Question

You are a MazeSolvingBot, dropped into the above maze. Unfortunately, while your motors are still functional (and you can tell which direction is north), your sensors are completely inoperable - you are effectively completely blind, and can't even tell if you are running into a wall. All you have available to you (as a knowledge base) is the above map. You are free to move between adjacent white cells (up/down/left/right) but black cells are blocked - and you get no feedback whether the move was successful or not:

Condition in Place

This is a maze navigation problem, but with the following conditions is place:

1. There is no feedback.
2. We are still aware of which is north.
3. We have a map of the maze with us.
4. Degree of freedom available to us is 4 (up/down/left/right).
5. Cannot pass thorough black cells.

Lets answers the following questions now:

a) You are somewhere in the above maze, with no prior knowledge. What is the probability you are at G?

The problem statement specifies that we are in a cell in the maze, without any prior information as to which cell it is. This is essentially saying that we have are at a random cell in the maze. It goes without saying that the cell we are in cannot be a wall.

Now let us assign a variable 'n' to the total number of cells in the maze and 'k' to the total number of black cells (which are the walls). This implies that we can be at any one of the (n-k) cells.
Hence the probability of us being in a cell and that cell being 'G' is the probability of us picking a random cell and it turning out to be 'G'. This can be expressed by the formula: <u>(1/(n-k))</u>.

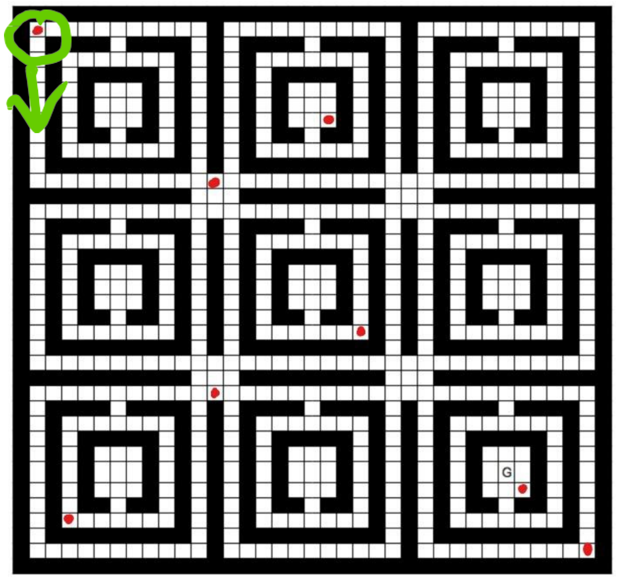
b) Argue that there is a finite sequence of moves that without knowing where you start, and without any feedback on your moves will result in you ending at location G with complete probability/certainty. Hint: What if you knew you started either at the top left or bottom right corner, but didn't know which?

The first thing to consider while answering this question is that the maze has a finite set of cells. Hence, it is quite easy to back track or predict our position after a finite set of moves.

