ASSIGNMENT 3
CS520

Ravi Teja Pothamsetty
Dipayan Dutta
Harsh Rajkumar Vaish
Vinodh Reddy Velemineti

PART 1

1. Given observations up to time *t* (Observations$_t$), and a failure searching Cell *j* (Observations$_{t+1}$ = Observations$_t$ ^ Failure in Cell$_j$), how can Bayes' theorem be used to efficiently update the belief state, i.e., compute: P (Target in Cell$_i$ | Observations$_t$ ^ Failure in Cell$_j$)?

Ans: Initially, the target can be present at any one of the (n^2) cells available to us, where n represents the size of the matrix. This probability of target being in a cell is equal to 1/($size^2$) for all cells. This is our initial belief.

Successively we search the cell with maximum belief and make the following changes in our belief matrix:

- If the target is present in the cell then we end our search and print the number of cells we had to search to accurately predict the target cell position.
- Else, we update the belief of the cell searched as following:

Belief(Cell[j]) = P(target in Cell[j] | Observation of failure in Cell[j]) (1)

We apply Bayes Rule to obtain the belief in terms of our known variables,

Belief(Cell[j]) = *beta* * P(Observation of failure in Cell[j] | target in Cell[j]) * P(target in Cell[j])

beta = P(Observation of failure in Cell[j]) = P(Observation of failure in Cell[j] and Target in Cell[j]) + P(Observation of failure in Cell[j] and Target not in Cell[j])

=> *beta* = P(Observation of failure in Cell[j] | Target in Cell[j]) P(Target in Cell[j]) + P(Observation of failure in Cell[j] | Target not in Cell[j]) P(Target not in Cell[j])

P(Target in Cell[j]) = Prior belief of Target in Cell[j]
P(Observation of failure in Cell[j] | Target in Cell[j]) = False negative value for Cell[j]
P(Observation of failure in Cell[j] | Target not in Cell[j]) = 1, If the target is not in Cell[j], then it must be present elsewhere in the board.
P(Target not in Cell[j]) = 1 - P(Target in Cell[j])

Using these known values we can calculate Belief(Cell[j]).

- Based on the change in belief at Cell[j], we calculate the belief in the remaining cells Cell[i] by proportionately distributing the change among the belief of the remaining cells.

  This is done by,

  Belief(Cell[i]) = $Belief(Cell[i]) + \frac{\Delta(Belief(Cell[j])}{1-Belief(Cell[j]} * Belief(Cell[i])$ **(2)**

In this question, we are asked to calculate the P (Target in Cell_i | Observations_t ^ Failure in Cell_j)

P (Target in Cell_i | Observations_t ^ Failure in Cell_j) = P(Target in Cell_i ^ Observations_t ^ Failure in Cell_j) / P(Observations_t ^ Failure in Cell_j)

= *beta* * P(Observations_t) * P(Target in Cell_i | Observations_t) * P(Failure in Cell_j | *T*arget in Cell_i ^ Observations_t)

where *beta* = *P(Failure in* Cell_j *^* Observations_t*)*

We marginalize the normalization factor for all possible cells where the target can be present.

So, *beta* = sum for all i (P(Failure in Cell_j ^ Target in Cell_i | Observations_t) * P(Observations_t))

Finally our equation is:
Obs_t: Observations till time t
P (Target in Cell_i ***|*** Obs_t ***^ Failure in*** Cell_j***) =***
{P(Target in Cell_i ***|*** Obs_t***) * P(Failure in*** Cell_j ***| Target in*** Cell_i ***^*** Obs_t***)} /***
{Σ_i ***(P(Failure in*** Cell_j ***^ Target in*** Cell_i ***|*** Obs_t***) * P(***Obs_t***))}***

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$_t$)?

