Algorithmic Trip Chartering

Program summarization

McIntire, Sairam, Shaikh

Background - 1/2

Current public transit is overcrowded. Eplane intends to offer a novel solution by introducing aerial public transportation. By circumventing conventional traffic, air taxis provide an expedited form of transportation.

Background - 2/2

Problem Statement

- How do we assign air taxis to the customers?
  - Air taxis are charged at vetihubs and vertiports
  - Rides are chartered to be picked up and dropped at vertiports and vertistops

Complexities

- On demand chartering
- Battery health
- No fly zoning
- CAA regulations

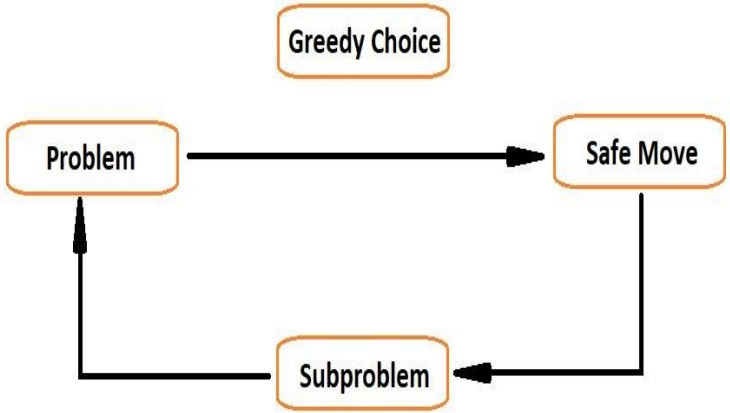
Literature Review

- OPTIMIZATION OF TAXI CABS ASSIGNMENT USING A GEOGRAPHICAL LOCATION-BASED SYSTEM IN DISTINCT OFFER AND DEMAND SCENARIOS
  - Introduces hybrid simulation goal programming (HSGP) for real-time air taxi dispatching in cities.
  - Focuses on New York City's centralized network.
  - Dynamically allocates air taxis after each customer drop-off.
  - Decides whether taxis should idle or pick up new customers and where they should go.
  - Concludes that 84 air taxis are needed for efficiency in New York City.
- Real-time dispatching of air taxis in metropolitan cities using a hybrid simulation goal programming algorithm
  - Evaluates methods for assigning taxis to customers in location-based systems.
  - Focuses on optimizing time and distance.
  - Combines GPS and pythagorean distance calculations.
  - Uses greedy and optimization-based vehicle assignment strategies.
  - Finds that using a shortest path algorithm with an optimization model significantly improves service efficiency.

Incorporated solution - 1/3

‘Greedy’ Approach

Named for its minimization wait time. The greedy approach is a relatively fundamental algorithm that assigns the closest taxi capable of completing the charter. A taxi is considered available if it is currently idle and has enough battery to complete the trip.



Greedy Algorithms Strategy

Incorporated solution - 2/3

Algorithm Overview

- Greedy Approach for Trip Assignment
- Calculated travel time and battery requirement for each trip
- Adjusted the start times based on the availability of the Air Taxi
- Updated the States of the Air Taxi after the trips

