Minimizing Flight Delay

Tanujit Dey*, David Phillips†, Patrick Steele‡

Abstract

In this paper, we model the problem of finding the minimum-delay route as a shortest path problem with stochastic costs. Our solution algorithm uses Monte Carlo simulation and network optimization techniques. Of particular novelty in our approach is the air network sampling that we use in order to accurately model the cascading effect of delays on flights. Our statistical models and simulations are based on real flight data obtained from the U.S. Department of Transportation.

Key Words: bootstrapping and ensemble regression, stochastic optimization, data mining, network optimization

1. Introduction

Air traffic delays are a current and growing problem with severe economic and environmental impacts. A recent congressional report has estimated that delays cost $41 billion in 2007 alone and caused an extra 740 million gallons of jet fuel to be burned (Schumer and Maloney, 2008). Moreover, a 2009 Government Accounting Office report has estimated that the number of flights is going to increase from 50 million in 2008 to 80 million by the year 2025 (Dillingham, 2009). We consider the problem of the air traveler determining how to best plan his travel.

Our specific contributions are as follows.

- We formulate the problem of finding a flight plan with minimum delay as a mathematical program we call the *shortest paths problem with correlated random lengths.* We also show how to extend our formulation to minimize more general functions such as overall flight time and flight costs.
- We develop a statistical model by using an ensemble technique to find significant variables for predicting delay for any given airline. The resulting prediction can then be used in cascade sampling.
- We give an algorithm to determine the relevant flight legs for the given origin and destination. Subsetting to the relevant flight legs allows us to tractably use Monte Carlo simulation in computing the flight plans with the highest probability of small delay.
- We implemented a visualization tool which displays results from our algorithms as well as other statistical analyses on the air transport graph. The visualization of both our algorithmic results and the statistical analyses makes it possible to discern meaningful trends in a large and complicated dataset.

*Mathematics Department, The College of William & Mary, tdey@wm.edu. Supported in part by NSF grant DMS-0703532 and a William & Mary summer grant

†Mathematics Department, The College of William & Mary, djphil@wm.edu. Supported in part by NSF grant DMS-0703532 and a NASA/VSGC New Investigator grant

‡Mathematics Department, The College of William & Mary, prsteele@wm.edu. Supported in part by a Chappell fellowship.

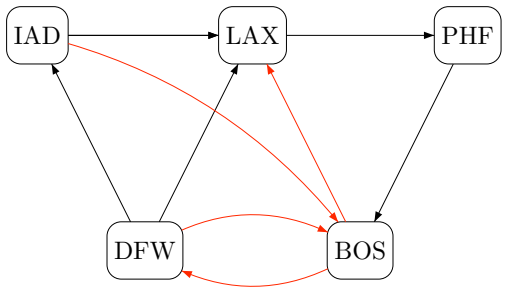

Figure 1: Two airport-flight graphs on 5 airports, represented as nodes, with 9 flights represented as edges. Note that the edge (DFW,BOS) is not the same as (BOS,DFW). The edges in red are on the path {(IAD,BOS), (BOS,DFW), (DFW,BOS), (BOS,LAX)}

Previous works focus on the scheduling decisions airlines can make to reduce overall delay. For example, Bratu and Barnhart (2006) solve integer programs that help airline companies determine flight rerouting plans that minimize the delay to passengers when exogenous events (e.g., inclement weather) disrupt the flight schedule. Delahaye and Odoni (1997) use stochastic optimization techniques to determine air traffic policies that reduce air space congestion. In related papers, Nilim et al. (2003) and Nilim and El Ghaoui (2005) present Markov decision process approaches to scheduling flights accounting for inclement weather.

In contrast, we focus on developing decision support tools that help individual travelers find flight plans that minimize expected delay. In formulating and solving the problem from the individual traveler's perspective our methods can also be used by airlines and air traffic policy makers to determine bottleneck airports with respect to delay.

