CS 440 MP1 Part 2

Philip Daian - daian2

September 29 2014

Question 1

Part A

After setting the parameters to the given values and simulating 10000 steps for 1000 episodes, it is clear that the agent's expected discount reward fluctuates across a few generally negative values for the first few episodes, settling on and finishing with an expected discount reward of 0.0 for the final episode of the simulation. This is because at first, the agent has not learned anything about the world, and sees each action as having an equal expected reward. With an epsilon of 0.0, it will attempt to greedily pick the best action each time. In the first few episodes, this greedy search will likely lead to a negative expected reward, as the bot will likely slip and drop or fail to deliver the packages as it randomly tiebreaks between available actions (as there are far more scenarios where this occurs than not). As a result, each action taken in the initial state will have a negative expected discount reward at first, and the greedy algorithm from the start state will begin to make choices that will not leave the start state. These actions by definition have a reward of 0, which is higher than the reward expected by the simulation for any potentially productive (state changing) actions which were initially probabilistically explored to have negative reward. They will thus be preferred by greedy every time, leading to the same result across all but the initial episodes: a policy yielding an expected reward of 0.0.

Part B

After setting an epsilon value of 0.1, the learning agent now finishes the simulation with a positive expected discount reward, generally converging on a value between 100 and 200. This is because rather than simply avoiding exploration as with the strictly greedy approach above, the randomness introduced ensures the agent explores policies it expects to provide a lower reward. Because these policies contain the paths actually advancing the state of the simulation and are the only way to get positive rewards, a policy with positive reward is eventually probabilistically explored. Once it is, the expected discount reward of previous states is updated, and similar policies are explored. The entropy introduced in the selection of the next action ensures that greedy does not ignore actions which could lead to states with both positive and negative rewards, and probabilistically guarantees the exploration of a wider range of policies.

Part C

With a large epsilon value, such as 0.5, the behavior generally settles on a negative expected discount reward in each episode. The effects of an epsilon of 1/2 imply that every second step, a completely random action is chosen. Because it is almost certain that there are a larger number of policies with negative expected discount rewards than with positive expected discount rewards (explaining the behavior in part a), such a large percentage of random choices likely destroys the structure of the iterative learning process and results in a policy selection that is nearly random and thus expected to have negative discount reward. Setting the epsilon value to 1.0

confirms this theory, exhibiting the same behavior of yielding polciies with negative expected discount rewards whose action choices are random in nature.

This implies that an optimal value of epsilon exists, large enough to encourage the greedy learning algorithm to explore policies it expects to yield a negative reward but small enough to preserve the structure and learning of the greedy process.

Question 2

Part A

With these parameters in WorldWithThief, the performance of our agent is poor, generally expecting rewards by the last episode that are between 0 and -15. Because the location of the thief also changes, the best action to take from a given state is affected by the position of the thief, and is not constant throughout the process. Thus, there is additional information in the environement with the potential to affect the reward (the position of the thief) not adequately encapsulated by the state, which is the model of the environment in our Markov process. While the agent seems to learn and expect a much better reward than with random policy selection, and in some cases sees an expected positive discount reward, it is impossible for the agent to learn a strategy for dealing with the thief (who is constantly moving) without knowing the location of the thief and thus impossible for the agent to effectively select the best action from a given state due to the inadequacy of our model.

Part B

With knows_thief set to y, the reward we expect with our learned policy is much higher than in the last episode, generally hovering around 280. The reason for this is explained in part a - because the position of the thief in a given state affects the optimal action to be taken from that state, our process model more closely models the environment by taking this into account and increasing our statespace accordingly. With the ability to know the position of the thief, the agent is able to avoid the thief consistently, and to take their position into account when estimating the expected discount reward in each state. This eventually yields a policy sensitive to the location of the thief and thus able to make more informed decisions about the next action to take in any given state.

