**(S1-21\_DSECLZG519) (Data structures and Algorithms Design)**

**Academic Year 2020-2021**

**Assignment 1 – PS12 - [Airport Connections] - Group 108**

**Overview:**

This design document covers the solution to the Route Finding Problem as Graphs Edge List Implementation based on a given set of routes as Source and Destination with constraints & finding the minimum number of flights (one-way flights) that need to be added so that the passenger can reach any destination city from the starting city using either direct or indirect flights with an algo design of Minimal Time Complexity.

**Goal:**

The goal is to create a python program using Python 3.7 by reading input file(.txt) and outputting the results into a file(.txt) & to perform an analysis for the operations, the running time in terms of input size: n.

**Algorithm:**

**Step 1:**

Read the Airports, Routes & Starting Airport from the **inputPS12.txt**

**Step 2:**

Created a Graph from the Airports and Routes list read from above.

**Step 3:**

Traverse all the routes using a hash table.

**Step 4:**

Using Airplane Names as keys to check if the given airport is reachable or not by visiting all the connections of a given Airport i.e visiting the connection and then visiting its connection’s connections.

**Step 5:**

Traverse every airport under the passed airport.

**Step 6:**

For the unreachable airports, calculate the score of how many unreachable airports

**Step 7:**

Add potential routes that make graph fully connected.

**Step 8:**

Output the results of minimum flights to be added into a file named outputPS12.txt

**Data Structure:**

Hash table is used which is fulfilling the requirements.

Hash table is one of the most important data structures that uses a special function known as a hash function that maps a given value with a key to access the elements faster.

**Run Time Analysis:**

**Function:**

**compute\_connects**

1. Time Complexity = O(V log(E) + (V + E))
2. Space Complexity = O(1)

**compute\_graph**

1. Time Complexity = O(V + E)
2. Space Complexity = O(V + E)

**compute\_unreachable**

1. Time Complexity = O(V + E)
2. Space Complexity = O(V)

**compute\_depth\_first\_search**

1. Time Complexity = O(V + E)
2. Space Complexity = O(V)

**unreachable\_score**

1. Time Complexity = O(V \* (V + E))
2. Space Complexity = O(V)

Where V is the number of Vertices/Nodes/Airports,

E is the number of edges/ routes

**So, overall time complexity**

**Alternate Modelling:**

An alternate approach to solve Graph Edge Implementation problem is to use Breadth First Seach Computation. Similar to the DFS approach, the BFS approach will be applied on the unvisited nodes. However, to solve it will be used better for finding the nearest ones while DFS is used to discover all the possible routes.