An essential part of our model is the use of the *airport-flight graph*. We consider a *graph*, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents *vertices* and $\mathcal{E}$ represents *edges* between pairs of vertices. An edge $(i, j) \in \mathcal{E}$ implies that the edge "starts" at the node i and "ends" at node j. By considering the airports as vertices and the flight legs as edges, a graph is a natural way to represent the air travel system (all references cited that consider air travel use variations of this model), e.g., see Figure 1. A flight plan in such a graph corresponds to a *path* in the network, which can be defined as a sequence of edges, $P = \{(i_1, i_2), (i_2, i_3), \ldots, (i_{k-1}, i_k)\}$ where every consecutive pair of edges share a vertex (e.g., again, see Figure 1) and no edge repeats itself.[1]

The organization of the article is as follows. The description of the data is in Section 1.1 and the statistical models and simulation are in Section 2. The outputs of the simulation form the inputs to the delay minimization formulation which we describe in Section 3. We present our algorithms to conduct the simulation and solve the delay minimization problem in Section 4. Computational experiments are discussed in Section 5. Proofs of our results are omitted in this extended abstract for space reasons and available in Dey et al. (2009).

1.1 Data

The data set used came from the U.S. Department of Transportation's Bureau of Transportation Statistics, and includes data on domestic flights from 1987 to 2008. Each flight has a database entry that includes information about the arrival, departure and delay times,

[1] We only consider directed edges and paths.

DelayLevel	0	1	2	3	4
Sum of all delays (minutes)	$0-15$	$15-30$	$30-60$	$60-120$	$120+$

Table 1: The response variable *DelayLevel*.

origin and destination airports and carrier. The data set contains close to 120 million flights and is over 12 gigabytes large.

We analyzed a subset of a data that consisted of the largest seven carriers in terms of number of flights, which were American Airlines (AA), Continental Airlines (CO), Delta Airlines (DL), Southwest Airlines (WN), American Eagle Airlines (MQ), United Airlines (UA), and SkyWest (OO) Airlines. Within these airlines, we analyzed 1987, 1992, 1997, and 2002 for trends over time, and constructed our regression models and simulation using the years 2005 to 2008. The variables we used were year, month, day of month, day of week, departure time, and arrival time of each flight, the carrier, the departure, arrival, weather, NAS, security, taxi in, taxi out, and late aircraft delays, and finally the origin and destination of the flight. Cancelled flights were omitted from the analysis.

2. Data analysis & simulation

Given the size of the data set and the time length over which the data spans, we choose to focus our analyses on recent data in order to best predict delay for current flights. We did, however, examine simple statistics from 1987, 1992, 1997 and 2002 in order to demonstrate how subtle differences in the air network graph can reveal information about the underlying data. As Figure 2 shows, the increased complexity of the air network can be seen with airport graphs. Other trends such as an evolution to a "hub-and-spokes" air network can also be seen.

We describe our statistical model for predicting delay in Section 2.1 and our simulation for generating delay in Section 2.2.

2.1 Statistical Model

The objective of our statistical analysis was to create an accurate model for predicting delay. Because of the characteristics of our data, we used a full multiple linear regression model, and in order to obtain a stable predictor, we used ensemble estimates via bootstrapping.

Due to the small number of variables relative to the sheer size of the data, we fitted a full model for multiple linear regression, and then selected the significant variables for predicting delay. There were several delay variables that corresponded to different types of delay. In order to use both the type and the magnitude of each type of delay as a predictor for a given day, we created the categorical variable *DelayLevel*, which is based on the sum of all the different delays, and is defined in Table 1.

In order to obtain a stable estimator, we used a bootstrapping technique. For each bootstrap draw, a multiple linear regression was performed on 70% of the data (sampled without replacement). The significant variables and their corresponding estimates were selected. These are then combined over the bootstrap draws to form an ensemble estimator for each given airline/year. By introducing bootstrapping, we injected extra randomness into the data that gets averaged over all the bootstrap runs which results in a estimator that is stable.

