Glenn Osborne
gosborn2@illinois.edu
Sept 29th, 2014
MP1 Part 2

Solutions

Part 2 1a

In this scenario, the agent accumulates negative rewards in its first episode (see Scatter Plot Appendix for this part). The negative rewards were caused by the slippery cells in column three. Because there is no exploration (epsilon is zero), the system converges on a policy that keeps the agent in the first column. If there was exploration, as we'll see later, the agent would eventually cross the slippery cells without penalty and find positive rewards.

Part 2 1b

In this scenario, the agent progresses from mostly episodes of negative rewards to mostly episodes of positive rewards. It appears from the scatter plot (see Scatter Plot Appendix for this part) that it takes roughly 25 episodes before the trend is mostly positive rewards. Even though it appears that convergence is on results that are mostly positive, the results across episodes vary greatly. I believe these variations are due to the solid wall of slipperiness that randomly deducts rewards. This scenario is an obvious demonstration of the "learning" effect of this algorithm. The epsilon value of 1/10 caused enough exploration to change the trend.

Part 2 1c

This scenario illustrates how too much exploration (too large of an epsilon) prevents convergence onto an optimal policy. The scatter plot for part 2 1c shown in the Appendix shows a random exploration without regard for rewards (positive or negative). The result is a policy that produces mostly negative rewards.

Part 2 2a

This scenario contains a thief that can add negative rewards in addition to the negative rewards possible through slipperiness. Because this particular scenario doesn't extend its state system to include the whereabouts of the thief, the agent cannot "learn" to avoid the thief. As illustrated in part 2 2a of the Appendix of Scatter Plots, the penalties are so severe that the system converges on a policy that prevents the agent from accumulating positive awards at all. Note that there is a whole in the wall of slipperiness, but it isn't enough to overcome the additional penalties incurred because of the thief.

Part 2 2b

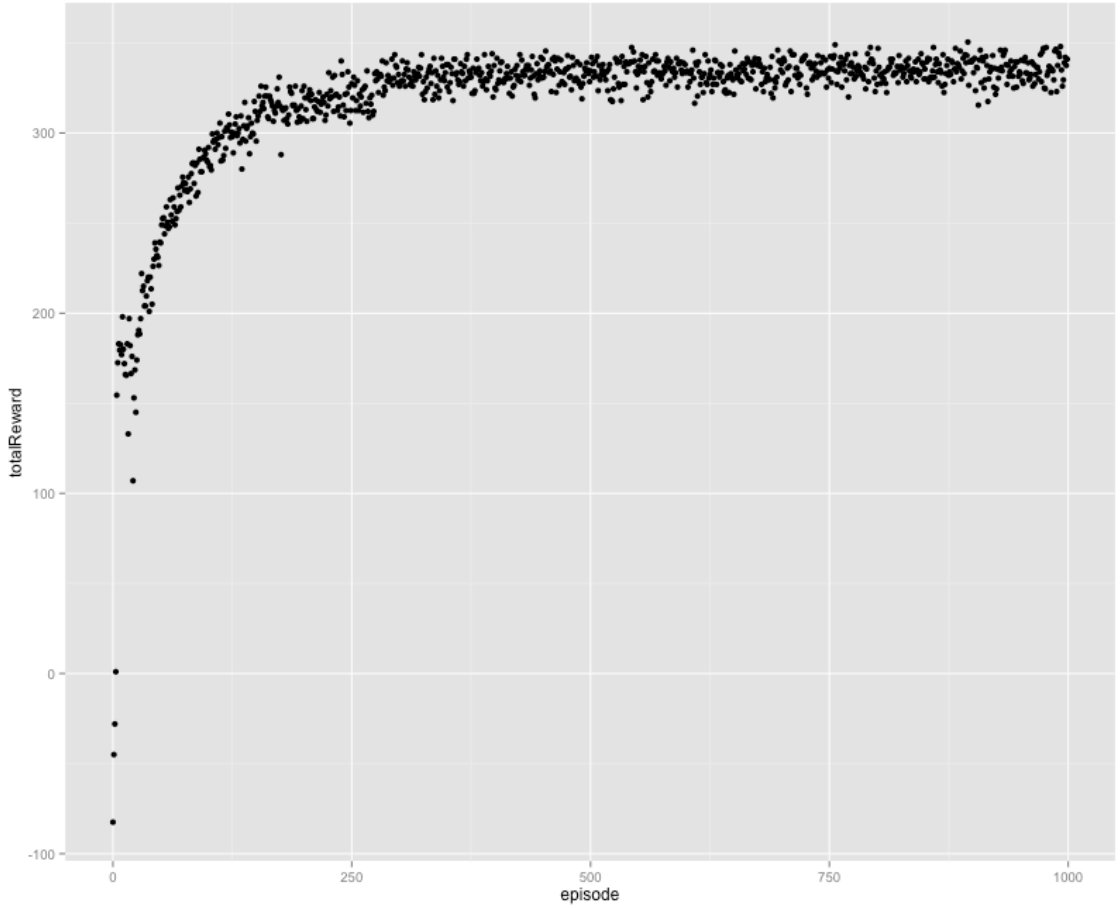
This scenario contains a thief, but the state system knows the thief. Knowledge of the thief allows the learning algorithm to account for the thief's whereabouts. The thief has a 50/50 chance of moving up or down, so it reasons that there would be enough room for the agent to move around the thief. Also, there is a whole in the wall of slipperiness. Part 2 2b of the Appendix of Scatter Plots shows convergence towards a policy that can navigate through the whole in the wall and avoid the thief most of the time.

