Reconstructing Complete Paths from Segment and Origin-Destination Data

Doug Altner, Lisa Vasko, David Klamm
Operations Research Department
The MITRE Corporation
McLean, Virginia 20111
{ daltner, lvasko, dklamm } @mitre.org
January 17, 2016

Abstract

Given the number of people who flew each flight leg and the number of people who ultimately flew between each origin and destination, how could we best reconstruct the complete itineraries of all passengers, including all layovers? We show how inferring complete passenger itineraries from such data can be modeled as a minimum cost path decomposition problem with path-based costs rather than arc-based costs, and solve this problem with column generation. Computational results on real airline data are included.

Key Words: Air Traffic; Column Generation; Network Optimization; Path-Based Costs; Path Reconstruction

1 Introduction

A number of non-airline organizations can use aggregate data on complete passenger itineraries—that is, data on the number of passengers who flew from A to B to C in month X on carrier Y—but this data is not publicly available. One remedy is to estimate this data by inferring complete itineraries from aggregate data on the number of people who flew each flight segment and the number of people who ultimately flew between each origin-destination pair. This information is largely publicly available for domestic flights [2].

At first, this inference might seem like a trivial flow decomposition problem, which can be solved in linear time [3]. However, the challenge is that shorter paths in this context are arguably more realistic than longer paths and so we prefer a flow decomposition that favors shorter paths and disfavors longer paths.

In this paper, we show how to estimate complete airline passenger itineraries from flight segment and origin-destination data. We formulate this problem as a path decomposition problem with path-based costs and then solve these linear programs with column generation. As a side
note, we are not aware of other researchers studying path decomposition problems with path-based costs even though researchers have been using column generation to solve path-based formulations of network flow problems since some of the first papers involving column generation. (E.g., Ford and Fulkerson 1958 [7].)

There are several reasons why researchers may want to reconstruct complete itineraries. This information would aid a variety of economic forecasts, traffic forecasts, and public sector capital improvement programs for airports and transit stations. Knowing complete airline itineraries can also help estimate incoming revenue for federal trust funds that are financed by taxes, which may include per-passenger taxes on flight segments that don’t apply to segments that are part of a multi-legged itinerary involving international travel or rural airports. Furthermore, complete passenger itineraries would also be valuable for checking the consistency of publicly available data sets published by the Bureau of Transportation Statistics [2].

Other researchers have studied similar problems, but to the best of our knowledge, none have proposed a similar kind of mathematical programming based solution for itinerary reconstruction. Coldren et al. 2003 [6] present a model for predicting airline ridership at the itinerary level, which is different from disaggregating historical data. Chu and Chapleau 2008 [5] and Gordon 2012 [8] propose methods for inferring complete passenger itineraries in a city bus network but they don’t use mathematical programming techniques.

Barnhart, Fearing and Vaze 2013 [4] also face the problem of inferring complete airline passenger itineraries from the BTS data sets. However, they also have access to some proprietary booking data and are interested in a more granular result: estimates on the number of passengers flying a specific itinerary, on a specific day, at a specific time, on a specific operating carrier. To accomplish this, they iteratively assign passengers to itineraries (which come from the proprietary data) according to a probability distribution weighted by a utility function used to model a passenger’s likelihood of choosing that specific itinerary based on a variety of factors such as a departure times and layover times. We don’t have access to proprietary data on itineraries, and our research purposes don’t require disaggregating this data by time and day, and so we have different alternatives to explore.

This paper is organized as follows: In section 2, we give a formal statement of this problem, and discuss why a standard path-decomposition approach won’t suffice. In section 3, we present our algorithms for solving this problem using column generation. In section 4, we present computational results using real data. In section 5, we draw some conclusions and list a few outstanding issues.

2 Problem Statement

In this section, we define some preliminary concepts and state the Itinerary Reconstruction Problem.