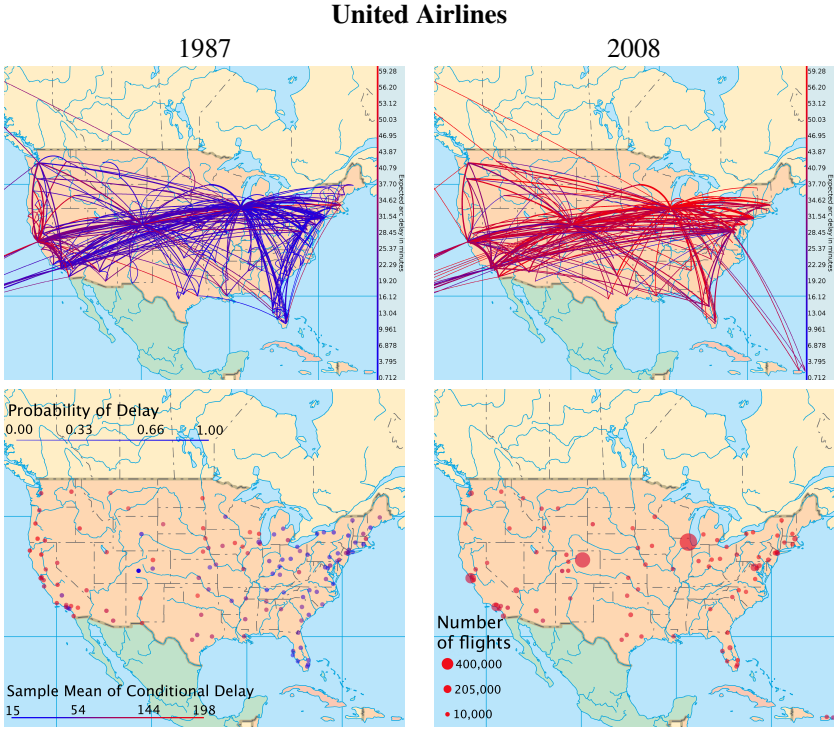

Figure 2: Historical delay averages for United Airlines in 1987 and 2008 are presented in two different graph depictions. In both graphs, the color indicates the average delay per flight leg. The top row has "edge"-focused graphs where the actual number of flights can be observed. Note that both the 1987 and 2008 arc graphs have the same connectivity but that expected delay has increased. The bottom row shows the "node"-focused graphs where the node size corresponds to number of flights and the node opacity indicates the probability of delay. The node focused graph shows how United has converted to the "hub-and-spokes" airport graph model

Conceptually, the method can be explained as the following multi-stage procedure:

1. Bootstrap the data. Run multiple regression on the bootstrapped data.
2. Find the significant variables and their corresponding estimates based on the output of step 1.
3. Repeat steps 1-2.
4. After finishing the bootstrap iterations, calculate an ensemble estimator for each significant variables in the model.

2.2 Simulation

In this section we describe our simulation. Our goal in the simulation was to accurately generate delay for each flight leg in a way which (a) is stable, (b) takes into account the impact of delays from the rest of the system and (c) is computationally efficient. In all of our simulations, we first determine the flight legs that are relevant to the desired flight plan. For example, if the flight plan is to travel from JFK to LAX on a Monday in November using Delta in 2 flights or less, we first must find all Delta flight legs that would allow

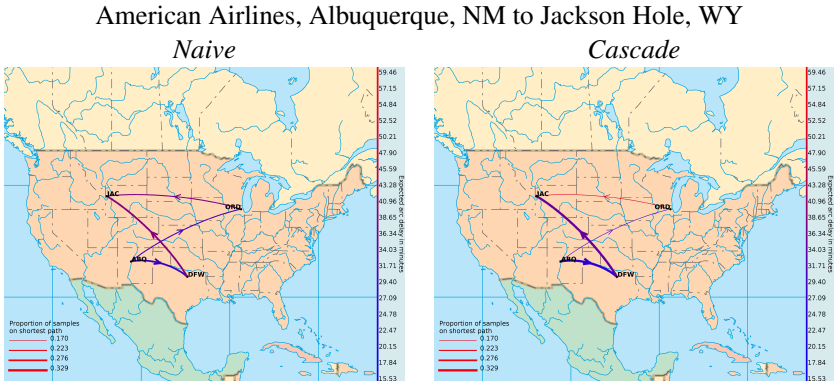

Figure 3: Line thickness indicates probability of being the shortest delay path. Color indicates the expected delay in minutes. *Naive* sampling indicates that both of the two flight plans have an equal probability of being the shortest delay path, whereas *Cascade* sampling indicates that flying through DFW has a much greater likelihood of being the shortest delay path.

a traveler to start from JFK and end up in LAX on a Monday in November within two flight legs. Determining the subset of relevant flight legs is nontrivial and we describe this fully in Section 4. We then sample variables from the generated subset in order to calculate predicted delay. We devised two sampling methods, labeled *Naive* and *Cascade*, and two ways of using the sampled variables calculate predicted total delay, labeled *Actual* and *Ensemble*. Thus, there were a total of four different simulations. As we shall see, our preferred sampling method was *Cascade-Ensemble*. We first describe the difference the different sampling methods and then the difference between calculated predicted total delay.

