CS520: AI Project 3-Probabilistic Search

Leena Dighole

1. Introduction

Aim: Probabilistic searching of cells with different rules to find target cell successfully.

The landscape contains cells with 4 terrains. The target is randomly located in the map and our aim is to search the target in minimum number of search rounds. The map of 10*10 grid can be viewed as below,

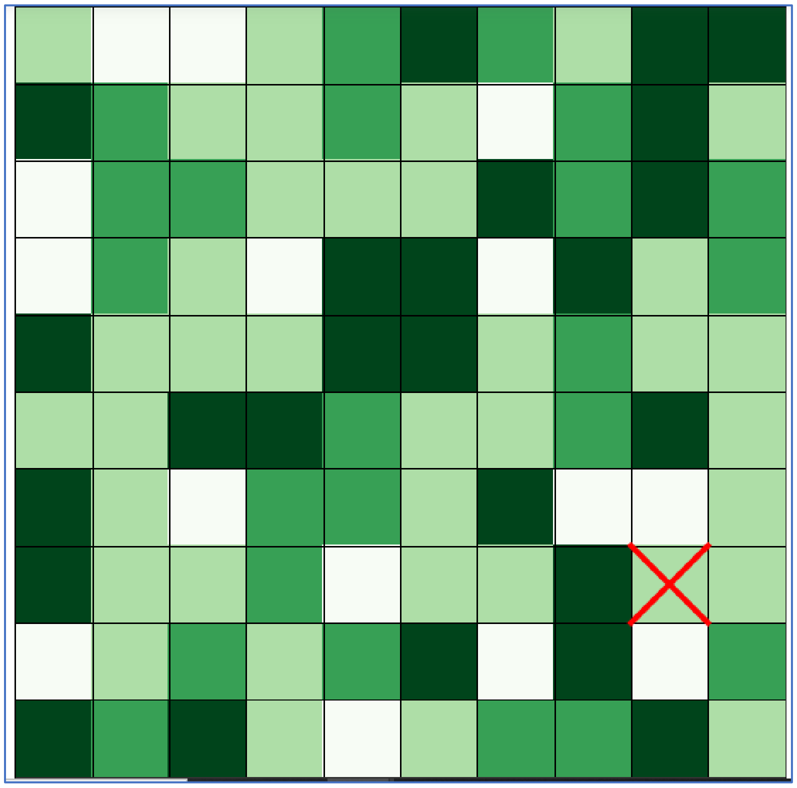


*Figure 1: 10*10 Map of cells*

For the 10*10 grid, the target is located at the cell indicated by the red cross in above figure.

The Target node is: (7, 8)

The Target terrain is: ('Hilly', 0.3)

The target terrain is Hilly which has false negative probability equals 0.3

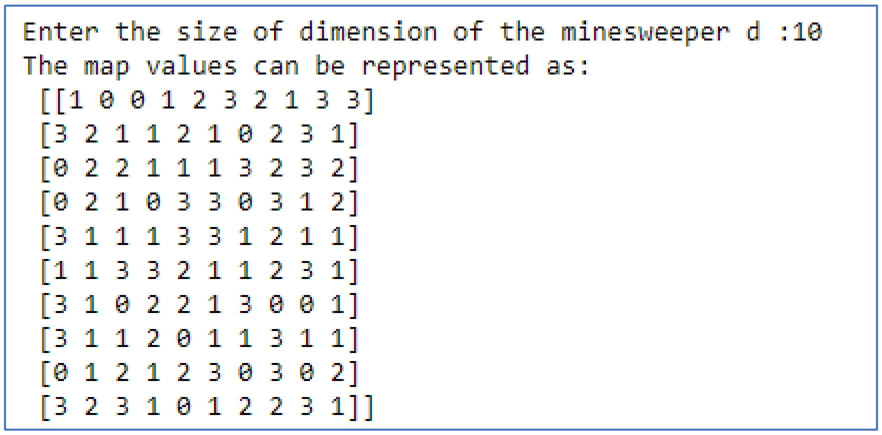
The map has 4 terrains and 4 different colors represents different terrains.