In this paper, we refer to the data set that indicates how many flew on the flight leg from airport A to airport B (in a given month, on a given carrier) as the segment data. Likewise, we refer to the data set that indicates the origin and final destination of each passenger as the market data. Given a passenger is traveling from A to B, an itinerary in this context would indicate the complete sequence of airports that the passenger visits in his trip from A to B,
We now informally state the Itinerary Reconstruction Problem as follows:

**Itinerary Reconstruction Problem** Given segment data and market data for a specific carrier and month, find the most realistic set of complete passenger itineraries that fits this data.

Of course, this problem statement raises the question: What is the standard for judging that one set of possible itineraries more realistic than another? Before giving the standard we chose, we first show that there are indeed options for reconstructing paths, and some may be less realistic than others.

Figure 1 illustrates how there could be options for reconstructing paths. In this example, each node corresponds to an airport and each arc corresponds to a flight segment. This data for this example indicates that two passengers are traveling through this network. The first starts at node A and ends up at node B. The second starts at node D and ends at node E. In addition, the segment data indicates that each segment has exactly one passenger.

There are two options for using this information to infer itineraries. The first option is to assume the first passenger traveled directly from A to B, and the second had an itinerary with four legs: D→A→C→B→E. The second option is to assume the first passenger traveled from A to B via C, and the second flew from D to A to B to E. Some may argue that the second option is more realistic, as it might be more believable that the passengers had 1 and 2 layovers respectively than for one passenger to have 3 layovers.

For this project, we decided that itinerary sets with few passengers on itineraries with a large number of legs are more realistic than the contrary. A precise objective function to model this informal rule of thumb is presented in the next subsection.
2.1 Mathematical Programming Formulation

We can define an Itinerary Reconstruction Problem for each carrier and each month. (In this context, months are always paired with a year, and so January 2014 counts as a month). Let $C$ be the set of all carriers of interest and let $M$ be the set of all months of interest. Let $N_{c,m} = (V, E_{\text{seg}})$ be the network that corresponds to carrier $c$ and month $m$ for each $c \in C$ and $m \in M$. In this series of networks, nodes correspond to airports, directed edges correspond to flight segments, and the capacities on the edge $(u, v)$ correspond to the total number of passengers who flew from airport $u$ to airport $v$ on carrier $c$ during month $m$. We call these networks the segment networks. Each path in the segment network corresponds to a possible itinerary.

Similarly, let $d_{s,t}$ be the number of passengers who bought a ticket for an itinerary that starts at airport $s$ and ultimately terminates at airport $t$ on carrier $c$ during month $m$ and let $M_{o-d}$ be the set of all origin-final destination pairs for carrier $c$ and month $m$. For simplicity of notation, we are leaving out the $c$ and the $m$ from the notation even though the market data and segment data differs with each carrier-month tuple.

We can then formulate the Itinerary Reconstruction Problem as the following linear program:

\[
\begin{align*}
\text{Minimize} & \quad \sum_{p \in P} |p|^2 x_p \\
\text{Subject to:} & \quad \sum_{p \in P; e \in p} x_p \leq c_e \quad \forall \ e \in E_{\text{seg}} \\
& \quad \sum_{p \in P_{s,t}} x_p \geq d_{s,t} \quad \forall \ (s,t) \in M_{o-d} \\
& \quad x_p \geq 0 \quad \forall \ p \in P
\end{align*}
\]

Here, $P$ corresponds to the set of all possible paths in the segment network, between any origin-destination pair in $M_{o-d}$. $x_p$ is the decision variable that corresponds to the number of passengers on the itinerary corresponding to path $p$.

The objective function aims to reduce the number of passengers assigned to large itineraries to make the solution more realistic. The objective coefficients are squared to ensure that the mathematical program favors, for example, assigning passengers to flights with three legs rather than dividing them equally between flights with two legs and flights with four legs. If these coefficients were not squared then each of these solutions would be equally desirable, which is not what we want.

The first set of constraints ensures the number of passengers assigned to each flight segment by our path reconstruction method does not exceed the actual number of passengers on that flight segment according to the segment data. The second set of constraints ensures that we identify an $s$-$t$ path for each passenger who bought a ticket from airport $s$ to airport $t$ according to the market data.

