Brett Glasner

DeJong CS440

MP1 Part 2: Experimentation

1.

1. In this simulation the agent gets a reward of -0.8 in the first episode, and then gets a reward of 0.0 for all of the other 999 episodes. Since the agent doesn’t look randomly explore at any stage with these settings it simply looks take actions that lead to the optimal reward. After several steps it sees a few actions as leading to negative rewards and all other actions as having 0 reward since it hasn’t explored very far. The greedy approach leads the agent to think that 0 reward is the maximum reward it can receive since it doesn’t look possible future rewards, and thus the agent gets stuck at locations with 0 reward. That is why we see the result of the agent getting total rewards of 0 for almost every episode
2. With epsilon set to a value of 0.1, the results for the agent are much more favorable. For the first 30 episodes or so the agent sees rewards ranging from -15 to 0, but after those 30 episodes the agent begins to see very positive rewards. For the rest of the episodes the total reward ranges from about 65 to 200 randomly changing between episodes but remaining constant. These positive rewards are due to the fact that since the robot explores randomly during some steps in each episode it doesn’t get stuck like with an epsilon of 0. It has more opportunities to take actions that could potentially lead it to a higher reward. Additionally, since the epsilon value is low the agent more heavily weights the QValues of potential actions so the agent is more likely to take actions leading to higher rewards. That is why the rewards for each episode are positive. The variability in rewards between episodes can be attributed to the randomness from taking random actions and from dropping the packages on slippery coordinates.
3. Changing epsilon to 0.5 causes the rewards at each episode to be variable and negative from about -10.0 to -50.0. With a high epsilon, the agent is taking a random action half of the time instead of taking actions based on the optimal rewards it has found at each state. The potential reward the agent could receive is valued less in the decision making process for taking an action, so the agent makes very inefficient actions frequently. Random actions are unlikely to lead to optimal rewards over time, so the reward amounts at each episode become more negative as the agent doesn’t look for the optimal action.

2.

1. The agent’s performance in this case is very bad with a consistent reward around -10.0 for every episode. Since the agent doesn’t know the thief exists it makes decisions on its action without taking into account that the thief is there and is almost guaranteed to run into the thief every time. The agent doesn’t know any states including the thief, so it doesn’t have any actions to account if it might run into the thief. If it runs into the thief consistently it is assuredly going to receive a negative reward, so over time the reward is very poor for not taking into account an additional factor of randomness.
2. In the case that the agent is aware of the thief, the performance of the agent starts off highly negative and then quickly becomes highly positive. Furthermore, the performance is much more consistent than in the simulations of the world without the thief. When the agent is aware of the thief it stores reward values for states involving the thief, so as the simulation runs the agent begins to become more aware of when it might run into the thief or not. The agent learns to avoid the thief and then consistently delivers the packages resulting in a high reward in each episode.
3. The optimal learning rate and epsilon I settled on were 0.1 and 0.01 respectively. After several trials with different values it became apparent that the learning rate had less of an effect on the reward received than epsilon did. A wide range of values for the learning rate showed little change throughout the episodes, however changing epsilon slightly had great effect. Using a small epsilon proved to be ideal for this simulation. Since the agent takes many steps in each episode it will still take a good number of random actions, but because the epsilon is small those random actions won’t affect the optimal path the agent learns over time much since it takes random steps so infrequently. Running the simulation with these settings saw episodes with consistent rewards of 340 which was greater than the 270-280 seen with the settings used in part b.

3.

This plot shows a very rapid increase in the reward gained by the agent in each episode suggesting that the agent learns very quickly which is the optimal path to take. The reward plateaus at about 350 although there is still a decent amount of variability around the value to which the graph converges.

This graph looks at the average of 10 trials using the same settings as the previous plot. The graph has a less steep incline than the previous plot. This suggests that in different trials the agent learns the optimal path in different amounts of time. This is to be expected given that how fast the agent learns is dependent on how lucky it gets with its random explorations. Another interesting feature is that this graph doesn’t plateau as quickly as the other graph. In fact, if this graph were to be extended to more episodes it appears as if the average reward would continue to increase. The average reward in this graph reaches a maximum of about 390 which is higher than the 350 seen previously. Lastly, this graph shows much lower variability than the graph of a single trial.