Part C

We begin by doing a binary search for the best possible learning rate, starting with the median value of 0.5 and analyzing the expected discount reward in the last episode of the simulation otherwise using the given parameters averaged over three runs. We fix epsilon at 0.05 as in the previous part. At 0.5, we experience an expected reward of 236. Trying both larger and smaller parameters, we see an expected reward of 270 at 0.25, and 179 at 0.75. This implies that a learning rate smaller than 0.5 is ideal, as there is a noticeably poor performance of learning rates larger than 0.5. This could be due to the loss of old information with high learning rates causing a loss in the ability of the model to sufficiently remember the structure of the problem.

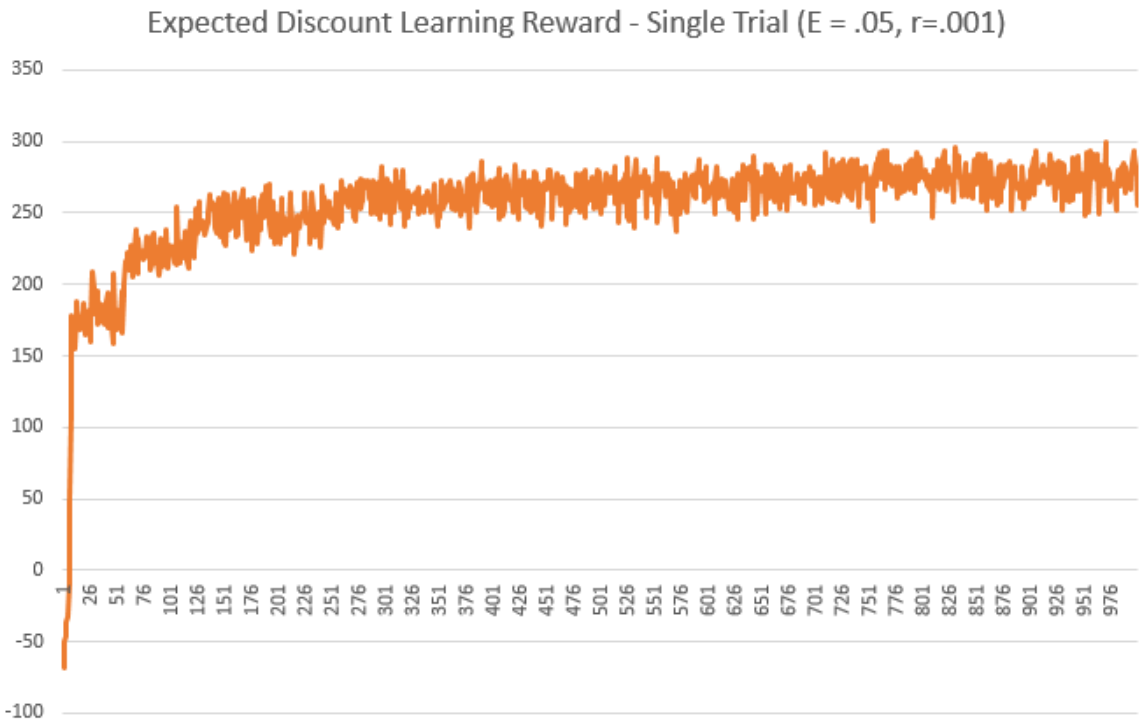
With a rate of 0.3, we see an average expected reward of 265. For 0.1 this number is 280, for 0.05: 278, for .01: 285, for .005: 280, and for .0005: 268. There is a clear peak of the reward expected somewhere between 0.1 and 0.01. Because running the simulation with learning rates in this range yielded very similar results, we will choose the last point at which we notice the expected reward to increase as we decrease the learning rate, which is 0.01. Smaller learning rates like 0.005 seem to minorly decrease the expected reward, while much smaller rates like .0005 yield noticeably and markedly inferior rewards than the optimal learning rate we choose, 0.01.