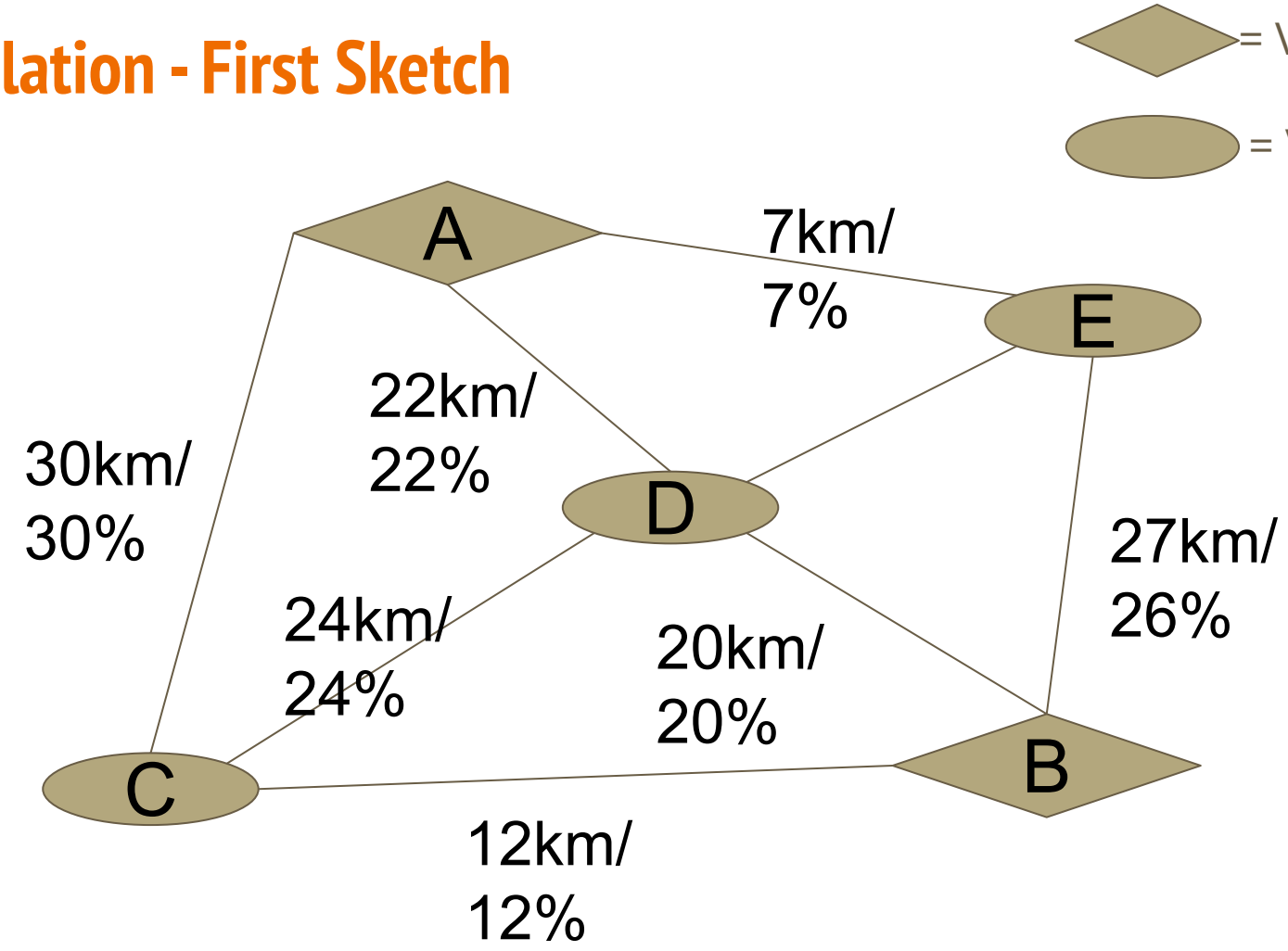
Key Functions

- calculate_travel_time - Assume one unit of distance will take one unit of time
- calculate _required_battery - Add Distance to Start and Distance of the trip, and assume that 1 unit of distance will take 1 unit of charge
- find_nearest_vertiport - Find nearest port to send for charging if battery falls below threshold
- simulate_air_taxi_operations - Simulate the operations based on the amount of time period specified

