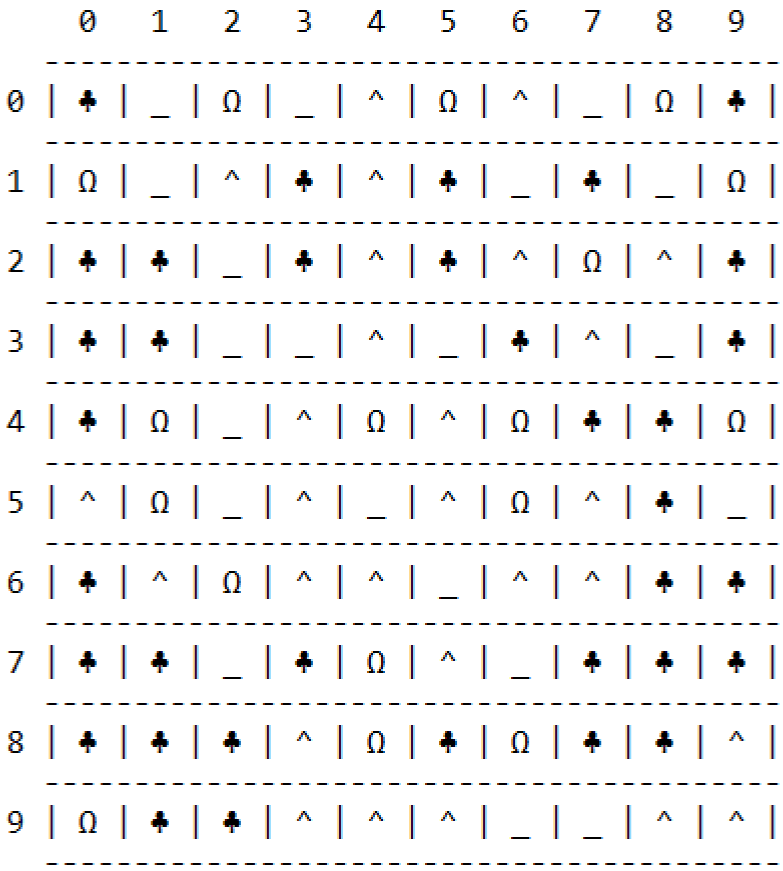
RUTGERS UNIVERSITY

Probabilistic Hunting

Prathamesh Kulkarni
November 19, 2020

1 INTRODUCTION

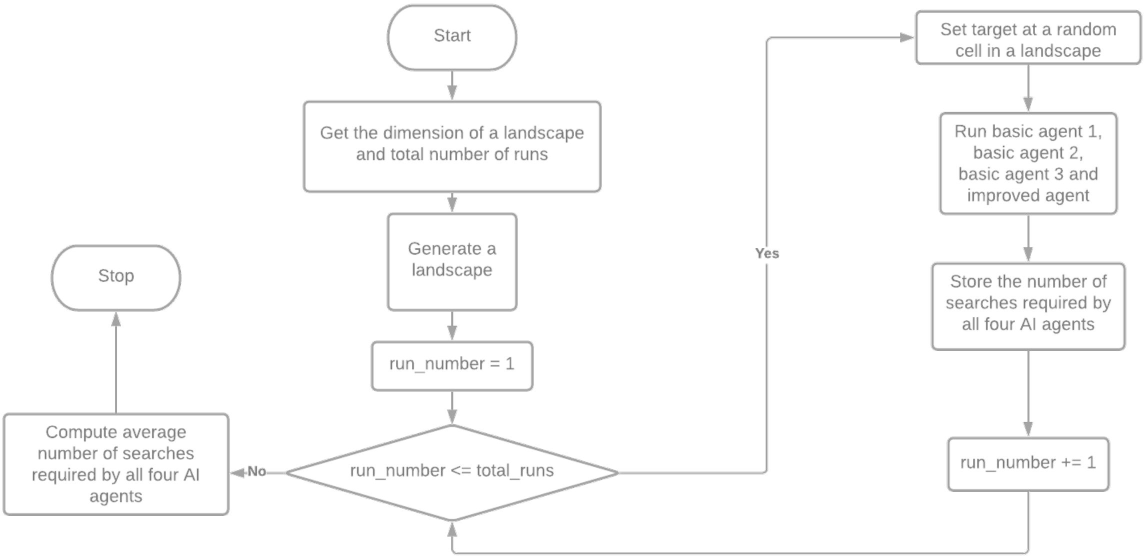
Landscape is a square grid of cells where each cell is flat, hill, forest or cave. A target is hidden in one of the cells in a landscape. There are four AI agents - basic agent 1, basic agent 2, basic agent 3 and improved agent - to find and hunt the target. Goal of AI agents is to locate the target in as few searches as possible by using information gained from failed searches.



```
     0   1   2   3   4   5   6   7   8   9
    -----------------------------------------
0 | ♣ | _ | Ω | _ | ^ | Ω | ^ | _ | Ω | ♣ |
    -----------------------------------------
1 | Ω | _ | ^ | ♣ | ^ | ♣ | _ | ♣ | _ | Ω |
    -----------------------------------------
2 | ♣ | ♣ | _ | ♣ | ^ | ♣ | ^ | Ω | ^ | ♣ |
    -----------------------------------------
3 | ♣ | ♣ | _ | _ | ^ | _ | ♣ | ^ | _ | ♣ |
    -----------------------------------------
4 | ♣ | Ω | _ | ^ | Ω | ^ | Ω | ♣ | ♣ | Ω |
    -----------------------------------------
5 | ^ | Ω | _ | ^ | _ | ^ | Ω | ^ | ♣ | _ |
    -----------------------------------------
6 | ♣ | ^ | Ω | ^ | ^ | _ | ^ | ^ | ♣ | ♣ |
    -----------------------------------------
7 | ♣ | ♣ | _ | ♣ | Ω | ^ | _ | ♣ | ♣ | ♣ |
    -----------------------------------------
8 | ♣ | ♣ | ♣ | ^ | Ω | ♣ | Ω | ♣ | ♣ | ^ |
    -----------------------------------------
9 | Ω | ♣ | ♣ | ^ | ^ | ^ | _ | _ | ^ | ^ |
    -----------------------------------------
```

