Flight Route Finder

Group member:

Zijing Ye

Yueqi Yan

Research Background

- **Shallow descriptive analyses**

Most Kaggle projects stop at simple EDA (line charts, bar graphs) or basic network maps—few tackle true route optimization.

- **Fragmented flight search tools**

Current platforms force travelers to juggle multiple services and fail to balance cost–time–convenience trade-offs in one unified optimizer.

- **Our passion for aviation data**

We're eager to apply Data Science and NetworkX to make flight planning smarter and more efficient.

Research Questions

How can we recommend the best route based on different user needs?

Given user input (origin, destination, current time):

- Find the **cheapest** route
- Find the **fastest** route
- Find the route with the **fewest transfers**

NetworkX-powered command-line interface (CLI) flight planner

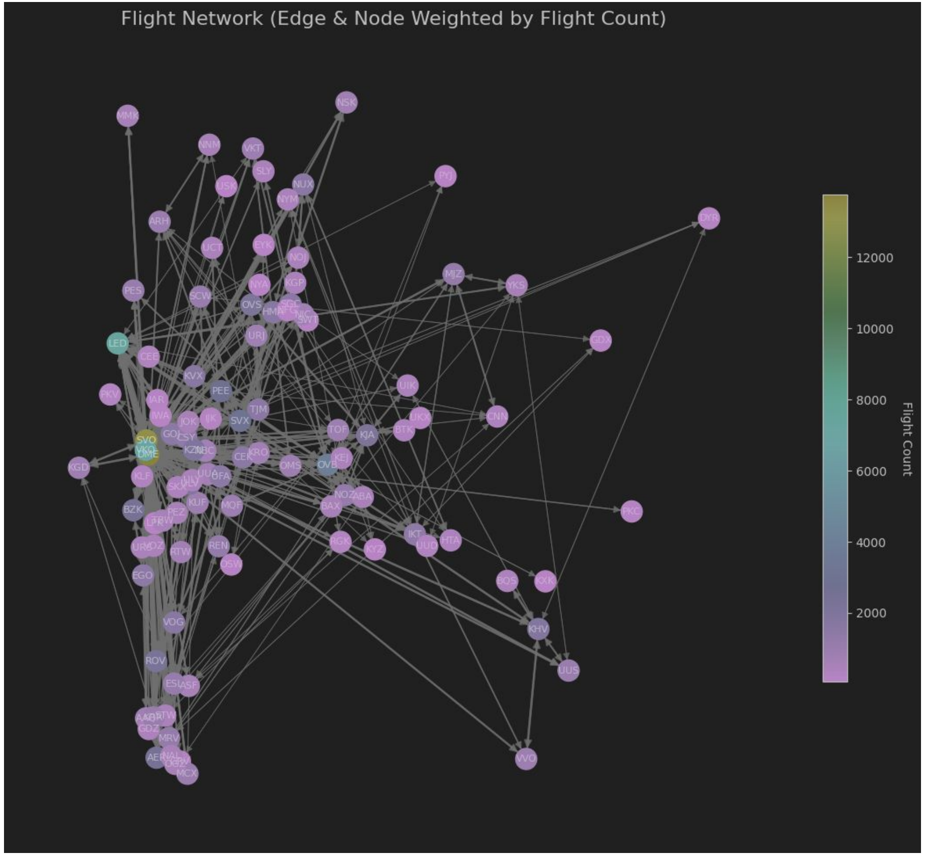
Dataset Overview

Source: Kaggle - Airlines Dataset (Russia, 2017)

