Grace Chong
DS210

Project Write Up

Data:
From the Stanford Network Analysis Project (SNAP)
https://snap.stanford.edu/data/ego-Twitter.html

I chose a social network dataset that is a txt file of two columns of Twitter users.
In a folder, you can also find datasets that include node features (profiles), circles, and ego networks from Twitter. It is said that this data set can be valuable for studying social network structures and dynamics, particularly focusing on how users organize their social circles on platforms like Twitter.

Use:
In my graph.rs I constructed a graph data structure using Struct, which uses an adjacency list where a hashmap tracks the nodes to a hashmap of edges. And an in degree list where my hashmap tracks the in degree number of incoming edges for each node. The methods I used where new() to create new 'Graph' with the empty adjacency list and indegree list, add_edge9src, dest) to update the adjacent list and in degree list, the 'number_of_ edges()' to calculate the total number of edges in the graph, number_of nodes() to calculate the total number of unique nodes in the graph, and a print summary to see what the first 5 nodes would be. The Function I used was the parse_dataset(file_path: &st) -> io::Result<Graph> in order to parse a dataset from file path to create the graph.

For my algorithms, I create 2 separate mods (algorithm.rs and algorithm2.rs). These modules focused on six degrees of separation and average distance, but it also covers the functionalities of Breadth-First Search (BFS)and six degrees calculation.
In algorithm.rs, I implemented BFS to calculate the distance from my given starting node to all the other nodes in the graph. I also implemented a function to calculate degrees of separation, which determines whether two nodes are within a specific number of degrees of separation. This was to see whether users are six or fewer social connections away from each other.
In algorithm2.rs, the function computes the average distance between pairs of nodes in the network. This calculation is significant as it offers insights into how close the overall network is.

For my main.rs, which is the initial creator of the graph and to the implementations of my algorithms, I included key operations that calculate the average distance between pairs of nodes in the graph and checked the degrees of separation between specific

nodes. This algorithm provides an understanding of the overall interconnectedness and reachability within the network.

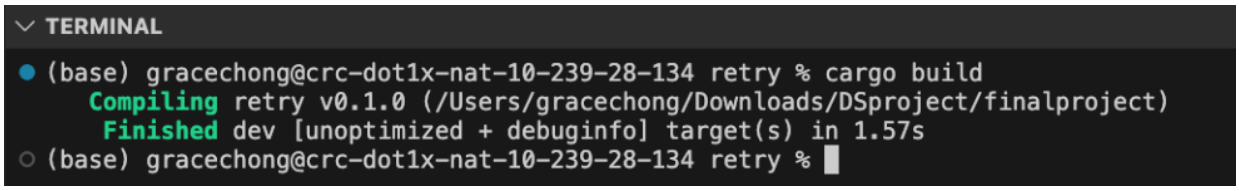
Tests

I included a total of 4 tests. 2 for algorithm.rs and 1 for algorithm2.rs. The tests I had for algorithm.rs was one using the twitter dataset and one from a constructed dataset. The test I had for algorithm2.rs was a constructed dataset. Not only were tests required for this project, but it helped validate the correctness of both the graph creation and the algorithms implemented.

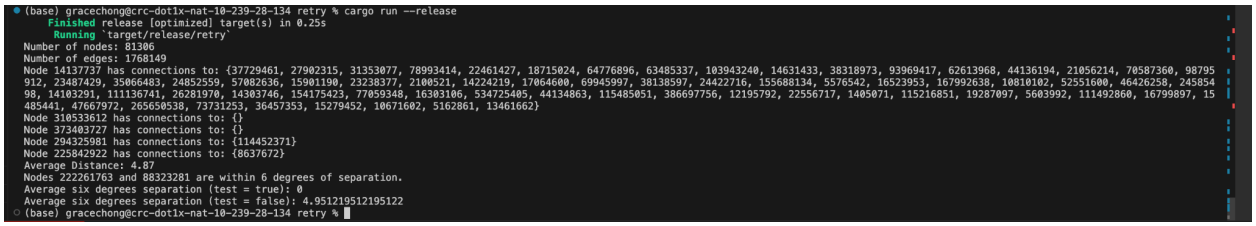
Results:

In order to get my shown results, I implemented cargo build, cargo run –release, and cargo test.

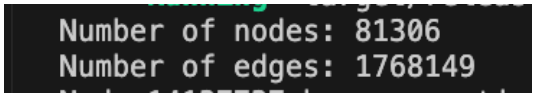
I used cargo build in order to compile the project and to manage my dependencies.



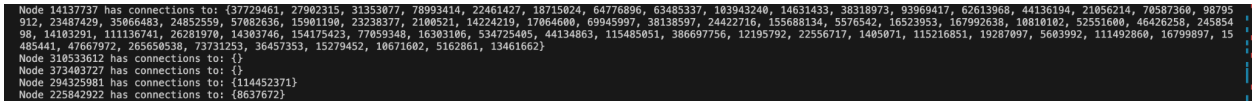
I used cargo –release in order to add optimization, and because it is ideal when testing the performance of my code.



It returned the accurate number of nodes and edges.



And the image below is a return of performing bfs to 5 different RANDOM nodes, so it is different every time I run it again. This was to find the shortest path in unweighted graphs, to help visualize how the nodes are connected, and its connections.



The average distance also changes every time I run it again, because it is random sampling without replacement. Though it may not be as accurate or efficient, it was a strategic choice to allow for a practical analysis of this large graph. It was to prevent expensive calculations and long waiting time.

```
Average six degrees separation (test = true): 0
Average six degrees separation (test = false): 4.640449438202247
```

The shown result of the image below indicates the average approximation of 4.64 degrees of separation which suggests a highly interconnected network. Which shows that in this graph, the twitter users are well connected.