Part 2 2c

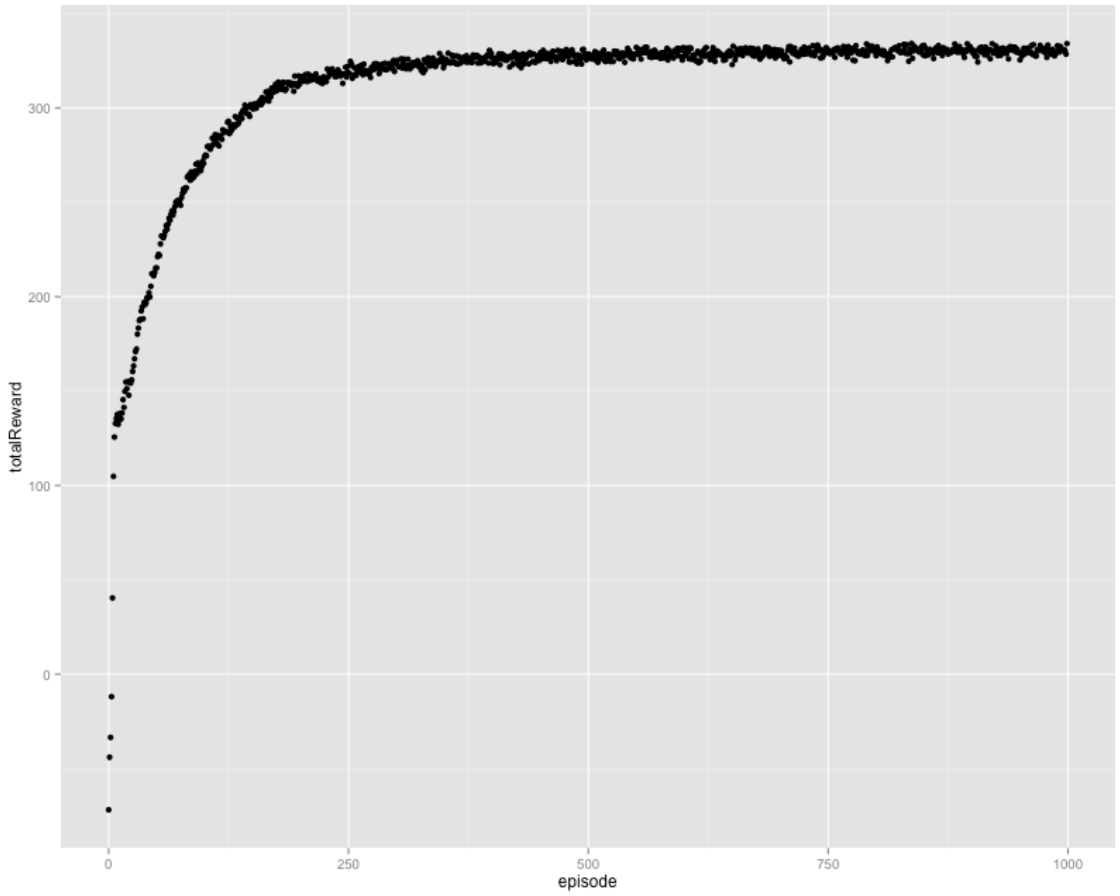
My experimentation seemed to produce the best results with a rate of 0.1 and an epsilon of 0.01. Because other experiments have shown that a certain amount of randomness is necessary to find the optimal policy, I spent a lot of time with epsilon values larger than 0.05. But for completeness I decided to try a relatively small number. I was surprised to see such good results with such a small epsilon. My thoughts on this are presented in the answer for Part 3, below.