In *Naive* sampling, the data for each flight leg is chosen independently and uniformly at random from the relevant data. Continuing the JFK to LAX example, suppose that a relevant flight leg was JFK to ORD. In *Naive* sampling, we uniformly select a flight at random from amongst all Delta JFK-ORD flights that occurred on a Monday in November from 2005 to 2008. Assuming that ORD-LAX was also relevant, the ORD-LAX flight leg predictor variables used would be independently and uniformly chosen at random from all Delta ORD-LAX flights that also occurred on a Monday in November from 2005 to 2008. Note that *Naive* sampling does not take into account interdependencies between flight legs, nor does it necessarily sample flights that are actually feasible to the travel plan if more than one flight leg is required.

In *Cascade* sampling we seek to address the flight-leg interdependencies and flight plan feasibility requirements. Instead of sampling flight legs independently, we sample a *day* uniformly from amongst the appropriate days of travel. Within the sampled day, the flight legs sampled must satisfy the feasibility constraint that they leave no sooner than the incoming flight leg arrives. In our previous example, if *Cascade* sampling was used, first a date would be chosen from all the Mondays of November from 2005 to 2008. Then, starting at JFK, a Delta JFK-ORD flight leg would be uniformly chosen at random. Then, the ORD-LAX flight would be uniformly chosen at random from all flights that took off after the JFK-ORD flight landed. Note that *Cascade* sampling generates flight plans that could actually occur within the desired dates. Also, *Cascade* sampling accurately models the correlations of delay that occur between flight legs by realistically generating the types of acute delay-causing events that occur in the airport system. As shown in Figure 3,

American Airlines, LaGuardia airport, NY to San Diego, CA

Actual *Ensemble*

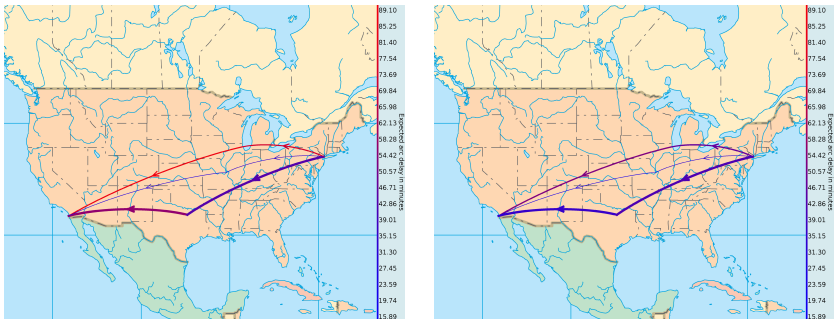

Figure 4: *Actual* versus *Ensemble*. Line thickness indicates probability of being the shortest delay path. Color indicates the expected delay in minutes. *Actual* predicts higher amounts of delay with an average standard error of 10.5% versus 6.9% in the *Ensemble* simulation over all flight leg delay time predictions after 500 iterations.

Cascade sampling can also lead to different conclusions of which flight plans cause the most delay.

We now describe our method for calculating delay based on the flight leg data sampled. When *Actual* is used, the delay used is just the sum of the various delay variables from flight leg sample. When *Ensemble* is used, the predictor variable coefficients generated from our statistical model are used to take a weighted sum of the predictor variables from our sampled flight leg. The weighted sum is a prediction of the response variable *DelayLevel* which was then linearly extrapolated to predict delay for the flight leg. In many of our experiments, there was not much difference between the best flight plans suggested by*Actual* and *Ensemble*. However, the delay prediction from *Ensemble* was much smoother than the *Actual* which generated much noisier (and, therefore, less stable) fits to the predictors. In addition, for some flight legs, the sample size was actually quite small, which means *Actual* would exacerbate the lack of stability in the delay prediction. As can be seen in Figure 4, the *Ensemble* simulation can be much more stable than the *Actual*.

Results are not reported in this extended abstract, but looking at the prediction error for delay, it was found that our ensemble predictor is very stable and the error rate is very low. We have also taken into consideration whether these ensemble estimators are good in predicting delay for a year when estimates are from a different year. For example, we calculated ensembles for an airline in 2007, and found that the predictions for year 2008 were very promising with very low prediction error when compared to the actual data.

