Project Goals Summary: Aparna Ayyah (aayyah2), Chaahat Gurnani (cgurna2), Joshmita Chintala (jchint2), and Aditi Adya (aditiaa2)

Dataset Summary

The dataset has a set of a variety of airport codes and flight routes with longitudes and latitudes dated from 2014. Each entry contains a specific amount of information that corresponds with each airport. For example, some of the characteristics of the entry include airport ID, name, city, country, IATA and ICAO codes, latitude, longitude, altitude, timezone, daylight savings time, time zones, type and source. When it comes to looking at the datasets, we will more particularly be looking at the airport codes and names with the longitudes and latitudes. We will be looking at the paths that each airport uses and generating directed graphs to represent the datasets.

Problem Statement

The problem we're trying to solve with this final project is figuring out the shortest path between two airports in the dataset. We define preferred route as the route with the shortest time in flight (the lowest in cost) between point A and point B. Point A and B will need to be valid airports with valid latitude and longitude values.

Our Base Algorithm

- We are planning on writing a shortest path algorithm that utilizes BFS and uses the distance between different airports as the weight. Our inputs are the starting airport and the destination airport in the form of their airport codes in IATA form (ex: ORD).
- Cost is defined by geographic distance between airports.
- We will be using the latitude and longitude coordinates provided in the dataset to calculate our distance measurement. Since earth is a sphere, we will be calculating the distance between different airports using the orthodromic distance formula:
    - D = 3963.0*arccos[(sin(lat1) * sin(lat2)) + cos(lat1) * cos(lat2) * cos(long2 – long1)]
- Once we calculate the distance between different airports, we will be storing that data in a graph, for our BFS to traverse. This graph will be superimposed on a world map so that it can be written to an output PNG.
- The path with the smallest total distance (time in air) from Point A to Point B will be our preferred route. If multiple preferred routes exist, we will use the path that comes first alphabetically.
- Once the shortest path is calculated, we will convert everything back to airport codes
- Our output will be a string containing the concatenated airport codes (in order) of our preferred route. It will use a pipe (|) as a delimiter.
    - Ex: For a trip from ORD to LAX, a direct flight is possible (which would be the lowest cost), so the output string would be "ORD|LAX"
- If traveling from Point A to Point B is not possible, then we will output "Not Possible".

Next Steps

- We foresee some airports and routes not being possible with just 1 flight, so in order to account for that, we want to allow for "jumps" (aka layovers) to find the shortest possible path from our dataset.
- If a user wants to travel via a specific airport, we want to rank paths that travel through that node as higher priority (despite the cost). This would mean that in such a case, we would select the path that travels through the given nodes(s) and has the lowest cost as the ideal route.
- In the case of a new airport/any dynamic changes to the dataset we will need to alter our graph data structure.
- We should display airport efficiency as that impacts the desirability of a route
    - We could color-code airports in our output graph visual by the ratio of airflow/airport capacity. Airports with a 1:1 ratio (same level of flow as capacity) would have little to know waits/delays whereas a ratio like 3:1 would indicate high delay risk (queues).
- By incorporating this metric of airflow/airport capacity into the cost calculation for best path selection, we could add an additional "pragmatic" best route that takes into account ease of route, not just geographic closeness.
- We could also analyze the causes of delays at various airports and include that in the graph visualization.