Time Range: 2017-07-16 to 2017-09-14

Geographic Scope: Domestic flights across Russia

Data Volume: 8 tables, 61,502 flight records

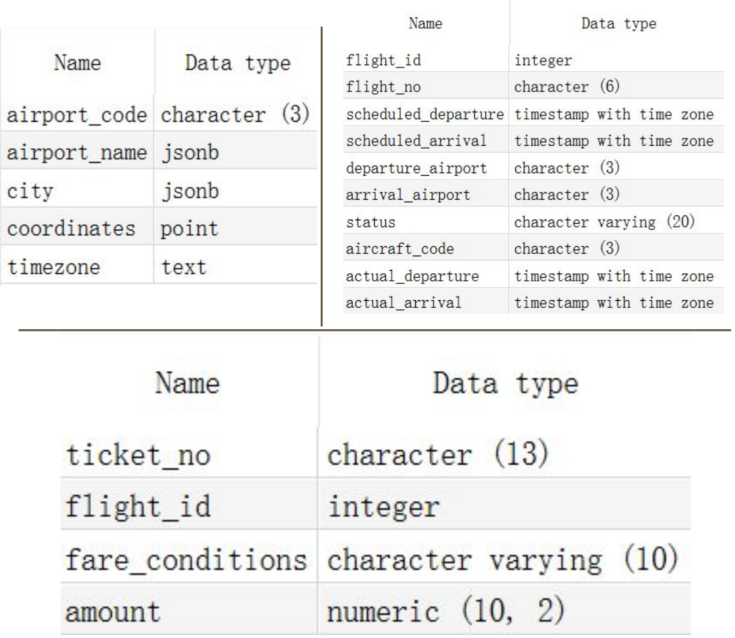


Flight Network Visualization

Nodes = Airports (colored by total flight volume)   Edges = Direct flights

Dataset Overview

Dataset Structure & Preprocessing



Name	Data type
airport_code	character (3)
airport_name	jsonb
city	jsonb
coordinates	point
timezone	text

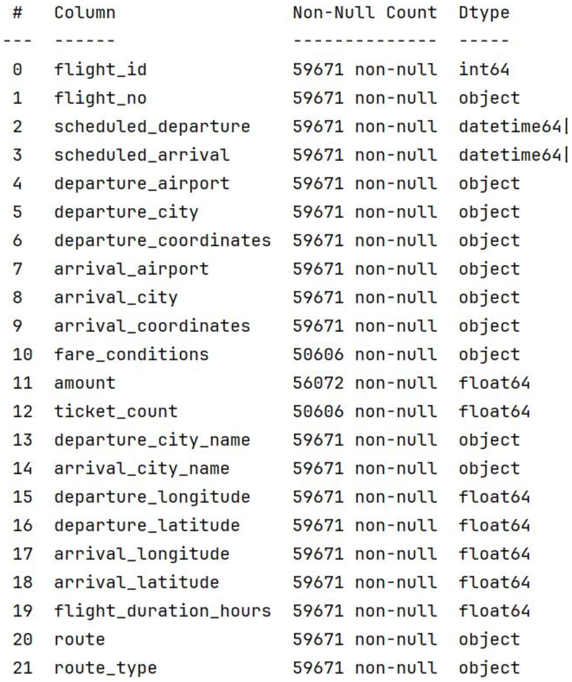
Name	Data type
flight_id	integer
flight_no	character (6)
scheduled_departure	timestamp with time zone
scheduled_arrival	timestamp with time zone
departure_airport	character (3)
arrival_airport	character (3)
status	character varying (20)
aircraft_code	character (3)
actual_departure	timestamp with time zone
actual_arrival	timestamp with time zone

Name	Data type
ticket_no	character (13)
flight_id	integer
fare_conditions	character varying (10)
amount	numeric (10, 2)

Three Source Tables (Raw Schema)

- merge table
- extract city names
- convert time columns

- handle missing prices
- create city-to-airport mapping



```
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   flight_id               59671 non-null  int64
 1   flight_no               59671 non-null  object
 2   scheduled_departure     59671 non-null  datetime64|
 3   scheduled_arrival       59671 non-null  datetime64|
 4   departure_airport       59671 non-null  object
 5   departure_city          59671 non-null  object
 6   departure_coordinates   59671 non-null  object
 7   arrival_airport         59671 non-null  object
 8   arrival_city            59671 non-null  object
 9   arrival_coordinates     59671 non-null  object
 10  fare_conditions         50606 non-null  object
 11  amount                  56072 non-null  float64
 12  ticket_count            50606 non-null  float64
 13  departure_city_name     59671 non-null  object
 14  arrival_city_name       59671 non-null  object
 15  departure_longitude     59671 non-null  float64
 16  departure_latitude      59671 non-null  float64
 17  arrival_longitude       59671 non-null  float64
 18  arrival_latitude        59671 non-null  float64
 19  flight_duration_hours   59671 non-null  float64
 20  route                   59671 non-null  object
 21  route_type              59671 non-null  object
```

