Jason Chen

DS 210

Prof. Kontothanass

5/4/22

Data Set :

The dataset that I picked was the “Pennsylvania road network”. This dataset consists of intersections and endpoints represented by nodes. Then there are the roads connecting to these intersections or endpoints that are represented by undirected edges. In total there are 1088092 nodes and 1541898 edges. Some other inconsequential information included that the largest WCC (weakest connected components) is 1087562, and the SCC (strongest connected components) is also 1087562 - which is surprising. The same applied for the WCC and SCC of edges it is both 1541514. The dataset is huge as it contains over 1000+ vertices, but it only consists of numbers without actual names, so we don't know the roads that the nodes correspond to.

Project:

The project read the files and create a graph based on that consist of edges. The edges will have the starting node, the ending node, and the distance which I set as 1. This is because the graph is considered unweighted so I wanted to make them the same weight. Based on the graph that is created, a algorithm for the shortest path will be run, and through the algorithm it will return Some((distance,path)). The algorithm for the shortest path was inspired by the BFS and Dijkstra's algorithm .The path will be vital to the later algorithm which calculates a simplified version of betweenness centrality. In the simplified version of betweenness centrality, based on the number of times the nodes appears it will be assigned a frequency value. For example, if the nodes appear 9 times in the paths of every nodes, then it will obtain a frequency value of 9. In the end, the top 5 node that appeared the most frequently will be return and be considered the most important. Then in my main function the top 4 nodes will be return instead of the top 5 because the top 4 have the same amount of frequency while the rest have similar frequency so it will varies. For this reason, I will return them based on biggest to smallest if the frequency is the same to keep the result consistent. The ending output of

the code will be  Node 859326 (frequency: 9), Node 847932 (frequency: 9),  Node 759553 (frequency: 9),  Node 674502 (frequency: 9). This means that the most important nodes to the graph will be these 4 as they appeared the most often in the graph. This is interesting because there isn't a single most important node, but rather 4. Since the graph only consisted of nodes without names, we won't be able to determine the actual road names essential to the network.