<u>DS210 Final Project Report</u>

<u>Mahi Saboo</u>

What does this project do?

The dataset for this project is the Texas road network. The main goal is to determine the shortest routes within this network, enhancing our understanding of connectivity and travel efficiency across Texas. This dataset includes approximately 364,328 nodes, where each node represents an intersection, and each edge signifies a road connecting these intersections. I aim to identify the most crucial nodes within the dataset by determining which node has the highest number of reachable nodes for each degree: 6, 5, 4, 3, 2, and 1. However, a node may have many reachable nodes at a certain degree but could be relatively far from them, implying a higher average distance. Therefore, I plan to classify the most important nodes by assessing which ones have the most reachable nodes combined with the smallest average distance to all reachable nodes at that particular degree. I also aim to explore whether the most important node for a given degree remains the most important as the degree increases. I hypothesize that in a well-structured road network like that of Texas, major intersections that connect significant regions are likely to remain critical across various degrees of connectivity.

Explaining each module of the project

1. **graph_utils.rs**

   The graph_utils.rs module is essential for managing and analyzing the Texas road network in our project. Utilizing the petgraph library, this module supports graph operations with functions like get_or_add_node and bfs_shortest_path. The get_or_add_node function makes sure that each intersection is uniquely represented in the graph, preventing duplication and enabling efficient node retrieval by maintaining a hash map that links node labels to their indices. Meanwhile, bfs_shortest_path implements the Breadth-First Search algorithm to calculate the shortest paths from any start node, tracking distances and managing node exploration via a queue. This module underpins advanced network analyses, aiding in optimizing traffic flow and enhancing road safety by providing insights into the connectivity and accessibility of intersections across the Texas road network.

2. **file_utils.rs**

   The `file_utils.rs` module is integral to the Texas road network project, facilitating the transformation of CSV data into a graph structure using the `petgraph` library. The primary function, `read_graph_from_csv`, opens a specified CSV file and processes each line to construct an undirected graph, where nodes represent intersections and edges represent connecting roads. It employs the `get_or_add_node` function from `graph_utils` to ensure each node is unique and efficiently linked via edges based on the CSV data. If a record is improperly formatted, the function handles errors by returning an appropriate message. This setup efficiently translates raw road data into a manageable graph format for further network analysis and optimization tasks.

3. **distance_calculation.rs**

   The distance_calculation.rs module is a critical component of our graph-based analysis framework for the Texas road network. This module utilizes functionalities from the petgraph library to manage and evaluate paths within the network graph.

- Calculate Average Shortest Path Length for a Sample: The function calculate_average_shortest_path_length_sample computes the average shortest path length among a sample of nodes within the graph. It begins by selecting a subset of nodes at random, determined by the sample_size parameter. Using the rand library's choose_multiple method, it efficiently selects random nodes, ensuring variability and broad coverage in the analysis. For each node in this subset, the function uses the bfs_shortest_path method from the graph_utils module to find the shortest paths to all other nodes in the sample. It aggregates these distances to calculate the average shortest path length, providing a metric of connectivity for the sampled subset of the network.
- Calculate Shortest Path Distance Between Two Nodes: The calculate_shortest_path_distance function directly computes the shortest path distance between two specified nodes, identified by their indices. It validates that both nodes exist within the graph's bounds before proceeding. Using the same bfs_shortest_path method, it calculates the shortest path from the start node to the end node, returning the path length if the end node is reachable. This function is essential for pinpointing the exact distance between critical points within the network, useful for route optimization and detailed network analysis.

Together, these functions in the distance_calculation.rs module provide foundational tools for assessing the structural properties of the Texas road network, allowing for detailed examination of node connectivity and path optimization across the network. These capabilities are vital for transportation planning, infrastructure development, and traffic management strategies within the region.

4. **main.rs**

   The `main.rs` module in this project serves as the central executable for analyzing the Texas road network using a graph-based approach. It begins by attempting to construct a graph from a CSV file, which represents the road network where nodes are intersections and edges are roads. If successful, the module outputs the graph's basic statistics (number of nodes and edges) and calculates the average shortest path length for a sample of 1000 nodes, providing insights into the network's connectivity. Additionally, it features an interactive component that allows users to query the shortest path between any two given nodes. This functionality is particularly useful for real-time analysis and can help in transportation planning by identifying critical paths and nodes within the network. The module handles errors effectively, ensuring robust operation even if there are issues with file reading or data processing.