We developed a path-based formulation for this problem rather than an arc-based formulation because 1.) we need to know the number of passengers on each itinerary, and 2.) the costs of assigning passengers to an itinerary are path-based rather than arc-based to ensure longer itineraries are disfavored.
The fact that we used a path-based formulation presents a challenge. A large carrier such as Southwest airlines serves around 150 airports, which means there is easily at least a billion possible paths in its segment network. Explicitly listing all possible decision variables is practically impossible. Hence, we use column generation so we only add variables on an as needed basis.

3 Solving the Itinerary Reconstruction Problem

In this section, we discuss how we solved the mathematical program (1). First, we present our greedy heuristic for generating an initial solution. Then we present our column generation technique.

3.1 Generating an Initial Solution

To construct an initial solution, we greedily assign as many passengers as possible to direct flights, then to flights with two legs, then flights with three legs, then flights with four legs, and finally, to flights with five legs. We stopped assigning passengers after flights with five legs to avoid spending too much time constructing an initial solution.

There is no guarantee that this algorithm will produce a feasible solution—that is, a solution in which every passenger is assigned to a path—even if we assume there are no errors in the data. However, as is discussed in the next subsection, the fact that this greedy algorithm usually doesn’t produce a feasible solution does not pose an issue.

Algorithm 1 contains pseudocode for our greedy algorithm. When we implemented this, we arbitrarily ordered the paths.

Algorithm 1: Greedy Algorithm for Constructing Initial Solutions

begin
for \( \ell \in \{1, 2, \ldots, 5\} \) do
  for \((s, t) \in M_{s-t} \) do
    for \( p \in \mathcal{P} : |p| = \ell \) do
      \( x_p \to \min \{d_{s-t}, \min_{e \in p} \{c_e - \sum_{q \in \mathcal{P}, e \in q} x_q\}\} \)
    end
  end
end

3.2 Column Generation Schema

In column generation, there is a master problem which contains our mathematical programming problem with the current subset of generated decision variables (i.e., columns), and then there is the pricing subproblem which is used to determine whether any of the other decision
variables need to be added to the master problem to create a better solution. Algorithm 2 contains an overview of the standard schema that we used for column generation.

Algorithm 2: Column Generation Schema

begin
1. Create an initial set of columns \( P_0 \) using Algorithm 1
   \( i \rightarrow 0 \)
do
   Solve the master problem using the columns in \( P_i \)
   \( P_{i+1} \rightarrow P_i \)
   for each \((s, t) \in M_{o-d}\) do
     Solve the pricing subproblem corresponding to \( s-t \)
     if \( A \) s-t column, \( p_{s-t} \), needs to be generated then
       \( P_{i+1} \rightarrow P_{i+1} \cup \{p_{s-t}\} \)
     end
   end
   \( i \rightarrow i + 1 \)
while Columns are still being generated
end

Let \( P_i \) be the cumulative set of paths we’ve added so far during column generation before iteration \( i \). At the beginning of iteration \( i \), we solve the following master problem:

Minimize \( \sum_{p \in P_i} |p|^2 x_p + \sum_{s-t \in M_{o-d}} 1000 \sigma_{s-t} \)

Subject to:
\[
\begin{align*}
\sum_{p \in P_i : e \in p} x_p & \leq c_e \quad \forall e \in E \\
\sum_{p \in P_{s-t}} x_p + \sigma_{s-t} & \geq d_{s-t} \quad \forall (s, t) \in M_{o-d} \\
x_p & \geq 0 \quad \forall p \in P_i \\
\sigma_{s-t} & \geq 0 \quad \forall (s, t) \in M_{o-d}
\end{align*}
\]

Here, \( \sigma_{s-t} \) is an auxiliary variable that corresponds to the number of passengers who cannot yet be assigned to a single path. 1000 is a prohibitively large coefficient we selected to discourage the solver from setting these variables to a non-zero value.

