Introduction to Artificial Intelligence

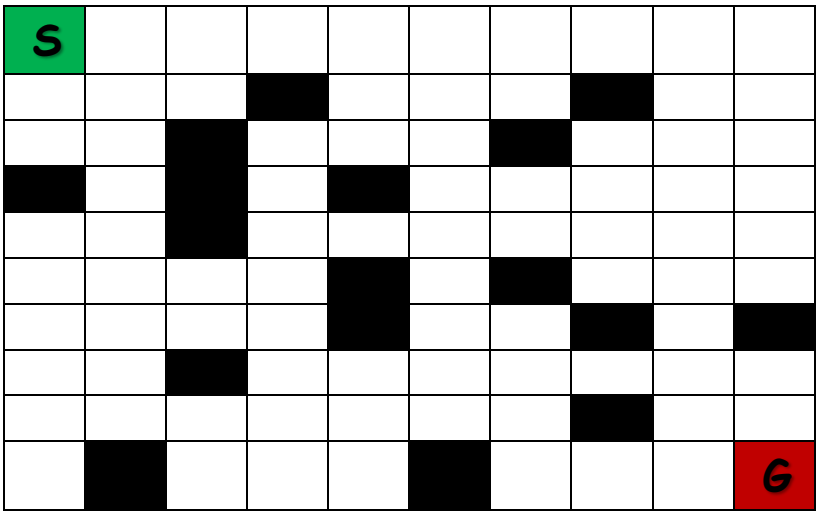
Assignment 1 2023

Amnon Meisels & Yair Vaknin

A* search on a Maze

Overview:

The goal of the algorithms you are requested to implement is to find a path on a maze, that leads from the starting square (green, denoted by S) to the goal square (red, denoted by G). The maze is very simple. It is a square grid with some black squares on it, that are the obstacles. The path needs to go through white squares only.



Moves are in either a vertical or a horizontal direction and each move is counted as a single step by your program. The goal of each algorithm you will implement in your program is to find a free path of empty (white) squares and to count its length and count also the total cost of the search to find it.

The output of the program will compare the solutions and the cost of finding them, by the different algorithms. Needless to say, it will describe the path found by the algorithm from the initial state to the goal state. Paths are a sequence of (x, y) coordinates, starting with (1,1) and ending with (n,n) for the above example.

A, Greedy and Uniform Cost search algorithms*

You are required to implement three different algorithms: The Uniform Cost (UCS) algorithm; the Greedy algorithm; and the A* algorithm. All algorithms have been defined in class and should be implemented as search on a tree. Use

the Manhattan Distance for the heuristic estimate of the distance to the goal state. Remember that A* uses the sum of the estimate and of the cost already paid as its heuristic function f().

Some objects for heuristic search

In order to perform tree search, the tree search needs to keep several objects that we have defined in class.

Frontier – a set of nodes on the tree that the algorithm performing search is currently looking at, in order to select the next node to expand. It starts with only the start node in it and each time a node is selected and expanded, the resulting nodes are added to it.

Your implementation must use a priority queue to represent the Frontier. In other words, order the candidate nodes for expansion by their priority whether heuristic function value or just path length. The frontier is an important concept, please make it clearly defined in your implementation.

Comment: To perform the search on a graph one needs to keep visited nodes in a set that can be called ***Closed***. Best to use the set data structure of Java.

Cost function – the function ***g(x)*** computes the cost paid up to the current node from the root (the ***start node***) of the search tree. For our path finding problem this is just the count of moves. The same cost function is used by all algorithms.

Heuristic function – the function ***h(x)*** computes an estimate of the cost to get from node **x** to the goal. The heuristics to be used by your implementation have been described above.

Input

The input to the program is a file that includes a Maze in the following format. The first line of the file defines the size of the maze. For example, 8 means an 8x8 square. Next, come rows in the number described by the size and each row has numbers describing the Maze in that row. The number 0 is a white square and 1 is a black one.

For example, the above figure has the following 3 first lines in its file:

10
0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 1 0 0
0 0 1 0 0 0 1 0 0 0

Output

A file that includes the solutions and costs found by each algorithm, where at the beginning of each row the cost of the steps and the length of the path found are given, in the following format:

Greedy (search 17 nodes; path length 9): (1,1),(1,2),(1,3),(2,3)...(n,n)
UCS (search 22 nodes; path length 19): (1,1)...(n,n)
Astar (search 32 nodes; path length 8): (1,1)...(n,n)

Deliverables

- The program and code sent to the grader.
- A document describing how to run the program and read the output.
- A couple of examples, demonstrating the output of the different algorithms (in the document).

Due date: Sunday, August 27, 2023.

Enjoy !!!