Terrain and corresponding value on the map are:

Sr No	Terrain	Probability of cells in map	Value on map	color on map
1	Flat	0.2	0	White
2	Hilly	0.3	1	Very light green
3	Forested	0.3	2	Light green
4	Cave	0.2	3	Dark green

Table 1: Different terrains and map probabilities

The white color on the map shows the flat terrain which is mapped with 0 value. Similarly, the Hilly terrain is mapped with very light green color and value 1. The Light green color represents Forested area with a value 2 and dark green color represents the cave terrain and has assigned value 3. For the above colored 10*10 map, the value matrix of the map will be:



```
Enter the size of dimension of the minesweeper d :10
The map values can be represented as:
 [[1 0 0 1 2 3 2 1 3 3]
 [3 2 1 1 2 1 0 2 3 1]
 [0 2 2 1 1 1 3 2 3 2]
 [0 2 1 0 3 3 0 3 1 2]
 [3 1 1 1 3 3 1 2 1 1]
 [1 1 3 3 2 1 1 2 3 1]
 [3 1 0 2 2 1 3 0 0 1]
 [3 1 1 2 0 1 1 3 1 1]
 [0 1 2 1 2 3 0 3 0 2]
 [3 2 3 1 0 1 2 2 3 1]]
```

*Figure 2: Map values of different terrains for 10*10 map*

Lets start searching the target in the map by randomly choosing initial node. The initial node is (3,5).

We have given the False Negative (FN) rate of the terrains. The FN shows the probability of finding the target in given cell if we search the cell. As the FN increases, the searching in the respective terrain increases. The number of rounds of searches increases as the False Negative rate increases. The False Negative rate is the probability with which a cell search result is negative knowing the cell contains the target. The False Positive rate is zero.

For different terrains, the FN rates are different.

Flat Terrain has FN =0.1

Hilly Terrain has FN =0.3

Forested Terrain has FN =0.7

Cave Terrain has FN =0.9

Looking at above FN rates of terrains, we can intuitively say that searching a target in a cave cell will need a lot of searches compared to Flat terrain as FN rate of Cave is more than the Flat terrain.

We will keep a track of target matrix and found matrix at each search iteration. At the start of search, there is equal probability of containing target in each cell of the map.

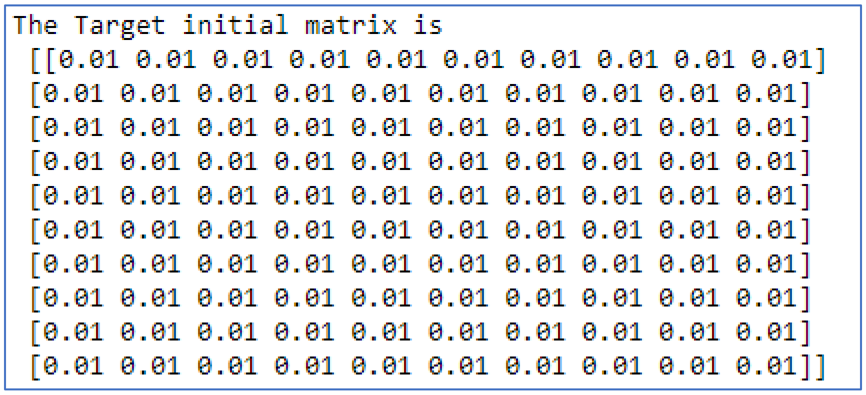
For 10*10 map,

Initially, The target containing matrix will have equal chances of containing target in each cell.

P(Target in cell i) = 1/No of cells

=1/100

=0.01.



```
The Target initial matrix is
 [[0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]
 [0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]
 [0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]
 [0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]
 [0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]
 [0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]
 [0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]
 [0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]
 [0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]
 [0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]]
```

*Figure 2: Initial Target matrix of 10*10 map*

Note that if we get failure in search of any cell then this target matrix gets updated by the additional information of knowing the failure in searched cell at any time t.

This target matrix will keep updating based upon bayes rule with the information of knowing a failure in particular searched cell. We will derive this formula and update the belief of cell containing target with each search iteration.