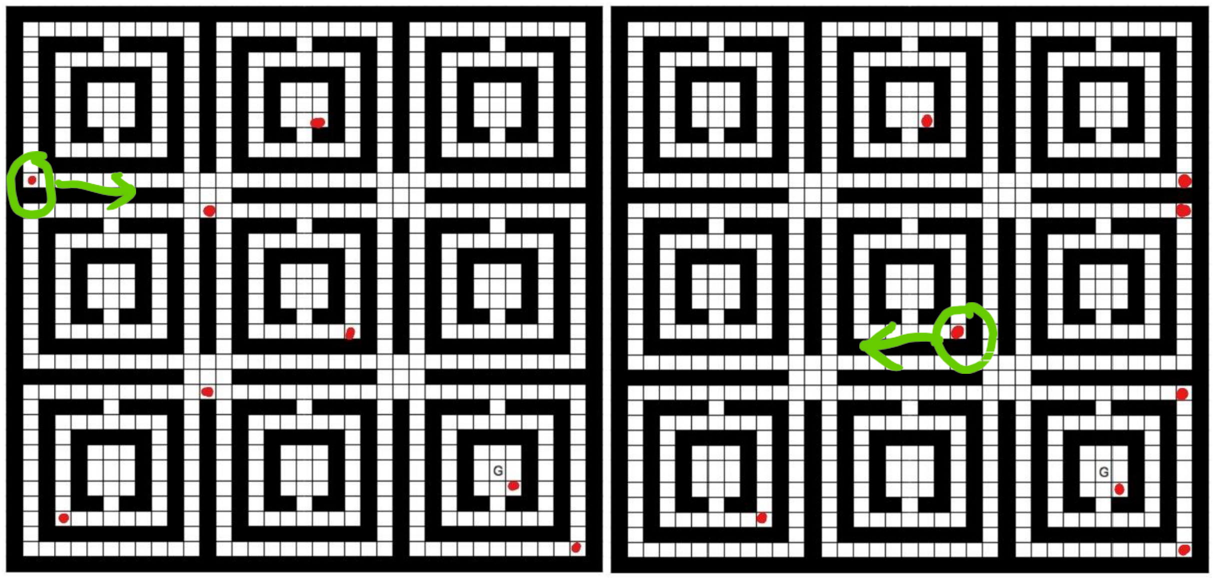
(<u>Note</u>: here we are assuming that we are moving a robot. Hence the words robot and us can be used interchangeably).

To explain the above statement and prove the solvability of the above question I believe the best was to achieve this is through pictorial representation. Hence below are a set of images in sequential manner that will finally result to the solution for any starting point with a 100 accuracy.

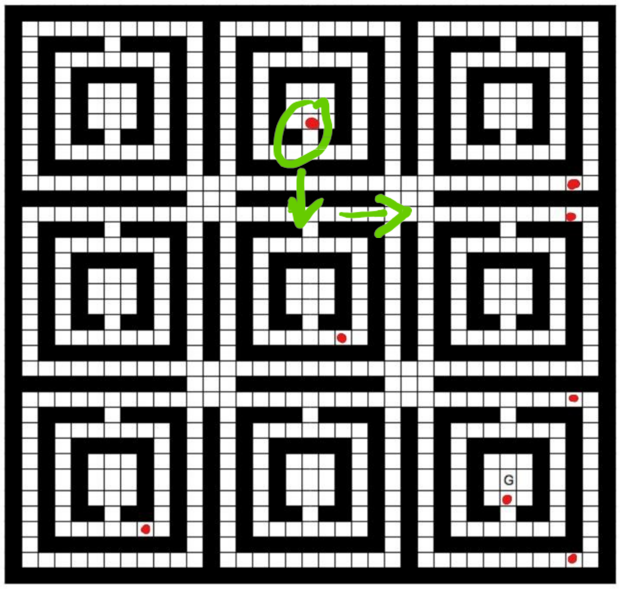
The green circles and arrows indicate the next transition state:

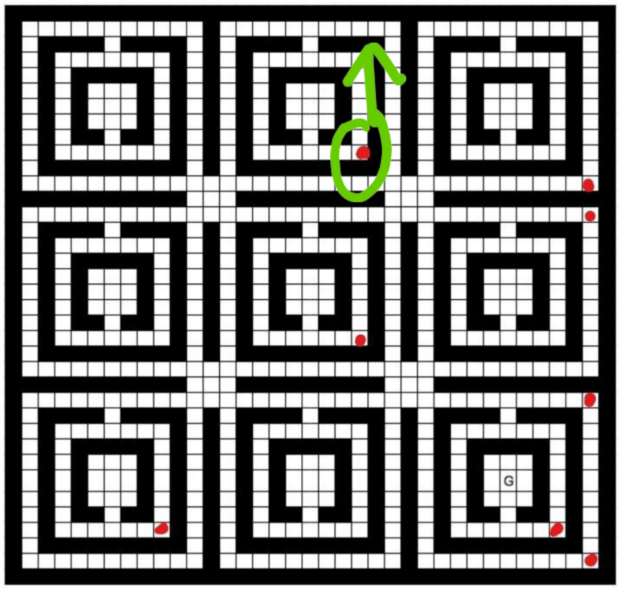


Step 1: Random start positions assigned.

