Dataset used: http://konect.cc/networks/subelj_euroroad/

DS 210 Project Report

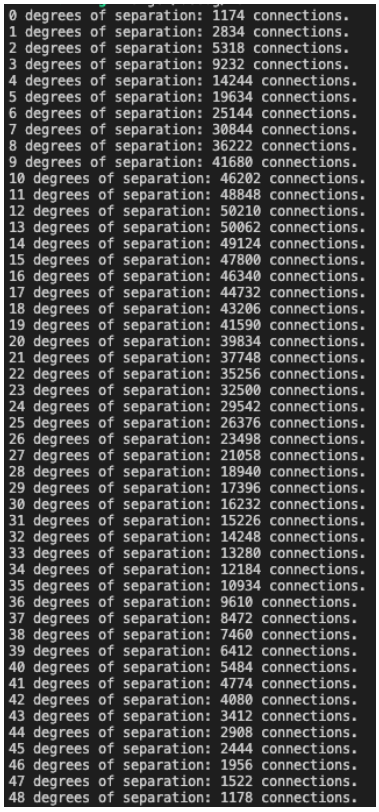
Paul Lee

The dataset I am using for the project is the international E-road network, a road network located mostly in Europe. I decided to use this dataset because road networks are fit to map and find degrees of separation between the nodes. Each node represents a designated code for each city and roads in Europe and the edges represent the connection between two nodes denoting that they are connected by an E-road. The objective of this project is to examine the distances and degrees of connection between nodes, specifically focusing on determining the shortest number of E-roads required to travel from one city to another in Europe.

My final project file contains two modules called ‘analysis’ and ‘graph_helpers’. In ‘analysis’, I created a RoadNetwork instance with a predefined road graph. This graph forms the fundamental structure of the analysis. The first algorithm I have used is Breadth-First Search (BFS) algorithm. It finds the shortest path from an origin city to a destination city. Next, I needed to understand the connectivity between the different parts of the road network. The ‘evaluate degrees’ function calculates the degrees of separation between all pairs of nodes (cities) in the graph through iterating over each possible pair of cities, and uses the BFS algorithm to determine the level of separation between them across the entire road network. In the ‘graph_helpers’ module, I utilized the petgraph crate to find the minimum path, in terms of the number of edges, between two cities in a graph. In terms of calculating the shortest distance, I used Dijkstra's algorithm. It finds the shortest paths from the starting city to all other nodes in the graph.

The main.rs opens up the dataset I have imported in src and constructs an undirected graph of the data. A HashMap is used to efficiently track and reference city nodes in the graph.

As the file is read line by line, cities are added to the graph, and connections are established between them. Once the graph is constructed, the analysis::RoadNetwork module is used to analyze the network. This analysis includes calculating degrees of separation between cities and summarizing the connectivity of the network. I have added some statistical information about the network as well, such as the total number of connections, the average degree of separation between cities, and the standard deviation of these degrees of separation.



Here are the outputs of the main function. As you can see, 25,144 connections of cities can reach each other within 6 different E-roads' combinations. This is 2.3246% of the total network between the cities. In order to reach the maximum connections you have to go through 12 degrees of separation, which are 50,210 connections. The average degree of separation is 18.35 with standard deviation of 9.43.

Sources:

https://gist.github.com/vTurbine/16fbb99225ad4c0ac80b24855dd61a7c

https://www.sotr.blog/articles/breadth-first-search

https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/

https://www.sheshbabu.com/posts/rust-module-system/

https://crates.io/crates/petgraph

https://users.rust-lang.org/t/using-petgraph-graphmap-to-create-sub-graphs/79041

https://users.rust-lang.org/t/advices-on-architecture-and-testing-in-rust/70630