3. Problem formulation

We reduce the problem of finding the flight time with the highest probability of minimum delay to a *shortest paths problem with uncertain edge lengths*.[2] For our graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we assume a set of edge lengths, $\ell_{ij}, \forall (i,j) \in \mathcal{E}$. For a path $P = \{(i_j, i_{j+1}) : j = 1, \ldots, k\}$, the total length of P is the sum of the edge lengths of edges on P, i.e., $\sum_{j=1}^{k} \ell_{i_j i_{j+1}}$. Then, given an origin vertex, $s \in N$, and destination vertex $t \in N$, the shortest path problem is to find the shortest length path in G that starts from s and ends at t.

For the case of deterministic edge lengths and graph structures, fundamental results in

[2]We refer to edge lengths, but note that the typical term in network optimization is edge *costs*.

shortest paths algorithms were given by Bellman (1958) and Dijkstra (1959). Their work is particularly noteworthy since variants of their algorithms are still the fastest known algorithms for solving the deterministic shortest paths problems over general (Bellman) and nonnegative (Dijkstra) edge lengths. The subsequent work on shortest paths problems is extensive, and for space reasons, we refer readers toAhuja et al. (1993) and Cormen et al. (2001) for several algorithms on the deterministic case. Early work on the case where the edge lengths were randomized include Dijkstra (1959) where the problem was reduced to the deterministic version by considering expected values of each edge lengths. For the problem where edge lengths were random, but independently generated, algorithms that find edges with probability guarantees include those by Loui (1983) and adaptive algorithms by Bertsekas and Tsitsiklis (1991) and Fan and Nie (2006). For the case where edge lengths were random and correlated but the density functions were known, Fan et al. (2005) recently described an algorithm to determine the path with the highest probability of being less than a certain length. Our problem differs from these previous results in that our probability density functions are unknown, although we are able to simulate the various edge lengths. Moreover, since the delay of a given flight leg causes delays to other flight legs, we cannot assume independence in flight delays.

Recall that $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a graph where $\mathcal{V}$ are vertices representing airports and $\mathcal{E}$ are edges (i.e., pairs of nodes) that represent possible flight legs. We use L_{ij} to denote the random amount of delay on a given flight leg $(i, j) \in \mathcal{E}$. For a given origin, s, and destination, t, we denote the set of paths from s to t in $\mathcal{G}$ as Ω. We wish to find α_P, which is the probability that $P \in \Omega$ represents the flight plan with the greatest probability of the lowest delay. To find this, we can solve the following stochastic mathematical program.

$$\begin{array}{lll} \min & \sum_{P\in\Omega}(\sum_{(i,j)\in P} L_{ij})\alpha_P & \\ \text{subject to} & \sum_{P\in\Omega} \alpha_P = 1 & \\ & \forall P \in \Omega, \qquad \alpha_P \geq 0. & \end{array} \tag{1}$$

We note that to solve even a deterministic version (i.e., when L_{ij} are constant) of (1) is complicated since the size of Ω is exponential in the number of edges. However, we can reformulate (1) by considering *arc flows*, x_{ij}, which represent the probability that (i, j) is on some path that has the probability of being the minimum delay path.

$$\begin{array}{lll} \min & \sum_{(i,j)\in\mathcal{E}} L_{ij}x_{ij} & \\ \text{subject to} & 0 \leq x_{ij} \leq 1 & \forall (i,j) \in \mathcal{E} \\ \text{(a)} & \sum_{\{j:(i,j)\in\mathcal{E}\}} x_{ij} - \sum_{\{k:(k,i)\in\mathcal{E}\}} x_{ki} = 0 & \forall i \in \mathcal{V} \setminus \{s,t\} \\ \text{(b)} & \sum_{\{j:(s,j)\in\mathcal{E}\}} x_{sj} = 1 & \\ \text{(c)} & \sum_{\{k:(k,t)\in\mathcal{E}\}} x_{kt} = 1 & \end{array} \tag{2}$$

Constraints (a) ensure that flow going into any vertex leaves the vertex, except for the origin and destination. In the context of our model, this ensures that the traveler leaves any airport she arrives at unless it's the origin or destination. Constraint (b) ensures the traveler leaves the origin and constraint (c) ensures the traveler arrives at the destination.

To convert an arc flow solution, x_{ij}, to a path solution, α_{ij}, we can use a flow decomposition algorithm (Ahuja et al., 1993). Such an algorithm runs in time that is linear in proportion to the number of edges. We note that the formulations are easily extended to random variables other than L_{ij}, such as total travel time and travel cost.

