Results

Mohit Ahuja

Ajith Senthil

The intention for this project was to utilize the OpenFlights data set to create a map-based visualization of all the airports and flight routes in 2012. Along with this, we intended to find the shortest path from one airport to another and draw the route onto our map. Finally, we planned to complete the breadth-first traversal requirement by traversing the graph and writing all the airport nodes to file. Our initial goal was to project only the airports and routes within the United States but we successfully completed our stretch goal of all the flights and routes globally using the data set of global flight paths and airports.

We had to make minor changes to our goals as our group turned out to be just two people instead of the expected four. Instead of using Djisktra's algorithm to get the shortest route, we opted to now use a breadth-first search to find the shortest number of flights between two airports and still decided to do the map projection as our complex algorithm.

While developing our program, we made a graph structure which utilized the unique airport ID's within the OpenFlights database as the keys to our vertex map (which is implemented by our hash map). The routes were stored in and edge list which is also stored in a hash map.

A discovery we made working through the project when increasing the efficiency of our breadth-first search algorithm was to use arrays to have a faster run time when searching for visited nodes. We initially used a vector that stored all visited airports and searched through it on each iteration to check if an airport had been visited. We realized that using a boolean array and indexing using the integer airport ID would significantly improve our visited lookup runtime by making it O(1).

The project utilized mercator projections to convert from spherical latitude and longitude coordinates to cartesian x and y coordinates for our map. This proved to be very effective with almost all of our airports being precisely graphed onto the map. There was one exception, the South Pole Airport, which is located at the south pole and does not work accurately with the calculations for a mercator projection. The airports were made to increase in size based on the number of incoming flights, which allowed for a good representation of how busy and important an airport was. The flight routes were drawn using a simple line algorithm based on the slope of

the line. This proved to work for the most part, but was very ineffective for lines that were close to vertical. Lines that were almost vertical did not draw completely and had many skips, which could be improved in the future with a better line drawing algorithm. The routes were also color coded based on their edge weights, the distance. The shortest routes were colored green and became increasingly red the longer the route is. This allowed for a good visual representation for the length of flights.

The map projection proved to provide a lot of insight on real-world flight data, even if it is from 2012. Because we were able to visually see the throughput of airports based on their size, we were able to observe that airports near the coast tended to be larger and more busy, likely due to more intercontinental flights going through them. Meanwhile, airports in the middle of continents tended to be smaller due to having mainly domestic flights.

The breadth first search for the shortest path turned out well as well, with a full user interface to find the starting airport and destination airport. It quickly finds the shortest number of flights to a destination, outputting it both to the console and drawing it on the map for the user to see. The breadth first traversal of the graph was similarly successful, covering all the airports in the graph and writing them to a file.

The project as a whole was useful in learning not only how to work on group projects but also how to use data sets and graph algorithms to return useful information to the end-user.

Final Map