

A RECURSION-BASED APPROACH TO SIMULATING AIRLINE SCHEDULE ROBUSTNESS

Marcial Lapp
Shervin AhmadBeygi
Amy Cohn
Omer Tsimhoni

Dept. of Industrial & Operations Engineering
1205 Beal Avenue, University of Michigan
Ann Arbor, MI 48109-2117, USA

ABSTRACT

Flight disruptions due to events such as inclement weather or mechanical failure are an increasing occurrence in today's air travel. It is important to develop flight schedules that are not only economically feasible, but also provide opportunities to absorb these disruptions so as to reduce downstream delays. In this paper, we present a simulation algorithm to evaluate a flight schedule's ability to mitigate disruptions by analyzing propagation effects on the flight network. This task is challenging for two reasons: the interdependence of flights, due to shared resources (e.g. cockpit/flight crews, aircraft), and the cyclic nature of the schedule, which repeats on a daily basis. We show how a recursion-based approach to the simulation enables us to overcome these challenges.

1 INTRODUCTION AND MOTIVATION

Flight delays are an increasingly common occurrence in today's air travel industry. The Air Transport Association estimates that there were a total of 134 million system delay minutes during 2007, which amounted to a \$8.1 billion increase in direct operating costs to U.S. airline carriers (Air Transport Association 2008). In addition, the U.S. Department of Transportation reports only 78.5% of flights as on time from 2003 until 2005, a number that decreased to 75.4% during 2006 and shrank even further to 73.4% during 2007 (U.S. Department of Transportation 2008). Many of these delays are what we refer to as root delays -- delays stemming from events which are intrinsic to the particular flight (e.g. a mechanical failure or weather delay). These delays are predominantly independent of the flight schedule. However, there is also a significant impact on the system stemming from propagated delays. These are delays passed on from one flight to its subsequent connections, which await its crew, aircraft, or connecting passengers. Clearly, these delays can

be impacted by the schedule, as the schedule determines the flight connections, as well as the slack between flights that can be used to absorb disruption.

In this paper, we present a recursive algorithm that simulates a daily airline schedule by generating root delays and measuring their propagation to downstream flights. Specifically, we are interested in the metric of total propagated delay minutes, which is the cumulative number of minutes that flights will be delayed due to the propagation of root delays. This metric is of great interest to an airline, as it provides them with a way to compare the expected operational performance of two different flight schedules.

2 RELATED LITERATURE

Simulation is frequently used in airline applications as a way to overcome network complexity and the impacts of stochasticity, both of which can make closed-form solutions intractable.

Barlow et al. (1997) presents an algorithm for simulating a flight reservation system. Specifically, the presented algorithm allows an airline to analyze how changes in flight schedule will alter demand, and eventually affect overall revenue.

Belobaba and Wilson (1997) present a simulation to determine the effects of yield management, the controlled allocation of seats (between business and leisure travelers) and their respective prices on high-demand flights.

Along the lines of flight schedule simulation, in Rosenberger et al. (2000) the authors introduce SIMAIR, a modular airline simulation built on three sub-components: a controller module, a recovery module and an event generator module. This airline simulation software allows for the evaluation of schedules and possible recovery scenarios with a variety of performance measures, the main focus of their investigation.

In Lee et al. (2003), the authors present further work in flight schedule simulation by extending SIMAIR. The authors develop a module that allows for the importation of different recovery scenarios based on what a specific airline carrier is capable of performing.

Our approach extends this literature by using simulation to assess the robustness of a fixed schedule. This is an important step towards improving the link between planning and operations.

3 KEY CHALLENGES

This section provides an introduction to flight schedule simulation, a technique used to evaluate a given flight schedule under various performance metrics. For example, network planners may be interested in analyzing a schedule’s performance under heavy delay, present during severe weather conditions.

Specifically, this section uncovers the challenges associated with such flight schedule simulations, and exposes why a simple first-in, first-out simulation (FIFO) fails to capture an accurate measure of the total amount of propagated delay that occurs in a flight network.

3.1 Flight Network Properties

Flight schedules feature intricate relationships among aircraft, crew, and possibly other resources. These relationships can be expressed as out-trees, as seen in Figure 1. In this example, we notice that the aircraft from Flight #515 proceeds to Flight #234, while the cockpit crew connects to Flight #562. Both of these flights subsequently influence additional flights as well. A successful simulation of such a flight schedule must take into account any effects that a flight can have on *all* flights in its corresponding out-tree.