The second matrix which is a found matrix contains the target finding probability of cells.

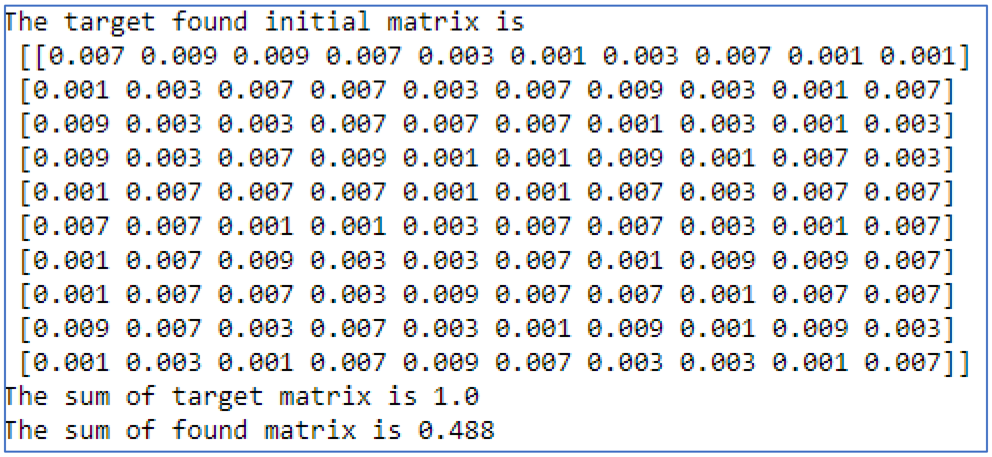
Let Target probability of cell i is ,

B(cell i) = P(target in cell i/Observations_t)

The P(Target found in cell i/Observations_t) = B(cell i)*(1-FN)

Where FN is the false negative rate of the cell knowing its terrain.

The initial found matrix for above 10* grid will be,



```
The target found initial matrix is
 [[0.007 0.009 0.009 0.007 0.003 0.001 0.003 0.007 0.001 0.001]
 [0.001 0.003 0.007 0.007 0.003 0.007 0.009 0.003 0.001 0.007]
 [0.009 0.003 0.003 0.007 0.007 0.007 0.001 0.003 0.001 0.003]
 [0.009 0.003 0.007 0.009 0.001 0.001 0.009 0.001 0.007 0.003]
 [0.001 0.007 0.007 0.007 0.001 0.001 0.007 0.003 0.007 0.007]
 [0.007 0.007 0.001 0.001 0.003 0.007 0.007 0.003 0.001 0.007]
 [0.001 0.007 0.009 0.003 0.003 0.007 0.001 0.009 0.009 0.007]
 [0.001 0.007 0.007 0.003 0.009 0.007 0.007 0.001 0.007 0.007]
 [0.009 0.007 0.003 0.007 0.003 0.001 0.009 0.001 0.009 0.003]
 [0.001 0.003 0.001 0.007 0.009 0.007 0.003 0.003 0.001 0.007]]
The sum of target matrix is 1.0
The sum of found matrix is 0.488
```

Figure 3: Target found matrix for 10 10 map*

We will keep track of target matrix and found matrix and update them with each searches.

The search results will be Failure in 2 cases:

- **When cell does not contain target then the search result should be failure.**
- **When cell contains the target and due to FN, the search result is Failure.**

The Success result will be when cell contains target and searches it correctly with given FN probability.

The sanity check of the algorithm is checking the sum of all the individual probability adds upto 1. The target matrix will add upto 1. But the found matrix may not because of the multiplication factor of False negative rate.

But the target matrix must adds up to 1 as the probabilistic belief of a cell containing target adds to 1. Knowing the failure in particular search, we will update our belief about the target located in that cell.

2.Baye's Theorem calculations

Initially we assume P(Target in cell i) =B(cell i)= 1/ Total no of cells.