Final Dataset After Preprocessing

Methodology

1. Flight Network Construction

Built a time-aware flight network using *NetworkX.MultiDiGraph*

- Nodes: Airports
- Edges: Individual flights between airports
- Attributes: Flight ID, Departure / Arrival Time, Duration, Ticket Price

➢ Supports multiple flights between the same airport pair

➢ Filters out flights departing before the user’s specified time

Methodology

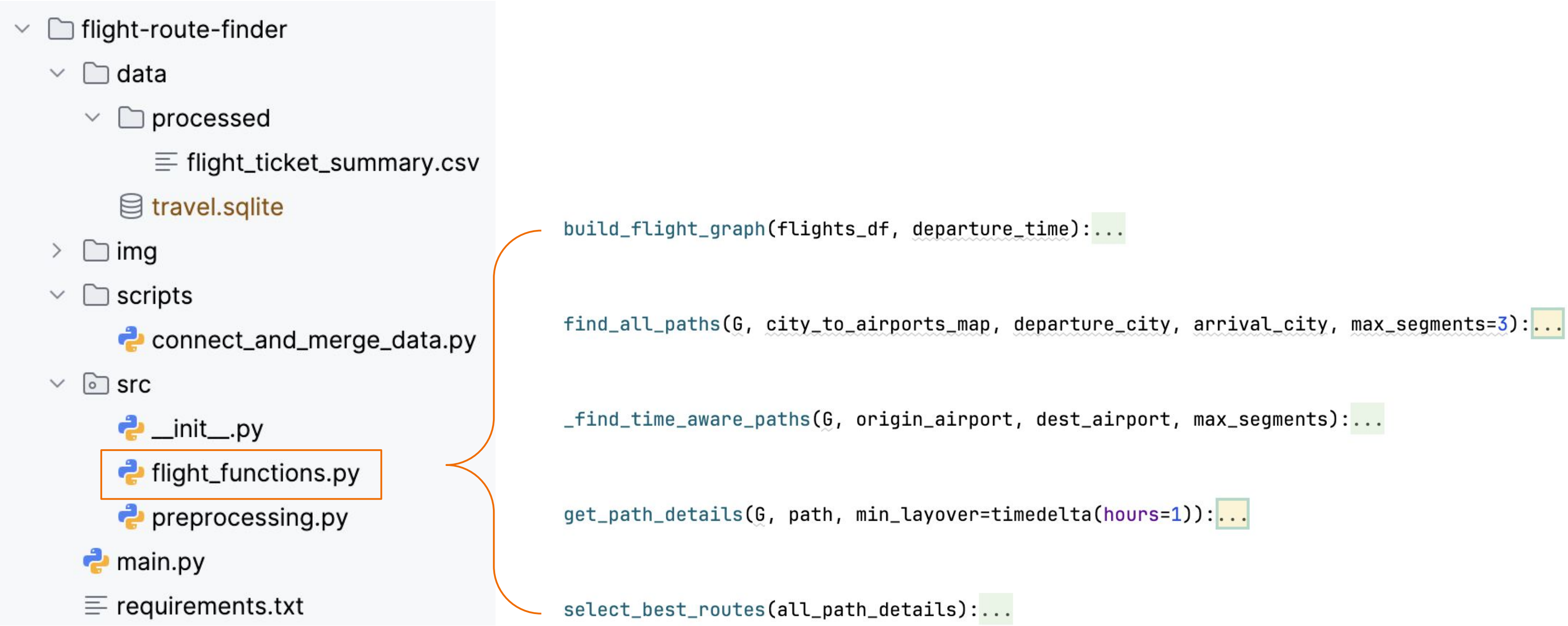
2. Time-Aware Path Search (Modified BFS)

Implements a *custom breadth-first search (BFS) algorithm*

- Explores all airport paths starting from the user’s origin city
- For each potential next flight:
  - Ensures its departure time is after the previous flight’s arrival
  - Requires a minimum layover buffer (e.g., 1 hour)

➢ What Makes It “Time-Aware”: This custom algorithm doesn’t just follow airport connections —it checks whether the schedule is physically feasible