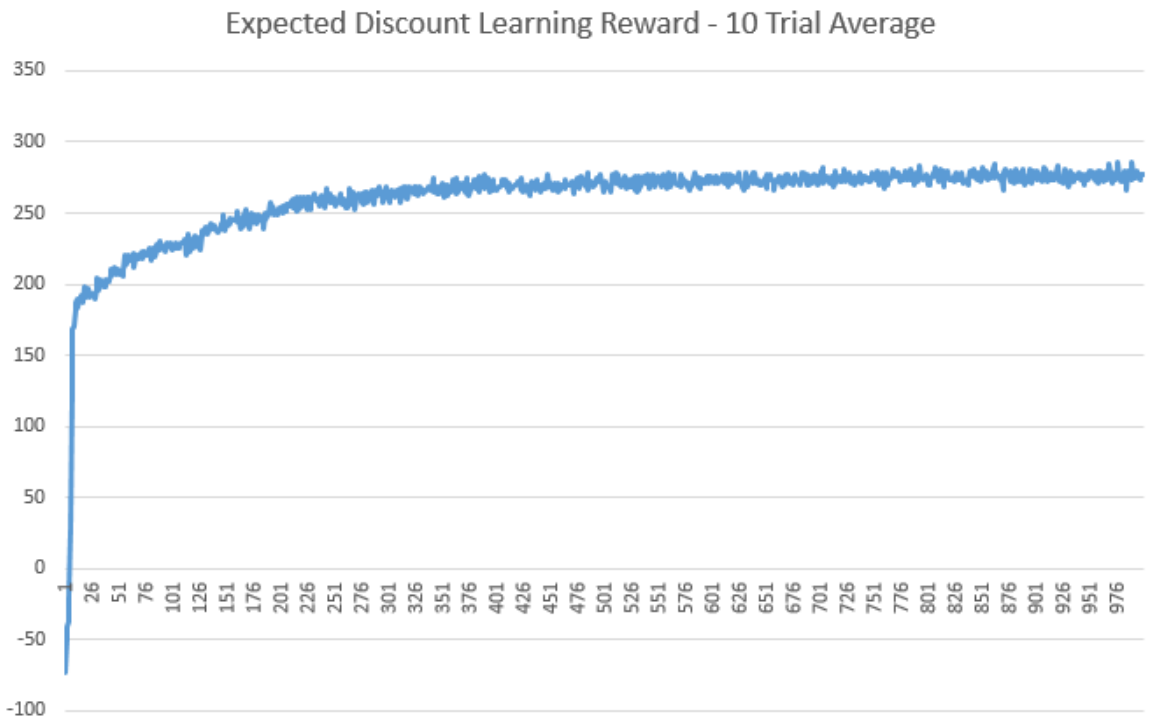
It is worth noting that observationally, there seems to be a decreased variation in the expected discount between epsiodes in a run as the learning parameter decreases. This is expected, as new information is less likely to drastically alter the policy with a low learning rate. This further justifeis our decision to choose the learning rate on the lower end of our acceptable range, as it will yield the most stable results across episodes in a given run.

