Flight Path Analysis and Airport Delay Heat Mapping in a Real-Time Air Traffic Simulation

Anthony Mainero, Tom Schmidt, Harley Sugarman

November 14, 2013

1 Introduction

Flight delays cost the global economy approximately $33 billion every year. According to the Federal Aviation Association, Half of this enormous sum is borne by the airlines in federal penalties, rebooking fees, and hotel rooms. The other half is borne by passengers in missed work, rescheduled connections, and cancelled meetings [1]. In 2010, over 37% of flights within the United States departed or arrived late, with an average delay of 29 minutes [5].

Delays are also notoriously difficult to deal with. The effect of a single bumped flight sends a ripple through the network of airports that can take hours to dissipate. It's not hard to see why this is the case: Turnaround times for planes are already short, and runway timetables (particularly in more popular airports) cannot accommodate unexpected landing times without creating even more congestion.

The goal of our project is to answer the question whether a given passenger can optimize his travel plans based on a specific set of parameters. For example, Bob needs to fly from Chicago O'Hare (ORD) to Santa Barbara (SBA) on the morning of 4th July [See Figure 1]. Which route minimizes the probability of a delay? Should he travel through New York (JFK) or Los Angeles (LAX)? What is the expected variance in delay time on each of these routes?

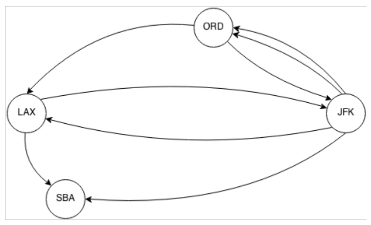

Figure 1: A Simplified Airport Network.

To answer these questions, we have built a simulation that approximates flight delay propagation in the United States' air traffic network. We treat the system as a multi-directed graph. Each airport is represented as a node, and a flight forms an edge between two distinct nodes. Figure 1 illustrates an example graph with four nodes and eight edges. Note that there can be multiple flights between two airports on a given day. Using existing historical flight data taken from the US Bureau of Transportation and Statistics (BTS), we train our simulation on a number of features using linear regression. Using this data, we create a graph (described above) which can produce reasonably accurate simulation of all flights in a given day. We can then use a variety of graph analysis techniques (detailed in the Algorithms section below) to determine the optimal path from between two nodes. A summary of our approach to this research can be found in Figure 2.

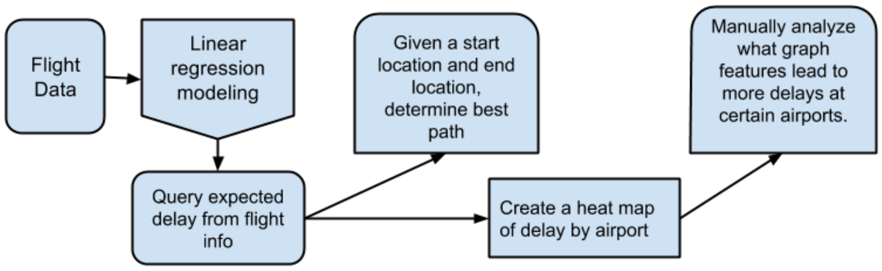
Flight Data
Linear regression modeling
Given a start location and end location, determine best path
Manually analyze what graph features lead to more delays at certain airports.
Query expected delay from flight info
Create a heat map of delay by airport

Figure 2: Goals of the project.

Finally, we take our analysis further and construct a heat map of the US air travel graph based on the total number and severity of delayed flights at each airport. This map can be analysed to identify best routes, airports to avoid, and congestion bottlenecks.

1.1 Prior Work

Much of the previous work in this field was discussed in the Project Proposal, so this section will serve primarily as a brief review of existing results, as well as introduce two new papers that contain techniques used in our algorithmic analysis.

Much previous work in this field has been concerned with modelling flight patterns and the viral spread of delays from one airport to another. Fleurquin [4, 5] found that the determining factor in the length of delays was flight connectivity (caused by passengers missing a connecting flight), and that the range of flight delays was extremely large, averaging at around an hour and peaking at approximately 700 minutes.