We now discuss the pricing subproblem. Let \( \lambda_e \) be the corresponding dual variable for the segment constraint associated with edge \( e \), and let \( \omega_{s-t} \) be the dual variable for the market demand constraint associated with origin-destination pair \( s-t \). Checking the reduced cost for a non-basic \( s-t \) path-variable \( p \in P \) is tantamount to determining if \( \sum_{e \in p} \lambda_e + \omega_{s-t} - |p|^2 > 0 \). (This comes from the dual constraint that corresponds to the primal variable \( x_p \).)

To avoid explicitly computing the reduced cost of a multitude of decision variables, we instead implicitly check the reduced costs by formulating a pricing subproblem as an integer program which will give us the \( s-t \) path with maximum reduced cost, for each \((s, t) \in M_{o-d} \). Its decision
variables as follows:

\[ y_e := \begin{cases} 
1 & \text{If edge } e \text{ is in our } s-t \text{ path} \\
0 & \text{Otherwise}
\end{cases} \quad (3) \]

\[ \xi_{|p|} := \begin{cases} 
1 & \text{If our } s-t \text{ path contains exactly } |p| \text{ edges} \\
0 & \text{Otherwise}
\end{cases} \quad (4) \]

For our pricing subproblems, we assume paths can have at most 10 edges. This is an engineering rule of thumb in which we assume itineraries with more than 10 legs are unrealistic. Even 10 may seem high to many, but a few individuals are willing to endure a surprising number of layovers to accumulate frequent flier rewards.

The integer programming formulation of the pricing subproblems is:

\[
\text{Maximize} \quad \sum_{e \in E} \lambda_e y_e - \sum_{|p|=1}^{10} |p|^2 \xi_{|p|}
\]

\[
\text{Subject to :}
\]

\[
\sum_{e \in FS(s)} y_e = 1
\]

\[
\sum_{e \in RS(t)} y_e = 1
\]

\[
\sum_{e \in FS(v)} y_e - \sum_{e \in RS(v)} y_e = 0 \quad \forall \ v \in N \setminus \{s,t\}
\]

\[
\sum_{e \in E} y_e + \sum_{|p|=1}^{10} (10 - |p| + 1) \xi_{|p|} = 10
\]

\[
\sum_{|p|=1}^{10} \xi_{|p|} = 1
\]

\[
0 \leq y_e \leq 1 \quad \forall \ e \in E
\]

\[
\xi_{|p|} \in \{0, 1\} \quad \forall \ |p| \in \{1, 2, \ldots, 10\}
\]

The objective function in this formulation is maximizing the reduced cost of column we are generating. Since we are solving one pricing subproblem for each \((s, t) \in M_{\sigma, d}\), we can drop the dual variable \(\omega_{s-t}\) from the objective function.

The first three sets of constraints are standard network flow constraints. The first requires that an edge come out of \(s\), the second requires an edge enter \(t\), and the third ensures flow balance at all non-terminal nodes. Here, \(FS(v)\) and \(RS(v)\) designate the forward star and reverse star of node \(v\). The fourth and fifth constraints ensure \(\xi_{|p|}\) takes the value 1 if and only if the path constructed by the \(y\) variables has exactly \(|p|\) edges in it.

The \(y\) variables are non-integers. We can treat these as bounded linear variables due to the following observation:

**Lemma 3.1** Suppose each of the \(\xi\) variables in subproblem 5 are restricted to integer values. Then all \(y\) variables are integral in every optimal extreme point of the subproblem induced by this restriction.

**Proof:** We prove this claim by contradiction. Suppose there is an optimal solution, say \(y^*\), such that some of the \(y\) variables assume fractional values. We show this cannot be an extreme
point solution.

