AI Project 3

Arpan Gupta and Fawaz Tahir

April 2021

1 Questions and Writeup

1.1 Problem 1

Problem 1 tasks us with figuring out a Probability calculation. This calculation will be key in figuring out how to implement Basic Agent 1. The first thing we take note of is the $\mathrm{Observations_t}$. It simply symbolizes all of the knowledge and information we have regarding the board. We can choose the expand the equation by "ignoring" $\mathrm{Observations_t}$ for now. This gives us

$$P(TargetinCell_i|FailureinCell_j)$$

Now, to continue our derivation, we can say this equals:

$$\frac{P(TargetinCell_i \wedge FailureinCell_j)}{P(FailureinCell_j)}$$

Using rules of probability, we can expand the numerator, and marginalize the denominator. This gives us a final statement of:

$$\frac{P(TargetinCell_i)}{P(TargetinCell_j) * P(Targetnotfoundincell_i)|Targetisincell_{\mathrm{i}}) + (1 - P(TargetinCell_{\mathrm{j}})}$$

The one main instance where this will change is when we want to update the probability of the cell we are currently in. This is because in this instance, i will be equal to j since the cell we just opened is the cell we are trying to update. This gives us the updated formula of:

$$\frac{P(TargetinCell_i) * P(Targetnotfoundincell_i)|Targetisincell_{\mathrm{i}})}{P(TargetinCell_j) * P(Targetnotfoundincell_i)|Targetisincell_{\mathrm{i}}) + (1 - P(TargetinCell_{\mathrm{j}})}$$

In terms of our program, we have all of this given information in order to be able to compute these probabilities. We will use this to update the probability that the Target is in each for every iteration of our agent.

1.2 Problem 2

Given the observations up to time t, the belief state captures the current probability the target is in a given cell. What is the probability that the target will be found in Cell i if it is searched:P(Target found in Cell i— Observations t)?

To understand this problem, we had to see what exactly does it mean for a Target to be found in $\mathrm{cell_i}$. A target can be found in a cell given two conditions. The first is that $\mathrm{cell_i}$ contains the target and the second is that the search is successful. This means that the P[Target found in $\mathrm{cell_i}$] = P[Target in $\mathrm{cell_i}$ $\wedge$ successful search in $\mathrm{cell_i}$]. The P[successful search in $\mathrm{cell_i}$] = 1-P[Target not found in $\mathrm{cell_i}$ | Target is in $\mathrm{cell_i}$]. This gives us the equation:

$$P(TargetFoundInCell_{\mathrm{i}}|Observations_{\mathrm{t}}) = P(TargetInCell_{\mathrm{i}} \wedge SuccessfulSearch_{\mathrm{i}}|Observations_{\mathrm{t}})$$

.
This can be split up using the Probability Rules. After splitting it up and distributing the Observations_t, we get this:

$$P(TargetFoundInCell_i|Observations_t) = P(TargetInCell_i|Observations_t) \wedge P(SuccessfulSearch_i|Observations_t)$$

.
From here we can determine that the second part of that right hand side of the equation is unnecessary since the probability of a successful search is independent of the observations at time t. This means that can be canceled out and we are left with the following

$$P(TargetFoundInCell_i|Observations_t) = P(TargetInCell_i|Observations_t) \wedge P(SuccessfulSearch_i)$$

where P(Target Found In Cell_i | Observations_t) is the probability expression we derived in problem 1. The P(Successful Search_i) is 1 minus the probability of a false negative or 1-P(Target not found in cell_i | Target is in cell_i)