2 SYSTEM DESIGN

2.1 SYSTEM ARCHITECTURE


Start
Get the dimension of a landscape and total number of runs
Generate a landscape
run_number = 1
run_number <= total_runs
No
Compute average number of searches required by all four AI agents
Stop
Yes
Set target at a random cell in a landscape
Run basic agent 1, basic agent 2, basic agent 3 and improved agent
Store the number of searches required by all four AI agents
run_number += 1

2.2 IMPORTANT MODULES

1. Landscape Generation
   It generates a square landscape of given dimension and sets the status of every cell to Flat, Hill, Forest or Cave. The choice of status is made according to the probability distribution: P(Flat) = 0.2, P(Hill) = 0.3, P(Forest) = 0.3, P(Cave) = 0.2.

2. Target setting
   It chooses row number and column number of a cell in a landscape uniformly at random. This becomes a position of a hidden target that is going to be searched and hunted by AI agents.

3. Probabilistic KB Generation
   It generates a probabilistic knowledge base that stores P(target_in_cell | observations) and P(target_found_in_cell | observations). These probabilities are updated by AI agents upon newer observations (failed searches) and used to decide a cell to be searched next.

4. Initial positioning of AI agent in a landscape
   It decides an initial position of AI agent in a landscape to begin the search. Basic agent 1 is positioned at a cell that has highest P(target_in_cell | observations) which turns out to be a random cell as all have equal probabilities. Other three agents are positioned at a cell that has highest P(target_found_in_cell | observations).

5. Searching the landscape
   It searches for a target in a given position and returns the boolean result of search. If target is not present in a given position, then search returns False with certainty. If target is present in a given position, then search returns result based on false negative rate of terrain type of a given cell. If search returns True, then AI agent hunts the located target and stops searching.

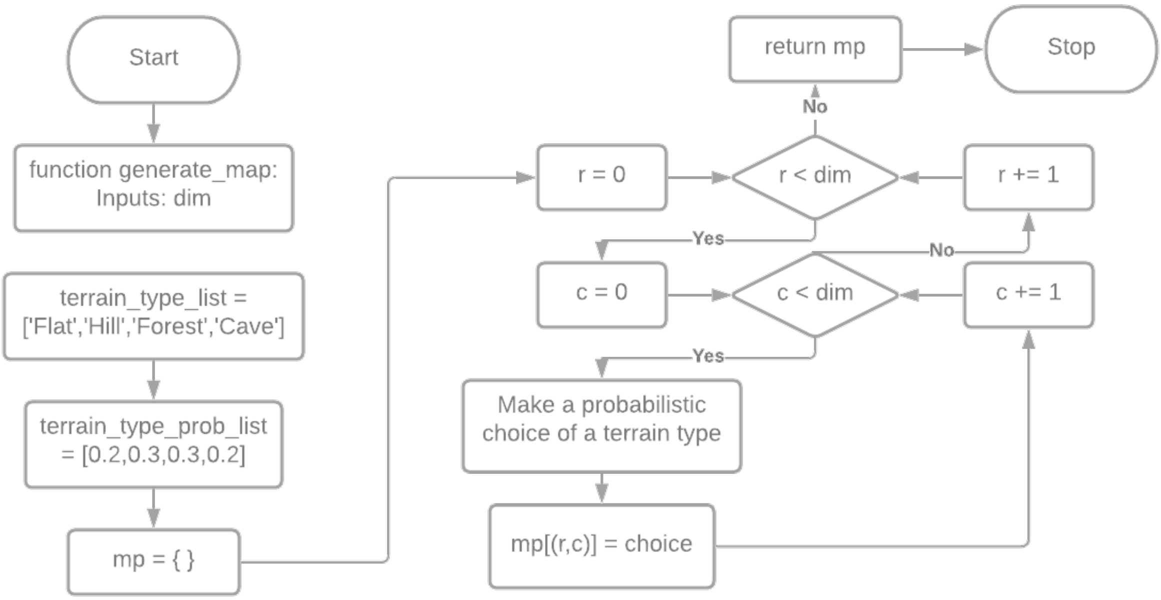
6. Updating probabilistic KB
   Failed searches are the newer information that is taken into consideration by AI agents to update the above-mentioned probabilities for every cell in a landscape. Probabilities are updated according to the Bayes rule.

7. Re-positioning of AI agent in a landscape
   The updated probabilistic knowledge base is used by AI agents to decide what position to search next. All four AI agents have their own strategies to make a choice of next cell. This choice also depends on whether the movements in a landscape incur cost.