Lets calculate **P(Target found in cell i),**

P(Target found in cell i) = P(Target found in cell i, Target in i)

+ P(Target found in cell i, Target not in i)

=P(Target in i) * P (Target found in cell i / Target in i)

+ P(Target not in i) * P (Target found in cell i / Target not in i)

…….. Using Baye's Theorem

=B(cell i) *(1- FN) + (1- B(cell i)) * FP

…. FN represents False Negative rate

…. FP represents False Positive rate

= B(cell i) *(1- FN) …… FP = 0 (given)

Thus, we can calculate found matrix using target matrix with above formula.

P(Target found in cell i) = B(cell i) *(1- FN) …………….(1)

Initially we choose a random cell j to be searched and we get failure after searching the cell j.

This Failure will update our target and found matrix for next search.

The next search cell i will be chosen using different rules proposed in the problem statement.

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.

Lets see how the failure in one cell search updates the target matrix and found matrix.

P(Target found in i/ fail j, Observations t)

= P(Target in i, fail in j, Observations t)/ P(fail in j, Observations t) ….Baye's Theorem

=P(Target in i/Observations t)*P(Fail in j/ Target in I, Observations t)/P(Fail in j/Observations t)

=B_t(cell i)* P(Fail in j/ Target in i, Observations t)/P(Fail in j/Observations t)

- Case 1: For i=j

  When we search the cell again in which we just got a failure

  As, P(Fail in i/ Target in i, Observations t) = FN of cell i

  B_{t+1}(cell i) = B_t(cell i) * FN / P(Fail in j/Observations t) …………….(2)

- Case 2: For i !=j

  When we search the new cell apart from the just failed search cell

  As, P(Fail in j/ Target in i, Observations t) = 1 …according to problem statement

  B_{t+1}(cell i) = B_t(cell i) / P(Fail in j/Observations t) …………….(3)

Lets calculate the denominator term using Baye's Theorem.

P(Fail in j/Observations t)

=P(Fail in j, Target in j/ Observations t) + P(Fail in j, Target not in j/ Observations t)

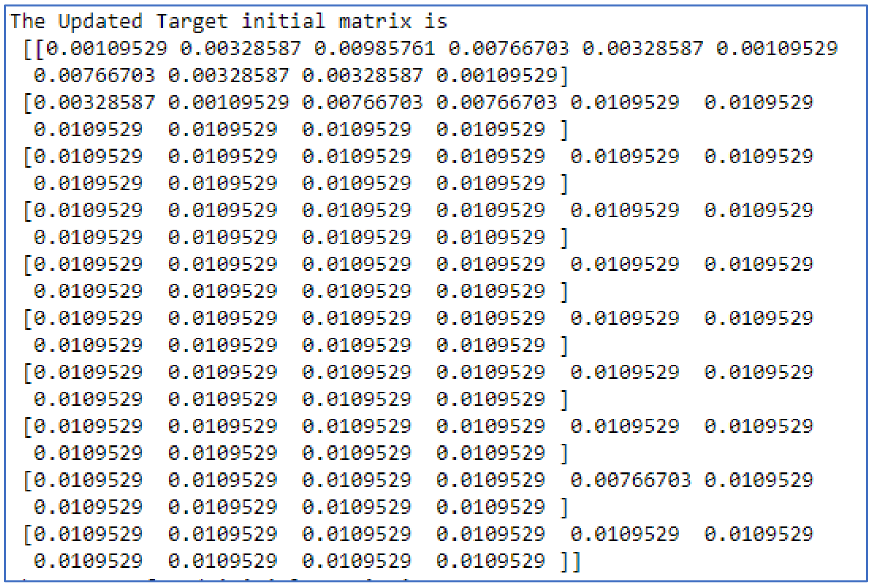
= B_t(cell j) * FN + 1 *(1- B_t(cell j)) …………….(4)

Where,

B_t(cell j) is the probability that the cell j contains target at time t

Using above calculations lets update our belief matrices knowing we got failure result in first search.

