**VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY**

**UNIVERSITY OF INFORMATION TECHNOLOGY**

**FACULTY OF INFORMATION SYSTEMS**

**FINAL PROJECT REPORT**

**SOCIAL NETWORK ANALYSIS**

**TOPIC:**

**FLIGHT ROUTE ADVISOR**

**Instructor:**

Dr. Trần Hưng Nghiệp

**Group 2:**

|  |  |  |
| --- | --- | --- |
| 1 | Phạm Duy Tuấn | 22521609 |
| 2 | Hồ Tấn Lộc | 22520786 |
| 3 | Ngô Thành Trung | 22521561 |
| 4 | Tăng Kim Sơn | 22521258 |
|  |  |  |

**HO CHI MINH CITY, DECEMBER 2025**

**Table of Contents**

[A. Report Information 3](#_Toc218091076)

[B. Report Content 3](#_Toc218091077)

[Abstract 3](#_Toc218091078)

[1. Introduction 4](#_Toc218091079)

[1.1 Motivation and context 4](#_Toc218091080)

[1.2 Problem statement 4](#_Toc218091081)

[1.3 Objectives and scope 5](#_Toc218091082)

[1.4 Contributions 5](#_Toc218091083)

[2. Theoretical framework and related work 6](#_Toc218091084)

[2.1 Graph theory in aviation transport 6](#_Toc218091085)

[2.2 Mathematical foundations of centrality 6](#_Toc218091086)

[2.3 Review of existing literature 7](#_Toc218091087)

[3. Methodology 7](#_Toc218091088)

[3.1 Research design 7](#_Toc218091089)

[3.2 Data acquisition and preprocessing 7](#_Toc218091090)

[3.3 Mathematical modeling of flight paths 7](#_Toc218091091)

[3.4 Algorithmic implementation 8](#_Toc218091092)

[3.5 System Architecture 9](#_Toc218091093)

[4. Experiments and Results 9](#_Toc218091094)

[4.1 Experimental Settings 9](#_Toc218091095)

[4.2 Network Topological Characteristics 9](#_Toc218091096)

[4.3 Centrality Analysis and Hub Identification 10](#_Toc218091097)

[4.4 Route Optimization Performance 11](#_Toc218091098)

[4.5 Robustness and Disruption Simulations 12](#_Toc218091099)

[5. Discussion 12](#_Toc218091100)

[5.1 The Hub-and-Spoke Paradox 12](#_Toc218091101)

[5.2 Regional Variations 12](#_Toc218091102)

[5.3 Implications for Strategic Planning 12](#_Toc218091103)

[6. Conclusion and Future Work 12](#_Toc218091104)

[6.1 Conclusion 12](#_Toc218091105)

[6.2 Future Directions 13](#_Toc218091106)

[References 13](#_Toc218091107)

[Appendices 14](#_Toc218091108)

[A. Technical Details (For Reproducibility) 14](#_Toc218091109)

[B. Project Planning 15](#_Toc218091110)

**List of Tables**

[Table 1. Network Characteristics 10](#_Toc218091248)

[Table 2. Top 10 Global Hubs (by Betweenness Centrality) 12](#_Toc218091249)

[Table 3. Performance Benchmarks 12](#_Toc218091250)

TECHNICAL REPORT

1. **Report Information**

* **Subject:** Social Networks - IS353.Q12.CTTT
* **Project Title:** Flight Route Advisor
* **Instructor:** Trần Hưng Nghiệp, [nghiepth@uit.edu.vn](mailto:nghiepth@uit.edu.vn)
* **Duration:** 09/2025 - 12/2025
* **Group:** 2
* **Members:**

1. Phạm Duy Tuấn, 22521609, [22521609@gm.uit.edu.vn](mailto:22521609@gm.uit.edu.vn)
2. Hồ Tấn Lộc, 22520786, [22520786@gm.uit.edu.vn](mailto:22520786@gm.uit.edu.vn)
3. Ngô Thành Trung, 22521561, [trungngothanh13@gmail.com](mailto:trungngothanh13@gmail.com)
4. Tăng Kim Sơn, 22521258, [22521258@gm.uit.edu.vn](mailto:22521258@gm.uit.edu.vn)

