Algorithms

Tanvi Pooranmal Meena

May 23, 2024

1 Approach

My approach to this problem is to leverage network graphs and a Breadth-First Search algorithm to find alternative flight paths for passengers affected by cancellations.

Here's how I'd tackle it:

1. Data Wrangling: Import passenger information, flight details, and canceled flight data.
2. Building a Flight Network: Create a graph with airports as nodes and flights as directed edges. Store flight information (ID, origin/destination, times, capacity) on each edge.
3. Identifying Stranded Passengers: Find passengers affected by canceled flights.
4. Rebooking Mission:
   (a) Iterate through stranded passengers.
   (b) For each passenger, use Breadth-First Search to explore the flight network from their original departure airport.
   (c) Prioritize paths that:
      - Minimize layovers.
      - Arrive at or before the original arrival time.
      - Respect layover time requirements.

   (d) Simulate seat allocation to avoid overbooking.
5. Success Evaluation:
   (a) Record new flight sequences for successfully rebooked passengers.
   (b) Track metrics:
      - Total affected passengers.
      - Number of rebooked passengers.
      - Average layovers for rebooked passengers.

    - Average arrival time deviation.

6. Reporting and Visualization (Optional):
    (a) Compile results: affected passengers, rebooked passengers, reallocation effectiveness.
    (b) (Optional) Create a visual representation of the flight network.

2 Mathematical Models

The code utilizes a graph-based approach to solve the flight rebooking problem. While not strictly mathematical modeling in the traditional sense (with complex equations and analytical solutions), it leverages mathematical concepts like graphs and Breadth-First Search (BFS) to achieve an optimal outcome.

Here's how the code incorporates mathematical ideas:

1. Graphs: The flight network is represented as a directed graph (using the networkx library). Airports are modeled as nodes, and flights are modeled as directed edges connecting these nodes. Each edge holds flight information like flight ID, departure/arrival airports, departure/arrival times, and capacity (relevant to simulating seat allocation).
   This graph structure allows for efficient exploration of connections between airports using BFS.

2. Breadth-First Search (BFS): This algorithm is employed to explore the flight network for potential rebooking paths for each affected passenger. BFS systematically visits nodes level by level, starting from the passenger's original departure airport. This ensures that paths with the fewest layovers (shortest paths in the graph) are prioritized.
   The algorithm considers layover time requirements and simulates seat allocation to avoid overbooking issues.

3 Roadblocks

As I embarked on the task of crafting a reallocation algorithm for passengers stranded by flight cancellations, I was determined to find a smooth and efficient solution.

I meticulously reviewed the code, line by line, ensuring it adhered to the principles of network graphs, Breadth-First Search, and passenger priorities.

Yet, to my dismay, the results yielded zero successful reallocations. This unexpected outcome forced me to delve deeper, confronting a hidden roadblock in my algorithmic approach.

There could be several reasons behind this outcome. Here are some possibilities I explored:

1. Data Inconsistencies: I double-checked the flight information, passenger details, and canceled flight listings for errors or typos. Everything seems to be in order.
2. Algorithmic Scrutiny: I meticulously reviewed the code for potential bugs, especially in the BFS implementation, path evaluation, and seat allocation simulation. At this point, everything appears to be functioning correctly.
3. Graph Construction: I confirmed that the graph representing the flight network is built correctly. Airports are connected as nodes, and flights are represented as edges with accurate details.
4. Algorithmic Bugs: I uncommented the print statements within the code to obtain valuable insights. The Breadth-First Search appears to be exploring the network efficiently, but it seems there just aren't suitable paths available for all passengers given the current constraints and flight network data.

In conclusion, despite a comprehensive examination, the zero reallocations remain a puzzle. The data appears accurate and complete, the algorithm seems to be functioning correctly and the graph representing the flight network is built accurately.

However, I gained valuable insights into the complexities of algorithmic problem-solving. It highlighted the importance of data quality, and the ever-present possibility of bugs.