Choose a pair of nodes \( u \) and \( v \) such that \( u \neq v \), \( 0 < \sum_{e \in FS(u)} y_e = \alpha \leq 1 \), \( \sum_{e \in RS(v)} y_e = \alpha \), and there exists a set \( P_{u-v} \) of at least two edge-independent \( u-v \) paths such that \( \sum_{p \in P_{u-v}} x_p = \alpha \) where \( x_p \) is the flow on path \( p \). Such a set of edge-independent paths can be chosen because the flows in this network are all s-t flow and obey flow balance. Thus, if we have a pair of nodes \( u \) and \( v \) that satisfy the first three but not the fourth of our listed conditions, then we can always recursively redefine \( u \) and/or \( v \) to be transshipment nodes along these \( u-v \) paths until we can find a pair that satisfies all four conditions.

Let \( p_1 \) and \( p_2 \) be two different paths in \( P_{u-v} \) and suppose that \( \alpha > x_{p_1} \geq x_{p_2} > 0 \) and \( x_{p_1} + x_{p_2} = \bar{\alpha} \leq \alpha \). Then we can represent \( y^* \) as a convex combination of two feasible solutions. The first, is the same as \( y^* \) except \( y_e = \bar{\alpha} \) for each \( e \in p_1 \) and \( y_e = 0 \) for each \( e \in p_2 \). The second, is the same as \( y^* \) except \( y_e = 0 \) for each \( e \in p_1 \) and \( y_e = \bar{\alpha} \) for each \( e \in p_2 \). This is a contradiction. \( \square \)

4 Computational Results

In this section, we describe our experiments and present our computational results.

For our data, we used the BTS’ T-100 data which offers aggregate segment data—that is, the number of passengers who flew from airport A to airport B for each operating carrier for each month—as well as aggregate market data—that is, the number of passengers who bought an itinerary that starts at airport A and ends at airport D for each operating carrier for each month\(^1\) [2].

We implemented our column generation code in Matlab, and conducted our experiments on a Dell laptop with 4 GB of RAM, 2.7 Ghz Intel processor, and 8 cores. We selected 25 airlines from the T-100 carrier list, and arbitrarily selected one month each for each carrier from 2014. We let our code run for up to 5 minutes per carrier. If the code did not find the optimal solution after 5 minutes, then we returned the best solution found so far.

Table 1 contains our results. The first column lists the operating carrier. The second column lists the number of passengers who flew on this operating carrier on this month, according to the T-100 data. The third column lists the numeric value of the 2014 month. The fourth column lists the percentage of passengers that were assigned to itineraries by our algorithm, rounded to two decimal places. The fifth column lists the number of passengers that were assigned to a direct flight. The sixth through ninth columns list the number of passengers who were assigned to an itinerary with 1 layover, 2 layovers, 3 layovers, and 4 layovers respectively. The last column lists the number of passengers who were assigned to flights with more than 4 layovers.

As the results show, the vast majority of passengers were assigned to direct flights. This seems right in part because many people tend to prefer direct flights when they are available. We