3 DETAILED DESIGN

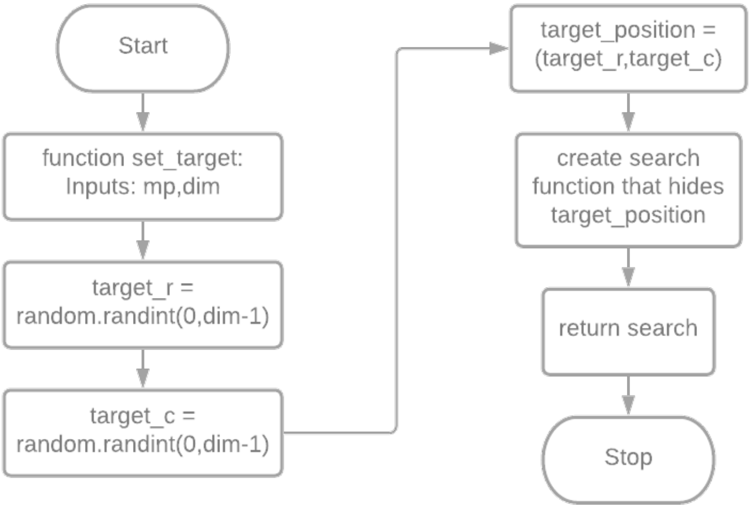
3.1 LANDSCAPE GENERATION

Landscape is a square grid of cells of given dimension where each cell is of one of the four terrain types: Flat, Hill, Forest and Cave. Landscape is stored as a dictionary of (row,col):terrain_type pairs. Choice of a terrain type of every cell is made randomly according to the following probability distribution: P(Flat) = 0.2, P(Hill) = 0.3, P(Forest) = 0.3, P(Cave) = 0.2.



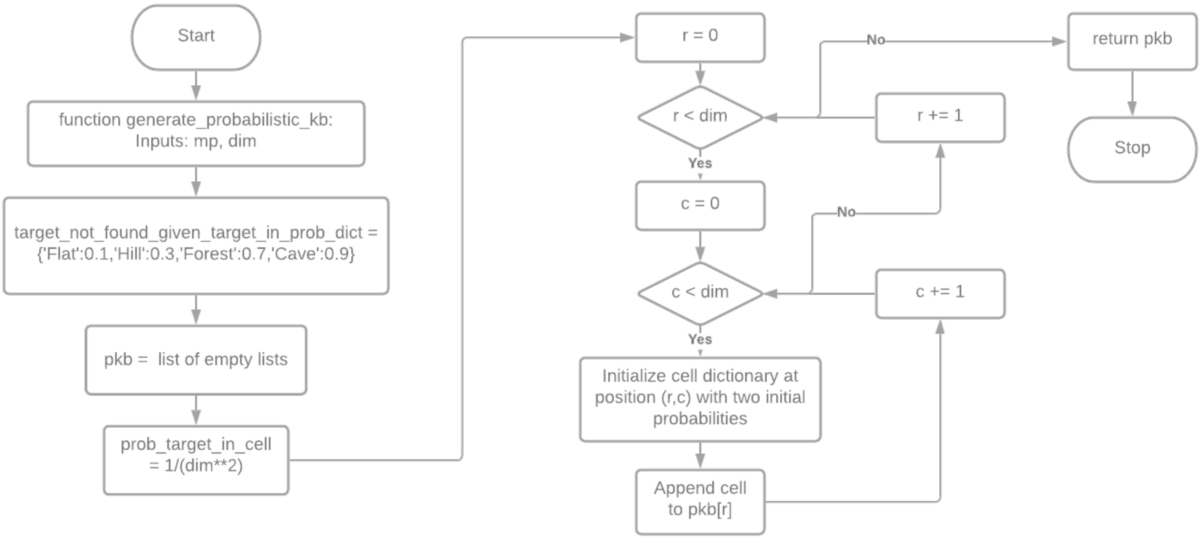
3.2 TARGET SETTING

A target is set in one of the cells in a landscape which is hidden from the AI agents. It randomly selects a row number and a column number of a cell to position a target. In order to hide this position from AI agent, it also creates and returns an inner function that stores target position in a scope of outer factory function. It returns an inner function which is later used by AI agents to perform search.



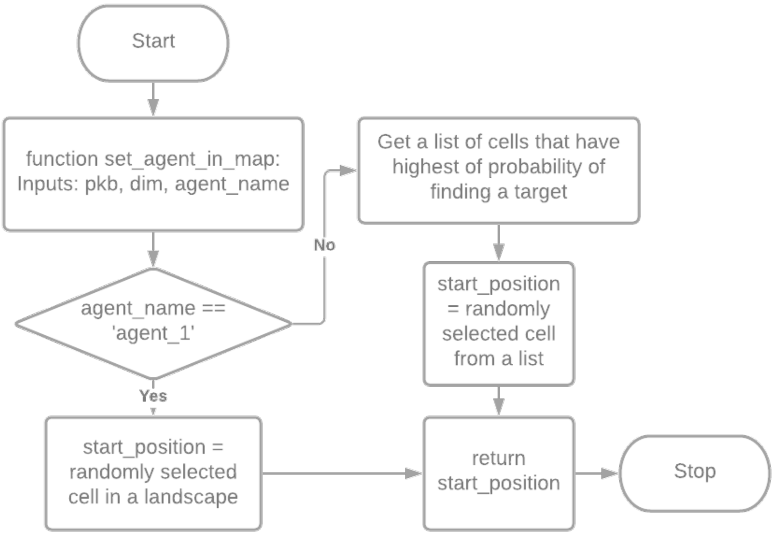
3.3 Probabilistic KB Generation

Probabilistic knowledge base is a two-dimensional list of cells where each cell is a dictionary. Each cell has following two key:value pairs- prob_target_in_cell : value and prob_target_found_in_cell : value. Initially, prob_target_in_cell has same value ($1/dim^2$) for every cell in KB. prob_target_found_in_cell is initialized according to the terrain type of a cell (false negative rate varies as per the terrain type).


