DS210 Final Project
Vivaan Mehta

Project Thesis

The primary goal of this project was to explore the structural properties and social connectivity patterns of a real-world social network using graph algorithms implemented in Rust. I selected the SNAP ego-Facebook dataset, which models user friendships in an anonymized Facebook network. My analysis was rooted in two key questions: How closely connected are individuals in a social network? and Which users occupy central or influential positions in that network? These questions directly align with classic ideas in graph theory, particularly the "six degrees of separation" hypothesis and the study of centrality measures.

To answer these questions, I implemented and analyzed the graph using core techniques from class, including Breadth-First Search (BFS), shortest path estimation, and centrality calculations. The idea was to gain insight into how real people are linked within a social graph, whether certain individuals play more important structural roles, and whether the graph demonstrates characteristics of "small-world" networks—those with short average path lengths and highly connected hubs. I also included several preprocessing functions that allowed me to clean, inspect, and summarize the dataset prior to algorithmic analysis.

Dataset

The dataset I used is the facebook_combined.txt graph from the Stanford Network Analysis Project (SNAP). It represents a single large connected component of the Facebook social graph, consisting of 4,039 users (nodes) and 88,234 friendships (edges). Each line in the file is an undirected edge connecting two user IDs, indicating a mutual friendship between them. There is no additional metadata such as age, location, or gender—this makes the dataset clean and focused purely on structural relationships, which is ideal for a graph-theoretic project.

What makes this dataset particularly interesting is its real-world relevance. Social networks like Facebook exhibit non-random patterns, with clustering, hub formation, and short paths between users. This dataset is known to follow the "small-world" phenomenon, meaning that any two people are connected through a small number of intermediate friends. Additionally, the size of the dataset is just right—it's large enough (4,000+ nodes) to demonstrate interesting behavior, but small enough to analyze efficiently using Rust.

Before running any algorithms, I verified the integrity of the dataset by checking for isolated nodes (those with zero connections), confirming the graph's size, and computing average degree and node connectivity. This preprocessing helped ensure that my analysis would run correctly and interpret meaningful patterns.

Code Explanation and Data Processing

The project consists of five main Rust modules: main.rs, graph.rs, bfs.rs, centrality.rs, and utils.rs.

In graph.rs, I implemented the Graph struct using an adjacency list (HashMap<usize, HashSet<usize>>). The function from_file() reads the facebook_combined.txt file, parses each line as an undirected edge, and constructs the graph. Supporting methods like neighbors(), node_count(), and edge_count() provide easy access to the graph's internal structure and size.

In main.rs, I began with data preprocessing functions to extract structural insights. I created a function find_isolated_nodes() to detect users with no connections (degree = 0). It returned an empty vector, indicating that all users in the graph are part of at least one friendship. This confirmed that the graph is connected. I also implemented find_most_connected() to identify the node with the highest degree—node 107, with 1,045 friends. Finally, I wrote average_degree(), which calculated the mean degree across all nodes. The result (~43.69) suggested that users, on average, have 43 friends.

The next component was "Six Degrees of Separation" analysis. I used a BFS-based approach in bfs.rs to compute shortest paths between randomly sampled node pairs. Specifically, I sampled 1,000 random pairs and measured the number of hops required to get from one to the other. The average shortest path length was 3.670, meaning that in this Facebook network, most users can be connected through fewer than 4 steps—strongly confirming the small-world hypothesis.

Following this, I implemented centrality measures in centrality.rs.

- Degree Centrality simply counts the number of neighbors each node has.
- Closeness Centrality calculates how close each node is to all other nodes in terms of average shortest path length.
- Betweenness Centrality estimates how often a node lies on the shortest paths between other node pairs—this was done using Brandes' algorithm.

The results were revealing. Node 107 was consistently ranked in the top 1–2 positions across all three centrality measures. This confirmed that it was not only the most connected node, but also a structural bridge and central figure within the network. This node likely represents the central ego node in the original Facebook crawl, and its dominance reflects real-world social hierarchy.

