**Shortest Path Finder using Parrarel Processing**

**Introduction**

Welcome to the Parallel Shortest Path Finder! This program is your go-to solution for finding the shortest routes between pairs of nodes in a graph. Using Dijkstra's algorithm, it efficiently navigates through graph connections represented by an adjacency list and a hash table-based distance matrix.

What makes this tool stand out is its use of MPI and OpenMP. MPI lets multiple computers work together, while OpenMP enables parallel processing within each computer. This means faster results, even for large graphs.

Whether you're mapping out transportation routes or analyzing complex networks, the Parallel Shortest Path Finder simplifies the task, making it fast and efficient. Let's dive in and discover the shortest paths in your graphs!

**Dataset**

About the dataset we are using three different datasets: Email-Euron, Email-EuAll and the DoctorWho dataset. All the two datasets are stored using hashmaps due to the datasets efficiency. After this the rest of process takes place

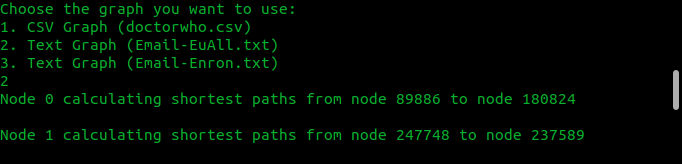
**How to run the code**

We have created a bash script file which upon running will automatically run the code three times on all three files. In the command line type:

**./driver.sh**

**Interface**

Upon running the driver code the terminal will tell you what to enter and what to enter once the GUI displays. The display will be as following :



This will display one by one for all 3 files.

**Pre Processing**

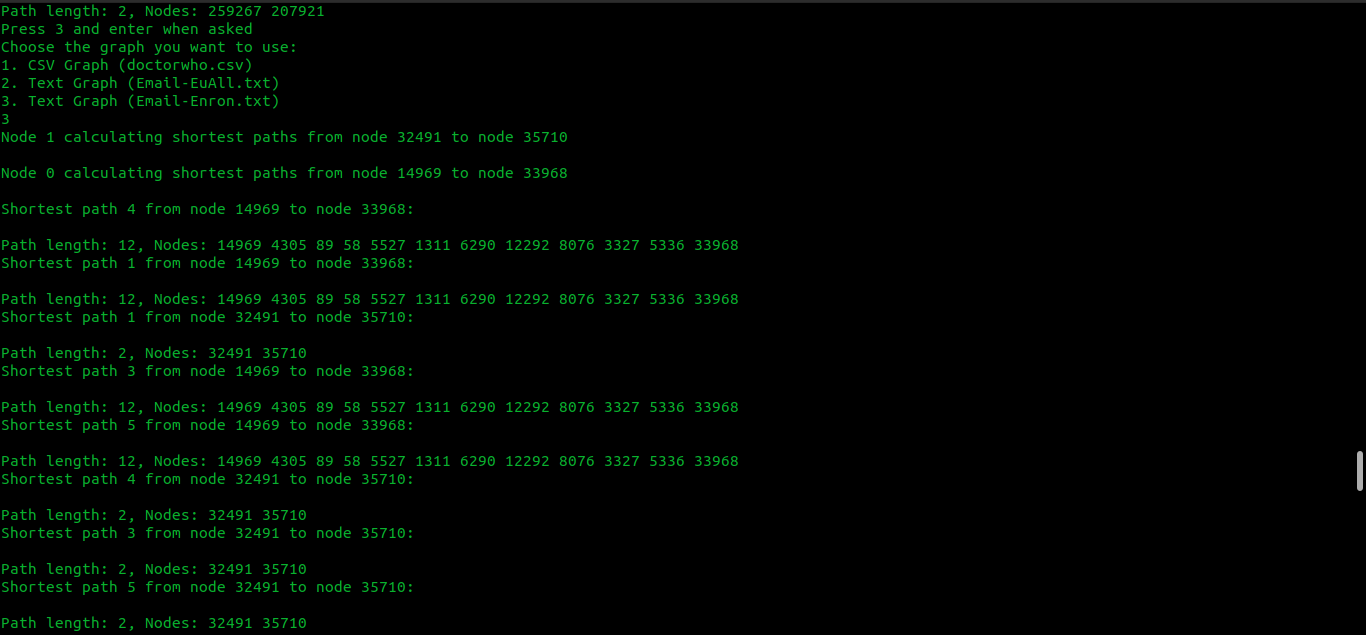
There wasn’t much need for pre-processing except for the skipping of the first 4 lines in the text files that contained the meta data of the graph.

**Storage**

We did face issues while storing the graphs as the graph had a huge number of nodes. At first we tried an adjacency matrix through arrays but it was not optimal as the due to the nature of the matrix as it was sparse so it took alot of memory to be stored. We switched to using hash maps as it was more efficient in storing the data which was much more efficient in storing the data.

**K shortest path**

We chose to use the dijkstra algorithm to find the shortest path as djkstra's algorithm stands out for its effectiveness in solving the shortest path problem in graphs. Its efficiency ensures that it can handle graphs of varying sizes with reasonable time complexity, making it suitable for both small and large-scale applications. Moreover, its accuracy guarantees the discovery of the shortest path between nodes, providing reliable results. Additionally, Dijkstra's algorithm is versatile and can accommodate different types of graphs, including directed and undirected graphs, as well as weighted and unweighted graphs.



**Optimization and Execution times**

For optimization we did the following:

* Used hashmaps for storage of the graph
* Used Dijkstra due to its strong ability to find shorted path
* Used bash script for execution of all three files with one command
* Used GUI for a user friendly experience

The Execution time for the DoctorWho csv and Email-EuAll took no time at all while Email-Enron takes some time to work completely

**Group Members:**

Muhammad Maemoon Farooq 21i1680

Abubakar 21i1691

Waleed Saeed 21i1698