Ans:

P(Target found in cell$_i$ | Obs$_t$) = P(Target found in cell$_i$ ^ Target in cell$_i$ | Obs$_t$) + P(Target found in cell$_i$ ^ Target not in cell$_i$ | Obs$_t$)

(Marginalizing based on whether the target is in cell i or not in cell i)

=> P(Target found in cell$_i$ | Obs$_t$) = P(Target found in cell$_i$ ^ Target in cell$_i$ | Obs$_t$)

(P(Target found in cell$_i$ ^ Target not in cell$_i$ | Obs$_t$) = 0)

=> **P(Target found in cell$_i$ | Obs$_t$) = P(Target found in cell$_i$ | Target in cell$_i$) P(Target in cell$_i$ | Obs$_t$)**

3) Consider comparing the following two decision rules:

Rule 1: At any time, search the cell with the highest probability of containing the target.

Rule 2: At any time, search the cell with the highest probability of finding the target.

For either rule, in the case of ties between cells, consider breaking ties arbitrarily. How can these rules be interpreted / implemented in terms of the known probabilities and belief states?

For a fixed map, consider repeatedly using each rule to locate the target (replacing the target at a new, uniformly chosen location each time it is discovered). On average, which performs better (i.e., requires less searches), Rule 1 or Rule 2? Why do you think that is? Does that hold across multiple maps?

Rule 1: At any time, search the cell with the highest probability of containing the target.
Rule 2: At any time, search the cell with the highest probability of finding the target.

How can these rules be interpreted / implemented in terms of the known probabilities and belief states?

- Rule 1:
  - This can be implemented using the belief values of each cell i
  - At each time t, we will select the the cell with the Maximum belief
- Rule 2:
  - This can be implemented using the formulations stated in Question 2
  - In this case, the next cell to be explored is decided based on the type of terrain with flat terrain explored compared to the cave terrain
  - **Exploration Order:** Flat > Hilly > Forested > Caves

For a fixed map, On average, which performs better (i.e., requires less searches), Rule 1 or Rule 2? Why do you think that is?

We used average number of searches required to locate the target as the performance metric based on the type of terrain in which target lies.

Average number of searches required to locate the target based on target position							
Flat		**Hilly**		**Forested**		**Caves**	
Rule 1	*Rule 2*	*Rule 1*	*Rule 2*	*Rule 1*	*Rule 2*	*Rule 1*	*Rule 2*
1243.96	251.19	1105.36	886.79	1283.52	1635.1	1238.09	5221.7

From the above table we observe that

1. Rule 2 beats Rule 1 when target is present in ***Flat*** terrain as Rule 2 starts exploration starting with ***Flat*** terrains compared to Rule 1 which looks at whole area. Since only 20 percent of cells are ***Flat,*** Rule 2 search space is greatly reduced
2. Rule 2 beats Rule 1 when target is present in ***Hilly*** terrain for the same reason as Rule 2 looks at ***Hilly*** terrains soon after exploring ***Flat*** areas while Rule 1 looks at whole area. Max number of unique cells Rule 2 looks at in this case is 50 percent ***(Flat plus Hilly)***
3. Rule 1 beats Rule 2 when target is present in ***Forested*** terrain because Rule 2 has to explore minimum 50 percent of cells ***(Flat plus Hilly)*** in any case where as it is not case for Rule 1.
4. Rule 1 beats Rule 2 when target is present in ***Caves*** terrain because Rule 2 has to explore minimum 80 percent of cells ***(Flat plus Hilly plus Forested)*** in any case where as it is not case for Rule 1.

On the whole, regardless of target location we can compare Rule 1 and Rule 2 based on Expected number of searches required .

Expected number of searches required = Sum of (P(terrain types) x Average number of searches required when target is located in that Terrain)

- Rule 1:
    - Expected Number = (0.2 x 1243.96) + (0.3 x 1105.36) + (0.3 x 1283.52) + (0.2 x 1238.09) = 1213
    - Average number of searches required = 1213
- Rule 2:
    - Expected Number = (0.2 x 251.19) + (0.3 x886.79) + (0.3 x 1635.1) + (0.2 x 5221.7) = 1851.14
    - Average number of searches required = 1851.14

We can see that, on the whole Rule 1 out performs Rule 2.

