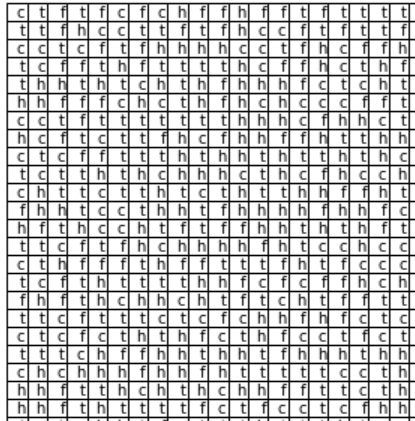
CS 520: Assignment 3 - Search and Destroy

Ruolin Qu(rq40),Changlin Jiang(cj360),Shengdong Liu(sl1563)

1. Representation

We are using 2-dimensional list to represent the map. Each cell has a component represent the “terrain” where “f” == ”flat”, ”h” == “hilly”, “t” == “forested”, “c” == “caves” and “x” == “target”.

The map generated by this code is represented as below:



1. A Stationary Target
   1. How can Bayes’ theorem be used to update the belief state?

Firstly, the algorithm starts with an initial state.

While t = 0, the probability of each cell is P(cellij) = 1/2500 in a 50X50 map.

Then, our algorithm searches a cell

if it’s the target, we done.

Else, reduce the probability of current cell, increase the probability of other cells as below.

Let’s say we search a flat cell at first and it’s not the target.

Which has a probability

False Negative(hit the target but not found),P(FN)current =

True Positive(hit and found), P(TN)=

True Negative(not hit and not found)， P(TP) = .

Based on this, we have total probability

P(total) = P(FN)current + = 0.1\*1/2500 +2499/2500 = 2499.1/2500

It is less than 1, so we normalize it, and have our belief state as below:

P(cellij | search current not found) = P(cell)ij/P(total).

We update this probability and find another cell to search.

* 1. P that the target in a searched cell.

The probability is just equal to the belief state \* the terrain probability which as below

P(Target in Cell | Observations ^ Failure in Cell ) = (1-P(Terrain))\* Belief State of cell.

* 1. Two decision rules

The two rules are:

- 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.

Rule 1 uses the belief state to choose where to search and (rule 2) uses the state we mentioned in 2.2. Rule 2 should require less search steps because it takes the terrain probability into consideration, so it is going to search the flat cell at first, searching a flat zone leads to more increase of the belief state of other cells. In this case, the searching progress may be less than using rule1.

We tested both rule1 and rule2 100 times in a fixed 50X50 map, calculate the average steps of finding the target. Here are the results, which support our hypothesis.

rule1avgsteps: 7924.91

rule2avgsteps: 7387.65

* 1. Step Moving searching

In this section, our algorithm has two actions:

1. Search the current cell
2. Move to another cell

After the normalizing and updating process we mentioned in section 2.1, we try to compute the Manhattan distance from current cell to all others. We calculate every other cells’ tendency of searching with the following function:

Tendency = Belief State / Manhattan Distance.

Then, we choose to move to the cell with the highest tendency and search from there.

The searching progress of rule 1 in this section will be like this.

1. the agent will search every node for 1 time to see if the target can be located.
2. If not, agent will search the node where the belief state is higher, which is able to be caves (with higher P(FN)).

It takes much more steps moving between such kinds of cells.

Let’s consider the algorithm runs with rule 2.

1. The agent will search the flat cells for 1 time to see if the target can be located, which may go through some cells directly without searching. We can say that flat cells have higher priority.
2. Following this rule, the agent will then search the hilly cells, then, forests, then caves, where it takes more steps travelling through these terrains. However, it will potentially find the nearest terrains, which indicates that it could take less steps than rule 1.

We tested these 3 algorithms for 100 times in 50X50 map.it took about 3 hours to do these test. And the results support our hypothesis.

Strategy local search: 17733.83

Strategy rule 1: 156699.9

Strategy rule 2: 89138.48

* 1. An old joke

The old joke leads to the searching strategy of rule 2. Where the agent searches the easier cells at first. However, we can consider that if the probability of finding the target under the lamp is relatively equals to the probability in the park, the better strategy is starting from litter zone.

I think this old joke reminds us that we should not ignore the hard cell and pass through it directly. Our local search algorithm tends to search every point in a local zone and slowly move to other zones which is different from ignoring the hard place.