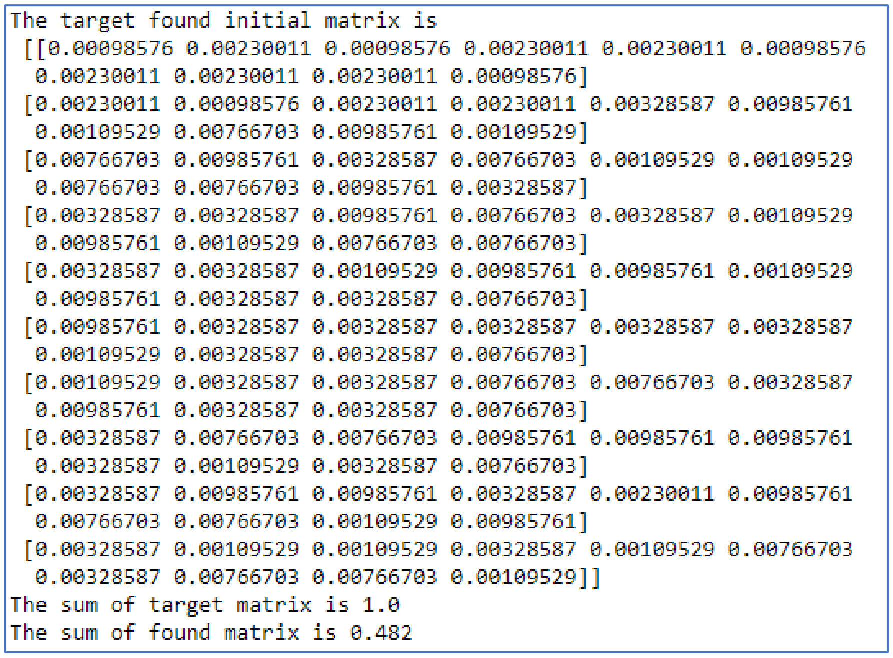
The updated target matrix at any time t will look like,



```
The Updated Target initial matrix is
 [[0.00109529 0.00328587 0.00985761 0.00766703 0.00328587 0.00109529
  0.00766703 0.00328587 0.00328587 0.00109529]
 [0.00328587 0.00109529 0.00766703 0.00766703 0.0109529  0.0109529
  0.0109529  0.0109529  0.0109529  0.0109529 ]
 [0.0109529  0.0109529  0.0109529  0.0109529  0.0109529  0.0109529
  0.0109529  0.0109529  0.0109529  0.0109529 ]
 [0.0109529  0.0109529  0.0109529  0.0109529  0.0109529  0.0109529
  0.0109529  0.0109529  0.0109529  0.0109529 ]
 [0.0109529  0.0109529  0.0109529  0.0109529  0.0109529  0.0109529
  0.0109529  0.0109529  0.0109529  0.0109529 ]
 [0.0109529  0.0109529  0.0109529  0.0109529  0.0109529  0.0109529
  0.0109529  0.0109529  0.0109529  0.0109529 ]
 [0.0109529  0.0109529  0.0109529  0.0109529  0.0109529  0.0109529
  0.0109529  0.0109529  0.0109529  0.0109529 ]
 [0.0109529  0.0109529  0.0109529  0.0109529  0.0109529  0.0109529
  0.0109529  0.0109529  0.0109529  0.0109529 ]
 [0.0109529  0.0109529  0.0109529  0.0109529  0.00766703 0.0109529
  0.0109529  0.0109529  0.0109529  0.0109529 ]
 [0.0109529  0.0109529  0.0109529  0.0109529  0.0109529  0.0109529
  0.0109529  0.0109529  0.0109529  0.0109529 ]]
```