Part 3

With the parameters of rate = 0.1 and epsilon = 0.01, the following graph from part 2 2c suggests that the optimal policy is a relatively narrow solution that is encountered quickly. Therefore, a small epsilon is sufficient exploration to find the optimal path, and it brings the added benefit of reducing the chance for wondering in later episodes, which explains the tight grouping of points in the scatter plot.



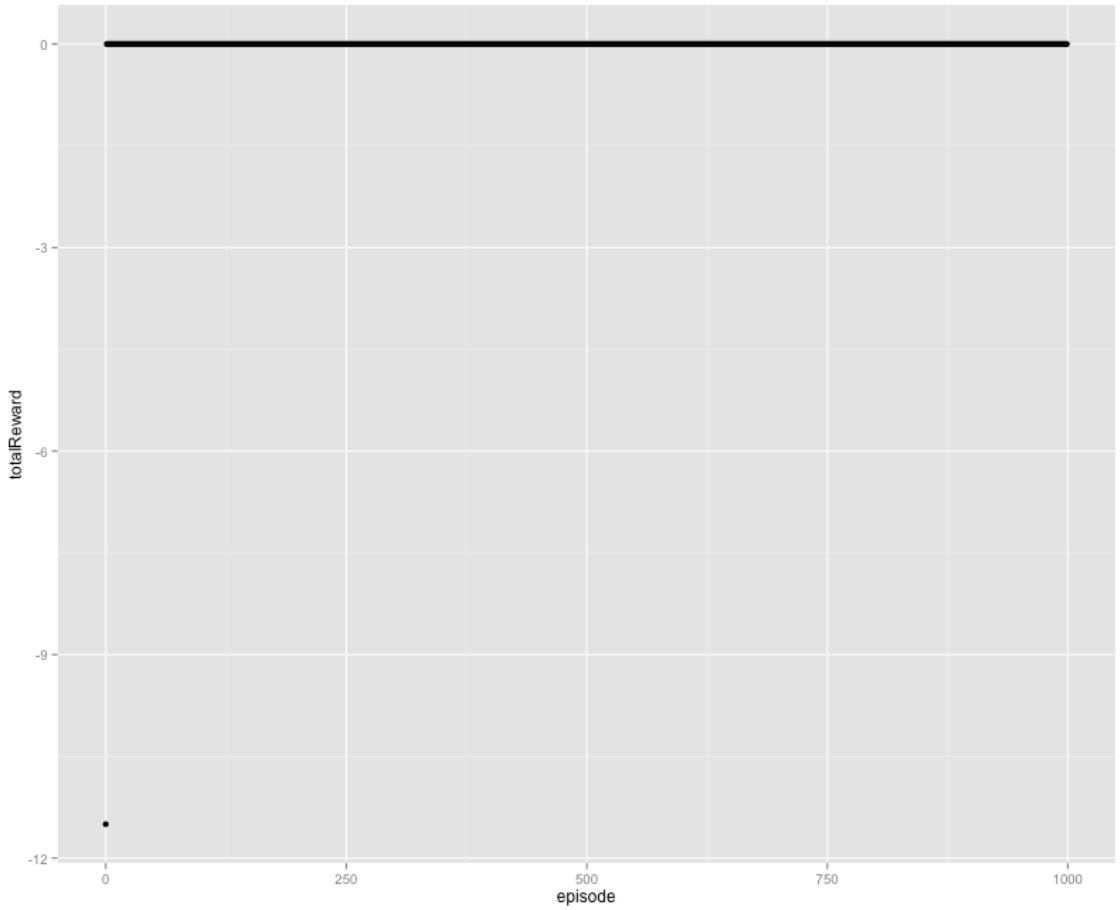
The following graph shows an average result of 10 simulations. Each simulation involved 10,000 steps and 1,000 episodes. The averaged graph is tighter than the graph produced from just one episode. The averaged graph suggests that averaging results over time will give us a better expectation of rewards produced by using this policy.