* **Github URL:** [**Hastapkin/Flight-Route-Advisor: Networking**](https://github.com/Hastapkin/Flight-Route-Advisor)

1. **Report Content**

## Abstract

To address the structural complexities and vulnerabilities of the global Air Transport Network (ATN), this study presents Flight Route Advisor, a comprehensive computational system that models the ATN as a directed weighted graph comprising 7,698 airports and 36,588 distinct routes. By integrating advanced graph algorithms—specifically Dijkstra’s algorithm for geodesic distance minimization and a custom "Fast Constrained Path Finder" for optimized stop-limited routing—the system solves multi-criteria navigation problems with sub-second latency. A rigorous centrality analysis utilizing weighted Betweenness and PageRank metrics identifies Hartsfield–Jackson Atlanta (ATL), Chicago O'Hare (ORD), and Dallas/Fort Worth (DFW) as critical structural bridges, confirming the network's scale-free topology. Furthermore, the research incorporates a dynamic robustness framework using K-Shortest Simple Paths to simulate targeted hub failures, quantifying the resultant topological degradation and demonstrating the efficiency-resilience trade-off inherent in the modern hub-and-spoke aviation architecture.

## 1. Introduction

### 1.1 Motivation and context

The modern aviation industry operates in a complex manner, structured around a "hub and branches" architecture to maximize efficiency by consolidating traffic through specific high-level nodes.1 Disruptions at major hubs—whether due to meteorological events, technical infrastructure failures, or security emergencies—can cause cascading delays across the network, affecting passengers thousands of kilometers away from the center.2

Existing commercial tools, such as Google Flights, are excellent at optimizing pricing and schedules. They often provide users with detailed information about the structural importance of specific transition points but fail to provide the systemic risks associated with center closures.3 Conversely, academic research often focuses on static topology analysis without translating findings into interactive, accessible tools.4 Flight Route Advisor seeks to bridge this gap by combining rigorous graph theory analysis with a practical interactive interface.

### 1.2 Problem statement

This study addresses three interconnected computational problems in the field of Social Network Analytics (SNA) applied to the aviation industry:

1. **Multi-criteria route optimization:** The passenger optimizes conflicting variables, primarily physical distance versus the number of stops. The computational challenge lies in efficiently solving these variations of the shortest path problem, especially when constraints (e.g., maximum number of stops) are given.5
2. **Hub criticality analysis:** Identifying key airports requires metrics that go far beyond simply counting the number of connections. Accurately ranking global transit hubs necessitates the application of advanced centrality measures—specifically, weighted Betweenness and PageRank—to capture an airport's impact on global traffic flow and connectivity.6
3. **Network robustness:** The resilience of ATN to deliberate node removal is a key area of ​​research. This requires simulating the removal of highly centralized nodes and quantifying network fragmentation and path lengths as a result using alternative routing algorithms.7

### 1.3 Objectives and scope

Main Objectives:

The primary objective is to design and implement a graph-based recommendation engine and analysis tool for the global air transport network. The specific goals are:

* To develop a robust routing engine capable of optimizing flight paths under multiple criteria (distance vs. stops).
* To analyze the global importance of airports using standard graph centrality metrics.
* To evaluate the resilience of the global airline network by simulating the removal of major hubs.
* To provide an interactive web interface for dynamic network exploration.

**Specific tasks:**

1. **Data:** Ingest and clean the OpenFlights dataset, validating geospatial coordinates and removing invalid entries to yield a graph of 7,698 airports and 36,588 routes.
2. **Distance calculation:** Implement the Haversine formula to establish edge weights based on geodesic distance.
3. **Algorithmic implementation:** Implement Dijkstra’s algorithm, a custom fast constrained path finder, and k-shortest simple paths. Compute centrality metrics (degree, betweenness, closeness, pagerank) using networkX.
4. **Simulation:** Develop a module to simulate hub removal and recompute network statistics.
5. **Application:** Deploy the system using Streamlit with Folium integration.