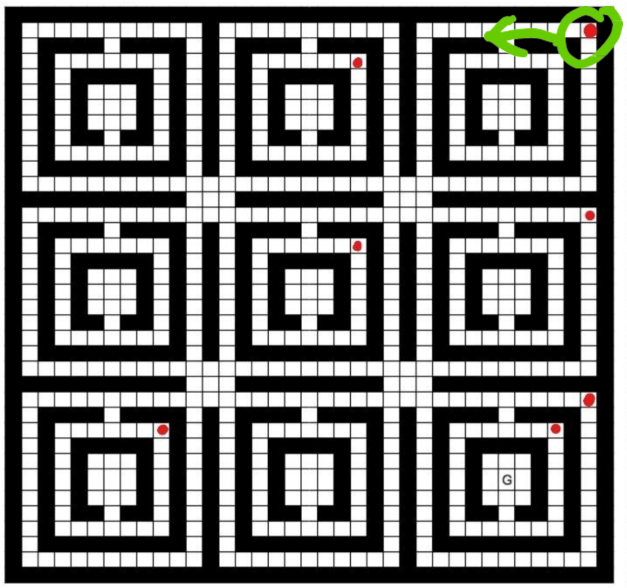


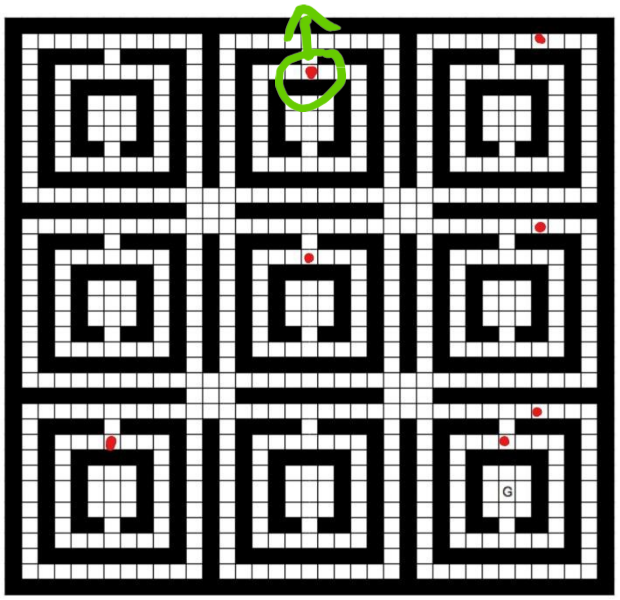
Step 2: Move down, as many moves as possible (height of the maze). This will result in the target hitting the wall no matter where it is. Post this move right as many moves as possible (width of the maze). This will result in the above diagram.



