272.

I then ran my code again on a more localized set of values, with the learning rate ranging from 0.05 to 0.25 and epsilon ranging from 0.0 to .20, with both values incrementing by 0.01 each time. The optimal values were rate=0.22 and epsilon=0.01 with an average reward per episode of 323.

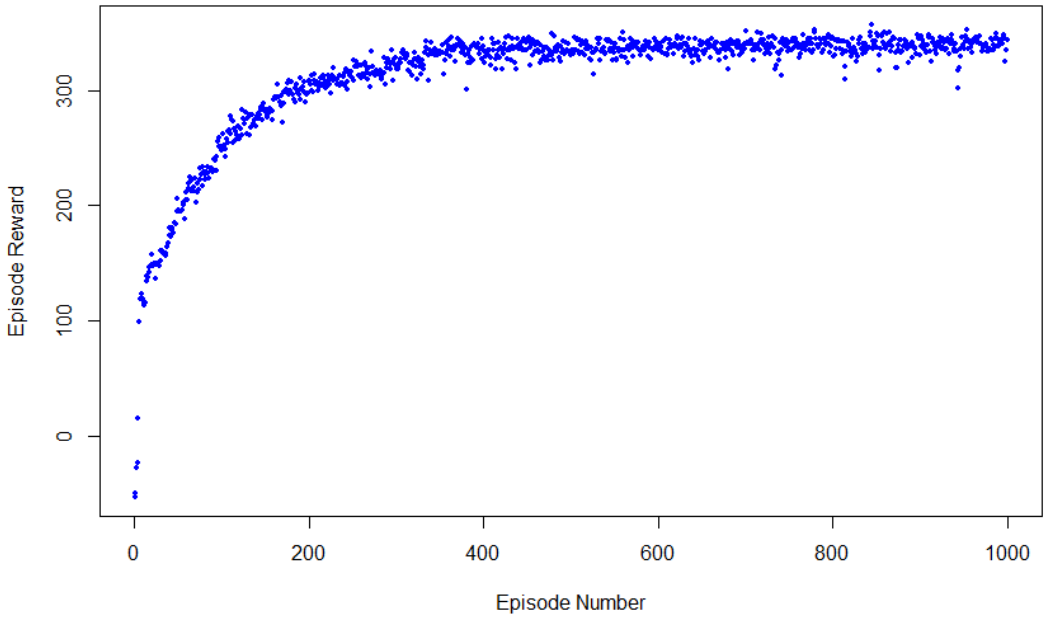


I ran my code once more with a more localized epsilon, since that seemed to require the more precise tuning out of the two. This time I ran it with the learning rate ranging from 0.05 to 0.25 incremented by 0.01 and epsilon ranging from 0.0 to 0.10 incremented by 0.005. The optimal values were rate=0.16 and epsilon=0.005 with an average reward per episode of 328.



I tested a few more values and the gain seemed to max out at around epsilon=.005. From the results it seems like learning rate should be between 0.1 and 0.2 and epsilon should be around .005. This is likely because there are a large number of steps in each episode, so a small epsilon is sufficient to create a decent policy in the first couple episodes and not bring the total average down overall by continuing to act semi-randomly throughout all the episodes.

3. Below is a plot of reward achieved using a learning rate of .16 and epsilon of .005. It appears to be asymptotic, leveling off after around 200 episodes to a reward of around 330.



Below is the average reward for each episode averaged over 10 simulations with the same settings. It is evident that the function is indeed asymptotic, and the rate of increase is approximately the same over multiple simulations.