Start
function generate_probabilistic_kb:
Inputs: mp, dim
target_not_found_given_target_in_prob_dict =
{'Flat':0.1,'Hill':0.3,'Forest':0.7,'Cave':0.9}
pkb = list of empty lists
prob_target_in_cell
= 1/(dim**2)
r = 0
r < dim
Yes
No
c = 0
c < dim
Yes
No
Initialize cell dictionary at
position (r,c) with two initial
probabilities
Append cell
to pkb[r]
c += 1
r += 1
return pkb
Stop

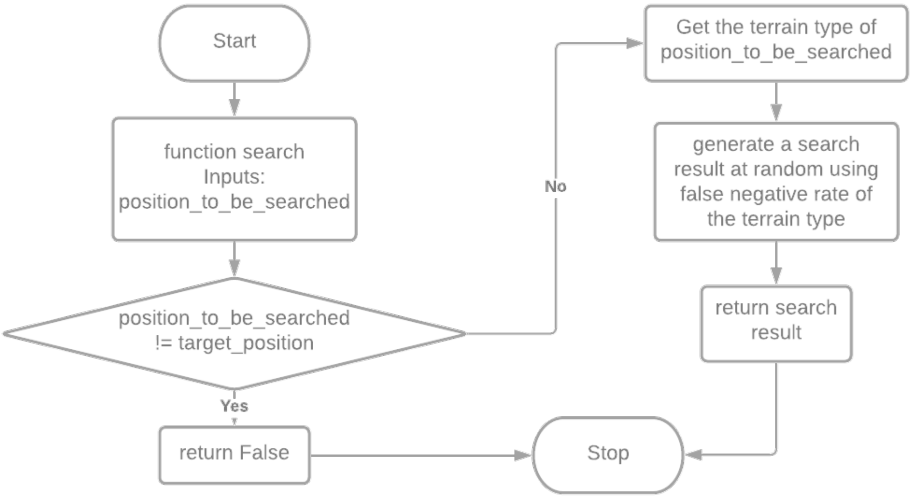
3.4 Initial positioning of AI agent in a landscape

AI agent needs to be positioned in a landscape to begin search. It selects a cell where agent begins searching for a target. Basic agent 1 begins searching in a cell that has highest probability of containing the hidden target. As all cells in a landscape initially have equal probability ($1/dim^2$) of containing a target, agent 1 chooses a cell at random to begin search. Basic agent 2, basic agent 3 and improved agent begin searching in a cell that has highest probability of finding a target. They choose a cell at random from a list of all cells that have highest probability of finding the target.



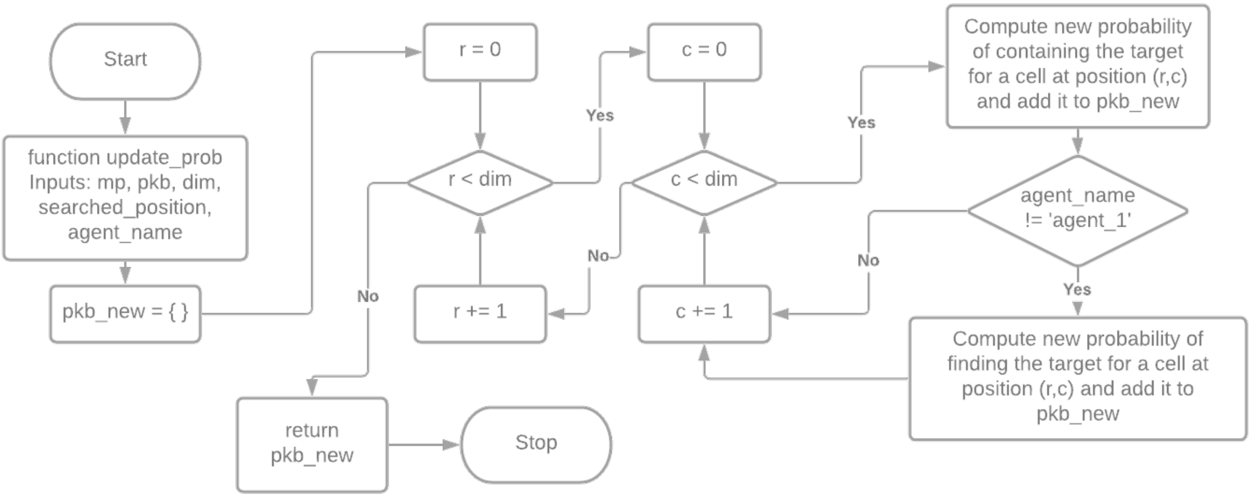
3.5 SEARCHING THE LANDSCAPE

It searches the hidden target in a given position and returns a boolean result of search. As false positive rate of the search is always 0, it returns False whenever target is not present in a given position. If target is present in a given position, then it generates a search result at random based on a false negative rate. False negative rate is determined by the terrain type of a given position.