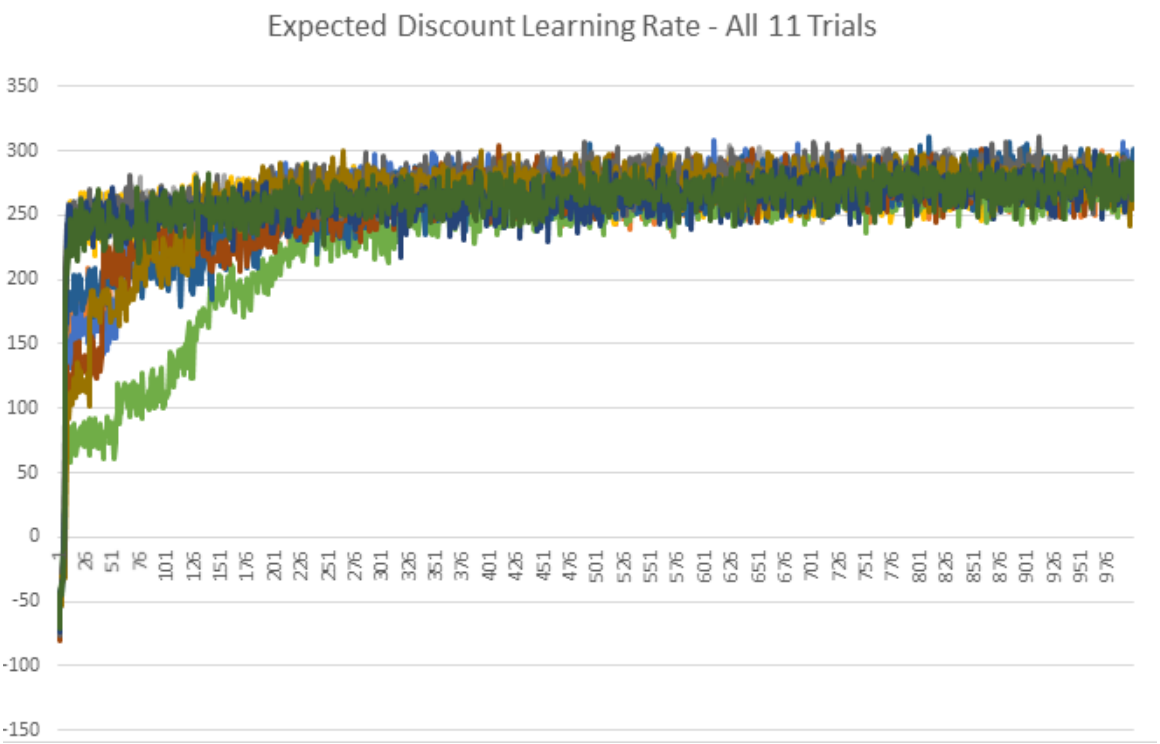
Fixing this learning rate, we perform the same experiment to determine the optimal value of ϵ. We start with the given value of 0.05, yielding an average expected reward of 282 averaged over three runs in the last episode. Increasing epsilon to .1 yields a noticeable decrease of the final expected reward to 235! This also comes with a large increase in the variation of values between episodes, with values within the last 10 episodes as low as 100 in some simulations. Because we know too large an epsilon parameter is not optimal, we also attempt to decrease epsilon. A value of .01, less than an order of magnitude away from our previous value, yields an average final reward of 240, a significant decrease from over 280. While there is much more consistency across episodes, with most episodes ending with an expected reward within 5 of the previous episode, this decrease in the performance of our agent is not optimal, so we decide on a final value of $\epsilon = 0.05$.

Question 3

With the final values of a learning rate of 0.01 and an epsilon of 0.05, in WorldWithThief with knowledge of the thief's location (raw data also attached to Compass in csv form, 11 runs in trial_results.csv:




Expected Discount Learning Reward - 10 Trial Average
350
300
250
200
150
100
50
0
-50
-100
1
26
51
76
101
126
151
176
201
226
251
276
301
326
351
376
401
426
451
476
501
526
551
576
601
626
651
676
701
726
751
776
801
826
851
876
901
926
951
976


Expected Discount Learning Rate - All 11 Trials
350
300
250
200
150
100
50
0
-50
-100
-150
1
26
51
76
101
126
151
176
201
226
251
276
301
326
351
376
401
426
451
476
501
526
551
576
601
626
651
676
701
726
751
776
801
826
851
876
901
926
951
976

The first graph shows exactly what we expect: for the optimal values of our parameters, each episode generally yields a progressively higher reward, eventually converging on some maximum expected value. The jitters in the growth of the expected reward are likely introduced by the randomness inherent in the policy creation, and as expected the initial expected reward is negative (as more policies yield negative than positve rewards).

What is more interesting is the graph of the expected discount after each episode, averaged over all simulations. As expected, as similar growth in the reward is noticed as the agent learns more about the environment after each episode. The jitters in the growth are reduced, likely stabilized by the increased amount of trials as the random probabilistic differences in policy average out across runs. The growth looks strikingly similar to that of a logistic function growing to a carrying capacity, perhaps suggesting that there is a theoretical maximum reward that can be expected to be achieved with these simulations in these models.

This is confirmed by the final graph, that of all ten runs, again which show growth converging on a maximum reward. While there is a probabilistic nature to the beginning of the simulation, especially in the initial stages of finding a policy, there is a clear trend of convergence on a reward between 250 and 300 given our parameters.

Increasing trials further would likely result in a continuation of this trend, smoothing out the average reward growth and further resembling the logistic function.