Tests

I included two unit tests in main.rs to validate correctness. One confirmed that the graph loads with the correct number of nodes and edges; the other tested bfs_distance() on a small hand-

crafted graph to ensure shortest path logic worked correctly. These tests gave me confidence that the algorithms would scale accurately on the full dataset.

```
running 2 tests
test tests::test_bfs_distance_small_graph ... ok
test tests::test_graph_load_and_count ... ok

test result: ok. 2 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.37s
```

Output Summary

```
Loaded graph with 4039 nodes and 88234 edges.
Number of isolated nodes: 0
Most connected node: 107 with degree 1045
Average degree of graph: 43.69
Average shortest path over 1000 samples: 3.703

Top 5 Degree Centrality:
 1. Node  107 → 1045
 2. Node 1684 → 792
 3. Node 1912 → 755
 4. Node 3437 → 547
 5. Node    0 → 347

Top 5 Closeness Centrality:
 1. Node  107 → 0.45969945355191255
 2. Node   58 → 0.3974018305284913
 3. Node  428 → 0.3948371956585509
 4. Node  563 → 0.3939127889961955
 5. Node 1684 → 0.39360561458231796

Top 5 Betweenness Centrality:
 1. Node  107 → 3916560.144440741
 2. Node 1684 → 2753286.6869082823
 3. Node 3437 → 1924506.1515714747
 4. Node 1912 → 1868918.2122567494
 5. Node 1085 → 1214577.7583604786
```

The output of the program offers a rich and structured overview of the social network's topology and reveals several meaningful patterns. Upon loading the dataset, the program first confirms that the graph contains 4,039 nodes and 88,234 edges, which aligns with the expected size of the SNAP Facebook ego network and ensures the file was parsed correctly. The next metric reported was the number of isolated nodes, which turned out to be zero. This means every individual in the graph is connected to at least one other person, reinforcing the idea that the dataset represents a single, fully connected community—an important prerequisite for analyzing path-based metrics like centrality or shortest paths.

The program then identifies the most connected individual as Node 107, who has a degree of 1,045. This is a remarkably high number of direct connections, suggesting that this person is either the central ego around whom the graph was built or someone who plays an exceptionally important role in the network. Following this, the average degree across all users was calculated to be 43.69, indicating a well-connected network where, on average, each user has roughly 44 friends. This degree density is characteristic of real-world online social platforms, where users commonly form many overlapping circles of friendship.

One of the most significant metrics in the output is the average shortest path length, computed from 1,000 randomly selected pairs of users. The result, 3.703, suggests that most users are just 3 to 4 connections away from any other user. This strongly validates the small-world phenomenon, which posits that most social networks have short average path lengths even if they contain thousands or millions of nodes. This result not only aligns with academic literature on network theory but also shows that Facebook users are remarkably close to one another in terms of degrees of separation.

The program also computes three centrality metrics to identify the most important or influential nodes in the network from different perspectives. The Top 5 Degree Centrality results highlight Nodes 107, 1684, 1912, 3437, and 0, all of whom have a high number of direct friends. These nodes represent the most socially connected users in the graph and are likely to be seen as popular or influential based on sheer volume of interactions.

Next, the Top 5 Closeness Centrality rankings—Nodes 107, 58, 428, 563, and 1684—indicate which users are most efficiently positioned in the network. Closeness centrality doesn't necessarily depend on how many friends a user has, but rather how quickly they can reach everyone else. The presence of nodes like 58 and 428, who were not in the top 5 for degree, reveals that some users are exceptionally well-placed within the graph, even if they aren't the most popular. This suggests a level of strategic positioning—users who are centrally embedded across many subgroups.

Finally, the Top 5 Betweenness Centrality results—107, 1684, 3437, 1912, and 1085—highlight users who act as bridges between otherwise distant parts of the network. These individuals may not have the most friends or be the fastest to reach everyone, but they serve as critical intermediaries in the flow of information or influence. Their structural importance lies in how many shortest paths depend on them, which is essential for understanding the resilience or vulnerability of the network.

Overall, these outputs reveal a layered picture of network influence and accessibility. Node 107 consistently appears at the top of every list, making it the undisputed superhub—a user who is not only highly connected but also centrally located and structurally essential. Meanwhile, the presence of different nodes in the closeness and betweenness lists underscores the idea that "important" nodes vary depending on the context—popularity, efficiency, or strategic control. This detailed output not only verifies known properties of social graphs but also provides an intuitive understanding of the Facebook network's internal structure**.**

Usage Instructions

To build and run the project:
cargo build
cargo run --release

To run tests:
cargo test

Place facebook_combined.txt in the root directory of the project (next to Cargo.toml). The program runs in under 15 seconds. The betweenness centrality computation takes the most time due to its algorithmic complexity.

Conclusion

This project demonstrated how graph algorithms can uncover deep insights in social networks using Rust. By applying BFS, shortest path sampling, and centrality analysis, I was able to quantify the structure of the Facebook ego graph and validate known phenomena like the "six degrees of separation." Rust proved to be an effective language for implementing efficient graph processing, and modular code organization helped keep logic clean and scalable.

My preprocessing functions, correctness tests, and detailed output formatting all contributed to the overall clarity and completeness of the project. This analysis highlights how structural properties—like degree, distance, and centrality—offer powerful ways to interpret real-world social behavior, even from anonymous network data.