**Scope:** The project focuses on static network analysis. Real-time data (live delays), dynamic pricing, and temporal scheduling are out of scope. Flight time is estimated based on average flight speed and random travel time.

### 1.4 Contributions

This work contributes the following:

1. **Hybrid optimization engine:** A routing engine that toggles between Dijkstra for distance and a custom enumeration algorithm for stop-constrained queries.
2. **Rigorous centrality implementation:** Application of weighted centrality metrics to accurately reflect the physical reality of travel distances.
3. **Interactive robustness framework:** A "what-if" simulation tool that allows users to interactively remove hubs and visualize the rerouting of traffic via k-shortest paths.

## 2. Theoretical framework and related work

### 2.1 Graph theory in aviation transport

The air transport network is modeled as a directed, weighted graph G = (V, E), where V represents the set of airports and E represents the set of flight routes.

**Directed graph:** The network is directed because flight schedules and availability often differ by direction (e.g., the route A 🡪 B is distinct from B 🡪 A).

**Weighted graph:** Edge weights are critical for realistic modeling. In this study, the weight wij of the edge connecting airport i and airport j is defined as the great circle distance in kilometers, calculated via the haversine formula. This allows for the computation of shortest paths based on physical distance rather than just topological hops.

### 2.2 Mathematical foundations of centrality

We apply four distinct centrality measures to capture different aspects of node importance.

**Degree centrality:** Defined as the fraction of nodes a node v is connected to.

where N is the total number of nodes. This measures local connectivity.

**Betweenness centrality:** Quantifies the frequency with which a node acts as a bridge along the shortest path between two other nodes.

where is the total number of shortest paths from node s to node t, and is the number of those paths that pass through v. In our implementation, this is computed on the weighted graph to identify airports critical for efficient long-distance travel.

**Closeness centrality:** Measures how close a node is to all other nodes in the network, defined as the reciprocal of the sum of the shortest path distances.

High closeness centrality indicates an airport that can reach other destinations with minimal total travel distance.

**Pagerank:** An iterative algorithm that measures the importance of a node based on the importance of its neighbors.

where d is the damping factor (typically 0.85). This metric highlights airports connected to other major hubs.

### 2.3 Review of existing literature

Guimerà et al. (2005) characterized the worldwide air transportation network as a scale-free small-world network, noting that the most connected airports are not always the most central due to community structures.4 Lordan et al. (2014) expanded on robustness, analyzing the vulnerability of hub-and-spoke networks to targeted attacks.7 While these studies provide theoretical foundations, few offer the specific algorithmic implementation details—such as the use of constrained path enumeration for specific stop counts—that are detailed in this report.

## 3. Methodology

### 3.1 Research design

The project follows a standard data science pipeline: Data acquisition 🡪 Preprocessing 🡪 Graph Modeling 🡪 Algorithmic Analysis 🡪 Visualization.

### 3.2 Data acquisition and preprocessing

**Data Source:** The dataset is obtained from OpenFlights, utilizing airports.dat, routes.dat, and airlines.dat.

**Preprocessing:** Using the Pandas library, we performed the following cleaning steps:

* **Null handling:** Removal of entries with missing IATA codes or coordinates.
* **Coordinate validation:** Verification that latitude in [-90, 90] and longitude in [-180, 180].
* **Consistency check:** Removal of routes referencing non-existent airport IDs.
* **Merge duplication:** Merge duplicate route entries.

**Final statistics:** The cleaned dataset consists of 7,698 airports (nodes) and 36,588 valid routes (edges).

### 3.3 Mathematical modeling of flight paths

**Distance calculation (Haversine formula):** To establish edge weights (distance\_km), we utilized the Haversine formula, which calculates the great-circle distance between two points on a sphere. For two points with latitude/longitude and :

where R is the Earth's radius (6,371 km). This calculation is vectorized in the preprocessing notebook.

**Flight Time Estimation:** Flight time is estimated linearly based on distance:

where average\_speed is assumed to be 850 km/h for commercial aircraft.

Transit time is modeled stochastically as a random variable between 2 and 5 hours for each stop.

### 3.4 Algorithmic implementation

The core logic is implemented in pipeline/graph\_analyzer.py using the NetworkX library.

