CS 440 / ECE 448: Introduction to AI

MP 1

Cassio dos Santos Sousa

September 29, 2014

1. (a) Using the data given in the enunciate ($\gamma = 0.9, \alpha = 0.1, \epsilon = 0.0$), it is observed. in the episodes' output, that the first reward is lower than zero (around -4.0 or -6.0), but quickly increasing to 0 (with only one or two other negative values), from where the rewards do not change until the end of the simulation.

```
-4.0
-3.5
-2.0
0.0
0.0
0.0
...
```

As $\epsilon = 0.0$, the movements of the agent follow the policy strictly. The first episode of this movement (Company) starts in a region with zero utility. The first movements of the agent also have zero utility, so a random variable is responsible for the tiebraker. At first, one can think that this would eventually guarantee

the optimal policy (reaching the customers). However, if you look at the map of **WorldWithoutThief**, you can see that the agent's starting point (Company) and the region of the costumers are separated by a slippery line. The first steps of our agent may be through these tiles, and as their utilities are (most probably) negative, the agent is prone to not return to that line. As the region of the Company is surronded by tiles of utility zero, any updates will also return zero, but this region has a higher utility than the slippery line, so the agent stays at that region during all the remaining movements (like if he had become "scared" of the line).

(b) Setting $\epsilon = 0.1$, the beginning of the simulation (shown below in fig. 1) is found inside a region of small values (not more than 4.0 or less than -4.0), from where it quickly starts increasing to the maximum reward (around 29.0).

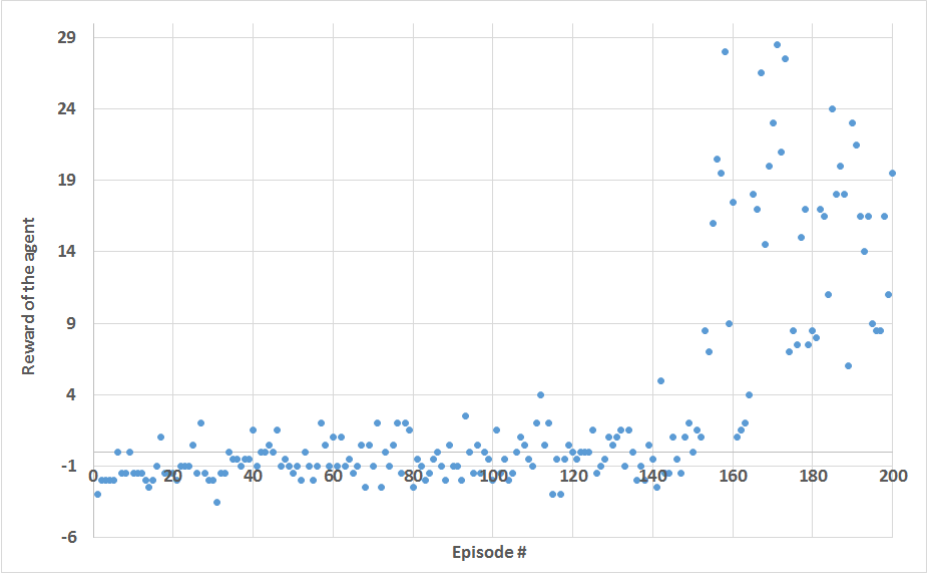

Figure 1: First episodes of a simulation using WorldWithoutThief and $\epsilon = 0.1$.

As the tendency of growth appears, the entire set of values is reached by the agent, and the graphic tendency (shown in fig. 2) starts reflecting the effects of randomness.

Figure 2: All the episodes of the simulation shown in Figure 1. The randomness is visible.

As $\epsilon = 0.1$, it is expected that a random movement is chosen 10 percent of the time, regardless of policies or utilities. At the beginning, this variable is responsible for taking the agent to a region of bigger rewards (the optimal state), as the slippery region secures this region. However, the effects of ϵ are present during the entire simulation, so even though the policy may be already optimal, the agent will eventually move to a slippery tile and receive a negative reward.

The graphic shows a positive average for the rewards (around 13.4). As the agent is capable of finding the optimal state more often than the slippery lines due to ϵ, a positive average was expected.

(c) Setting $\epsilon = 0.5$, the movement of the agent was random during half of the episodes. The results are shown in fig. 3.

