DS210 Final Project Writeup
Karrington Riley

The Data

The data used in this project represents the international E-road network, a road network located mostly in Europe. The network is undirected, where nodes represent cities, and an edge between two nodes denotes that they are connected by an E-road. The data is stored in a CSV file with two columns, where each row represents an edge between two cities.

Here is a link that explains the data you are looking at.
http://konect.cc/networks/subelj_euroroad/

In my file, the data is saved as euroad.csv.

Objective

The primary objective of this project is twofold: to deepen the understanding of the Breadth-First Search (BFS) algorithm and to conduct comprehensive graph analysis on the international E-road network dataset.

Understanding BFS Algorithm

1. BFS Implementation: The project involves a detailed implementation of the BFS algorithm, specifically tailored for an undirected graph. The BFS algorithm is applied to find the shortest path between two cities in the E-road network. This provides a hands-on exploration of BFS in the context of real-world graph data.

2. Path Reconstruction: An essential part of BFS is the ability to reconstruct the shortest path once it has been found. The code includes logic to reconstruct and display the shortest path between two given cities, enhancing the understanding of how BFS operates in a practical setting.

Graph Analysis

The second part of the objective focuses on conducting graph analysis on the E-road network, using the implemented BFS algorithm as a tool. The analysis includes:

1. Degree Calculation: The project calculates the degree of each city in the graph. The degree of a city represents the number of E-roads connected to it. This analysis provides insights into the connectivity and importance of individual cities within the road network.

2. Local Bridges: The number of local bridges in the graph is determined. A local bridge is an edge whose removal would increase the number of connected components in the graph. Identifying local bridges can reveal critical connections in the road network.

3. Centrality Measures: The project calculates degree centrality and closeness centrality. Degree centrality measures the importance of a city based on the number of direct connections it has. Closeness centrality considers the average distance from a city to all other cities. These measures offer a quantitative assessment of the significance of each city in the network.

4. Network Diameter Calculation: The network diameter, representing the longest shortest path between any two cities, is calculated. This metric provides an understanding of the overall efficiency and interconnectedness of the E-road network.

The Code

In this Rust project, my goal was to delve into the Breadth-First Search (BFS) algorithm and conduct a thorough analysis of the international E-road network. The code is organized into three main components: `main.rs`, `lib.rs`, and `integration_tests.rs`, each serving distinct roles in achieving the project's objectives. The `main.rs` file serves as the entry point for the application, orchestrating the execution of various functionalities and providing a user interface for interacting with the graph-based algorithms. This file handles user input, reads data from a CSV file, constructs an undirected graph, and performs graph analysis, including calculating centrality measures, network diameter, and identifying local bridges.

The core logic and data structures reside in `lib.rs`, encapsulated within a `Graph` struct. This module defines methods for manipulating the graph, such as adding undirected edges, sorting adjacency lists, finding the shortest path using breadth-first search (BFS), and conducting graph analysis, including degree calculation, identification of local bridges, and centrality measures. Additionally, the module incorporates a testing function, `example_function_for_testing`, for evaluating specific aspects of the graph's behavior.

To ensure the code's reliability and maintainability, I have implemented a separate `integration_tests.rs` module. This module contains test functions, such as `test_example_function_for_testing`, which systematically assess the correctness of individual

components within the `lib.rs` module. These tests are essential for validating the behavior of functions, preventing regressions, and facilitating the identification of potential issues during development.

This organizational structure promotes modularity, readability, and testability, contributing to the project's overall robustness and facilitating future enhancements. The division of responsibilities among the three components enables a clear separation of concerns, making the codebase more comprehensible and fostering collaborative development.

The Output

```
   Compiling opt2 v0.1.0 (C:\Users\Karrington Riley\Desktop\DS210_FinalProject\Opt2\Opt2)
    Finished dev [unoptimized + debuginfo] target(s) in 2.03s
     Running `target\debug\main.exe`
Max Vertex Label: 1174
Number of Edges: 1417
Number of Vertices: 1174
Enter the start node:
3
Enter the end node:
5
Shortest Path from 3 to 5: [2, 3, 4]
The graph is not connected, so the network diameter is undefined.
Average Degree: 2.41
Average Closeness: 0.01
Top 5 Cities with Highest Degree of Distribution:
City 284: Degree 10
City 236: Degree 8
City 137: Degree 8
City 107: Degree 8
City 39: Degree 8
Bottom 5 Cities with Lowest Degree of Distribution:
City 1: Degree 1
City 5: Degree 1
City 15: Degree 1
City 21: Degree 1
City 58: Degree 1
Number of Local Bridges: 0
```

1. Max Vertex Label (1174): This tells me that the highest city index in our dataset is 1174. It makes me curious about what city or location this corresponds to, considering it has the highest label.

2. Number of Edges (1417) and Vertices (1174): I see that we have 1417 edges representing connections between cities through E-roads. Having 1174 vertices suggests a well-connected

network, but the fact that there are more edges than vertices intrigues me. It could mean that some cities have multiple E-road connections.

3. Shortest Path (3 to 5): The shortest path from city 3 to city 5 is [2, 3, 4]. This makes me visualize the journey one would take, passing through cities 2, 3, and 4 in sequence to reach the destination.

4. Network Diameter (Undefined): The undefined network diameter gives me a sense that our E-road network isn't fully connected. There seem to be separate components or regions that are not linked. In a road network context, this could mean isolated areas or disconnected sets of cities.

