1) WorldWithoutThief

a. With epsilon at 0, nearly all the episodes end with 0 reward because without ever going in new directions, the agent ends up doing the same thing every time

b. With epsilon at 0.1, the average reward by the end of the 1000 episodes is approximately 100. This is because allowing the agent to test different routes gives it the opportunity to find optimal routes.

c. With epsilon at 0.5, the average reward is in the negatives. This is because there is a 50% chance the agent will make a completely random move each step. With such a high chance of randomness, the agent often makes poor decisions.

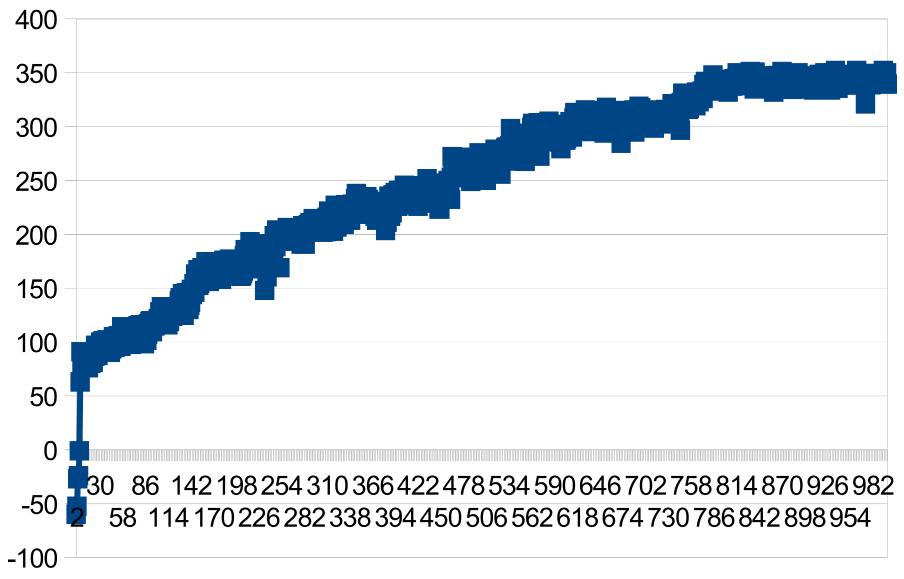
2) WorldWithThief

a. The performance is poor with epsilon at .05 and no knowledge of the thief because the thief's random movements make it very difficult for the agent to learn a good path to take.

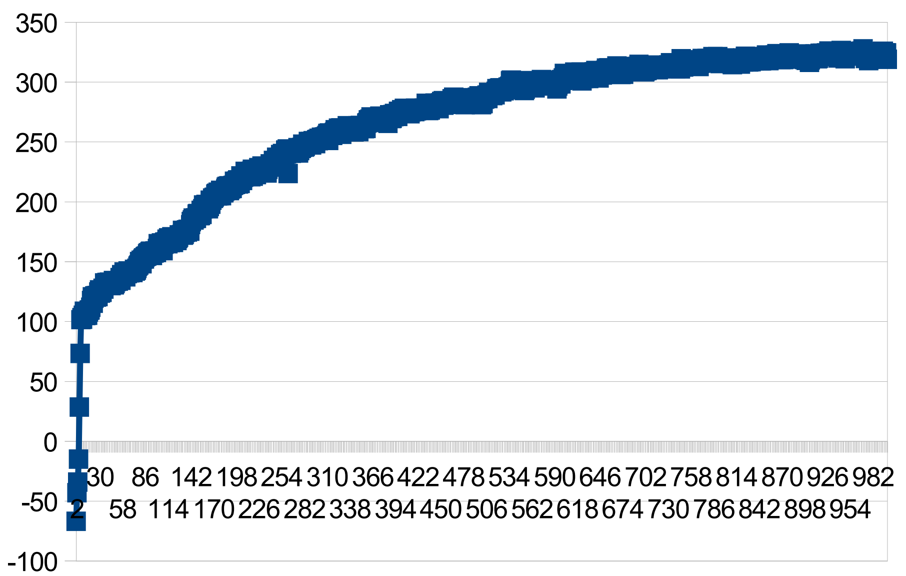
b. Just by allowing the agent to know where the thief is, the average reward jumps up to over 250. This is because the agent can learn to take a path that makes it very unlikely it will run into the thief.

c. The learning rate didn't seem to have much effect in my tests. Values from .1 to .5 all resulted in about the same average reward. I ended up putting the learning rate at .2. Changing epsilon had a much more noticeable effect, with smaller values improving the average reward. I found the optimal value to be about .001. With so many steps per episode, it is better to have a small epsilon to prevent the agent from making too many random choices, while still making enough to explore most possible paths. Very small learning rates prevent the agent from learning the best path within the 10000 steps, and large learning rates make it harder to learn better paths as they are found.

3) Plot results



The agent very quickly improves in the beginning, continues improving somewhat linearly for most of the episodes, then plateaus at the end.



The average of 10 trials behaves much in the same way, but with a smoother curve after the initial massive improvement. It now most closely resembles a y=ln(x) graph.