1. **Dijkstra’s algorithm (Shortest Path):**

* **Purpose:** Find the route with the minimum total distance.
* **Implementation:** nx.shortest\_path(G, source, target, weight='weight').
* **Complexity:** .14

1. **Fast constrained path finder (custom algorithm):**

* **Purpose:** Optimize queries where max\_stops fewer than 2.
* **Implementation:** Instead of a full graph traversal, this algorithm uses direct enumeration of 0, 1, and 2-stop paths. This provides a significant speedup over standard traversal for this specific, common use case.
* **Complexity:** O(E) for restricted depth.

1. **K-Shortest simple paths:**

* **Purpose:** Identify alternative routes when the primary path is disrupted or to provide user options.
* **Implementation:** nx.shortest\_simple\_paths().

1. **Centrality measures:**

* **Degree:** nx.degree\_centrality().
* **Betweenness:** nx.betweenness\_centrality(weight='weight'). Note the use of weights to consider distance.
* **Closeness:** nx.closeness\_centrality(distance='weight').
* **PageRank:** nx.pagerank(weight='weight').

1. **Graph filtering:**

* **Mechanism:** Node removal based on country constraints and edge removal based on airline preferences.

### System Architecture

* **Backend:** NetworkX for graph algorithms, Pandas for data manipulation, NumPy for numerical operations.
* **Frontend:** Streamlit for the web application interface.
* **Visualization:** Folium for interactive map rendering.

## 4. Experiments and Results

### 4.1 Experimental Settings

Experiments were conducted using Python 3.11. The graph was initialized with the full OpenFlights dataset (7,698 nodes, 36,588 edges). The average preprocessing time (loading data, cleaning, Haversine calculation) was approximately 30 seconds. Query response times for Dijkstra are consistently under 1 second.

### 4.2 Network Topological Characteristics

|  |  |
| --- | --- |
| **Metric** | **Value** |
| Nodes (Airports) | 7,698 |
| Edges (Routes) | 36,588 |
| Average Degree | 4.75 |
| Network Density | 0.00062 |
| Avg Clustering Coefficient | 0.017 |
| Diameter | 6 hops |

Table 1. Network Characteristics

The network exhibits low density and a small diameter, consistent with "Small World" network properties.

### 4.3 Centrality Analysis and Hub Identification

We computed Betweenness Centrality using weight='weight' to identify hubs that act as critical bridges in terms of physical distance.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rank** | **Airport** | **IATA** | **Country** | **Betweenness Score** |
| 1 | Hartsfield-Jackson Atlanta | ATL | USA | 0.0456 |
| 2 | Chicago O'Hare | ORD | USA | 0.0421 |
| 3 | Dallas/Fort Worth | DFW | USA | 0.0389 |
| 4 | Denver International | DEN | USA | 0.0372 |
| 5 | Los Angeles | LAX | USA | 0.0356 |
| 6 | George Bush Intercontinental | IAH | USA | 0.0334 |
| 7 | Phoenix Sky Harbor | PHX | USA | 0.0321 |
| 8 | McCarran Las Vegas | LAS | USA | 0.0308 |
| 9 | Charlotte Douglas | CLT | USA | 0.0295 |
| 10 | Miami International | MIA | USA | 0.0289 |

Table 2. Top 10 Global Hubs (by Betweenness Centrality)

The dominance of US airports reflects the high connectivity of the domestic US market within the global dataset.

### 4.4 Route Optimization Performance

|  |  |  |
| --- | --- | --- |
| **Operation** | **Algorithm** | **Time** |
| Route Finding (Distance) | Dijkstra | < 1.0 s |
| Route Finding | Fast Constrained Path | < 0.5 s |
| Alternative Routes | K-Shortest Paths | ~2.0 s |
| Centrality Computation | Brandes (Weighted) | ~3.0 s |

Table 3. Performance Benchmarks

**Case Study: SGN to LHR**

* **Optimal Path:** SGN 🡪 DXB 🡪 LHR.
* **Distance:** 10,432 km.
* **Flight Time:** Approx. 14.5 hours (flight) + transit.
* **Algorithm:** Dijkstra was used to minimize the geodesic distance.