To solve for x_{ij} in (2), we adopt a Monte Carlo simulation (Spall, 2005). For each arc, we randomly generate a sample length and then we solve a (deterministic) shortest paths problem. By repeating this several times, and counting up the number of times an arc appears on a shortest path, we can estimate x_{ij} with a controllable sample. Here is a pseudo-code description of our approach.

(a) Construct a relevant network, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, for the given origin, destination, flight date, airline(s) and number of flight legs.

(b) Initialize arc counters c_{ij} to zero for each $(i, j) \in \mathcal{E}$.

(c) Repeat steps (d)-(f) k times

  (d) Generate valid flight delays to find sample realizations of L_{ij} for all $(i, j) \in \mathcal{E}$.

  (e) Find the shortest path with respect to the sample realizations.

  (f) For all arcs (i, j) on the shortest path found, increment c_{ij} by 1.

(g) Set $x_{ij} = c_{ij}/k$ and return the x_{ij}.

To solve for the shortest path problem with the sample realizations in step (d), we use Dijkstra's algorithm since we assume the flight delays (i.e., arc lengths) to be nonnegative. We could relax this assumption and use Bellman's algorithm instead. Accomplishing step (a) is complicated, and we describe our algorithms to construct the network in Section 4.

4. Constructing and visualizing graphs

In this section, we first describe our algorithm for subsetting our flight network $\mathcal{G}$ to the airports (i.e., vertices) and flight legs (i.e., edges) relevant to the origin-destination, airline, date and flight-leg constraint we are interested in. We then describe our visualization tool.

4.1 Constructing relevant graphs

Since our runtime scales with our problem size, we seek to reduce the flight network by as much as possible. The network created only contains flight legs that are a part of a path from the origin to the destination. We let k denote the constraint on the number of flight legs our flight plan is allowed.

To create this network, we begin with the full network and apply an algorithm we call REDUCE. REDUCE subsets the network by selecting only the vertices and edges that satisfy the following conditions. Only airports that can be reached in k flight legs or fewer from the origin or destination are kept. Then, we only include a flight (i, j) if j is within $k-1$ flight legs of the destination and i is within $k-1$ flight legs of the origin. To conduct the removal of these flight legs, we use Breadth-First Search on $\mathcal{G}$ and on the *reverse graph* of $\mathcal{G}$, which is $\mathcal{G}_R = (\mathcal{V}, \mathcal{E}_R)$ where $\mathcal{E}_R = \{(i, j) : (j, i) \in \mathcal{E}\}$. We can accomplish REDUCE using four passes of Breadth-First Search. To conclude this section, we state our main result.

Theorem 1. *Given a graph $\mathcal{G}$, source node s and destination node t,* REDUCE *produces a graph that contains only arcs that can be traversed as part of a k-path from the origin node s to the destination node t, and runs in $O(|\mathcal{V}| + |\mathcal{E}|)$ time.*

4.2 Visualizing our results

Using a combination of Java, Python and c, we implemented an applet which takes the origin, destination and flight date. The tool calls our algorithms that sample the data and solve the stochastic optimization problem. It then draws the subgraph associated with the paths that had any positive sample probability of being the minimum delay flight plan where the arc thickness corresponds to the probability. The tool also color codes arcs to indicate the degree of expected delay on the different flight legs.