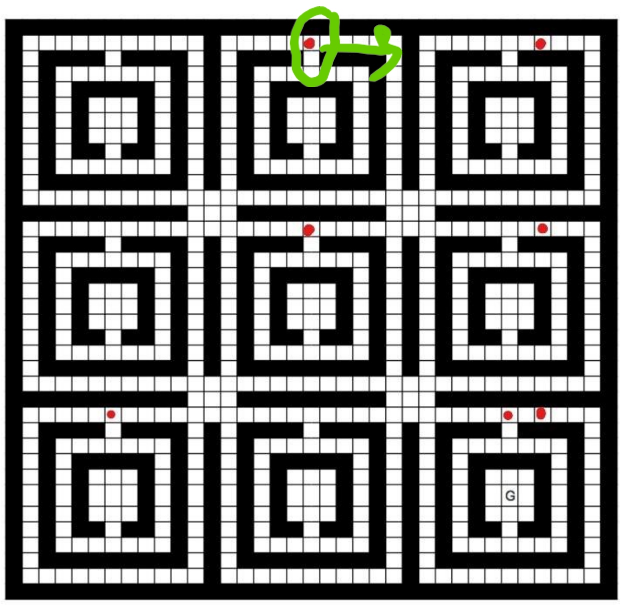


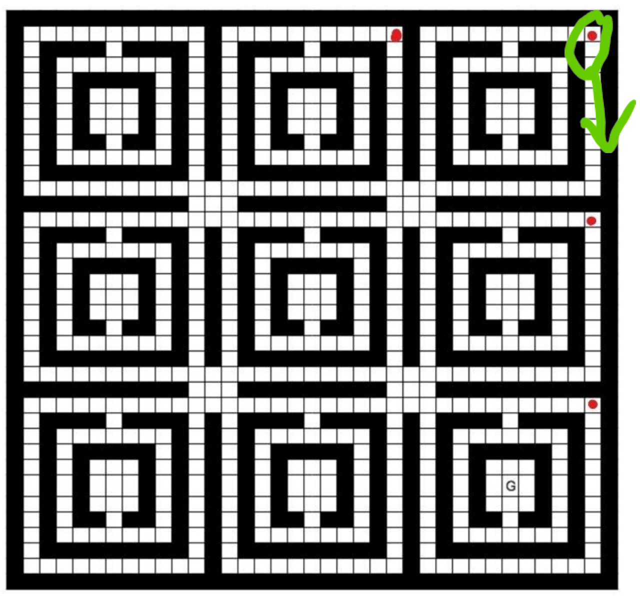
Step 3: Move one step to the left followed by two steps down. Then move as to the right as possible (width of the maze).



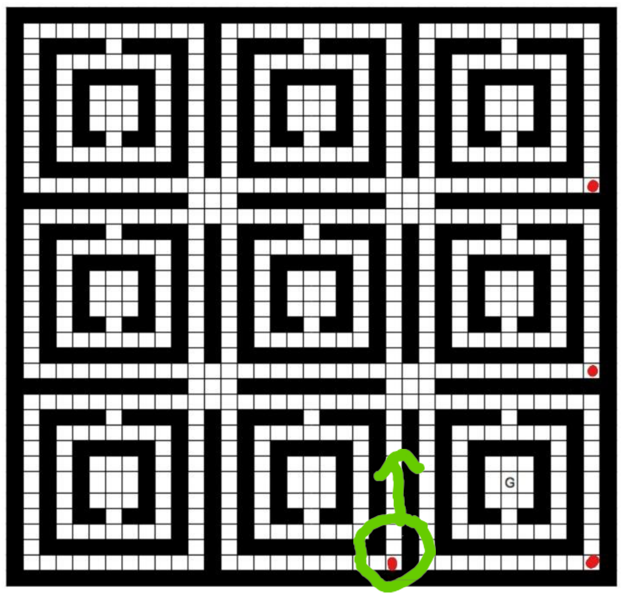


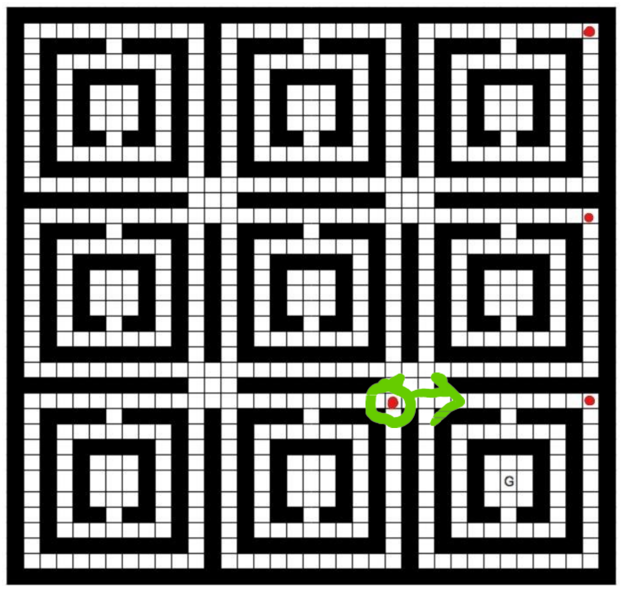
Step 4: Now we need to move as up as possible (the height of the maze) and now to get all the instances of the robot that might be in the 2nd smallest boxes out we need to move 3 steps left.