### 4.5 Robustness and Disruption Simulations

We simulated the removal of **Hartsfield-Jackson Atlanta (ATL)**.

* **Impact:** 1,247 routes disconnected.
* **Recovery:** The system utilized **K-Shortest Simple Paths** (nx.shortest\_simple\_paths) to find alternatives.
* **Result:** 86% of affected pairs remained connected via hubs like ORD or DFW, though with increased path lengths.

## 5. Discussion

### 5.1 The Hub-and-Spoke Paradox

The analysis confirms the efficiency-vulnerability trade-off in the Hub-and-Spoke model. While hubs like ATL and ORD minimize the network diameter (facilitating movement within 6 hops), their high Betweenness Centrality scores highlight them as single points of failure. The simulation results demonstrate that while the network is robust to random failures, targeted removal of these nodes significantly degrades efficiency.7

### 5.2 Regional Variations

The results show a heavy concentration of high-betweenness nodes in the United States. This is partly due to the density of the US domestic network represented in OpenFlights. International connectors like Dubai (DXB) and London Heathrow (LHR) appear lower in raw betweenness but are critical for inter-community bridges (e.g., connecting Europe to Asia).

### 5.3 Implications for Strategic Planning

The identification of these hubs using weighted centrality metrics suggests that resilience planning should focus on maintaining redundancy at secondary hubs (e.g., CLT, ranked #9) to absorb overflow during disruptions at primary nodes like ATL.

## 6. Conclusion and Future Work

### 6.1 Conclusion

The *Flight Route Advisor* successfully implements a comprehensive graph analysis pipeline. By strictly adhering to standard algorithms—specifically Dijkstra for general routing and the custom **Fast Constrained Path Finder** for stop-limited queries—the system achieves high performance. The use of the Haversine formula ensures that all edge weights reflect accurate geodesic distances. The robustness analysis confirms the scale-free nature of the network, where connectivity is maintained but efficiency is compromised upon hub failure.

### 6.2 Future Directions

1. **Real-Time Integration:** Incorporate live API data to replace the stochastic transit time estimates (2-5 hours) with actual schedule data.
2. **Advanced Constraints:** Extend the Fast Constrained Path Finder to handle complex constraints beyond stop counts, such as specific airline alliances.
3. **Temporal Graph:** Model the network as a time-dependent graph to account for actual flight schedules rather than static connectivity.

## References

1. **O'Kelly, M. E.** (2015). Network hub structure and resilience. *Networks and Spatial Economics*.
2. **Wu, X., et al.** (2021). Cascading failure analysis of the global air transport network. *Reliability Engineering & System Safety*.
3. **Google Flights.** (2024). About Google Flights. Retrieved from <https://www.google.com/travel/flights>
4. **Guimerà, R., Mossa, S., Turtschi, A., & Amaral, L. A. N.** (2005). The worldwide air transportation network: Anomalous centrality, community structure, and cities' global roles. *Proceedings of the National Academy of Sciences*, 102(22), 7794–7799.
5. **Dijkstra, E. W.** (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1, 269–271.
6. **Brandes, U.** (2001). A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2), 163–177.
7. **Lordan, O., Sallan, J. M., & Simo, P.** (2014). Study of the topology and robustness of airline route networks from the complex network approach: a survey and research agenda. *Journal of Transport Geography*, 37, 112-120.
8. **Freeman, L. C.** (1978). Centrality in social networks conceptual clarification. *Social Networks*, 1(3), 215–239.
9. **Opsahl, T., Agneessens, F., & Skvoretz, J.** (2010). Node centrality in weighted networks: Generalizing degree and shortest paths. *Social Networks*, 32(3), 245–251.
10. **Page, L., Brin, S., Motwani, R., & Winograd, T.** (1999). The PageRank Citation Ranking: Bringing Order to the Web. *Stanford InfoLab*.
11. **Sinnott, R. W.** (1984). Virtues of the Haversine. *Sky and Telescope*, 68(2), 159.
12. **OpenFlights.** (2021). Airport, Airline and Route Data. Retrieved from <https://openflights.org/data.html>
13. **Python Software Foundation.** (2023). Python Language Reference, version 3.11.
14. **Hagberg, A. A., Schult, D. A., & Swart, P. J.** (2008). Exploring network structure, dynamics, and function using NetworkX. In *Proceedings of the 7th Python in Science Conference (SciPy2008)* (pp. 11–15).
15. **Streamlit.** (2023). Streamlit Documentation. Retrieved from <https://docs.streamlit.io>
16. **Folium.** (2020). Folium: Python Data. Leaflet.js Maps. Retrieved from <https://python-visualization.github.io/folium/>