3.6 UPDATING PROBABILISTIC KB

AI agents use observations (failed searches) to update their probabilistic knowledge base in order to make a better choice of a cell to be searched. Agents use Bayes rule to update the probabilities. Basic agent 1 updates only the probability of containing the target. Basic agent 2, basic agent 3 and improved agent update probability of containing the target as well as probability of finding the target.

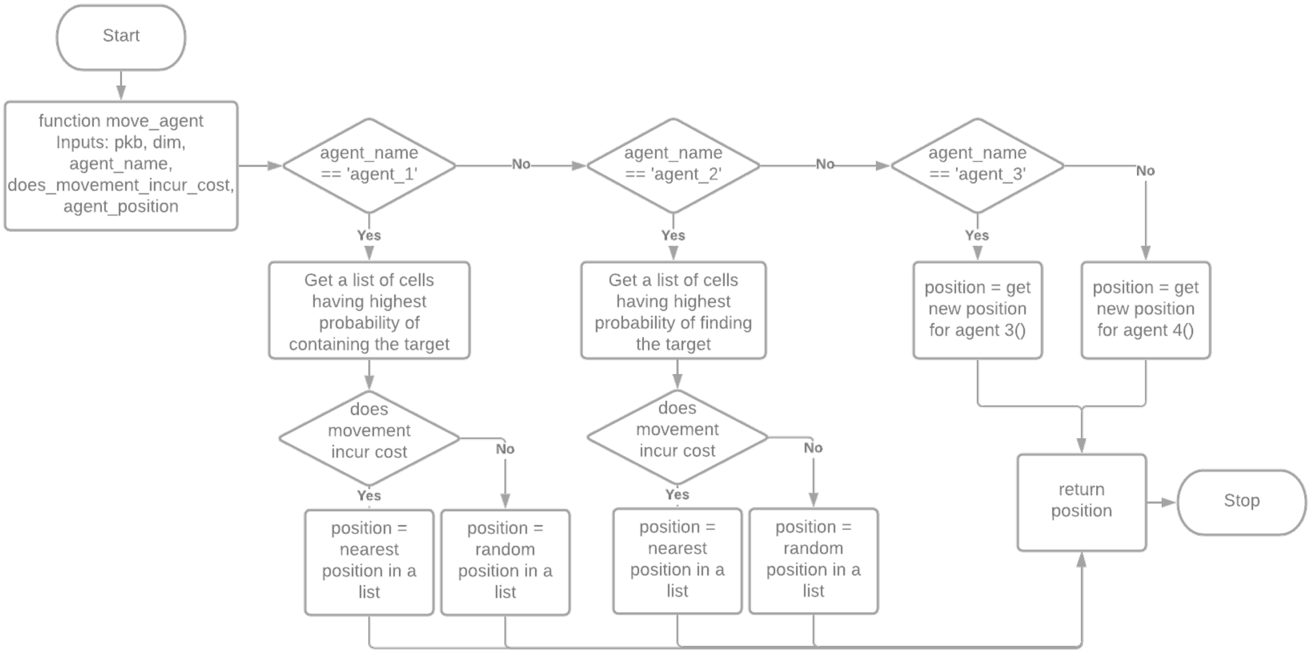


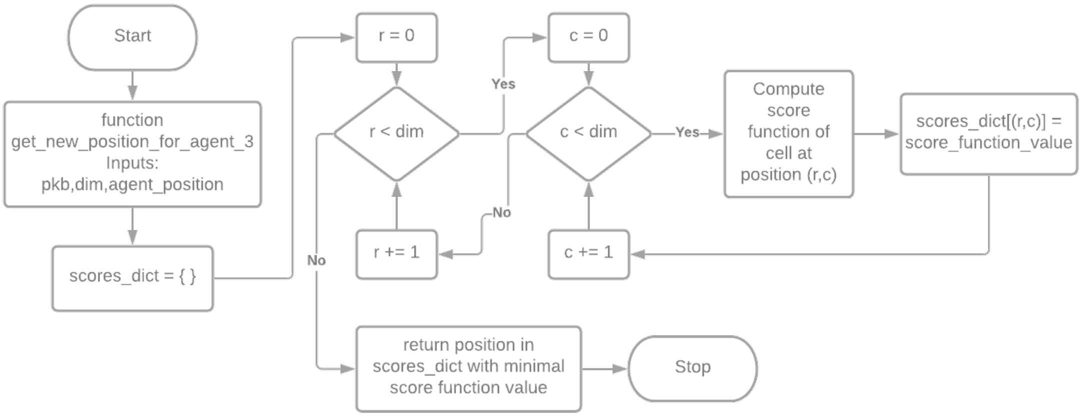
3.7 Re-positioning of AI agent in a landscape