Does that hold across multiple maps?

No, this kind of pattern doesn't hold for multiple maps because expected number of required searches depends on the distribution of terrain type.

For example, Suppose the target is in ***Cave*** terrain,

We take two different maps, Map 1 and Map 2

Distribution of terrain type in Map 1: All of the cells are of terrain type **Cave**

Distribution of terrain type in Map 2: Only 10 percent of cells are of terrain type **Cave**

Rule 1 and Rule 2 perform the same in case of Map 1,

Rule 1 outperforms Rule 2 in case of Map 2 because Rule 2 has to search 90 percent of cells before searching **Cave** Type cells.

Hence terrain type distribution determines which rule outperforms, So, we can't determine a definite relation for multiple maps between Rule 1 and Rule 2.

4) Consider modifying the problem in the following way: at any time, you may only search the cell at your current location, or move to a neighboring cell (up/down, left/right). Search or motion each constitute a single 'action'. In this case, the 'best' cell to search by the previous rules may be out of reach, and require travel. One possibility is to simply move to the cell indicated by the previous rules and search it, but this may incur a large cost in terms of required travel. How can you use the belief state and your current location to determine whether to search or move (and where to move), and minimize the total number of actions required? Derive a decision rule based on the current belief state and current location, and compare its performance to the rule of simply always traveling to the next cell indicated by Rule 1 or Rule 2. Discuss.

Ans:
In this case, we will move if the expected number of searches at current cell is less than the cost incurred to travel to a location plus expected number of searches at that cell. (we will choose that cell which has minimum value of travel plus number of searches combined).
So, we use a hypothesis to decide whether to move or search that cell.
For Rule-1:
New_probability_matrix = Belief_matrix **/** (1+log(distance_to_cover))
For Rule-2:
New_probability_matrix = Prob_find-matrix **/** (1+log(distance_to_cover))

Using our proposed approach

Average number of searches required to locate the target based on target position							
Flat		**Hilly**		**Forested**		**Caves**	
Rule 1	*Rule 2*	*Rule 1*	*Rule 2*	*Rule 1*	*Rule 2*	*Rule 1*	*Rule 2*
1098.0	497.0	1590.7	1343.4	1781.8	3243.6	1920.1	5320.2

Simple Shifting to best position

Average number of searches required to locate the target based on target position							
Flat		**Hilly**		**Forested**		**Caves**	
Rule 1	*Rule 2*	*Rule 1*	*Rule 2*	*Rule 1*	*Rule 2*	*Rule 1*	*Rule 2*
46391.0	8225.4	47380.2	29017.2	39261.1	51469.1		76542.5

						32020.32	

5) An old joke goes something like the following:
A policeman sees a drunk man searching for something under a streetlight and asks what the drunk has lost. He says he lost his keys and they both look under the streetlight together. After a few minutes the policeman asks if he is sure he lost them here, and the drunk replies, no, and that he lost them in the park. The policeman asks why he is searching here, and the drunk replies, "the light is better here". In light of the results of this project, discuss.

Ans:
Let **$P(F_i|O_t)$ = P(finding the target at cell i | Observations upto time t)**
The above joke refers to the phenomenon known as ***"observational bias"***. Even though he knows that he lost his keys in the park where it's dark(analogous to belief in our project used in Rule 1), the drunkard looks only at the places where there is light, i.e. where it's easiest to find the keys(analogous to selecting the cell having highest probability of finding the target & consequential in choosing the next cell in Rule 2). In our project too, we introduce observational bias by choosing cells which have the highest probability of finding the target. While doing this, we ignore cells that actually contain the target but don't have a high $P(F_i|O_t)$.

Using the statistics mentioned in the Question 3, we can clearly observe that if the target is present in easier terrains, i.e. where the probability of finding target is higher, Rule 2 performs better than Rule 1 as Rule 2 prefers easier terrains to harder ones. Rule 1 ,however, doesn't harbour any such bias and prefers cells based on their belief whatever their terrains might be.

