Brandon Holland

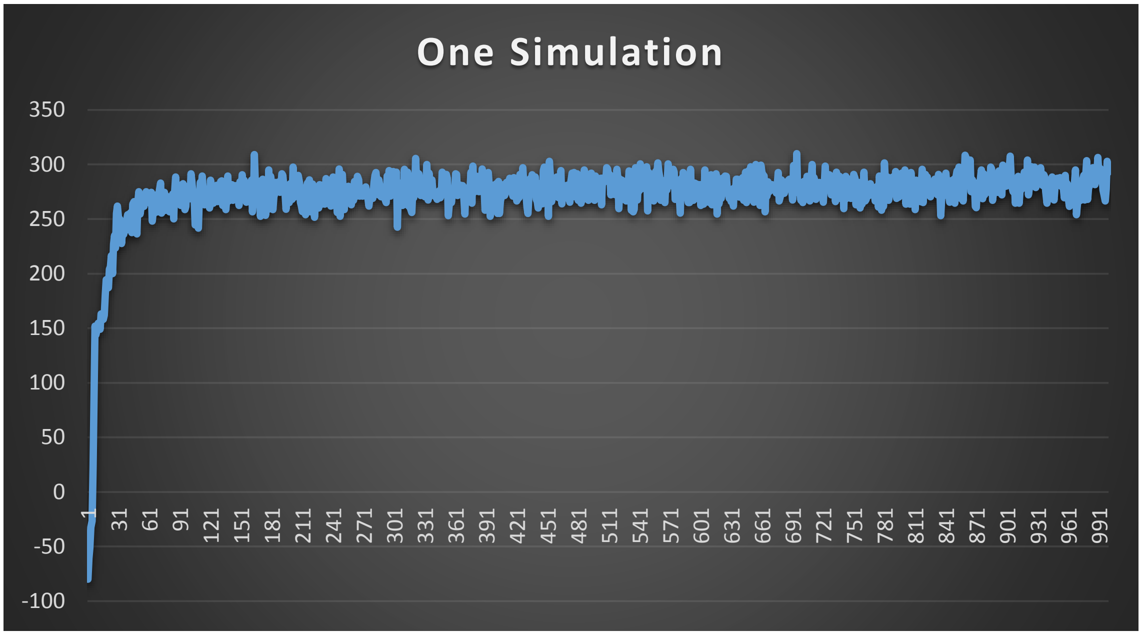
Beholla2

ECE 448

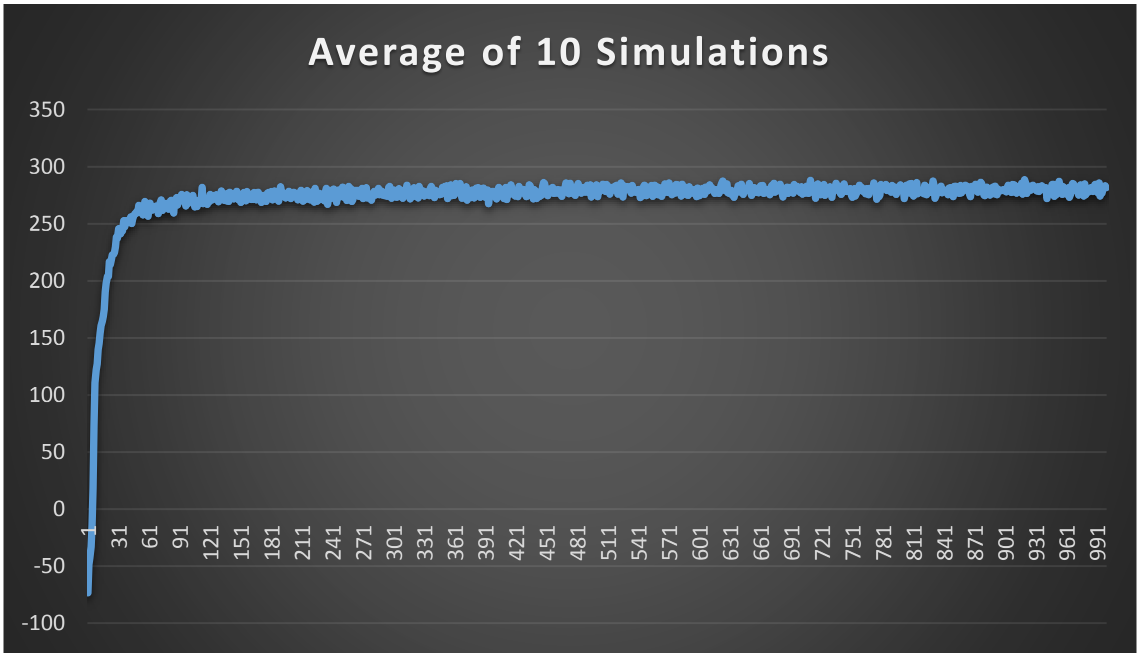
MP1.2

1. World Without Thief
    a. When the epsilon value is set to 0, we yield an initial negative reward and then the rest are all o. The policy that is generated just forces the robot to move to the upper left corner of the map. It attempts to move up another square, but it just remains in a fixed spot for the entire execution.
    b. When the epsilon value is set to 0.1, we yield rewards are negative for approximately the first 30 steps. After that, we receive positive rewards that range from 50-200. With this epsilon value, the packages are actually able to be delivered correctly.
    c. When we set the epsilon value to 0.5, we yield rewards that range from about -10 to -50. If we set the epsilon value to 0.9, we receive even smaller numbers in the range of -60 to -70. With epsilon values this large, we are expecting too random of numbers and for this reason, we are not able to implement the learned policy.
2. World With Thief
    a. When we set the knows_thief parameter to no, the performance of the agent is very poor (values from around -10 to -20). The reason for this is because the agent is at a disadvantage since it does not know where the thief is and is not able to learn how to avoid it.
    b. When we set the knows_thief parameter to yes, the performance of the agent is very good (values from around 200 to 300). The function yields much larger reward values throughout all of the episodes because the robot knows of the thief and can implement the learning policy.
    c. For both the learning rate and the epsilon value, I found that values between 0.05 and 0.1 yielded are best results. These values gave me the largest average reward per episode. Anything smaller or larger, I saw a quick loss in performance.

3. Graphing Expected Reward



Graph for One Simulation



Graph for an Average of 10 Simulations

Observation: For both graphs, I observe that the learning rate is very quick at the beginning of the simulation. It only takes about 20-30 episodes to reach its steady state. Once it reaches this point, the award stays relatively constant for the rest of the simulation.