Appendix of Scatter Plots

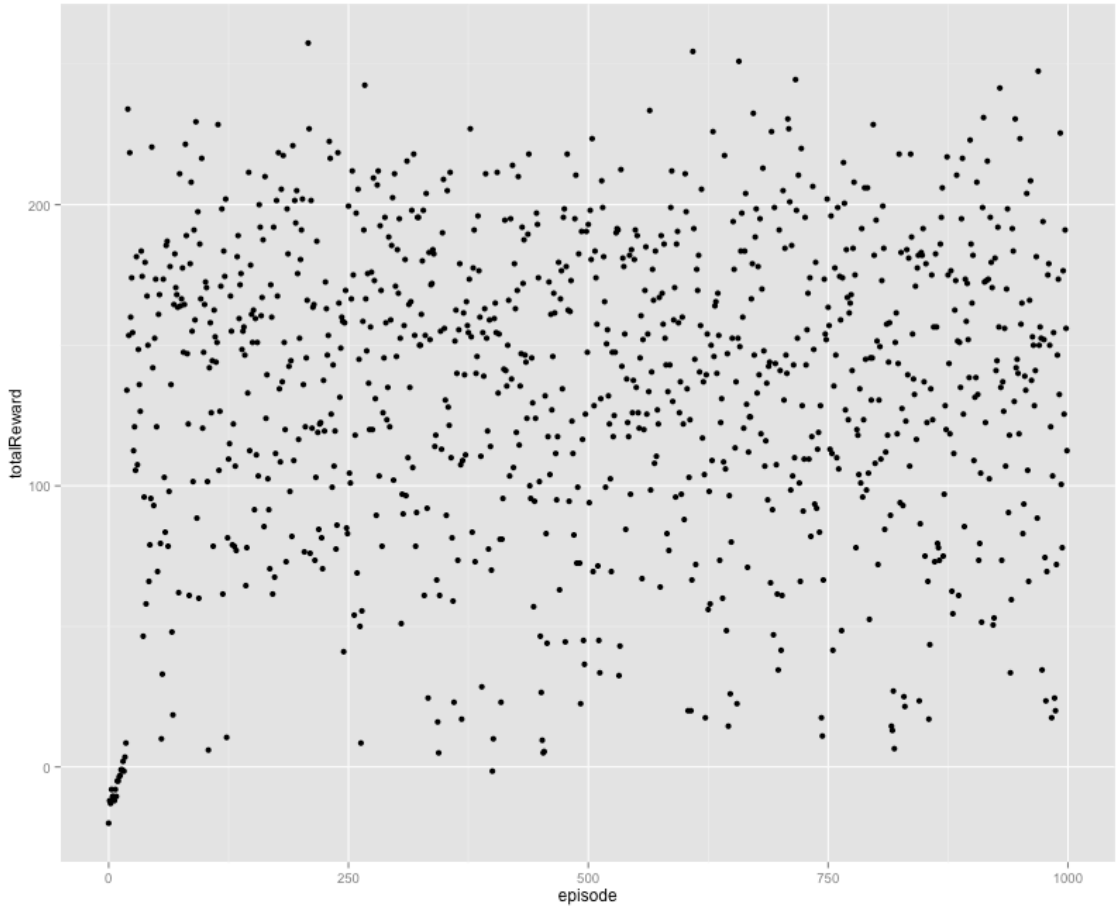
Part 2 1a

WorldWithoutThief QLearningAgent n 10000 1000 policy.txt episodes.txt
discount = 0.9
rate = 0.1
