DCS 229: Data Structures and Algorithms April 15th, 2024

Aru Poleo Lab 5

Reflection

In my implementation of DFS and BFS in C++, BFS consistently outclassed DFS in pretty much every metric, which I am a little bit confused by (perhaps I did something wrong regarding the implementation in a way that escaped my tests?).

			Maze Size		
%	Alg	Metric	5x5	20x20	50x50
.2	DFS	Duration (ms)	19	107	127
		Pushes	14	66	69
		Path Length	3	27	61
	BFS	Duration (ms)	13	81	88
		Pushes	10	47	45
		Path Length	2	8	19
.5	DFS	Duration (ms)	15	6	2
		Pushes	3	3	1
		Path Length	1	6	6
	BDS	Duration (ms)	4	5	2
		Pushes	3	3	1
		Path Length	1	3	6

Fig 1. Results from benchmarks

I ran 10,000 trials per maze per algorithm, and I did so with seed variation but the same 10,000 seeds for each of the iterations. At the higher levels of blocked %, we see that both the DFS and BFS algorithms seem to perform exceptionally well.

However, the reason they do so is because they only mazes where they were able to determine the path to the goal were the ones where the start and end were already close to each other. Distance from the start to the end seems to correlate (to an extent) with how likely there is to be a path to the end of the maze.

Consistently— and this was to be expected— BFS outperformed DFS in path length at every instance. This is because, unlike DFS, BFS will find a path that is the shortest possible length.