Incorporated solution - 3/3

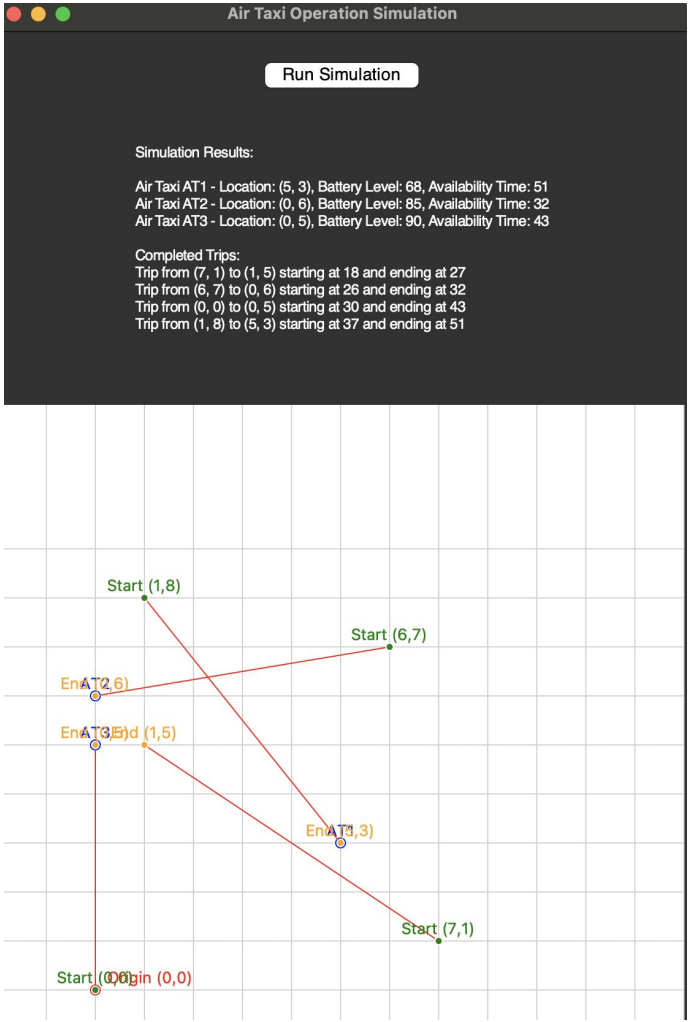
Advantages	Disadvantages
• Easy to implement • Speed and Efficiency • Good initial solution that can be improved upon • Can handle large data sets more efficiently • Provides good approximate estimates	• Future customers can get affected, since it ignores how current choices could affect the future • Suboptimal Routing for the air taxi, since it selects the nearest option not considering the overall route

Simulation - First Sketch



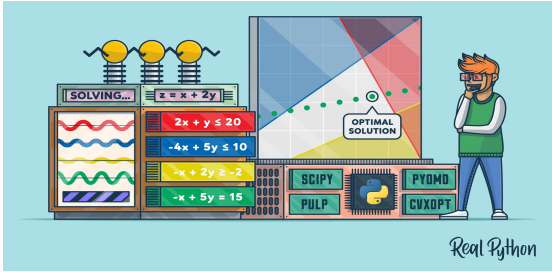
Simulation - Current Look

- Simulation runs for selected period of time
- Demand is generated using the poisson distribution
- Vertiports are created at random locations in a 2d grids
- Trips are assigned to air taxis
- Assumptions:
    - Each kilometer = 1% battery loss
    - Trip path was simplified and all paths were assumed straight



Integer Programming

- Choose a variable to optimize
- Aims to find most optimal solution to the problem
- Easier to incorporate constraints such as trip start and end timings
- Can optimize multiple variables such as minimizing travel time and maximizing battery health
- Is computationally intensive
- Might overfit trying to find the most exact solution



Trip Assignment Updating

- Sorted the trips by start time
- Found the best Air Taxi by arrival time and battery level
- Adjusted the start time if the air taxi will be late

Updating the Air Taxis

- Update location battery level and availability time after each trip
- Handle Charging if battery falls below threshold

Future Developments

- Implement and compare different approaches such as integer programming and heuristic algorithms
- Add actual vertiport data, in a 3d map and plan out flight routes depending on altitudes and no fly zones
- Add a 3d simulation of the flight path and optimize battery charging by deciding whether to send it to fast, slow or rapid charging
- Implement mechanisms to adapt to on demand change
- Update simulation to display curved paths for trip
- Implement algorithm into application with group 1’s vertistop project