*Figure 4: Updated Target Matrix for 10*10 map*

The updated found matrix will look like,



```
The target found initial matrix is
 [[0.00098576 0.00230011 0.00098576 0.00230011 0.00230011 0.00098576
  0.00230011 0.00230011 0.00230011 0.00098576]
 [0.00230011 0.00098576 0.00230011 0.00230011 0.00328587 0.00985761
  0.00109529 0.00766703 0.00985761 0.00109529]
 [0.00766703 0.00985761 0.00328587 0.00766703 0.00109529 0.00109529
  0.00766703 0.00766703 0.00985761 0.00328587]
 [0.00328587 0.00328587 0.00985761 0.00766703 0.00328587 0.00109529
  0.00985761 0.00109529 0.00766703 0.00766703]
 [0.00328587 0.00328587 0.00109529 0.00985761 0.00985761 0.00109529
  0.00985761 0.00328587 0.00328587 0.00766703]
 [0.00985761 0.00328587 0.00328587 0.00328587 0.00328587 0.00328587
  0.00109529 0.00328587 0.00328587 0.00766703]
 [0.00109529 0.00328587 0.00328587 0.00766703 0.00766703 0.00328587
  0.00985761 0.00328587 0.00328587 0.00766703]
 [0.00328587 0.00766703 0.00766703 0.00985761 0.00985761 0.00985761
  0.00328587 0.00109529 0.00328587 0.00766703]
 [0.00328587 0.00985761 0.00985761 0.00328587 0.00230011 0.00985761
  0.00766703 0.00766703 0.00109529 0.00985761]
 [0.00328587 0.00109529 0.00109529 0.00328587 0.00109529 0.00766703
  0.00328587 0.00766703 0.00766703 0.00109529]]
The sum of target matrix is 1.0
The sum of found matrix is 0.482
```

*Figure 5: Updated Found Matrix for 10*10 map*

Note the sanity check parameter that all the entries **in updated target adds up to 1**. This is not mandatory for updated found matrix.

3.Performance Analysis

Lets select the next cell to be searched at any time t based upon the rule 1 and rule 2.

1. **Terrain and No of searches**

Lets implement the rule 1 for different terrains in which target is located and compare their results.

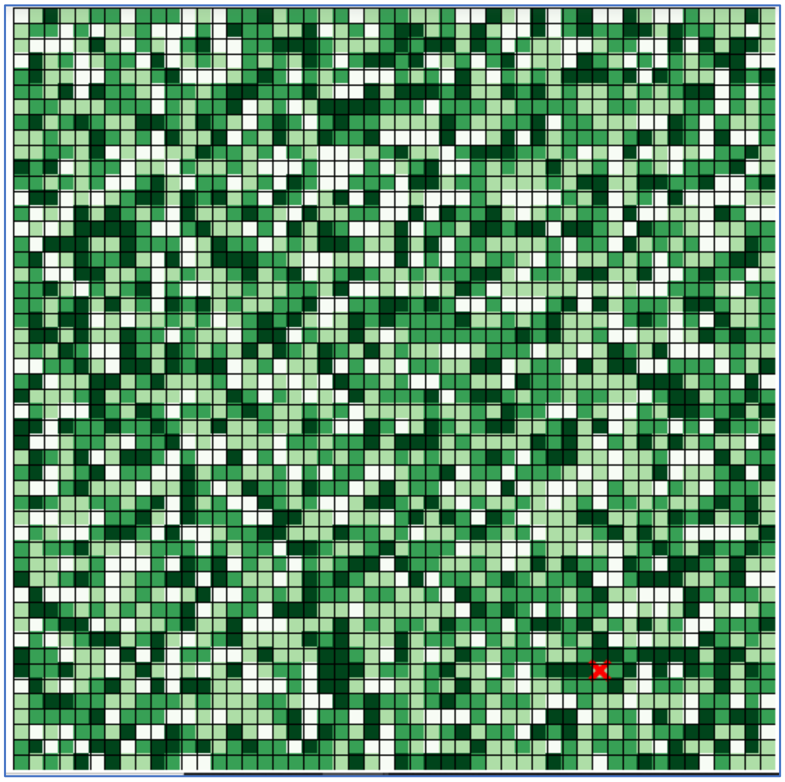


*Figure 6: Terrain map of 50*50 dimensions*

Case 1: Flat terrain

FN rate = 0.1

No of rounds = 1316.72 = 1317

These number of rounds are the average of 10 iterations with the target located in Flat terrain.

Case 2: Hilly terrain

FN rate = 0.3

No of rounds = 2190.13 = 2191

These number of rounds are the average of 10 iterations with the target located in Hilly terrain.