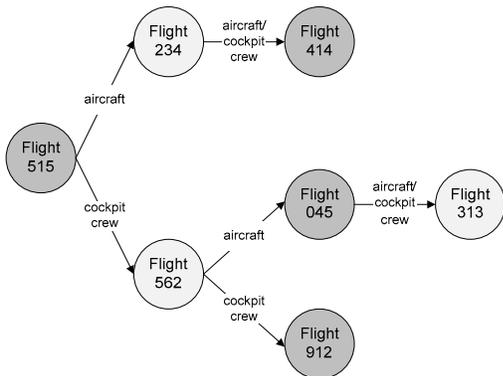


Figure 1: Outbound Connecting Flight Effects

In addition, flight relationships can also be represented as in-trees, where resources from different flights merge. This occurrence is depicted in Figure 2. In this ex-

ample, we notice that the aircraft from Flight #623 combines with the cockpit crew from Flight #1262 to conduct Flight #763. In this case, when a simulation is performed, all possible effects from incoming flights must be taken into consideration.

Because flight schedules contain both in- and out-trees, the network cannot be defined as a pure tree structure.

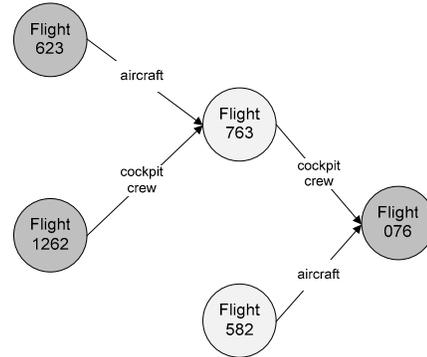


Figure 2: Inbound Resource Effects

3.2 Flight Schedule Time Issues

In addition to the intricate propagation relationships presented in the previous section, flight schedules also exhibit complicating timing properties that do not allow for first-in, first-out simulation approaches. Specifically, flight schedules repeat daily, making the schedule circular rather than linear. Thus, there is no “first” flight, and *any* flight in the day’s schedule can be delayed by upstream flights, and also cause delay to downstream flights.

Consider the example in Figure 3. A first-in, first-out algorithm would simulate this schedule in the order of departure time. It would start by simulating Flight #466 and propagating its effects to its connection, Flight #098. Next, Flight #098 would be evaluated and again, the respective propagation effects measured. Finally, after simulating Flight #231 (at the end of the day), the algorithm would terminate. However, in some cases this termination would be premature, because Flight #231 has the potential to propagate delay back to Flight #466.

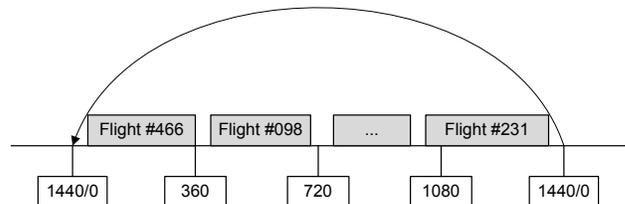


Figure 3: Recurring Schedule Example

As seen from this example, using an iterative, first-in, first-out simulation methodology when comparing two or more flight schedules is ineffective, since the concept of a particular “first flight” in a flight schedule does not exist.

Furthermore, Figure 3 depicts only a simple example, with two resources remaining paired throughout the day. As presented in Figure 2, a flight may be preceded by two distinct flights, leading to further difficulty during the propagation time calculation.

A successful simulation must take into consideration the fact that there is no predetermined start and end for a given simulation, such that when a flight is simulated, it can potentially suffer from incoming delays, or propagate delays to flights occurring on another day. In addition, the simulation routine must be able to accurately accrue the amount of total propagated delay from distinct incoming flights.

4 RECURSION-BASED APPROACH

Prior to presenting our simulation algorithm, we describe several assumptions made during the creation of this model.

A1 – We only consider root delays of 60 minutes or less in duration. Although longer root delays certainly occur (e.g. when a major storm impacts an airport, or an aircraft becomes unusable to a malfunction), such delays are typically not appropriate to plan for in the scheduling process.