In *Minimizing Flight Delay* [3], Dey successfully models a similar problem of finding the minimum delay route between two airports using a Monte Carlo simulation and some basic network analysis. He also develops a version of Dijkstra's algorithm that we draw from in our own analysis of the data.

Ulrik Brandes [2] has also developed a faster method for betweenness centrality that runs in $O(nm + n^2 \log n)$ time, improving from $O(n^3)$, which we use in our feature selection for linear regression.

2 Data and Analysis

Fortunately, the Bureau of Transportation Statistics publishes a CSV for every year they have collected flight statistics, starting in 1987. The CSV consists of an entry for every flight that contains its origin, destination, timestamps for important milestones (taxi time, departure time, arrival time) and a handful of pre-calculated metrics (minutes of delay, air time, distance between airports, etc.). The BTS spreadsheet also contains one extremely important (and useful) column: the reason for delay. If a flight is late within the United States, the FAA assigns blame to one of a number of reasons (security, weather, aircraft malfunction, previous flight delay, and National Air System (NAS) errors) and logs this data. We use this data as a feature to predict which planes or airports are affected in our simulation, and more accurately depict the effects of different kinds of delay. For example, a malfunction may take one plane out of commission, but bad weather may shut down an entire airport for several hours.

Although all of this data is publicly available, we chose to use more recent years (2009-2012) in training and testing our program. Due to rapid advances in technology over the past 30 years, the number of delays (and many of the reasons for those delays) simply aren't relevant today. Furthermore, several of the values used in our algorithm only started being tracked relatively recently. The dataset is also extremely large - a single month's worth of flight information is over 200MB in size - so using more than a few years of data would result in an unwieldy several-hundred gigabyte large database.

There was initially a minor issue with our initial data where a flight's departure and arrival timestamps were given in the timezone of the origin and destination airports, respectively. To circumvent this problem, we use the data provided by [www.timezonedb.com] to lookup the timezone of each airport and convert the timestamps to the uniform UTC standard.

3 Mathematical Background and Algorithms

3.1 Linear Regression

We use information from our flight schema to build training data (input variables and an output scalar) to build a regression model for determining delay due to five different causes described above. Thus, we train on the same set of independent variables on five different models, one for each of these causes. The dependent variable Y we train on for each of these models is the delay due to the respective cause. Thus, the expected delay for a given flight is the sum of the predicted delays returned by the five different models.

The independent variables X for determining flight delay include some given directly from our dataset (flight time, distance, arrival and departure locations, month/day, time of day, airline, etc.) and some data that we extrapolate from our dataset (airport degree, closeness centrality, and betweenness centrality). However, since the edges in our network are not static (flights only happen at certain times of day), we expect that network properties should not be calculated across the entire graph for a given day, but rather for the network as it exists in a time window (i.e. the network of all flights active 90 minutes before expected departure up to 90 minutes after expected arrival).

Once the five separate linear regression models are trained, we then have a way to query, given flight information as a set of the independent variables X, what is the sum of the expected delays from each type of delay Y.

We have chosen to use linear regression as opposed to other models for a couple of reasons:

1. Linear regression predicts output as a scalar value, rather than as a binary or otherwise qualitative category.
2. Linear regression is a somewhat efficient algorithm, which is important given that our dataset consists of millions of points, each representing a flight.

3.2 Closeness Centrality and Betweenness Centrality

As an input variable for logistic regression (described above), we're using the algorithm for betweenness centrality from Brandes' paper [2].

For closeness centrality, we simply find the shortest paths to all other nodes in the graph using Djikstra's Algorithm (described below), sum them, and the return the inverse of the result.