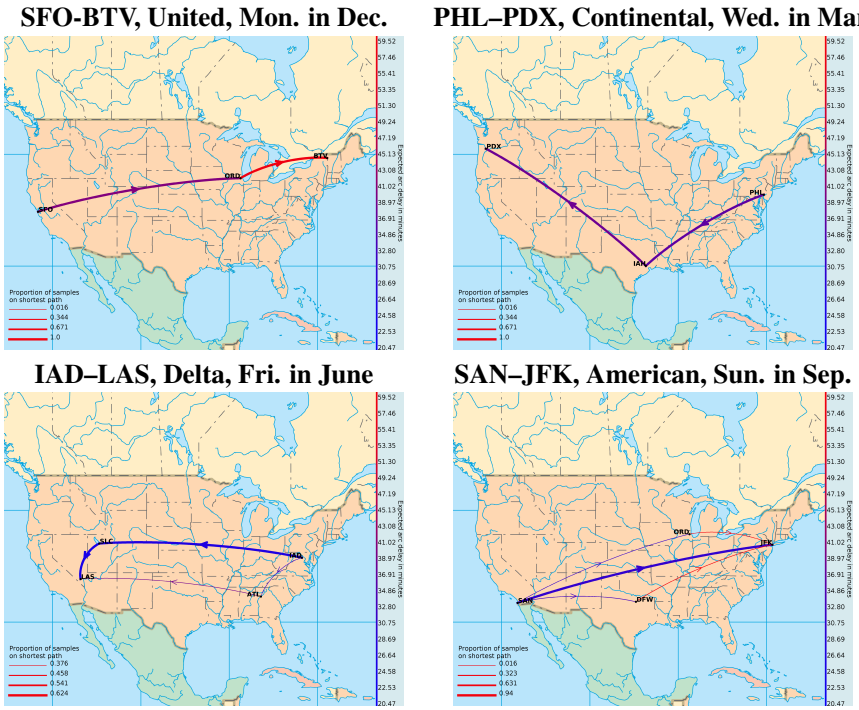

Figure 5: A variety of flight plans generated with our methods.

5. Experimental results

Our algorithms and models generate stable predictions of flight plans that have small amounts of delay. As can be seen in Figure 5, our algorithms can generate flight plans for any possible date, origin and destination on the seven largest air carriers of our dataset. Our algorithms will also work if the data is not specified or a range of dates is specified. All of our reported results computed 500 iterations of the Monte Carlo simulation on a 3.0 GHz Intel machine with two dual-core processors and 4 GB of RAM. Our algorithms took less than minute to find flight plans when no more than 2 flight legs were allowed. However, when 3 flight legs were allowed, the runtime increased to 7 minutes. In future work, we plan on implementing methods that scale more efficiently with the flight leg constraint.

References

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1993), *Network Flows : Theory, Algorithms, and Applications*, Englewood Cliffs, NJ: Prentice Hall.

Bellman, R. (1958), "On a routing problem," *Quarterly of Applied Mathematics*, 16, 87–90.

Bertsekas, D. and Tsitsiklis, J. (1991), "An analysis of stochastic shortest path problems," *Mathematics of Operations Research*, 580–595.

Bratu, S. and Barnhart, C. (2006), "Flight operations recovery: New approaches considering passenger recovery," *Journal of Scheduling*, 9, 279–298.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001), *Introduction to Algorithms*, The MIT Press and McGraw-Hill, 2nd ed.

Delahaye, D. and Odoni, A. (1997), "Airspace congestion smoothing by stochastic optimization," *Lecture Notes in Computer Science*, 163–176.

Dey, T., Phillips, D. J., and Steele, P. (2009), "Minimizing flight delay," Tech. rep., Dept. of Math, College of William & Mary, contact authors for manuscript.

Dijkstra, E. W. (1959), ""A Note on Two Problems in Connection with Graphs"," *Numerische Mathematik*, 1, 260–271.

Dillingham, G. (2009), "Next Generation Air Transportation System, Status of Transformation and Issues Associated with Midterm Implementation of Capabilities," Testimony before the Subcommittee on Aiviation, Committee on Transportation and Infrastructure, House of Representatives GAO-09-479T, United States Government Accountability Office, Washington, DC.

Fan, Y., Kalaba, R., and Moore, J. (2005), "Shortest paths in stochastic networks with correlated link costs," *Computers and Mathematics with Applications*, 49, 1549–1564.

Fan, Y. and Nie, Y. (2006), "Optimal routing for maximizing the travel time reliability," *Networks and Spatial Economics*, 6, 333–344.

Loui, R. (1983), "Optimal paths in graphs with stochastic or multidimensional weights," *Communications of the ACM*.

Nilim, A. and El Ghaoui, L. (2005), "Robust solutions to markov decision problems with uncertain transition matrices," *Operations Research*, 53, 780–798.

Nilim, A., El Ghaoui, L., and Duong, V. (2003), "Multi-Aircraft Routing and Traffic Flow Management under Uncertainty," in *5th USA/Europe Air Traffic Management Research and Development Seminar, Budapest, Hungary*, pp. 23–27.

Schumer, C. and Maloney, C. (2008), "Your flight has been delayed again: flight delays cost passengers, airlines, and the US economy billions," *The US Senate Joint Economic Committee*.

Spall, J. (2005), *Introduction to stochastic search and optimization: estimation, simulation, and control*, Wiley-Interscience.