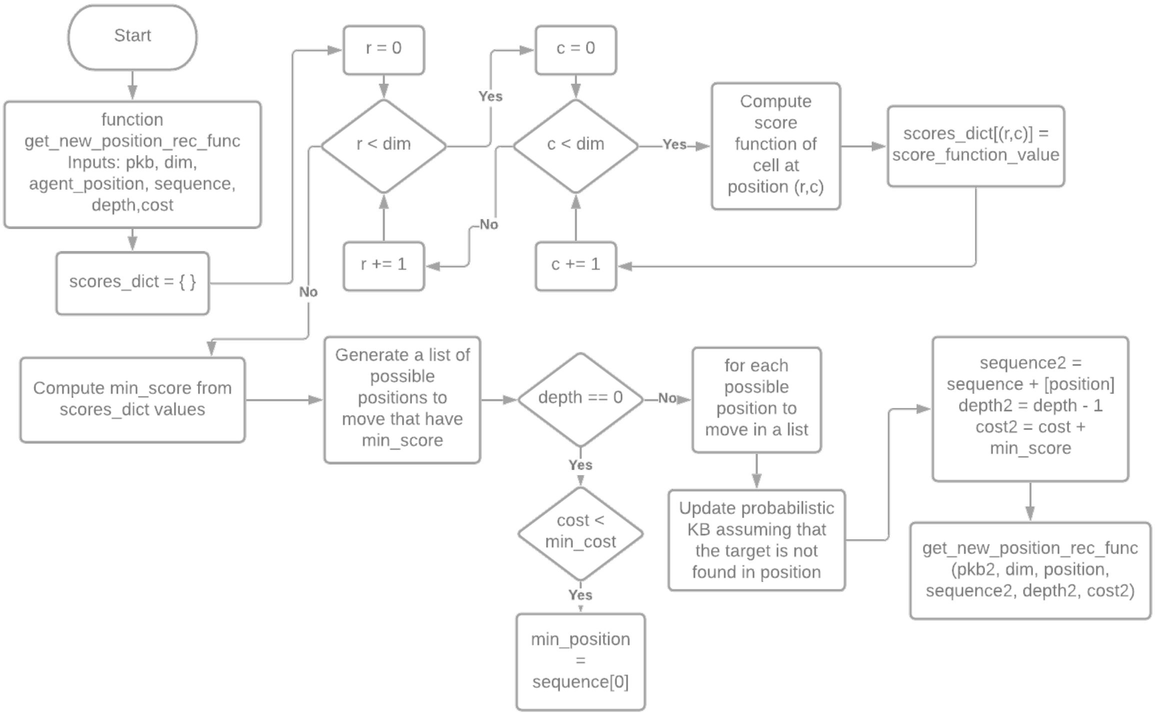
After every failed search, AI agents update the probabilistic knowledge base and decide where to search next. All four agents differ in the strategy used to make this decision.

When motion does not incur cost:
Basic agent 1: It always moves to and searches a cell that has the highest probability of containing the target. If there are more than one cells having highest probability of containing the target, then agent makes a random choice amongst them as the motion does not incur cost.
Basic agent 2: It always moves to and searches a cell that has the highest probability of finding the target. If there are more than one cells having highest probability of finding the target, then agent makes a random choice amongst them as the motion does not incur cost.

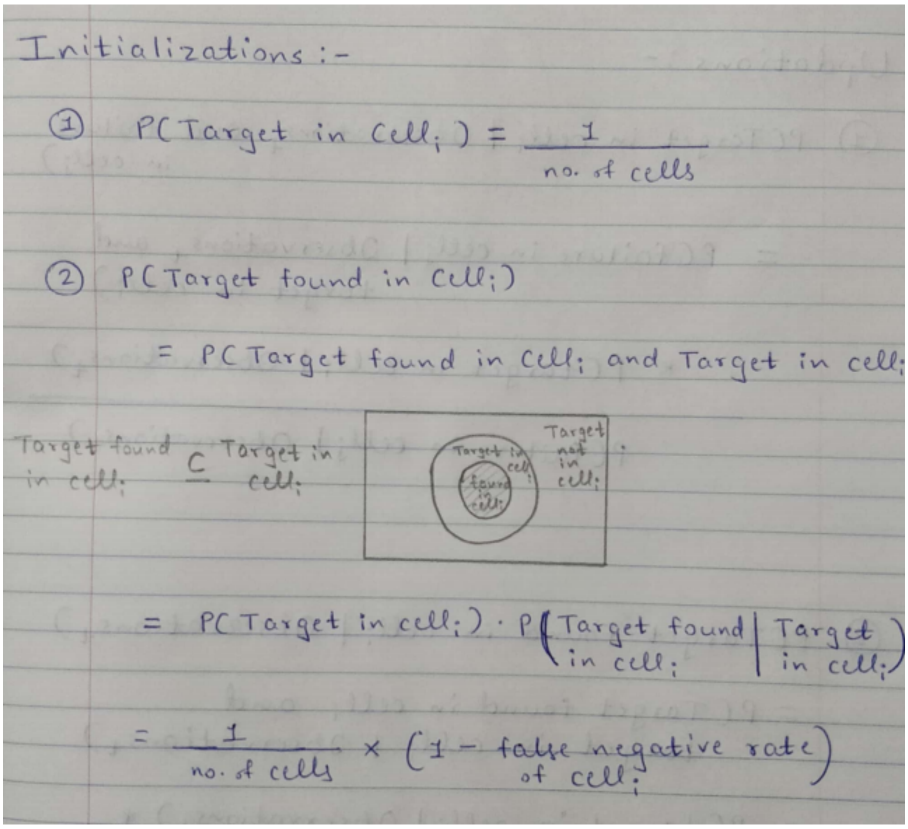
When motion incurs cost:
Basic agent 1: It always moves to and searches a nearest cell that has the highest probability of containing the target. If there are more than one nearest cells having highest probability of containing the target, then agent makes a random choice amongst them.
Basic agent 2: It always moves to and searches a nearest cell that has the highest probability of finding the target. If there are more than one nearest cells having highest probability of finding the target, then agent makes a random choice amongst them.
Basic agent 3: It computes a score function given as $(1+manhattan\_distance\_from\_current\_cell)/prob\_target\_found\_in\_cell$ for every cell in a landscape. It always moves to and searches the cell that has minimal score function value.
Improved agent: It chooses a next cell to search by looking ahead more than one step. It uses a recursive implementation of depth-limited search to perform the look-ahead where the number of look-aheads can be tuned as input parameter. Along the path in a search tree, it always updates the probabilistic knowledge base by assuming the worst case that the target is not found in current state in a tree. It computes path cost from root node to leaf node as the sum of score function values of nodes along the path from root to leaf. It decides to take the first action of a path that has minimal total score function value.