3.3 Modified Djikstra's Algorithm for Fastest Routes

In order to create a service that will return the fastest route for one to take from a starting airport to an ending airport given flight delays, we need a modified shortest-path-finding algorithm. Djikstra's Algorithm is a well-documented algorithm that, given a starting location and an end destination, returns the least-cost path between them. We're implementing a modified version of Djikstra's algorithm in the sense that, for each node in the path, we're setting it's cost to be not the given flight time and layover time, but rather the flight time, layover time, and expected delay. The set of airports reachable from an airport A in our path is the set of airports E where flights to an airport B∈E are *available* after the expected arrival time, given delay, of the flight in our path to A. We calculate this by, before running Djikstra, iterating through all flights in a given day in order start time, calculating their expected delay given the state of the graph to that point. We then change the departure and arrival times of the graph accordingly. If a plane arrives late to its destination, it must have at least 45 minutes for refueling, maintenance, and boarding before taking off again. The available flights are determined by the expected arrival time at airport A from their previous location, also given delay.

In the algorithm, we create a working path WP beginning at our starting location, then create a set of new paths that contain flights from our starting location to every reachable airport B∈E. We enqueue in PQ each of these paths, with the cost in the priority queue being the travel time + expected delay. Then, we dequeue each path WP from the queue one at a time. If WP ends at our final destination, we return that as the path with least cost factoring in expected delay. Otherwise, we create new paths that are copies of WP, each with a new flight and airport appended to them from E, and enqueue those.

3.4 Creating a Delay Heat Map

In order to create a delay heat map for our entire set of airports in the network, we simply iterate through each airport A, sum the expected delays of all flights that depart from A and all flights that arrive at A, and normalize by dividing by the total number of flights into and out of A. That normalized value is the heat map value.

3.5 Mathematical Background

Our logistic regression model utilizes least-squares, as discussed in class. For any function Y = F(X) and a set of data points D, where Y is our dependent variable and X is our independent variable, we can calculate the orthogonal distance from each data point in D to F(X,Y). The least-squares function returns for D the function F(X,Y) such that the sum of squares of those distances is the lowest. This is referred to as the *best fit* of a function to the data.

4 Difficulties

The biggest problem we have encountered has been determining what to do with our function for expected delay. We initially wanted to use it to rewire the graph to be more resilient to delays, but we ran into several issues: Determining the constraints under which we can rewire is very difficult, if not impossible, without data we don't have. In real world flight systems, flight paths are chosen primarily for economic reasons, and have to satisfy constraints based on customer demand, economic feasibility, and a limited number of planes. We can't just remove edges and add them elsewhere because that would imply that we could teleport planes. Our method for calculating expected delay relies on the propagation effects of a plane's delay on the rest of that plane's flights that day. For these reasons, we feel that delay optimization is not actually a reasonable problem to solve. We have instead decided to tackle the problem of travel optimization.

The other biggest difficulty so far has been finding a good dataset and processing that data. We eventually found data from the Bureau of Transportation Statistics. The data is extremely unwieldy, however, and parsing it alone takes plenty of time. Performing regression analysis on the data is highly time-consuming and may actually be too processor-intensive to execute reasonably on our own computers, so we may move to clusters. This factor has ruled out even more time-consuming and sophisticated techniques, such as Random Forests and SVMs. So far, we have only performed regression on a small fraction of the data.

We still need to perform regression on the entire data, hook the resulting model into a querying model, generate a heatmap, and write up a manual analysis of what features of the graph contribute to a higher heat signature at specific airports. We have written code to get all of our features, including the ones we extrapolate from the data.

References

[1] Ball, Michael; Barnhart, Cynthia; et al.; *Total Delay Impact Study*; 2010.

[2] Brandes, Ulrik; *A Faster Algorithm for Betweenness Centrality*; 2001.

[3] Dey, Tanujit; Phillips, David; and Steele, Patrick; *Minimizing Flight Delay*; 2011.

[4] Fleurquin, Pablo; Ramasco, Jose J.; and Eguiluz, Victor M.; *Systemic delay propagation in the US airport network*; 2013.

[5] Fleurquin, Pablo; Ramasco, Jose J.; and Eguiluz, Victor M.; *Characterization of delay propagation in the US air transportation network*; 2010.

[6] Pastor-Satorras, Romualdo; and Vespignani, Alessandro; *Immunization of complex networks*; 2002.