epsilon = 0.0



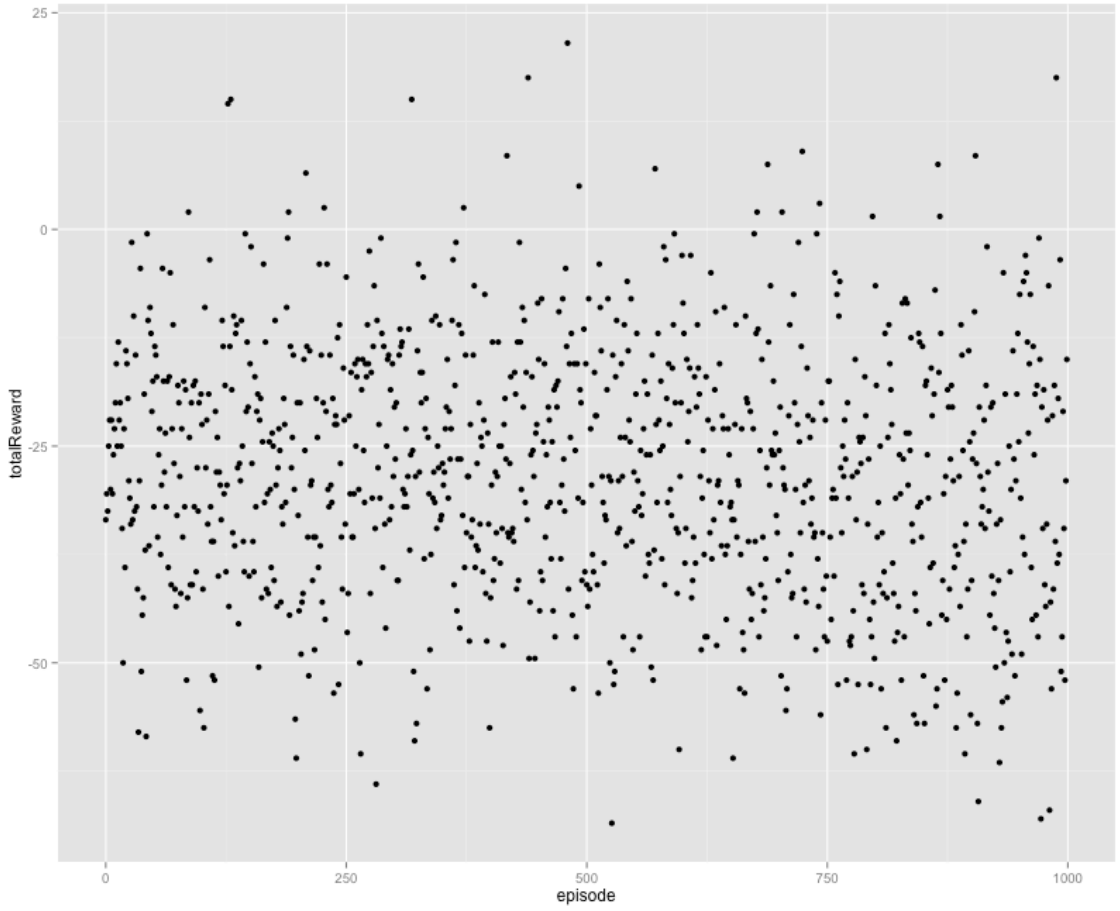
Part 2 1b

WorldWithoutThief QLearningAgent n 10000 1000 policy.txt episodes.txt
discount = 0.9
rate = 0.1
epsilon = 0.1



