B"H

CDS DS 210 - FINAL PROJECT

Moshe Rosenstock

Report:

Data:

- I selected my Graph from a Dataset made by Huawei, which represents a Twitter Communication Network, the Graph is Directed and Labeled, it has 1000 Nodes and 50153 edges. Huawei uses this kind of dataset to perform Customer Analysis in their Marketing Strategy through Social Network Analysis.

Implementation of the Project:

- What I did for the Final Project was the following. First I imported and cleaned the data. To do so, I had to convert the .xlsx file which contained the names and connections of 1000 people, into a text-file that replaces each user/person name with one unique node (labeled from 0-999) and then create edges for the nodes that had connections between them. Secondly, I had to read the text-file (named edges_huawei.txt) and convert it into a Vector of tuples in rust (using files I/O). Then I had to create a type struct Graph and its implementations, in order to convert my vector of tuples into our desired version of a Graph. After I created my Graph, I thought that the best way to filter the vectors was to perform the Page-Rank algorithm for all the nodes inside our Graph, based on their Page-Rank score select the Top 50 nodes of the graph.
- After selecting the Top 50 vertices, I created a new graph, that contained only the connections/edges between the Top 50 nodes (exclusively between them). This gave us a new graph that contained only the links between the Top 50 users of our dataset, which is based on filtered data and can be useful to generate many interesting conclusions. With this new graph, which contains only the top vertices, I performed Breadth-First Search (BFS) algorithm. The BFS algorithms helped us to find the shortest path inside our graph; our function performed a BFS on a graph starting at a given source node. It traverses each of the source node's neighbors, and then each of their neighbors, until all vertices in the graph have been visited. And then returns a vector that contained all the vertices in the order they were visited. Then, the final part was to calculate the distance between each of the Top 50 nodes, as it can give us a lot of insides into our data and graph.

(Note that the graph containing the Top 50 edges, could vary each time we run our code, as it contains a finite amount of randomness.)

Conclusions:

- After running our algorithm, I could appreciate that for most nodes inside the Top50, the distance between them was between 1 and 2 (at most 3). Which is a very small distance, compared to other social networks. I got to the conclusion that the regular distance between two popular nodes tends to be considerably less than two regular nodes, as usually, the popular/top nodes tend to be connected or have at least one connection in common.