Part 2

The target is no longer stationary, and can move between neighboring cells. Each time you perform a search, if you fail to find the target the target will move to a neighboring cell (with uniform probability for each).

However, all is not lost - whenever the target moves, surveillance reports to you that the target was seen at a Type1 x Type2 border where Type1 and Type2 are the cell types the target is moving between (though it is not reported which one was the exit point and which one the entry point.

Implement this functionality in your code. How can you update your search to make use of this extra information?
How does your belief state change with these additional observations? Update your search accordingly, and again compare Rule 1 and Rule 2.

Initially, we don't have any evidence to assign different belief values to each cell i. The target can be present at any one of the (n^2) cells available to us, where n represents the size of the matrix. This probability of target being in a cell is equal to 1/(size^2) for all cells.

We start with exploring a random $cell_i$, Now we have evidence whether the cell moved from type 1 to type 2 or vice versa using surveillance reports. Below is how we update our beliefs based on the evidence,

- Now we know that target will either be present in Type 1 or Type 2.
    - ***Beliefs of Terrain types other than Type 1 or Type2 (Indicated in Red):*** So, we can mark belief of the cells (target present in that cell) of terrain types other than this as **Zero** like P(Cells of Type 3) = 0, P(Cells of Type 4) = 0

Type3	***Type 1***	***Type 1***	***Type 4***
Type3	***Type 2***	***Type 1***	***Type 4***
Type3	***Type3***	***Type3***	***Type 4***
Type3	***Type 4***	***Type 2***	***Type 4***

    -
    - ***Beliefs of Cells of Type 2 and No Adjacent cells of Type 1 (Indicated in Red):*** P(Target in Cell with Type 2 **and** No adjacent Cells of Type 1) = 0 because if the target is present in that cell, then it can't move, but according to our evidence it moves which contradicts. So target cannot be present in this cell

Type3	***Type 1***	***Type 1***	***Type 4***
Type3	***Type 2***	***Type 1***	***Type 4***
Type3	***Type3***	***Type3***	***Type 4***

Type3	***Type 4***	***Type 2***	***Type 4***

- ***Beliefs of Cells of Type 1 and No Adjacent cells of Type 2 (Indicated in Red):***
  Similarly, for Type 1, P(Target in Cell with Type 1 **and** No adjacent Cells of Type 2) = 0

Type3	***Type 1***	***Type 1***	***Type 4***
Type3	***Type 2***	***Type 1***	***Type 4***
Type3	***Type3***	***Type3***	***Type 4***
Type3	***Type 4***	***Type 2***	***Type 4***

- ***Beliefs of Cells of Type 2 with Adjacent cells of Type 1 (Indicated in Red):***
  In this case, we distribute equally the belief of cell to its adjacent of cells of desired Type. For example, a cell has probability 0.3 and 3 adjacent cells, then 0.1 is added to each of its adjacent cells

Type3	***Type 1***	***Type 1***	***Type 4***
Type3	***Type 2***	***Type 1***	***Type 4***
Type3	***Type3***	***Type3***	***Type 4***
Type3	***Type 4***	***Type 2***	***Type 4***

For a fixed map,

Average number of searches required to locate the target based on target position							
Flat		**Hilly**		**Forested**		**Caves**	
Rule 1	*Rule 2*	*Rule 1*	*Rule 2*	*Rule 1*	*Rule 2*	*Rule 1*	*Rule 2*
536.9	1762.3	837.1	4204.5	1477.9	2264.7	857.7	2615.1

From the above table calculated using the evidence information,
we observe that Rule 1 outperforms Rule 2 when target is present in any terrain type

For the given fixed map,

- Expected Average number of searches for Rule 1 = 973.42
- Expected Average number of searches for Rule 1 = 2816.24
- On the whole, Rule 1 outperforms Rule 2

Average number of searches required to find the target is less when we use the surveillance reports.