Optimal Air Route

Pranesh Senthilvel

Abstract

This document outlines my approach to solving the "Optimal Air Route" competitive programming problem, which requires finding the most fuel-efficient path for an airplane navigating through a network of airports.

1 Introduction

In an age where air travel is increasingly common, manual route planning has become inefficient. This problem aims to optimize air travel by calculating the most fuel-efficient route using the given network of airports, distances, and fuel constraints. Airports are positioned using distance and angle (in polar coordinates) from the origin, making the input more realistic.

2 Problem Understanding

Given:

- Origin and destination airport codes.
- Initial fuel available.
- A list of airports with their distance and angle from the origin.

Fuel consumption is 1 unit per 1 unit of distance. The plane can refuel to full capacity (5000 units) at any airport.

3 Approach

Part 1: Cartesian Conversion

Using the given distance and angle, convert all airports from polar to Cartesian coordinates using:

$$x = D \cdot \cos(\theta \text{ in radians})$$
$$y = D \cdot \sin(\theta \text{ in radians})$$

Part 2: Graph Construction

Treat each airport as a node in a graph. The weight of the edge between two nodes is the Cartesian distance between them. This allows us to use graph algorithms to solve the problem.

Part 3: Choosing an Algorithm

We need a shortest-path algorithm that handles edge weights well. Dijkstra's algorithm is a natural choice due to its efficiency and suitability for dense graphs (since the number of airports is small). A* was avoided because its performance degrades in dense graphs and we do not have an effective heuristic.

Part 4: Implementation

After converting coordinates and constructing the weighted graph, apply Dijkstra's algorithm from the origin node to find the shortest path to the destination. At each node, if fuel is insufficient for the next leg, simulate a refuel.

4 Example

Input

```
DEL NYC 2000
3
BLR 1800 0
DXB 1000 90
NYC 3200 0
```

Output

```
DEL->BLR->NYC
```

Explanation: From DEL to BLR uses 1800 fuel, leaving 200. Refuel to 5000 at BLR. From BLR to NYC uses 1400 fuel. Total: 3200.

5 Conclusion

This problem elegantly combines geometry, graph theory, and classic algorithms. Efficient implementation ensures quick computation even under constraints. The design allows extension to larger datasets if needed.