Start
function move_agent Inputs: pkb, dim, agent_name, does_movement_incur_cost, agent_position
agent_name == 'agent_1'
No
agent_name == 'agent_2'
No
agent_name == 'agent_3'
No
Yes
Get a list of cells having highest probability of containing the target
Yes
Get a list of cells having highest probability of finding the target
Yes
position = get new position for agent 3()
position = get new position for agent 4()
does movement incur cost
No
Yes
position = nearest position in a list
position = random position in a list
does movement incur cost
No
Yes
position = nearest position in a list
position = random position in a list
return position
Stop


Start
function get_new_position_for_agent_3 Inputs: pkb,dim,agent_position
scores_dict = { }
r = 0
r < dim
Yes
No
r += 1
c = 0
c < dim
Yes
No
c += 1
Compute score function of cell at position (r,c)
scores_dict[(r,c)] = score_function_value
return position in scores_dict with minimal score function value
Stop


Start
function get_new_position_rec_func Inputs: pkb, dim, agent_position, sequence, depth,cost
scores_dict = { }
r = 0
r < dim
Yes
No
r += 1
c = 0
c < dim
Yes
No
c += 1
Compute score function of cell at position (r,c)
scores_dict[(r,c)] = score_function_value
Compute min_score from scores_dict values
Generate a list of possible positions to move that have min_score
depth == 0
No
Yes
cost < min_cost
Yes
min_position = sequence[0]
for each possible position to move in a list
Update probabilistic KB assuming that the target is not found in position
sequence2 = sequence + [position] depth2 = depth - 1 cost2 = cost + min_score
get_new_position_rec_func (pkb2, dim, position, sequence2, depth2, cost2)

4 PROBABILITY CALCULATIONS



Initializations :-

① $P(\text{Target in Cell}_i) = \dfrac{1}{\text{no. of cells}}$

② $P(\text{Target found in Cell}_i)$

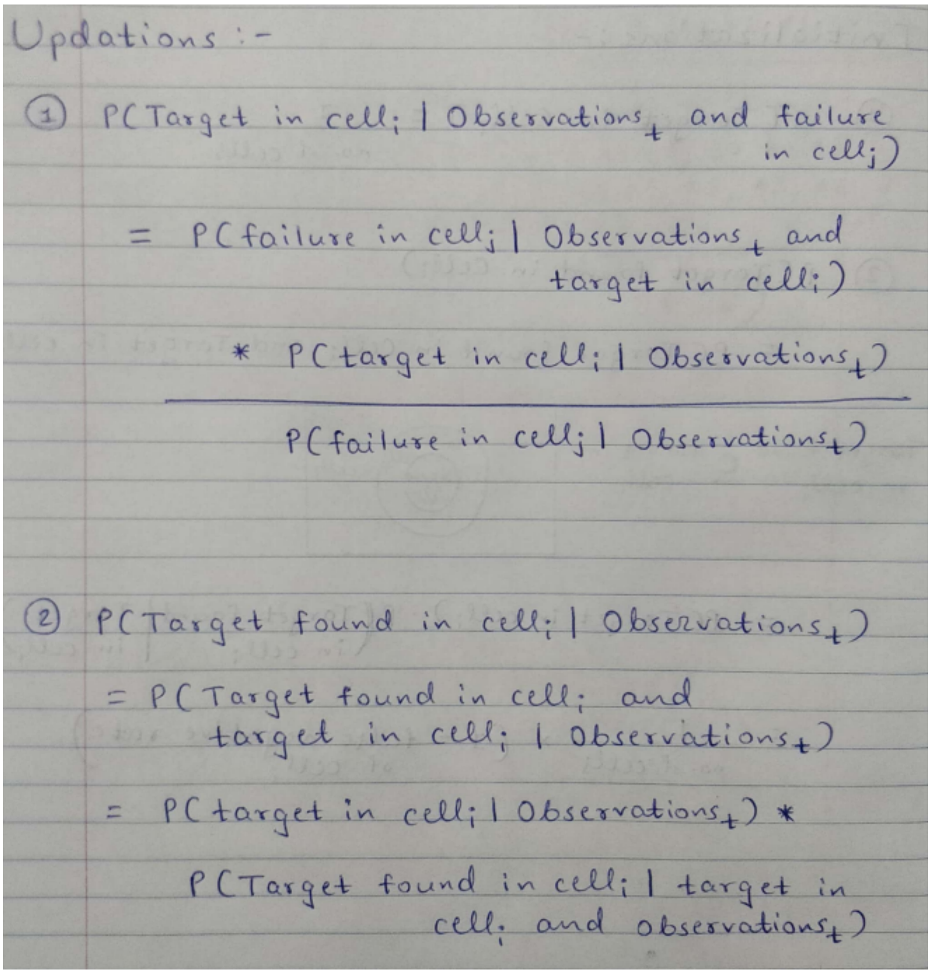
$= P(\text{Target found in Cell}_i \text{ and Target in cell}_i$

