Alex Li

Praval Telagi

Kevin Zhao

Development Log

Week 1: (11/18/2020 - 11/25/2020)

11/22/2020: First, we re-clarified our goals based on our mentor's feedback. We made the project more applicable to the real world. Next, we created some template files that we will be using in our final project. Afterwards, we implemented graphs and edges similarly to lab_ml. We imported the necessary data files needed for our final project. We also created a new header file that included the struct (vertex) of an Airport (using data files.)

For the upcoming week, we hope to have a working makefile and we hope to continue modifying our graph and edge files to satisfy our needs.

Week 2: (11/25/2020 - 12/2/2020)

11/27/2020: First, we created our makefile so we can test our code and run our code. Next, we created a class called vertex_parser to create the Airport vertices in our graph. In this file, we parsed the data into a vector of Airport structs. We ran into a few issues with converting strings to ints/doubles, so we made sure to handle cases where a string would not properly convert to a number.

For the upcoming meeting, we plan to add these vertices to the graph. We also plan to parse the edges (routes).

11/28/2020: First, we created an edge_parser class to parse the routes.dat file and to grab the routes from airports by using the airport id. Next, we created an unordered map with key airport id and value Airport pointer. We also created a skeleton function to calculate the euclidean distance using the latitude and longitude between two airports. We also handled edge cases where the airport ID in the routes.dat were unknown.

For the upcoming meeting, we hope to implement the distance formula (taking into account the curvature of the earth.) We also plan to implement BFS traversal.

11/29/2020: First, we successfully implemented a calculating distance function. We use this to add weights to the edges on our graph corresponding to the distance between two airports. Next, we created a new class called BFS that implements BFS traversal. We will use this to check if a given airport ID is on the map. This was very difficult; it took lots of conceptual understanding on how to implement BFS search in our specific project. We spent a few hours thinking of different ways to modify the classes to implement the BFS search, and we finally figured out a method that works! We also deliberated the possible applications of BFS for finding air travel paths.

For the upcoming meeting, we would really like to meet with our mentor to check in our progress. We would also like to gain a better understanding of how to approach the implementation of Dijkstra's algorithm.