\(^1\) Technically speaking, the T-100 market data splits itineraries by flight number, and so even this data set has limited information on the origins and final destination pairs of passengers. So, for example, an individual who is flying from A to C via B would be recorded in one A to C row in the T-100 market data if his flight from A to B had the same flight number as his flight from B to C. But it would be recorded in two separate rows, one from A to B and one from B to C, if these flights had different flight numbers.
<table>
<thead>
<tr>
<th>Carrier</th>
<th>Passengers</th>
<th>Mo.</th>
<th>%</th>
<th>1 Leg</th>
<th>2 Legs</th>
<th>3 Legs</th>
<th>4 Legs</th>
<th>5 Legs</th>
<th>≥ 6 Legs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Air Canada®</td>
<td>2292240</td>
<td>2</td>
<td>100.00%</td>
<td>424178</td>
<td>1195</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Alaska®</td>
<td>10042510</td>
<td>3</td>
<td>100.00%</td>
<td>1698306</td>
<td>35775</td>
<td>7353</td>
<td>725</td>
<td>361</td>
<td>21</td>
</tr>
<tr>
<td>Allegiant®</td>
<td>4225351</td>
<td>3</td>
<td>100.00%</td>
<td>861275</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>American®</td>
<td>43820722</td>
<td>4</td>
<td>99.99%</td>
<td>7249532</td>
<td>30342</td>
<td>1485</td>
<td>907</td>
<td>196</td>
<td>0</td>
</tr>
<tr>
<td>Asiana®</td>
<td>696797</td>
<td>6</td>
<td>100.00%</td>
<td>123583</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>British Airways®</td>
<td>3329901</td>
<td>4</td>
<td>100.00%</td>
<td>589414</td>
<td>2082</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Chautauqua®</td>
<td>1892221</td>
<td>3</td>
<td>100.00%</td>
<td>405098</td>
<td>427</td>
<td>24</td>
<td>13</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Delta®</td>
<td>61706989</td>
<td>1</td>
<td>99.99%</td>
<td>8688606</td>
<td>53508</td>
<td>4272</td>
<td>1272</td>
<td>680</td>
<td>0</td>
</tr>
<tr>
<td>ExpressJet®</td>
<td>15440078</td>
<td>1</td>
<td>99.95%</td>
<td>2222190</td>
<td>2093</td>
<td>42</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Frontier®</td>
<td>5567220</td>
<td>6</td>
<td>100.00%</td>
<td>1108583</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Hawaiian®</td>
<td>4892791</td>
<td>1</td>
<td>99.81%</td>
<td>821702</td>
<td>629</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>JetBlue®</td>
<td>15511774</td>
<td>6</td>
<td>100.00%</td>
<td>2747577</td>
<td>5265</td>
<td>713</td>
<td>149</td>
<td>124</td>
<td>0</td>
</tr>
<tr>
<td>Korean Air®</td>
<td>1268592</td>
<td>5</td>
<td>100.00%</td>
<td>213769</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Lufthansa®</td>
<td>2520967</td>
<td>6</td>
<td>100.00%</td>
<td>546253</td>
<td>1790</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Mesa®</td>
<td>4252089</td>
<td>2</td>
<td>99.50%</td>
<td>612574</td>
<td>2579</td>
<td>22</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PSA®</td>
<td>2475934</td>
<td>1</td>
<td>99.99%</td>
<td>384997</td>
<td>1238</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Qantas®</td>
<td>595522</td>
<td>1</td>
<td>100.00%</td>
<td>93746</td>
<td>12240</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Republic</td>
<td>6191890</td>
<td>5</td>
<td>100.00%</td>
<td>1162679</td>
<td>8628</td>
<td>451</td>
<td>22</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>SkyWest®</td>
<td>13429789</td>
<td>2</td>
<td>100.00%</td>
<td>1936426</td>
<td>3850</td>
<td>115</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Southwest®</td>
<td>60969264</td>
<td>2</td>
<td>99.90%</td>
<td>7989841</td>
<td>332286</td>
<td>42239</td>
<td>8384</td>
<td>3783</td>
<td>0</td>
</tr>
<tr>
<td>Spirit®</td>
<td>6680062</td>
<td>4</td>
<td>99.99%</td>
<td>1086358</td>
<td>16121</td>
<td>2690</td>
<td>493</td>
<td>143</td>
<td>0</td>
</tr>
<tr>
<td>United®</td>
<td>44277734</td>
<td>3</td>
<td>99.99%</td>
<td>7811115</td>
<td>54054</td>
<td>2995</td>
<td>1093</td>
<td>598</td>
<td>0</td>
</tr>
<tr>
<td>US Airways®</td>
<td>28914858</td>
<td>5</td>
<td>99.97%</td>
<td>5057496</td>
<td>51941</td>
<td>4992</td>
<td>1952</td>
<td>545</td>
<td>0</td>
</tr>
<tr>
<td>Virgin America®</td>
<td>3209000</td>
<td>5</td>
<td>100.00%</td>
<td>576595</td>
<td>600</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Virgin Atlantic®</td>
<td>1452976</td>
<td>4</td>
<td>100.00%</td>
<td>276370</td>
<td>1586</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 1: Sample results on T-100 data

