CS 225 Final Project Results

Alyssa Licudine (alyssal3), Albert van Valkenburg (albertv2), Gavin DeBrun (gdebrun2)

Our group implemented a graph to analyze the OpenFlights/Airline Route Mapper Route Database [1]. We implemented a graph using an adjacency matrix that handles directed, weighted graphs. The weights of each edge are the geographical distance between the corresponding airports. We implemented methods that compute geographical distance, perform breadth-first search traversal, compute shortest paths using Dijkstra's algorithm, and a landmark path algorithm that computes the shortest path between two locations that pass through a landmark location. We implemented a main executable that demonstrates the implemented algorithms and a test executable that tests the flight graph class. The user can customize the origin, landmark, and destination airport used by the main executable by modifying `options.txt`.

Graph Class

Our Flight Graph class utilizes several member variables to store the airport and route data. The class reads an airport input file and a route input file to generate the graph. We create two maps: one from the airport codes to their latitude-longitude coordinates and another from the airport code to its unique index. Further, we create a list of edges in the graph that represent unique routes. The adjacency matrix is implemented using a 2-dimensional vector of distances, where the first coordinate corresponds to the origin airport of a route and the second coordinate corresponds to the destination airport of a route.

Implemented Methods

Geographical Distance:

We implemented a method that computes the geographic distances between two latitude-longitude coordinates. This method is used to compute the weights of edges in the graph. The function computes the great-circle distance between the coordinates using the haversine formula [2]. This formula is numerically well-conditioned but assumes that the Earth is a perfect sphere, so the final computed distance is not exactly correct. However, the formula is accurate up to a relative error of 1%.

Incident and Adjacent Airports:

We implemented a method that determines whether there exists a route from an origin airport to a destination airport. We also implemented a method that returns all the airports incident to an origin airport.

Breadth First Search:
Our breadth-first search algorithm takes a root airport code and traverses the connected component containing the root. Because our dataset contains disconnected components, our breadth-first search function returns a vector of traversals, where each traversal corresponds to a distinct component.

Dijkstra's Shortest Path:
Our implementation of Dijkstra's algorithm uses the edge weights (the geographical distances between airports) and finds the shortest paths from a given source node (an airport) in the graph to a destination node.

Shortest Landmark Path:
This algorithm finds the shortest path between the origin and destination airports that passes through a desired landmark airport. It operates much like the original shortest path algorithm but considers two paths to minimize: the path between the source node and the landmark node and the path between the landmark node and the destination node.

Discoveries and Issues

We discovered that some airport names contained extra commas, so we used an offset to deal with this issue when processing the input files.

Some airports did not have coordinates, so we decided to set the location of those airports to the South Pole so they still have real but out-of-the-way geographical locations; this would ensure that these airports will not impede with the shortest path algorithms.

Because the Earth is not a perfect sphere, the distances calculated in the coordinates-to-distance method are not exactly accurate but should be fair estimates of the actual distances.

References

[1] *OpenFlights: Airport and airline data.* (2019). Retrieved November 16, 2020. OpenFlights.org. https://openflights.org/data.html

[2] *Great-circle distance.* (2020). Retrieved November 16, 2020, from https://en.wikipedia.org/wiki/Great-circle_distance