Case 3: Forested terrain

FN rate = 0.7

No of rounds = 9339.54 = 9340

These number of rounds are the average of 10 iterations with the target located in Forested terrain.

Case 4: Cave terrain

FN rate = 0.9

No of rounds = 23769.3 = 23770

These number of rounds are the average of 10 iterations with the target located in Cave terrain.

Observations:

- As the False Negative rate of any terrain increases, the total number of steps required to locate target in the terrain increases.
- For Flat terrain with FN = 0.1, the average total no of rounds required to locate target is 1316.72
- For cave terrain with FN =0.9, the average total no of rounds required to locate target is 23769.3
- The searching in a Cave terrain is much harder compared to the Flat terrain due to the high value of FN rate.
- The searching no of rounds for Flat, Hilly, Forested and Cave terrains are different for the same map due to differences in FN rates.

2. **Rule 1 Performance:**

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

For 50*50 map,

The array of 50 rounds searches for map is,

[4370, 8990, 613, 23769, 718, 2442, 3795, 1869, 2642, 19912, 6349, 2128, 7172, 597, 2390, 1943, 632, 481, 10997, 2709, 576, 2451, 17637, 8644, 480, 2264, 648, 2308, 10261, 6347, 9156, 1748, 2106, 14261, 970, 1194, 12026, 11792, 10406, 29555, 18422, 2453, 2416, 544, 783, 6654, 4448, 7812, 1834, 4668]

These 50 iterations have randomly chosen target cell and terrain.

The average final rounds using rule 1 is: **5987.64**

3. **Rule 2 Performance:**

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

For 50*50 map,

The array of 50 rounds searches for map is,

[1075, 2137, 13651, 970, 4501, 19359, 1081, 1544, 5154, 265, 8441, 1096, 8595, 995, 4224, 1541, 675, 16196, 3157, 67, 2934, 8020, 17086, 1673, 4776, 2755, 786, 2796, 7, 7714, 4136, 9316, 817, 2685, 1925, 1283, 1553, 426, 7922, 257, 614, 41842, 12643, 3299, 369, 10316, 942, 6500, 459, 18393]

The average final rounds using rule 2 is: **5379.36**

These 50 iterations have randomly chosen target cell and terrain.

Observations:

- The Rule 2 average no of rounds is **5379.36** and Rule 1 average no of rounds is **5987.64**
- The Rule 2 has less no of search rounds compared to rule 1

- Searching the cell with maximum probability of finding the target is more significant than choosing the cell with the maximum probability of cell containing target.

4. Other maps performances:

The different dimensions map is created and they are iterated for 50 times to check whether the Rule 1 or rule 2 performs better.

Lets check the performances!

<u>case 1: for 50*50</u>

The average final rounds using rule 1 is: **7203.6**

The average final rounds using rule 2 is: **4347.7**

<u>case 2: for 60*60</u>

The average final rounds using rule 1 is: **4014.2**

The average final rounds using rule 2 is: **4359.5**

<u>case 3: 40*40</u>

The average final rounds using rule 1 is: **3929.3**

The average final rounds using rule 2 is: **4697.6**

<u>case 4: 10*10</u>

The average final rounds using rule 1 is: **696.2**

The average final rounds using rule 2 is: **281.0**

<u>case 5: 30*30</u>

The average final rounds using rule 1 is: **2324.2**

The average final rounds using rule 2 is: **2303.4**

Observations:

- For dim = 10,30,50 we can see that based upon 50 iterations, Rule 2 outperforms Rule 1.
- For dim = 40,60 we observed that Rule 1 outperforms Rule 2
- As the no of observations increases, we may get the different results.
- As the dim size increases, the searching rounds increases.
- We cant specifically say that the Rule 2 will always outperform Rule 1 because of the uncertainty and randomness in choosing start node, randomness in the result due to FN rate etc.
- Using large no of iterations, we may expect consistency in the results across different maps.
- Intuitively Rule 1 and Rule 2 gives almost same results across different maps.
- On an average Rule 2 performs better than Rule 1 from above observations.
- As the FN rate increases, the corresponding no of rounds to locate target successfully increases.