A2 - Our probability distributions, which were derived from actual carrier data (see Section 4.1), are premised on the assumption that root delays are independent and identically distributed. Although root delays certainly do show some correlation effects (e.g. flights originating at a common airport during a common time period will experience the same inclement weather), we did not have adequate data to include this in our probability distributions. Our approach however, allows for the substitution of other probability distributions, including those exhibiting correlations.

A3 - We assume that all delays continue to propagate until they are fully absorbed by slack in the network. In fact, this is not always the case, and recovery options such as canceling flights or swapping aircraft can also be used to absorb delay. Such recovery decisions are difficult to replicate, however, as they are typically made by individual operations controllers and thus vary from person to person. Furthermore, many of these recovery options are not appropriate for delays of lesser magnitude and instead are used for higher-impact delays.

A4 – Our recursive algorithm requires that the total amount of propagated delay does not exceed 24 hours. This is a reasonable assumption given that root delays do not exceed 60 minutes, and that there are typically downtimes throughout the schedule (especially overnight) that

can absorb residual delays. This assumption held true in all data sets that we were provided.

A5 – When a flight incurs separate propagated delays, both from its incoming aircraft and incoming crew, we assume the propagated delay is the maximum of the two rather than their sum. In other words, as soon as both resources are available, the flight can depart. However, we then *add* any root delay in addition to this root delay. For example, a mechanical problem would not be observed until the aircraft was available.

4.1 Generating Delay Data

In developing the delay generation unit of our model that is used to simulate randomness in our flight schedules, we follow a similar approach to Rosenberger et. al. (2000), who use an Event Generator Module that implements a probability distribution to generate random events/delays in a network.

As previously mentioned, our probability distributions are based on actual U.S. carrier data. Initially, this data set included different types of delays, such as mechanical failure, weather, delays due to late incoming aircraft, etc. We filtered this data so as to omit all types of delay that indicate propagation, leaving us with only root delay data.

Using the available, filtered data, we attempted to fit various distributions for each of the origin airports that appeared in the flight schedule. However, since deviations from this set of possible distributions tended to exceed statistical thresholds, we reverted to empirical distributions instead. The delays are separated into bins of 0, 10, 20, 30, 40, 50, and 60 minutes of delay, each bin with a respective probability of occurring.

In developing these empirical distributions, we assume that the origin airport is responsible for a root delay of a given flight. Arguably, there are situations in which the destination airport is the cause of a late departure or possibly a mid-flight weather problem causes a late arrival. Nonetheless, given the fact that most of the delays of inclement weather/mechanism failure occur prior to take-off, we feel that this assumption is valid, but could benefit from further research. Again, as mentioned previously, our simulation algorithm does not depend on the probability distributions.

4.2 Algorithm Design

Before presenting the actual algorithm, we define several terms commonly used in the airline industry. *Minimum required turn time* is the amount of time deemed necessary to “turn around” an aircraft or crew before their next flight. This may include the time required to clean the aircraft, or for the pilot to perform inspection. *Block time* refers to the difference in time from the point at which an

aircraft has departed from its origin gate until its arrival at its destination gate.

In order to overcome the problems associated with the first-in, first-out simulation model, we implement a recursive algorithm. This algorithm explores all possible flight connections until the total amount of propagated delay has dissipated. We encourage the reader to follow along with the diagram presented in Figure 4 that visually explains the simulation algorithm.

We begin by adding all flights in the schedule to our future event list, in arbitrary order, initializing each flight's propagated delay and root delay to zero. We then do the following:

We select the next flight in the future event list to process and call upon the random delay generator to provide a root delay for this particular flight. If this root delay is non-zero, we update the departure time of the current flight to be the sum of the (current value of the) propagated delay and this additional root delay. We then consider all outbound connections from this particular

flight. For each connection, we determine how much (if any) the current flight's delay would propagate to this child by subtracting the existing slack from the current delay value. If this residual delay is non-negative, we then check whether it is larger than the connecting flight's current propagated delay.

If it is, we replace the connecting flight's propagated delay with this new, larger value. Next, we add this to the connecting flight's root delay (which would be 0 if that flight hadn't been processed yet) to determine its new departure time. Finally, we recursively examine the current flight's children to evaluate the impact of the residual delay on these flights, repeating until the flight delay is fully absorbed.

Once the algorithm has explored all flights in the out-tree of the root flight, it moves to the next flight in the future event list and the process repeats.

