Universitat de Lleida
Grau en Tècniques d'Interacció Digital i de Computació
Computing Techniques

Assignment 1

Solving a maze

Cristian Oprea and Xavier Vila

13th November, 2020

1 Testing both algorithms on different mazes

Test both algorithms on different mazes. Try different sizes (at least 10x10, 15x15 and 20x20) with different wall compositions and start and end nodes. Write a short report with these results and explain why you think one algorithm beats the other in terms of finding a shorter solution. Hint: BFS provides shorter solutions.

We tested both algorithms with the examples provided aswell as three examples created by us. As stated in the hint, the BFS algorithm always gives shorter solutions. The following list shows exactly what we mean:

- Test_maze's lenght was 18 in BFS and 24 in DFS
- Test_mazev2's lenght was 6 in BFS and 12 in DFS
- Test_mazev3's lenght was 45 in BFS and 47 in DFS
- Test_maze_10x10's lenght was 11 in BFS and 19 in DFS
- Test_maze_15x15's lenght was 35 in BFS and 47 in DFS
- Test_maze_20x20's lenght was 62 in BFS and 172 in DFS

As we can see, the DFS algorithm usually is much longer (except in test_mazev3's case, which is similar, but still longer), this meaning that BFS is the algorithm to use in these types of mazes. Take test_maze_20x20's example. As we can see, BFS solved the maze in 62 steps, which is almost one third of the steps of DFS's solution (172)

This is due to how the algorithms work. The BFS algorithm always searches for the shortest path. The DFS algorithm, on the other hand, starts searching in the left direction because of the way it's programmed, so this doesn't guarantee that it's the shortest path.

2 $\mathcal{O}$ costs

In terms of the following values:

b $\rightarrow$ branching factor: maximum number of children a node can have
d $\rightarrow$ depth: length of the shortest path between the start and end nodes
m $\rightarrow$ maximum length: length of the maximum path that can be achieved

What is the $\mathcal{O}$ cost in terms of visited nodes of finding the solution for the BFS algorithm?
If we imagine the labyrinth as a graph, every cell will become a node, and the children of the cell, will be the children of every node. This way we can clearly see that every level of nodes, would multiply their sons. This would be $(b * b)$ and would happen d times. So the final cost of the BFS algorithm will be $\mathcal{O}(b^d)$

What is the $\mathcal{O}$ cost for the DFS algorithm?
This time will be the same but due to that DFS doesen't find the shortest path, we will not multiply $(b * b)$ d times, instead will be, at most, m times. To sum up, the final cost of the BFS algorithm will be $\mathcal{O}(b^m)$