1 Task - Search

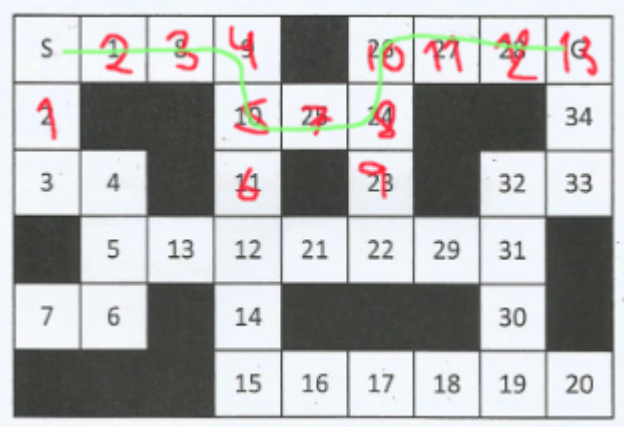
1.1 Represent the robot navigation problem from location S to location G as a search problem with all the relevant elements necessary for its description

Not sure how it should be represented but I believe that since it's a search problem, we will need to use data structures to able to use the information for planning. There should probably be a two dimensional grid including the numbered cells and the blocked cells. The robot has its position, and while planning should check if next cell is passable or not.

1.2 What path would be found from S to G with the following algorithms:

1.2.1 Depth-First search

Depth-First search starts from S and explores each branch of the cell until the goal. An expected path would be:

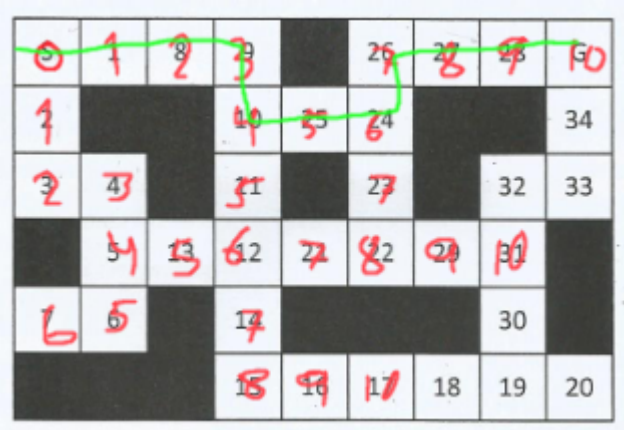


1. [S]
2. Expanding [S] -> [1, 2]
3. Expanding [1] -> [(1-8), 2]
4. Expanding [18] -> [(1-8-9), 2]
5. Expanding [189] -> [(1-8-9-10), 2]
6. Expanding [18910] -> [(1-8-9-10-25), (1-8-9-10-11), 2]
7. Expanding [(1-8-9-10-25)] -> [(1-8-9-10-25-24), (1-8-9-10-11), 2]
8. Expanding [(1-8-9-10-25-24)] -> [(1-8-9-10-25-24-26), (1-8-9-10-25-24-23), (1-8-9-10-11), 2]

9. Expanding [(1-8-9-10-25-24-26)] -> [(1-8-9-10-25-24-26-27), (1-8-9-10-25-24-23), (1-8-9-10-11), 2]
10. Expanding [(1-8-9-10-25-24-26-27)] -> [(1-8-9-10-25-24-26-27-28), (1-8-9-10-25-24-23), (1-8-9-10-11), 2]
11. Expanding [(1-8-9-10-25-24-26-27-28)] -> [(1-8-9-10-25-24-26-27-28-G), (1-8-9-10-25-24-23), (1-8-9-10-11), 2]
12. Goal reached. Final path: (1-8-9-10-25-24-26-27-28)

1.2.2 Breadth-First search

Starts at some arbitrary node of the graph and explores the neighbouring nodes first, before moving to the next level neighbours.



1.2.3 A* search

What would be a good A* heuristic? Since the scenario involves a maze, it would be good to use manhattan distance as heuristic because it is more accurate when dealing with vertical and horizontal paths.

The distance to the goal is calculated for each cell. So the robot will follow the path that is least costly to the goal.

- D = Distance to goal
- H = Heuristic for each cell

1. [S]
2. Expanding [S] -> [1(D+H), 2(D+H)]
3. Expanding [1] -> [(8(1(D+H)) + D+H), 2(D+H)]
   (a) Expand the lowest cost

4. Expanding [2] -> [8(D+H) + (1(D+H)), 3(D+H) + (2(D+H))]
    (a) And expansion continues till goal is reached

1.2.4 Explain the major differences between state-space search and local search. Does it make sense to solve the robot navigation problem using local search.

1. With State-space search, there's a path produced from the search of a graph, however with local search this doesn't apply since local search is only working on the current and the neighbouring states.
2. Using state-space search, the algorithms often keep track of the history and the visited nodes. With local search, there's no history.

One can use local search to solve the problem. However, it might not present the fastest route to the goal from the initial state. If one uses a gradient search (Greedy search together with an heuristic function to determine the cost to the goal from the position of the robot), it may be possible for the robot to reach the goal at some point, perhaps even using the same route as the previous algorithms presented.

2 Task 3 - Knowledge representation

2.1 Transform the predicate logic formula to CNF

$\forall x Human(x) \Rightarrow \exists t FallsAsleep(x,t)$	$t \longrightarrow G1$	$Human(x) \Rightarrow FallsAsleep(x,G1))$
$Human(x) \Rightarrow FallsAsleep(x,G1))$		$\neg Human(x) \vee FallsAlseep(x,G1)$