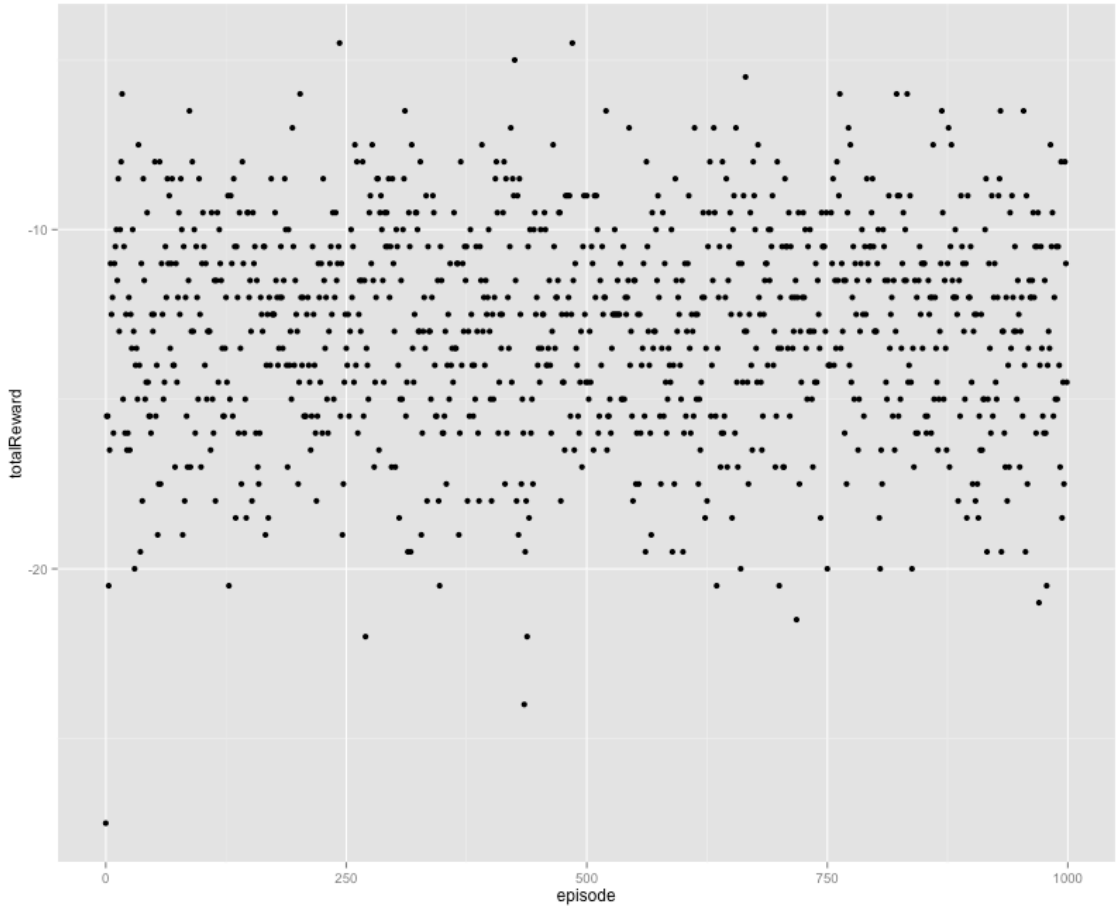
Part 2 1c

WorldWithoutThief QLearningAgent n 10000 1000 policy.txt episodes.txt
discount = 0.9
rate = 0.1
epsilon = 0.5