As the random choices happen more often, it is hard for the agent to reach higher values. As soon as the agent discovers a tile with bigger reward, the next movement can lead him to go through a slippery tile (and vice versa). As there are many random choices, it is also hard to find the best utility, even if the agent

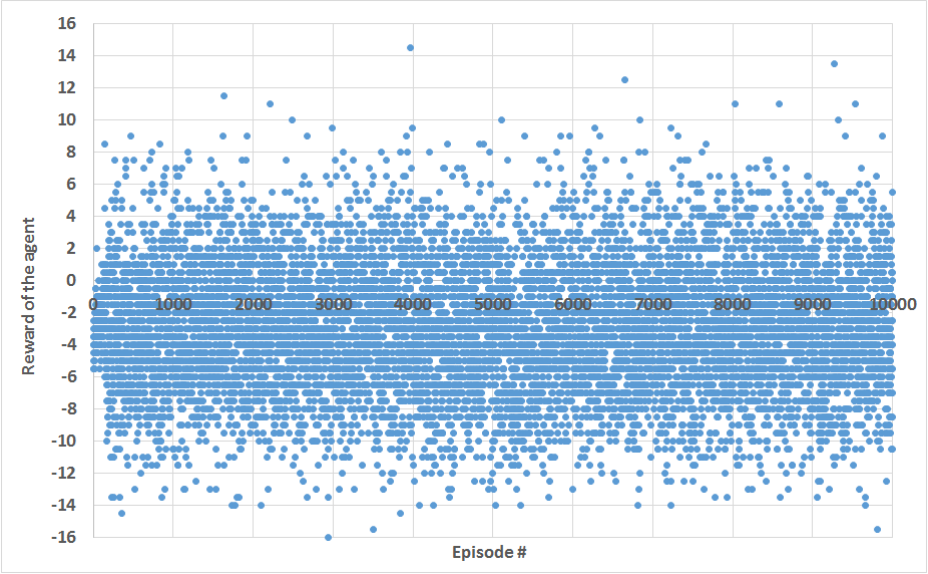

Figure 3: Episodes of a simulation using WorldWithoutThief and $\epsilon = 0.5$.

is prone to find it.

The graphic shows a negative average for the rewards (around -2.94). As ϵ is bigger, the effects over the average are now decided by the amount of tiles that can reduce or increase the total reward. As there are more slippery tiles than customers, a negative average was expected.

2. (a) Using $\epsilon = 0.05$ in WorldWithThief without knowing where the thief is, the episodes showed the rewards in fig. 4. The performance of our agent was poor, reaching rewards lower than -9.5 but never reaching rewards higher than 2.0. The average reward of this simulation was -1.32.

As our random variable is not null, the agent will eventually find the slippery tiles and the thief. As soon as the agent discovers them, a negative utility is set for these tiles. However, there is one path that only has the thief blocking the agent's way. As soon as the policy discovers this tile, the agent tries to go through this tile. Four fifths of the tries will bring a positive reward for the agent, but one fifth of the tries will bring a substantially negative reward for the agent (finding

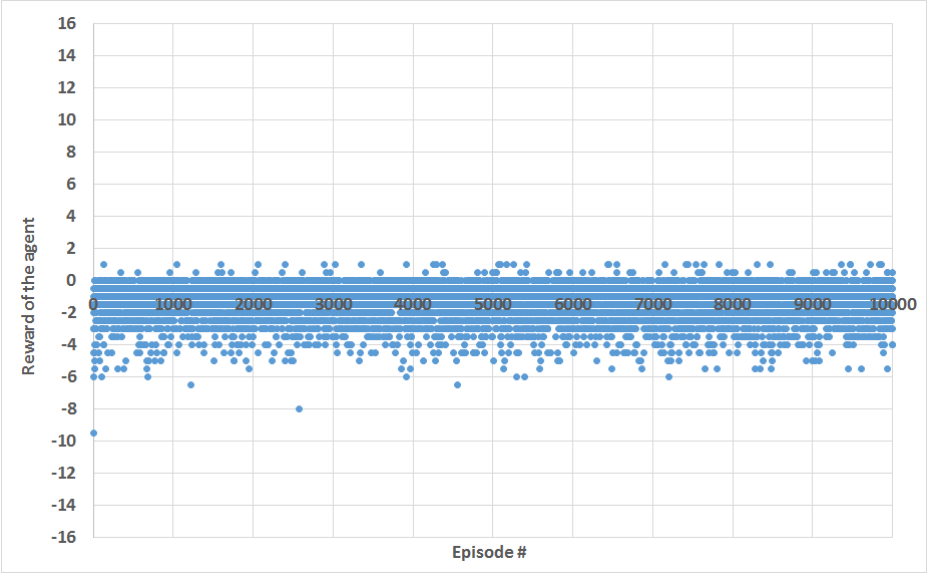

Figure 4: Episodes of a simulation in WorldWithThief not knowing the thief's location.

the thief). For those times that make the agent find the thief, the utility of that one path reduces, and the agent's policy starts avoiding it.