1.3 Problem 3

We can start the discussion with Basic Agent 1. This agent draws on the formula we computed for Problem 1. The agent wants to travel to the cell containing the highest probability of containing the target, and then search it. This means it does not generally account for the fact that the cell may return a false negative. Because of this, we can expect to see poor results when we run the agent. In order to check if the agent searched the target cell, and actually found the target, we use a random number between 0 and 1 and compare it to the probability of a false negative. If that random number is greater than the probability of a false negative, then we say the agent returns Success. In each iteration, we search the current cell, check if it returns Failure or Success, if it returns Failure, we update the rest of the board with our calculations from Problem 1, as well as the cell we just opened. We then look for the cell that has the highest probability, and move to that cell, repeating the process till a cell returns Success.
We can now move on to Agent 2. Agent 2 differs slightly from Agent 1 in the fact that it cares about travelling to the cell with the highest probability of actually finding the target. In Agent 1, we just cared if the cell contained the target and at which probability. This introduces the idea of the equation we calculated in Problem 2, where we care about what terrain the cell actually has, since having a terrain of a complex maze has a very high chance of being a false negative, whereas flat terrain has a relatively low change of being a false negative. We also implemented this agent in the same way with the change in how we pick the next cell. The key note to make here is the probabilities for each cell still remain the cell, the only difference occurs in how we pick the next cell to open. We pick the next cell based on the chances that we find the target in that cell.
After implementing both agents, we came across some interesting findings. In most cases, with a big map and a lot of repitions, Agent 2 was almost always better than Agent 1. On the other hand, for some of the smaller maps and fewer repetitions, Agent 1 was sometimes coming out better. After tracing through some maps, we realized this was due to a couple of factors. The first one being the fact that the computer becomes very unprecise the more times a number get modified. In order to try and fix this issue, we converted all of our numbers to hex and then used them. This greatly helped fix the issue, but we were still seeing similar results with Agent 1 sometimes beating Agent 2. Our explanation for this relied on two main factors: the fact that this had a high factor of luck in where we started and what cells we went to. This would be addressed in our advanced agent. Furthermore, the other factor is the terrain type. When the target was in cells like Maze of Caves and Forest, we were noticing better results in Agent 1 then Agent 2. This made sense because Agent 2 would almost avoid the cells with low probabilites of being found whereas Agent 1 did not care. We tried addressing this issue in the advanced agent as well.
For the data that we recieved on a maze of size 50x50, and after running 40 repitions, we got an average score of:
Agent 1: 20,493
Agent 2: 15,753

1.4 Problem 4

To improve the agent from basic agents 1 and 2, we decided to add two new features. The first was to add a factor that took into account the direct next move of the agent. To do this, we modified the action the agent would take if it came across a scenario where more than one cell contains the next highest probability of the target being found in the cell. When this situation comes up we currently look at the Manhattan distance of the new cell and the current cell and pick the one that is closest. Instead, we now take a look of the sum of Manhattan distances of the current cell to the next cell and the next cell to the cell with the second highest probability. The reason we search for this is because we want to account for the impact of how selecting a cell affects future decisions. With our implementation, the agent will search for its next cell with future moves in mind. It will want to select a cell that requires as little movement as possible in its next following move. The other feature we added was that we didn't want the agent to move more than 50 spots in one move. We felt that moving this much diminished the value of the cell with the highest probability so instead the agent would select the cell with the highest probability within a range of 50 Manhattan distance. This improved agent gave us an average score of 8841.48 when run 25 times on a 50x50 board.

1.5 Acronym

For the acronym we came up with the MAD Agent. This stands for the Map Adjusted Distance Agent due to the impact of the Manhattan distance on our improved agent.

1.6 Statement and Contributions

Arpan:

Statement: "All of my work is my own and I did not copy or take from online or any other student's work."

Contributions

1. Worked on Improved Agent
2. Assisted on Basic Agents 1 and 2
3. Answered Problem 1 and 3
4. Assisted on writing report and everything Fawaz did

Fawaz:

Statement: "All of my work is my own and I did not copy or take from online or any other student's work."

Contributions

1. Worked on Basic Agents 1 and 2
2. Assisted on Improved Agent
3. Answered Problem 2 and 4
4. Assisted on writing report and everything Arpan did