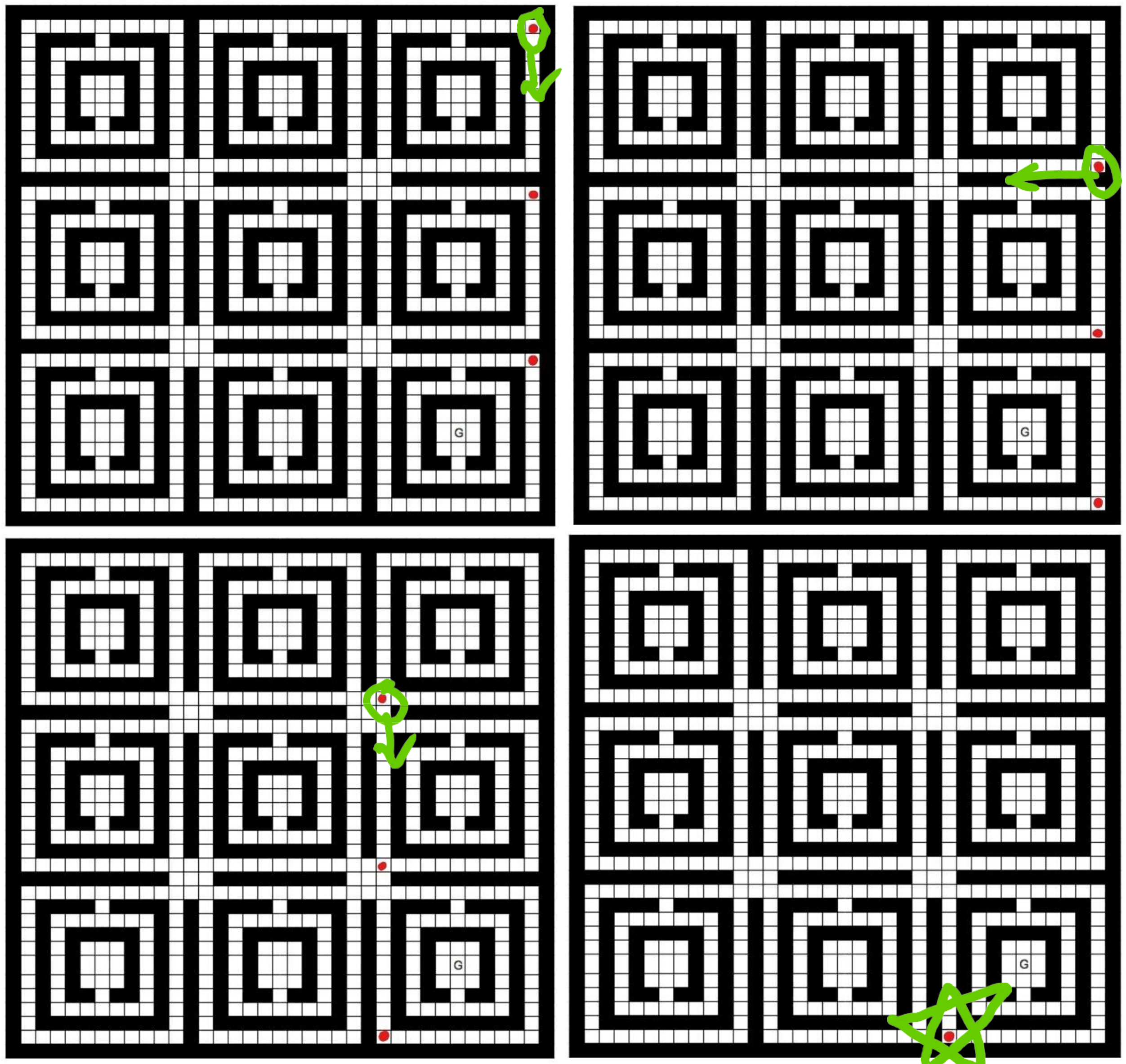


Step 5: Now move up by two steps and this should bring all the instances of the robot that are inside the 2nd smallest boxes out. Post this move as right as possible (which is the width of the maze). This will result in the position of the robot as shown in the above diagram.





Step 6: Now move all the instances of the robot as down as possible (height of the maze). Post this move, move all the instances of the robot positions 10 places up. This is indicated by the green circle. Then we need to move the robot as right as possible (in this case 36)



Step 7: We now move the robot down (the height of the maze). This will result in the instances of the robot being as seen in the second above picture. Now move the robot 10 places to the left and then down by the height of the maze. Now we can see how all the random possible positions of the robots have consolidate into one single point.

The above set of diagrams and steps show that irrespective of the start position of the robot we can bring the robot to the position indicated by the start in the figure above just applying a finite set of moves.

Now that we know where any instance of the robot will consolidate to all we have to do to reach the Goal cell with a 100% probability is:

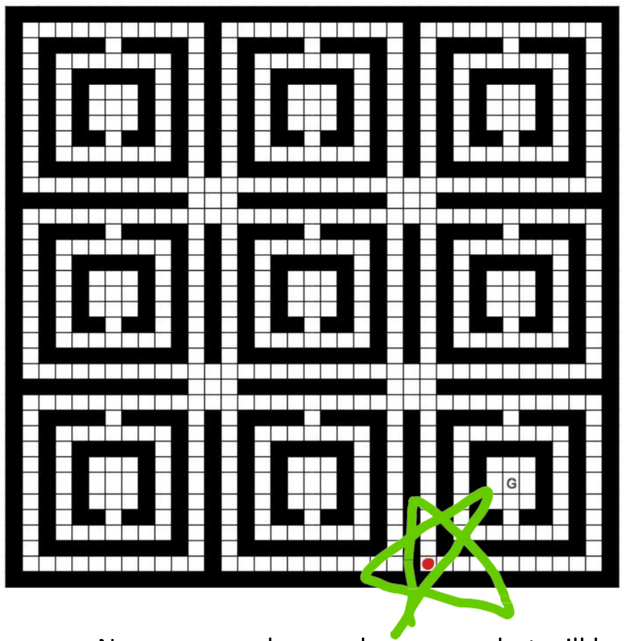
1. Use the above mentioned set of steps to fist be sure about the position of the robot irrespective of the start point.
2. Now move the robot 10 places up followed by 5 places right followed by two places down followed by 3 places right followed by 6 places down followed by 3 places left and then 3 places up.

By following the above mentioned procedure, we can be sure that the robot is at the Goal cell irrespective of it start position.

c) How could you find such a sequence? How could a computer find such a sequence?

The main idea behind solving this problem or finding a sequence that can solve this problem is generalizing the problem. The solution should be able to find a solution or sequence of steps to the goal cell irrespective of the start point. Hence, I took a more generalized approach whereby I took into consideration multiple instances of the position of the robot and found an algorithm or set of transition states that result in the same end result.

Upon applying the above-mentioned transition states irrespective of the start state of the robot it will reach the point indicated by the start in the below figure.



Now once we know where any robot will be after applying the above described algorithm, all we need to do post this is use the transition steps that will get to the Goal cell. Which is pretty straight forward: move up 10 steps, then to the right by 5, followed by 2 steps down, then 3 steps right, followed by 6 spaces down, then 3 spaces left and then 3 spaces up.

d) Write an algorithm to find the shortest sequence of moves you can to reach G independent of where you begin and without feedback. Describe your algorithm in detail, including any design choices you made. What is the sequence of moves?