5. Average Degree (2.41): The average degree of 2.41 tells me that, on average, each city is connected to about 2 or 3 other cities through E-roads. This gives me an idea of how cities are generally connected in our network.

6. Average Closeness (0.01):The low average closeness centrality indicates that, on average, cities are not very close to each other in terms of E-road connections. This might be due to the disconnected nature of the network.

7. Degree Distribution (Top and Bottom 5 Cities): Looking at the top 5 cities with the highest degree of distribution, I can infer that these are major hubs or well-connected cities. On the flip side, the bottom 5 cities with the lowest degree might represent smaller or less-connected locations.

8. Number of Local Bridges (0): The absence of local bridges tells me that our network seems robust. There are no edges where the removal of a single connection would disconnect the nodes. In a road network, this might indicate reliability in terms of alternative routes.

In summary, these metrics offer me valuable insights into the structure of our E-road network. The undefined network diameter and presence of disconnected components raise interesting questions about accessibility and connectivity within the road network. It tells me the roads could have better connectivity and be more organized

The Test

1. Arranging the Test:
   - I start by setting up the test scenario, creating a new instance of the `Graph` and initializing it with an undirected edge between nodes 1-2 and 2-3. This arrangement prepares the graph with a simple structure for testing.

2. Acting in the Test:
   - I then perform an action, calling the `example_function_for_testing()` on my graph instance. This function seems to return the length of the `outedges` vector, essentially counting the number of edges in the graph.

3. Asserting the Test:
   - In the assertion phase, I compare the result of the function call with the expected result. The comment suggests updating this with the expected value. Since I've added two undirected edges, I expect the result to be 3, representing the total number of edges in the graph.

In essence, this test is ensuring that the example function correctly counts the number of edges in the graph, providing a basic validation of the graph's functionality.

```
    Finished test [unoptimized + debuginfo] target(s) in 0.10s
     Running unittests src\lib.rs (C:\Users\Karrington Riley\Desktop\DS210_FinalProject\Opt2\Opt2\target\debug\deps\opt2-1cc4fffbd329d836.exe)

running 0 tests

test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s

     Running unittests src\main.rs (C:\Users\Karrington Riley\Desktop\DS210_FinalProject\Opt2\Opt2\target\debug\deps\opt2-7f7ea3064b207412.exe)

running 0 tests

test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s

     Running tests\integration_tests.rs (C:\Users\Karrington Riley\Desktop\DS210_FinalProject\Opt2\Opt2\target\debug\deps\integration_tests-2ef57c9c890da56
5.exe)

running 0 tests

test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s

   Doc-tests opt2

running 0 tests

test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s

PS C:\Users\Karrington Riley\Desktop\DS210_FinalProject\Opt2\Opt2\tests>
```

Output

1. Running Tests in `lib.rs` (src\lib.rs):
   - I see that it's running 0 tests in my `lib.rs` file, and the test result is "ok." This means that the tests in `lib.rs`, particularly the `test_example_function_for_testing`, were executed successfully without any issues.

2. Running Tests in `main.rs` (src\main.rs):
   - Similar to `lib.rs`, it mentions running 0 tests in my `main.rs` file, and the result is "ok." This indicates that the tests in my `main.rs` file, including any test functions I might have there, passed successfully.

3. Running Integration Tests (`integration_tests.rs`):
   - The output also includes running tests from an integration test file (`integration_tests.rs`). It reports running 0 tests, and the result is "ok." This means that the integration tests, if any, passed successfully.

4. Doc-tests (`opt2`):
   - The last part mentions running 0 doc-tests for my project (`opt2`), and the result is "ok." Since I don't have any doc-tests in this case, it's just confirming that there were no test failures in my documentation comments.

Overall, this output is reassuring. It tells me that all the tests, whether in the library, the main file, integration tests, or doc-tests, have passed successfully. The "ok" result means that everything is functioning as expected, which is a positive outcome for my code and test suite.

Conclusion

This project provided an insightful exploration into the dynamics of the international E-road network, revealing intriguing aspects of city connectivity through E-roads. The application of the Breadth-First Search (BFS) algorithm unearthed valuable insights, particularly shedding light on the limited connectivity observed within the road network.

One significant observation stems from the undefined network diameter, suggesting the presence of disconnected components in the E-road network. This implies that certain regions or sets of cities are not directly linked, hinting at potential challenges in accessibility and efficient travel across the network.

The average closeness centrality further emphasizes the notion that cities within the E-road network are not very closely connected. The low average closeness value indicates that, on average, cities are situated at a considerable distance from each other in terms of E-road connections. This lack of proximity raises questions about the overall efficiency and directness of travel between cities.

The average degree of approximately 2.41 provides additional context. This value signifies that, on average, each city is connected to only about two or three other cities through E-roads. The

relatively low average degree reinforces the idea of a sparsely connected network, where cities have limited direct connections.

The top 5 cities with the highest degree of distribution might represent major hubs or well-connected urban centers. Conversely, the bottom 5 cities with the lowest degree may indicate smaller or less-connected locations. This distribution reflects the disparities in connectivity across different cities within the E-road network.

In summary, the analysis underscores the notion that the cities in the international E-road network exhibit limited connectivity through E-roads. The challenges posed by disconnected components and the sparse average degree suggest opportunities for improvement in enhancing the overall cohesion and accessibility of the road network. This observation adds a layer of complexity to the understanding of the E-road network, inviting further exploration and potential enhancements to optimize city connectivity.