4.Improved Agents

We are restricting our cell search movements to choosing a neighbor cell of the current position with different rules.

We are using 4 agents and deciding the cell surfing based upon 4 different rules.

Agent 1: Travel to the nearest cell chosen by Rule 1 and search it.

Agent 2: Travel to the nearest cell chosen by Rule 2 and search it.

Agent 3: At each time, score each cell with (manhatten distance from current location)/(Probability of finding the target in that cell)

Agent 4: At each time, take the average score of 3 cells with (manhatten distance from current location)/(Probability of finding the target in that cell)

Lets visualize the Improved agent logic graphically,

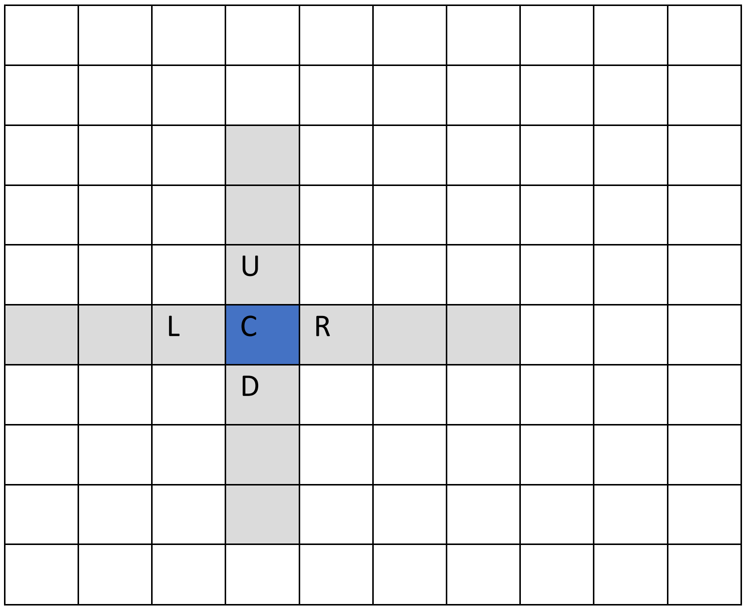


*Table 2: Improved Agent Logic with 10*10 map*

The improved agent will average the score of the next 3 cells of the current neighbor cell and chooses the cell with minimal average score.

In above figure,

C represents Current cell position

And U is UP,D is Down, L is Left and R is Right

The cell will calculate the score described in agent 3 and will take the average of the 3 Up cells. It will repeat the similar calculations for neighbors. Then it will choose to travel to the neighbor cell which has lowest score.

The motivation of this strategy is to choose the neighbor of current cell which has highest probability of finding the Target.

Lets analyze the results of 4 agents for 50*50 grid.

Total no of iterations are 50 for each agent.

We also analyzed the results of 30*30 and 10*10 maps to compare the consistency in the results.

The average no of rounds for different agents are:

Agent	50*50	30*30	10*10
Agent 1	11998.4	4728.3	228.5
Agent 2	8081.4	2147.4	123.0
Agent 3	10535.2	3069.7	201.5
Agent 4	9669.2	1988.8	156.3

Observations:

- For 50*50 map, Agent 3 and 4 gives better results than agent 1.
- Agent 4 requires less no of rounds compared to agent 3.
- If we increase the no of iterations we will get consistency in the results.
- As the FN increases, the no of rounds for the terrain increases.
- Agent 1 performs worst in all 4 agents
- Agent 2 and Agent 4 performs better compared to other 2 agents.
- The probability of finding a cell in a map proves helpful in searching the target in minimum no of rounds.
- The agent 4 strategy tries to improve on agent 3 and agent 1.

5.Future Scope:

We can design our improved agent with taking average of 5 cells and take an average of the scores calculated using agent 3 and try to head in the direction in which our belief of finding the target in higher in order to locate target successfully!

We can consider a small grid from current location as our improved agent and try to average the scores of all cells in the small grid and make a move in the direction in which the average score value id lower.

The intuition behind calculating the score is to balance the tradeoff between distance of the cell from its current location and the probability of finding the target in that cell.