(b) Knowing the location of the thief in the previous item sharply increases the performance of our agent. As shown in fig. 5, the initial tendency of the rewards is ascendant, and at the end of the simulation, rewards as big as 36.0 begin to appear. The average reward of this simulation was 26.9.

As the agent knows where the thief is, it is easy to avoid him. As soon as the agent finds the line that's only blocked by the thief one fifth of the time, he knows when to pass. As soon as the agent discovers how to use that path, the reward increases substantially. Some reward drops in the simulation may happen due to either ϵ or dead ends (having to pass through slippery tiles to avoid the thief, who has higher penalty).

(c) One way to find the best learning rate and ϵ (considering the previous case) is to fix one of the variables and find the optimal value for the other one. This method only works if our function mapped in this region ($\{\alpha, \epsilon\} \in [0,1] \times [0,1]$)

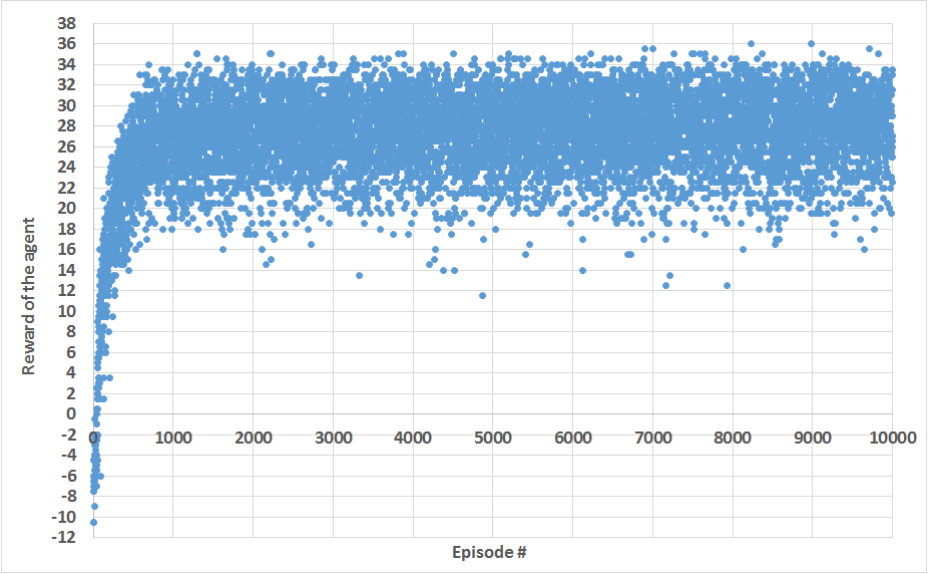

Figure 5: Episodes of a simulation in WorldWithThief knowing the thief's location.

actually has this extreme value being either concave (for maximum) or convex (for minimum), which is a reasonable assumption.

Let $\epsilon = 0.05$ be the closest to the optimal value, and let's change the learning rate accordingly. Our references will be the mean reward and the biggest reward over seven simulations of each value (keeping only the biggest value for each one):

```
alpha Average Biggest
0.01 26.29 36.0
0.05 27.02 36.0
0.08 27.27 36.0
0.09 27.39 36.0
0.1 27.16 37.0
0.2 26.91 36.0
0.5 23.78 35.0
0.8 18.66 32.0
```

Using the average as reference (as the biggest values were close to each other), and using our assumption, we can use $\alpha = 0.09$ as our optimal value. Doing the same process with ϵ:

```
epsilon Average Biggest
0.001 22.38 33.0
0.005 30.56 37.0
0.008 31.46 38.0
0.009 31.51 38.0
0.01 31.69 38.0
0.011 31.13 38.0
0.03 29.78 37.0
0.05 27.39 36.0
0.15 14.87 30.0
```

Using our assumptions, we can use $\epsilon = 0.01$ as out optimal value. A legitimate proof of optimality needs to consult every point in our domain, and these calculations would iterate over hundreds or even thousands of simulations (so our average gets closer to the statistical average). However, this raw calculations give us an idea of where the optimal values are.