$\text{Target found in cell}_i \subseteq \text{Target in cell}_i$

Target in cell
found in cell$_i$
Target not in cell$_i$

$= P(\text{Target in cell}_i) \cdot P\left(\text{Target found in cell}_i \,\middle|\, \text{Target in cell}_i\right)$

$= \dfrac{1}{\text{no. of cells}} \times \left(1 - \text{false negative rate of cell}_i\right)$



Updations :-

① $P(\text{Target in cell}_i \mid \text{Observations}_t \text{ and failure in cell}_j)$

$$= \frac{P(\text{failure in cell}_j \mid \text{Observations}_t \text{ and target in cell}_i) * P(\text{target in cell}_i \mid \text{Observations}_t)}{P(\text{failure in cell}_j \mid \text{Observations}_t)}$$

② $P(\text{Target found in cell}_i \mid \text{Observations}_t)$

$= P(\text{Target found in cell}_i \text{ and target in cell}_i \mid \text{Observations}_t)$

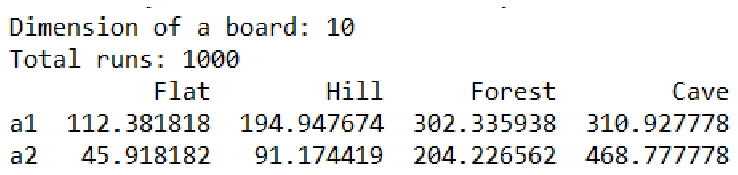
$= P(\text{target in cell}_i \mid \text{Observations}_t) *$

$P(\text{Target found in cell}_i \mid \text{target in cell}_i \text{ and observations}_t)$

5 Results

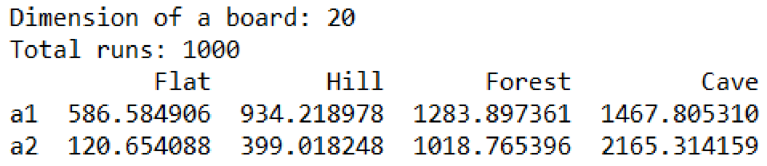
5.1 Average number of actions when motion does not incur cost

This is a table of average number of searches required by basic agent 1 and 2 for all four terrain types in a landscape of dimension 10.



```
Dimension of a board: 10
Total runs: 1000
          Flat        Hill      Forest        Cave
a1  112.381818  194.947674  302.335938  310.927778
a2   45.918182   91.174419  204.226562  468.777778
```

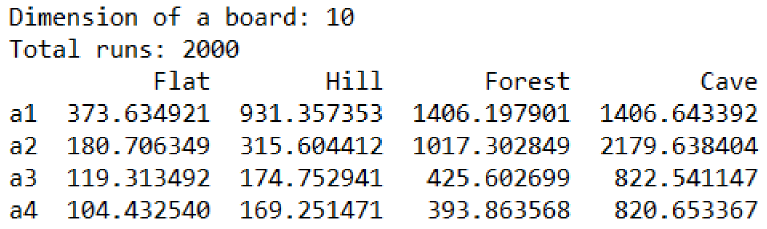
This is a table of average number of searches required by basic agent 1 and 2 for all four terrain types in a landscape of dimension 20.



```
Dimension of a board: 20
Total runs: 1000
          Flat        Hill       Forest         Cave
a1  586.584906  934.218978  1283.897361  1467.805310
a2  120.654088  399.018248  1018.765396  2165.314159
```

5.2 Average number of actions when motion incurs cost

This is a table of average number of searches required by basic agents 1, 2, 3 and improved agent for all four terrain types in a landscape of dimension 10.



```
Dimension of a board: 10
Total runs: 2000
          Flat        Hill       Forest         Cave
a1  373.634921  931.357353  1406.197901  1406.643392
a2  180.706349  315.604412  1017.302849  2179.638404
a3  119.313492  174.752941   425.602699   822.541147
a4  104.432540  169.251471   393.863568   820.653367
```

6 Conclusion

When motion does not incur cost, it is observed that the basic agent 2 requires lesser number of searches to hunt the target as compared to basic agent 1 when terrain type of the target location is Flat, Hill or Forest. When terrain type of the target location is Cave, basic agent 2 needed more number of searches as compared to basic agent 1.
When motion incurs cost, it is generally observed that the number of actions (searches + movements) required to hunt the target decrease from basic agent 1 to improved agent. This decrease is due to the differences in strategies used by AI agents to select a cell to search next.

7 References

1) https://docs.python.org/3/
2) https://docs.python.org/3/library/random.html
3) https://www.oreilly.com/library/view/learning-python-5th/9781449355722/
4) https://en.wikipedia.org/wiki/Bayes%27_theorem
5) https://pandas.pydata.org/
6) https://docs.python.org/3/library/pprint.html
7) https://en.wikipedia.org/wiki/Iterative_deepening_depth-first_search