CS2040S: Data Structures and Algorithms

Tutorial Problems for Week 8

For: Oct 11, 2019

Problem 1. Multiverse Malarchy
Professor Paths has discovered something amazing: we live in a *multiverse*. There exist n parallel universes that are connected via m inter-dimensional wormholes. These wormholes allow for near instantaneous travel between the universes but some wormholes allow travel in only one direction. This means that you can travel between any two universes (A and B) as long as there is a path from A $\rightarrow$ B and back again (from B $\rightarrow$ A).

Problem 1.a. Describe the most efficient algorithm you can think of to determine if a path exists from one universe to another, and vice-versa. Remember the wormholes may only allow travel in one direction so, the path from A $\rightarrow$ B may not be the same from B $\rightarrow$ A. State the time complexity of your method in terms of n and m.

Stability Woes. Professor Paths has called for an emergency meeting. Unfortunately, it looks like the multiverse may be unstable. If it is unstable, there will be a multi-dimensional collapse on Dec 4th 2019 at 9:13am, ending all existence as we know it!

The multiverse is *stable* if every universe is reachable from every other universe via some path through the wormholes. Otherwise, the multiverse is *unstable*. If it turns out the multiverse is unstable, we can all go home and enjoy our final weeks before the collapse.

Problem 1.b. Describe the most efficient algorithm you can think of to determine if the universe is stable. State the time complexity of your method.

Problem 2. A-Maze-ing Rescue! The diabolical Dlorah Hos has stuck you in a maze! If you manage to find your way out, you can steal her most prized posession, the *Tome of Computer Science*. Otherwise, you get eaten by her pet python. But there's a twist, she's also trapped your friend Naruto in the maze. You're going to have rescue Naruto *before* heading to the exit. Good luck!

The maze is represented as A, a 2D array of pixels of size $n \times n$. Each pixel $A[i][j]$ can either take on one of four values: zero, one, two, or three. $A[i][j] = 0$ represents an open area, $A[i][j] = 1$ represents a wall, $A[i][j] = 2$ represents Naruto, and $A[i][j] = 3$ is the exit. You can move into open areas but cannot walk through walls. You start at $A[0][0]$ and are only allowed to move in four directions: up, down, left, and right; you cannot move diagonally. The figures below show you example mazes and shortest paths to Naruto and then to the exit.

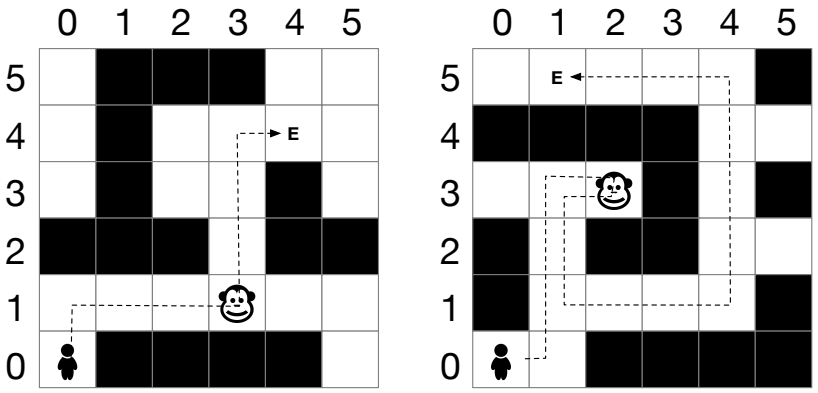

Figure 1: Two examples showing a 6×6 maze and a path to the exit. Here, we have illustrated open spaces as white cells, walls as black cells, Naruto as the monkey face icon, and the exit as 'E'.

Problem 2.a. Given A, provide an algorithm that will find *the shortest path* to rescue Naruto and then to the exit.

Problem 2.b. What is the running time and space cost of your algorithm in terms of n? Explain why.