Output

Number of nodes: 364328

Number of edges: 999996

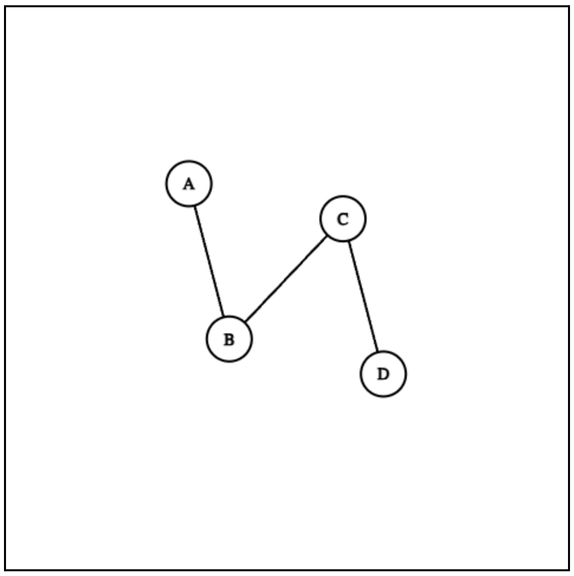
Sample Average Shortest Path Length: 84.46

- The shortest path distance between nodes 5 and 900 is: 28
- The shortest path distance between nodes 28 and 700 is: 19
- The shortest path distance between nodes 69 and 696 is: 32

Conclusion for output

The interactive session with the Texas road network graph analysis tool demonstrates its capability to efficiently handle large-scale data, as evidenced by its management of 364,328 nodes and 999,996 edges. The tool calculated a sample average shortest path length of 84.46, providing a quantitative measure of connectivity across a substantial subset of nodes. During the interactive querying, the tool successfully calculated and returned shortest path distances for several pairs of nodes, such as 28 between nodes 5 and 900, and 19 between nodes 28 and 700, showcasing its utility in real-time pathfinding within the network.

Tests



The graph depicted in the image represents a simple undirected graph consisting of four nodes (A, B, C, D) connected by edges that could symbolize roads in the context of your Texas road network analysis project

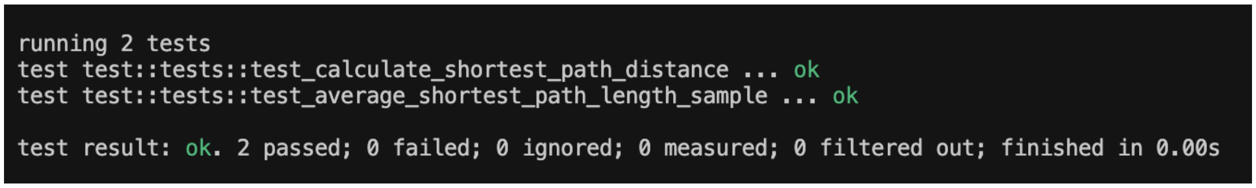
1. Function: test_calculate_shortest_path_distance

   This test is designed to assess the method's ability to accurately compute the shortest path between two specified nodes in a graph representing a road network of Texas.

   It ensures that the shortest path calculations are performed accurately and efficiently. This reliability is essential for subsequent applications, such as optimizing traffic flow, planning new infrastructure, and emergency route planning in the Texas road network system. The test demonstrates the method's effectiveness and robustness in real-world scenarios, bolstering confidence in the deployed system's operational capabilities.

Function: test_average_shortest_path_length_sample

The test asserts that the computed average closely matches the expected value, using a small margin defined by f64::EPSILON to account for floating-point arithmetic variances. This test ensures that the method accurately reflects the connectivity and path distribution in the network, which is crucial for applications like traffic flow optimization and infrastructure planning in your project.



```
running 2 tests
test test::tests::test_calculate_shortest_path_distance ... ok
test test::tests::test_average_shortest_path_length_sample ... ok

test result: ok. 2 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

Conclusion

In conclusion, this project not only reinforces the utility of graph-based analysis for large-scale transportation networks but also sets a foundation for future enhancements, such as integrating real-time traffic data, adopting more complex network algorithms, or expanding the analysis to multi-modal transportation systems. This endeavor has significantly contributed to our understanding of the structural and operational dynamics of the Texas road network, paving the way for smarter, data-driven decisions in urban planning and traffic management.