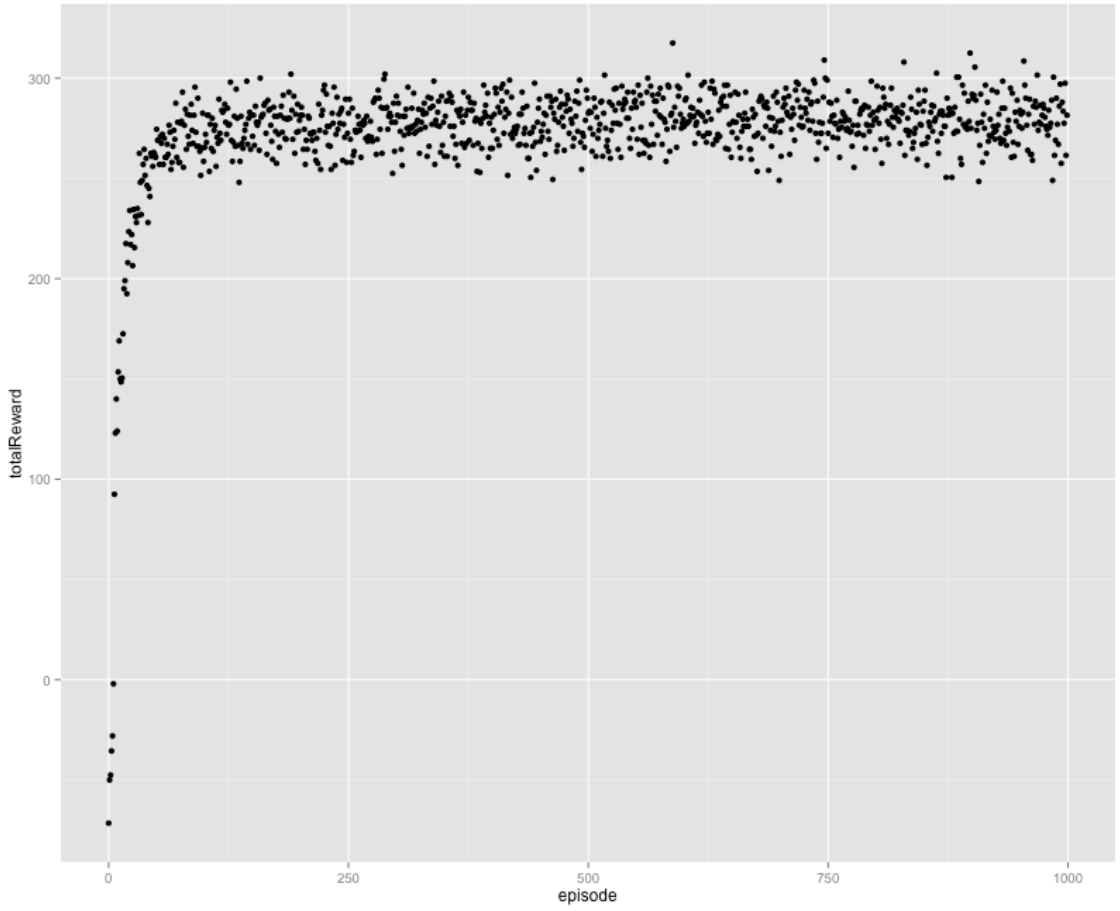
Part 2 2a

WorldWithThief QLearningAgent **n** 10000 1000 policy.txt episodes.txt
discount = 0.9
rate = 0.1
epsilon = 0.05



Part 2 2b

WorldWithThief QLearningAgent **y** 10000 1000 policy.txt episodes.txt
discount = 0.9
rate = 0.1
epsilon = 0.05



Part 2 2c

WorldWithThief QLearningAgent **y** 10000 1000 policy.txt episodes.txt
discount = 0.9
rate = 0.1
epsilon = 0.01