The algorithm to solve this problem has already been explain in detail in question “b”. But, to reiterate and put the algorithm in a consolidated form:

Algorithm:

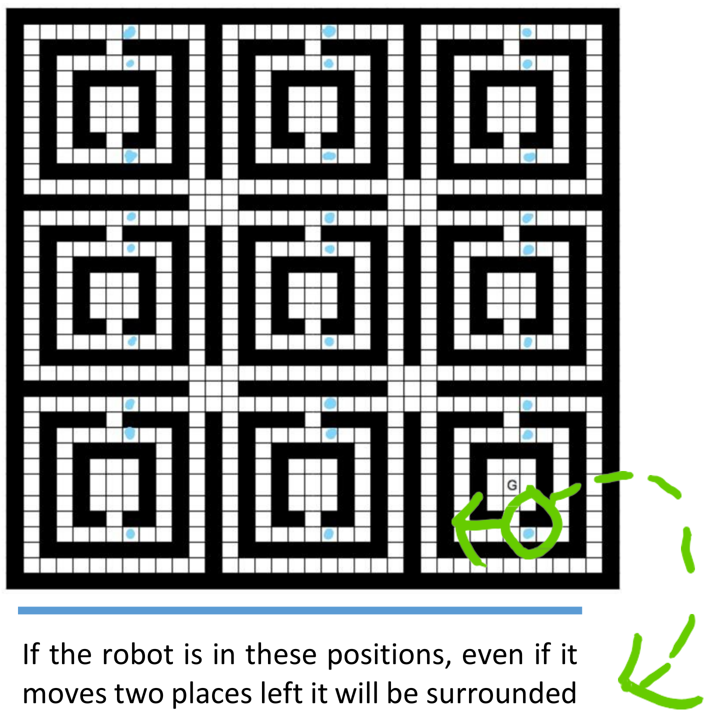
1. Move down, as many steps as possible (height of the maze) in the above case it is 36. This will result in the target hitting the wall no matter where it is.
2. Next move right as many moves as possible (width of the maze) in this case it is 36.
3. Now most of the robot positions should be locked. The next step is to get any instances of the robot that might be with in the smallest squares out. For this we first move 1 step to the left followed by 2 steps down. Now all instances of the robot should be out of the smallest square.
4. The next step will be to get any instance of the robot out of the next perimeter. (Note: changes made to other instances of the robot that are already outside are made redundant by the following moves. Also, we have used the walls to lock out some of the unnecessary transitions of the robot.). Now we need to move as up as possible (the height of the maze) and now to get all the instances of the robot that might be in the 2nd smallest boxes out we need to move 3 steps left. Now move up by two steps and this should bring all the instances of the robot that are inside the 2nd smallest boxes out.
5. Post this move as right as possible (which is the width of the maze). Now move all the instances of the robot as down as possible (height of the maze). Post this move, move all the instances of the robot positions 10 places up. Then we need to move the robot as right as possible (in this case 36)
6. We now move the robot down (the height of the maze). This will result in the instances of the robot being as seen in the second above picture. Now move the robot 10 places to the left and then down by the height of the maze. Now we can see how all the random possible positions of the robots have consolidate into one single point.
7. Now once we know where any robot will be after applying the above described algorithm, all we need to do post this is use the transition steps that will get to the Goal cell. Which is pretty straight forward: move up 10 steps, then to the right by 5, followed by 2 steps down, then 3 steps right, followed by 6 spaces down, then 3 spaces left and then 3 spaces up.

e) Suppose that after each move, you receive an observation / feedback of the form Yt = the number of blocked cells surrounding your location. Let Y0 be the number of blocked cells surrounding your starting location. Again, you get no feedback if the move was successful or not, simply the number of blocked cells surrounding your current location.

e.1) You initially observe that you are surrounded by 5 blocked cells. You attempt to move LEFT. You are surrounded by 5 blocked cells. You attempt to move LEFT. You are surrounded by 5 blocked cells. Indicate, for each cell, the final probability of you being in that cell.

e.2) Write an algorithm to take a sequence of observations {Y0, Y1, . . . , Yn} and a sequence of actions {A0, A1, . . . , An−1} and returns the cell you are most likely to be in.

