CS 225 Project Goals

Dataset: OpenFlights

Imported as graphs
Airports are the vertices
Flight routes are the edges

Traversals

BFS

Covered Algorithms

Shortest Path: Dijkstra's algorithm

Uncovered Algorithms

Landmark algorithm

Goal

We will use the adjacency list implementation for the graphs. Then, we will utilize the graph functions to complete the implementation of the selected algorithms. The primary goal of our project is to find the shortest route between two destinations. If there already exists a route between two airports, then the shortest path is simply the line between the two destinations. However, if there does not exist a direct path between the two destinations, then our algorithm will find the shortest distance accounting for layover flight destinations.

We will first convert the .dat files from OpenFlights and convert it to .txt files. This way, we will be able to read the data in our code and add the datasets into our data structures. After we store all the data into our code, we will create a graph with a node representing our home airport - Chicago O'Hare International Airport. Then, we will map our root node to its direct destinations. For instance, if there is a direct flight from O'Hare International Airport (ORD) to Los Angeles International Airport (LAX), then we will create a node LAX and connect it directly to the root node (home airport ORD). We will do this with every airport and eventually arrive with a completed tree.

To traverse the tree, we decided that the most complete way of accessing each node in the tree is by using Breadth-First-Search (BFS). Then, we will find the shortest path between two nodes by using Dijkstra's algorithm. Dijkstra's algorithm is optimal for finding the shortest path as it is uniform cost search. However, in the case where there does not exist a direct path between two

nodes, we will utilize the Landmark Path algorithm. The Landmark Path algorithm will allow us to find the shortest path between two non-direct nodes by traversing through additional nodes that result in the shortest path.

Ultimately, our project will be able to create a tree consisting of nodes as airports and edges as routes between airports, then find the shortest distance between two destinations using Dijkstra's algorithm and the Landmark Path algorithm.