Project Structure



```
flight-route-finder
    data
        processed
            flight_ticket_summary.csv
        travel.sqlite
    img
    scripts
        connect_and_merge_data.py
    src
        __init__.py
        flight_functions.py
        preprocessing.py
    main.py
    requirements.txt
```

```
build_flight_graph(flights_df, departure_time):...

find_all_paths(G, city_to_airports_map, departure_city, arrival_city, max_segments=3):...

_find_time_aware_paths(G, origin_airport, dest_airport, max_segments):...

get_path_details(G, path, min_layover=timedelta(hours=1)):...

select_best_routes(all_path_details):...
```

Results

```
Departure city: Moscow
Arrival city: Novosibirsk
Departure date (YYYY-MM-DD, default: today): 2017-9-4
Departure time (HH:MM, default: now): 17:00
```

```
CHEAPEST ROUTE:
  Total price: 28100.00
  Total duration: 0 days 05:15:00
  Transfers: 1

  Flight segments:
  1. DME → KVX
     Departure: 2017-09-12 15:50:00
     Arrival: 2017-09-12 18:20:00
     Price: 7700.00
  2. KVX → OVB
     Departure: 2017-09-14 11:40:00
     Arrival: 2017-09-14 14:25:00
     Price: 20400.00
```

```
FASTEST ROUTE:
  Total price: 30700.00
  Total duration: 0 days 03:25:00
  Transfers: 0

  Flight segments:
  1. DME → OVB
     Departure: 2017-09-05 11:05:00
     Arrival: 2017-09-05 14:30:00
     Price: 30700.00
```

```
LEAST_TRANSFERS ROUTE:
  Total price: 30700.00
  Total duration: 0 days 03:25:00
  Transfers: 0

  Flight segments:
  1. DME → OVB
     Departure: 2017-09-05 11:05:00
     Arrival: 2017-09-05 14:30:00
     Price: 30700.00
```

Results

```
Departure city: Kaliningrad
Arrival city: Krasnoyarsk
Departure date (YYYY-MM-DD, default: today): 2017-8-10
Departure time (HH:MM, default: now): 08:00
```

```
CHEAPEST ROUTE:
  Total price: 45200.00
  Total duration: 0 days 07:00:00
  Transfers: 2
  Flight segments:
  1. KGD → DME
     Departure: 2017-09-12 17:05:00
     Arrival: 2017-09-12 18:35:00
     Price: 11000.00
  2. DME → OVB
     Departure: 2017-09-13 11:05:00
     Arrival: 2017-09-13 14:30:00
     Price: 27900.00
  3. OVB → KJA
     Departure: 2017-09-14 12:20:00
     Arrival: 2017-09-14 14:25:00
     Price: 6300.00
```

```
FASTEST ROUTE:
  Total price: 48300.00
  Total duration: 0 days 05:55:00
  Transfers: 1
  Flight segments:
   1. KGD → SVO
     Departure: 2017-08-17 12:00:00
     Arrival: 2017-08-17 13:30:00
     Price: 11700.00
  2. SVO → KJA
     Departure: 2017-08-21 10:25:00
     Arrival: 2017-08-21 14:50:00
     Price: 36600.00
```

```
LEAST_TRANSFERS ROUTE:
  Total price: 48300.00
  Total duration: 0 days 05:55:00
  Transfers: 1
  Flight segments:
   1. KGD → SVO
     Departure: 2017-08-17 12:00:00
     Arrival: 2017-08-17 13:30:00
     Price: 11700.00
  2. SVO → KJA
     Departure: 2017-08-21 10:25:00
     Arrival: 2017-08-21 14:50:00
     Price: 36600.00
```

Limitations and Future Enhancement

	Limitation	Future Enhancement
Route Finder Criteria	● Only supports shortest flight duration or lowest price	● Customizable Multi-criteria Optimization ● Additional Optimization Criteria
Data sources	● No real-time flight or gate data; relies solely on static schedules.	● Real-time Flight Status Integration ● Expanded Travel Options
Connection Parameters	● Uses generic transfer time, ignores terminals and delays.	● Airport-specific Transfer Times
Time Zones	● Time shown in UTC format, not adjusted to user’s time zone.	● Enhanced Time Zone Management ● Geographic Route Visualization

THANKS!