3. The first plot, shown in fig. 6, has the rewards of the first simulation using the optimal values. The second plot, shown in fig. 7, has the average of ten simulations for each episode.

As previously seen in fig. 5, the agent in fig. 6 starts with an ascending tendency, reaching the biggest rewards after a thousand episodes. However, considering the effects of ϵ and eventual dead ends, the rewards scatter over each episode, even when the tendency becomes stable at the final episodes.

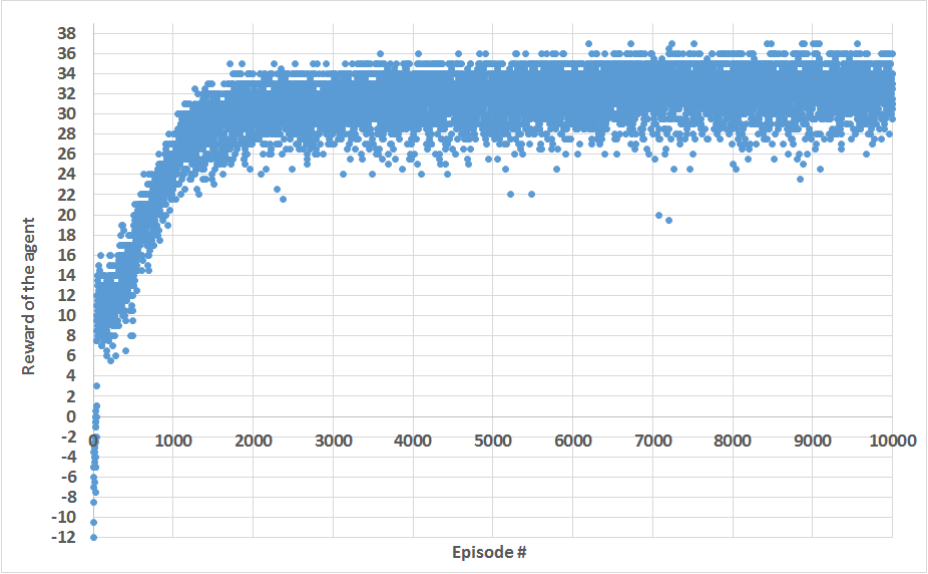

Figure 6: Rewards for each episode of the first simulation using optimal values for (α, ϵ).

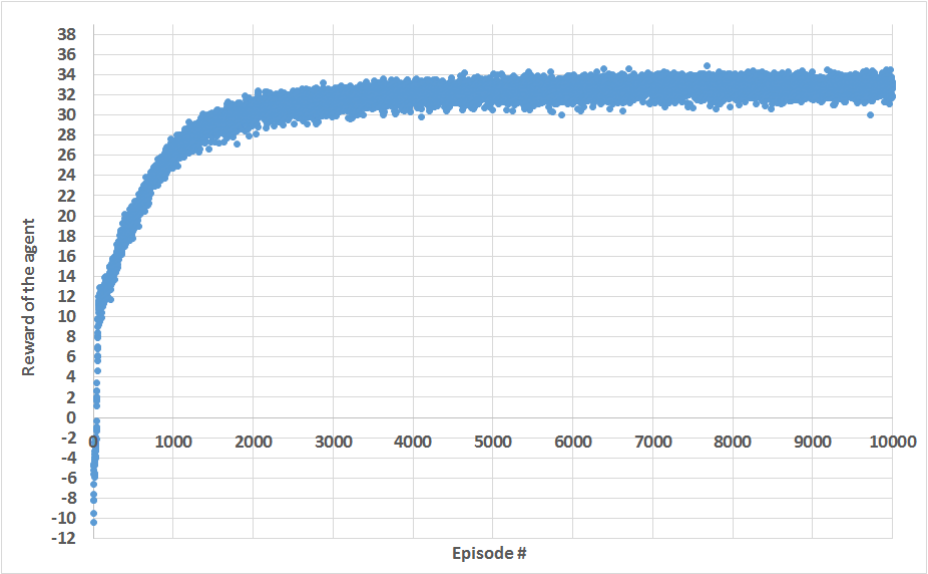

Figure 7: Rewards for each episode of ten simulations using optimal values for (α, ϵ).

In fig. 7, we see that the scattering reduces visibly. This happens because, as an effect of the average, residual effects become less significant.