Ryan Kennedy

Professor Clark

CS 3353

October 17, 2019

Analysis of Search Algorithms

I believe I normalized my data wrong since the numbers are very small and seem to not be the right result. I looked up how to normalize data and used this site to normalize the data in excel <https://yourbusiness.azcentral.com/normalize-excel-8475.html> .

A screenshot of a computer

Description automatically generated These are the graphs and table of the normalized data and a table of averaged data for the adjacency list. From this data you can tell that A\*, Dijkstra, and BFS recursive/iterative had a small amount of nodes in its path. This makes sense for A\* and Dijkstra sincethere are finding the most optimal path. This also makes sense for both BFS algorithms since BFS searches neighbor nodes and can result A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedin a smaller path than DFS which goes down an entire path until hitting the end. I can also tell from this data that A\* and Dijkstra had the least amount of distance in their pathes. This makes sense since A\* and Dijkstra are searching for the least amount of distance in their paths. From the averaged data, I can tell that Dijkstra and A\* took more than double the time on average than the other algorithms. This is due to the fact that Dijkstra and A\* explore much more nodes on average since the algorithms are looking for the most optimal route not just the first route found.

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA close up of a map

Description automatically generatedThese are the graphs of the normalized data for the adjacency matrix. The algorithms are slower on average using an adjacency matrix than the algorithms are using an adjacency list. I believe this is due to the fact the algorithms must iterate through the whole matrix’s size to find sibling nodes. I also found that the algorithms mostly functioned the same except for the DFS algorithms. The DFS iterative implementation using an adjacency matrix had ~500 more nodes on average than the implementation using an adjacency list. I believe this is due to the adjacency matrix looping through the siblings differently and the DFS algorithm went down a different path using the adjacency matrix.