In order to solve the first part of this question we need understand that only a finite set of cells satisfy this condition. The below image indicates all the cells that are still surrounded by 5 walled cells even after two consecutive left moves. (Note: here we consider even walls that are in diagonal to the current position of the robot).



If the robot is in these positions, even if it moves two places left it will be surrounded by 5 walls at both the instances.

So essentially the probability of the robot being any of the positions other than these boils down to zero. The total number of such positions are 27. Therefore, the probability of the robot being in one of these positions is (1/27).

Now give a set of actions and set of observations finding the current position of the robot essentially boils down to finding and isolating all the points that satisfy a observation for a particular action. This concept has been demonstrated above. This can be mathematically model as shown below.

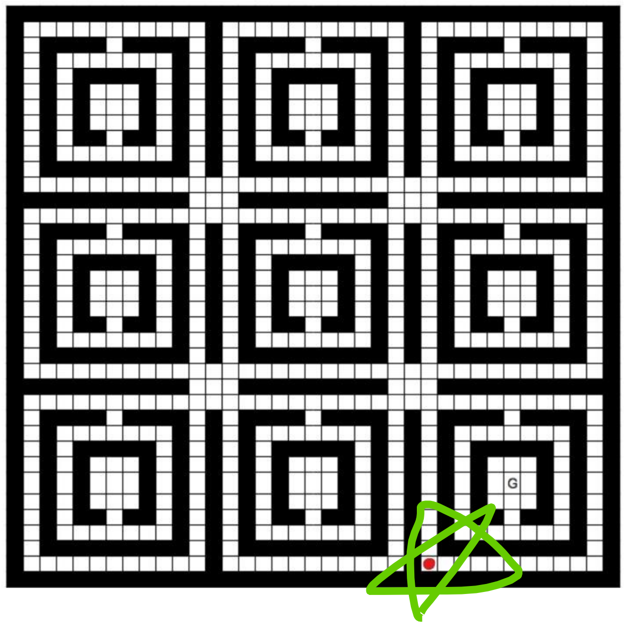
$f(\text{Cell}_t+1)$ = max over all $f(\text{Cell}_t)$:

$P(\text{Cell}_t/\text{Cell}_t\text{-}1)*P(Y_n/A_nY_n\text{-}1]*f(\text{Cell}_t)$

Where, $f(\text{Cell}_t)$ =Maximum Likelihood of Cell being in that state at time t.

Bonus: G was chosen arbitrarily. In the no-feedback case, if you want to determine where you are as efficiently as possible, what square should you try to get to, and what moves should you take to get there?

As stated in solutions to questions (b) and (d) I believe the best approach towards solving this question or problem is to first implement a generalized algorithm that primarily focuses on bringing any instance of the robot to a fixed point on the maze in a fixed number of steps.
Hence, the first step would be to implement the algorithm stated in question (b) or (d) and post this we now know where the robot is located as indicated by the figure below.



The red dot is the point of convergence. This is where any instance of the robot will end up after going through the algorithm. Now that we know where any random instance of the robot is all we need to do from here is apply a finite set of move that will take us to goal node.
Now as it is stated in the question that 'G' has been chosen randomly we need then starting from the point above slowly traverse the grid until we find point 'G'. This can be done in finite time as there are only a finite number of options for the location of 'G'.
(Note: Hence the primary focus through-out all the questions in this assignment is to develop a generalize algorithm that will work for any starting point of the robot. That is why we focused on creating a finite step algorithm that will bring all the instances of the robot to a single point in the maze. From here we can apply a direct path to 'G'.)