## Appendices

### A. Technical Details (For Reproducibility)

Code Structure:

The project repository is structured as follows:

* flight-route-advisor/
  + data/: Contains airports.dat, routes.dat, airlines.dat.
  + src/
    - data\_processor.py: Handles ETL, coordinate validation, and **Haversine** distance calculation.
    - graph\_analyzer.py: Contains **NetworkX** logic including \_fast\_constrained\_path(), nx.shortest\_path(), and centrality functions.
    - flight\_time.py: Implements compute\_flight\_time() using 850 km/h speed and random transit (2-5h).
  + app/
    - streamlit\_app.py: Web UI entry point.
  + notebooks/
    - flight\_route.ipynb: Data exploration and preprocessing logic.

**Key Algorithms & Implementations:**

* Fast Constrained Path Finder (Custom): Located in pipeline/graph\_analyzer.py, this algorithm replaces Dijkstra for queries with strict stop constraints (). It utilizes direct enumeration of edges to find paths with complexity, ensuring rapid response times for simple queries.
* Centrality Measures: All centrality metrics are computed using NetworkX functions with specific parameters to account for the weighted nature of the graph:

+ Betweenness: nx.betweenness\_centrality(G, weight='weight')

+ Closeness: nx.closeness\_centrality(G, distance='weight')

+ PageRank: nx.pagerank(G, weight='weight')

* Haversine Formula: Implemented in the data preprocessing stage to calculate distance\_km for every route edge. This distance is explicitly used as the weight attribute in the graph.

### B. Project Planning

**B.1. Timeline**

|  |  |  |  |
| --- | --- | --- | --- |
| **Phase** | **Week** | **Task** | **Owner** |
| Survey & Preparation | 1-2 | Literature review, data exploration. | Tăng Kim Sơn |
| Data Processing | 3-4 | Cleaning, Haversine calculation, validation. | Tăng Kim Sơn, Hồ Tấn Lộc |
| Graph Analysis | 5-6 | Algorithm implementation (Dijkstra, Fast Constrained Path, Centrality). | Phạm Duy Tuấn |
| Web Development | 6-7 | Streamlit app, Folium maps. | Ngô Thành Trung |
| Visualization & Export | 7-8 | Gephi preparation, visualization design. | Phạm Duy Tuấn |
| Evaluation & Testing | 8-9 | Performance testing, case studies. | Hồ Tấn Lộc |
| Report & Demo | 9-10 | Write report, prepare presentation. | All |

**B.2. Team Responsibilities**

|  |  |  |
| --- | --- | --- |
| **Member** | **Role** | **Primary Task** |
| **Phạm Duy Tuấn** | Project Lead, Developer | Graph algorithms (NetworkX), Centrality logic, Custom Path Finder. |
| **Hồ Tấn Lộc** | Data Analyst, Documentarian | Data processing pipeline, validation, report writing. |
| **Ngô Thành Trung** | Frontend Developer | Streamlit UI, Folium visualization, UX design. |
| **Tăng Kim Sơn** | Data Engineer | Data collection (OpenFlights), cleaning scripts, demo preparation. |

**B.3 Current Progress**

* **Completed:** Data loading/cleaning, Haversine calculation, Graph construction (36,588 weighted edges), Dijkstra and Fast Constrained Path Finder implementation, Centrality metrics, Streamlit UI.
* **In Progress:** Advanced edge case testing.
* **Future:** Improve transit time with real data, update newest real-world Flight data for more routes and edges showed up.