also expect this algorithm to find more direct flights than probably actually took place because of the nature of the T-100 data. As stated in a previous footnote, the T-100 data would split up a single itinerary into multiple market orders, if the itinerary consisted of more than one flight number. This means, for example, that an individual who is flying on Southwest® from SNA to DCA via PHX would be recorded in one SNA to DCA row in the T-100 market data if his flight from SNA to PHX had the same flight number as his flight from PHX to DCA. But it would be recorded in two separate rows, one from SNA to PHX and one from PHX to DCA, if these flights had different flight numbers.

Some of the results in which less than 100% of the passengers are because the algorithm was terminated prematurely after 5 minutes, instead of being allowed to be run to completion. For example, the algorithm assigns 99.97% of the passengers in the Southwest® instance (rather than 99.90%) if it is allowed to run for 10 minutes instead of 5. Sometimes, however, the less-than-100% results are explained due to inconsistencies in the data. This was the case in the Hawaiian Airlines® instance, in which the number of passengers leaving some airports in the market data was greater than the number of passengers leaving that same airport according to the segment data.

The needs of our project enabled us to take a shortcut with regards to international layovers. Even though we don’t have access to international-to-international segment data, we didn’t
need to know the exact international part of any itinerary, only that it terminated at an international airport. Hence, we assumed that any passenger at an international airport can reconnect to any other international airport. If needed, this assumption can be restricted to a more specific network of realistic international layovers.

5 Conclusions and Future Work

We presented a practical method for reconstructing complete passenger itineraries using segment and origin-destination data. We also showed how our methods could estimate complete airline passenger itineraries for U.S. carriers of various sizes using the publicly available T-100 data published by the Bureau of Transportation Statistics [2]. But there are still opportunities to build upon this research.

On reconstructing airline itineraries in particular, one challenge is that the T-100 data currently records market orders that were split according to flight number, meaning that a single itinerary involving two separate flight numbers would appear as two itineraries in the T-100 market data. It would be useful to identify how to accurately piece together these itineraries. One option is to try to use the Bureau of Transportation Statistics’ Origin and Destination Survey, which is also known as the DB1B data [1]. This data splits itineraries based on “flight breaks,” essentially sufficiently long lapses in time between flight segments, rather than flight numbers. However, one complication is this data only presents 10% of the total volume of passengers, and the manner in which the subset is selected is not publicly known. (It certainly does not appear to be drawn uniformly at random.)

Another research opportunity is to identify a way to assure the quality of the itineraries generated by this method, given data on the actual itineraries is not publicly available. Furthermore, there may also be opportunities in identifying better standards for guessing whether one set of itineraries is more realistic than another. We used squared, path-based costs but perhaps there is a better metric.

More broadly, there are likely opportunities to research algorithms for network optimization problems in which the cost of a path cannot be decomposed into the sum of the costs of the edges composing the path. Another context in which this may arise is aircraft routing. (See Royset, Carlyle and Wood [9] for an example of an aircraft routing problem.) If the goal is to find the most fuel efficient path from A to B, the fuel consumption along each arc $(u, v)$ in the path is a function of the weight of the aircraft, and the weight of the aircraft is a function of how much fuel remains in the aircraft, and the remaining fuel is a function of how the aircraft first arrived at node $u$. One heuristic method for addressing this issue is to keep track of how much fuel has been consumed thus far during a multi-label, Dijkstra-like algorithm, but it might be worthwhile to investigate if this issue can be more accurately addressed by modeling certain aircraft routing problems as network optimization problems with path-based costs.

6 Acknowledgements

We thank our sponsor, who funded the project in which this was one part of our analysis. We also thank the Operations Research Department at the MITRE Corporation, which partially
supported the writing of this article through professional development funding. Finally, we thank our colleague Justin Nave, who originally came to us with this research question.

References