It should be noted that some flights are updated more than once in our algorithm. This is essential in accurately determining the amount of delay that could be propagat-

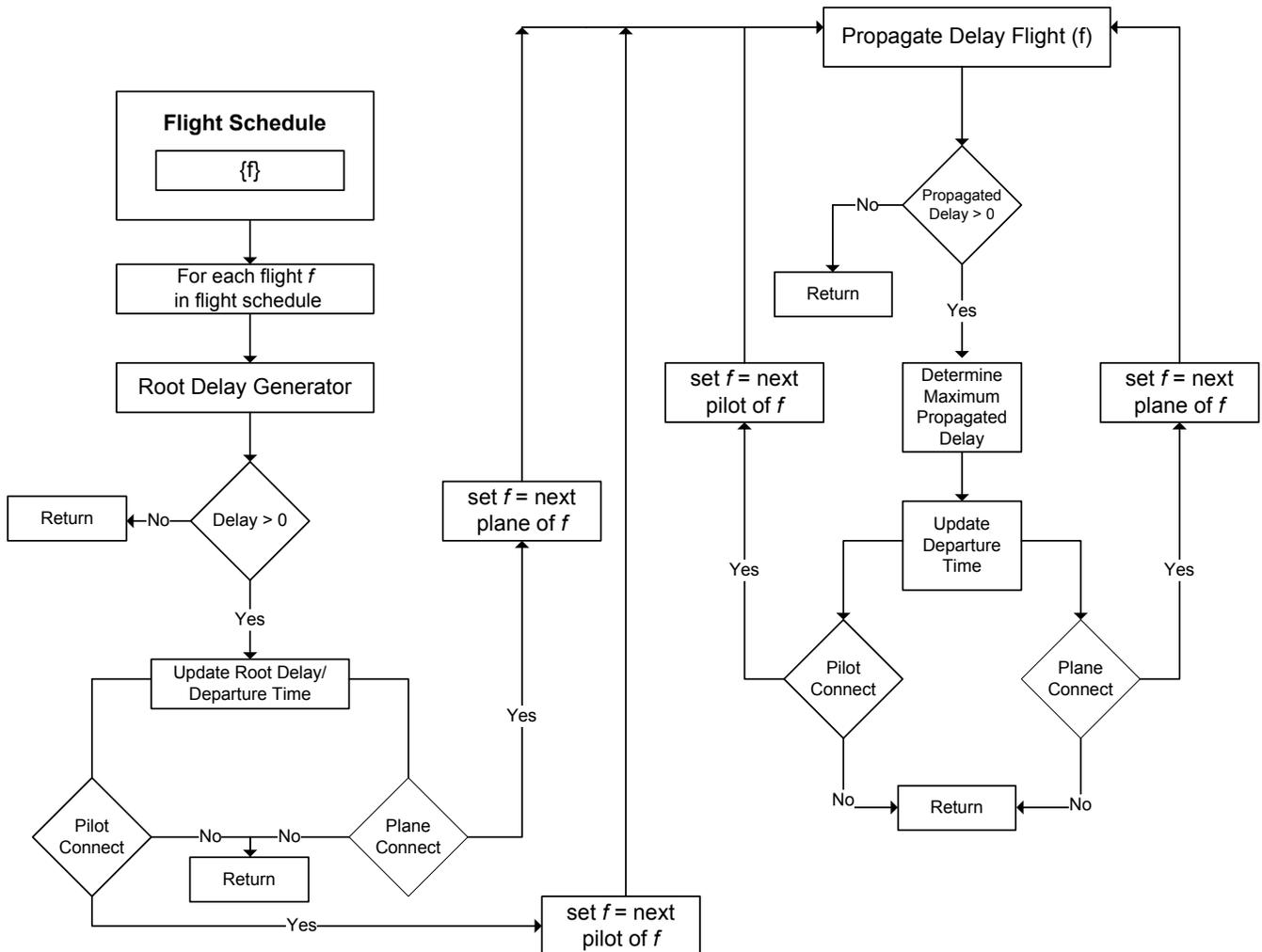


Figure 4: Simulation Order of Execution

ing from two different parent flights. In addition, each flight in the schedule will be exposed to a possible root delay from the our random delay generator. This delay will be added to any propagated delay that has already been simulated and downstream flights will be updated as well.

4.2.1 Simulation Example

This algorithm, due to its recursive nature, overcomes the challenges presented in Section 3, and accurately models the propagated delay in the flight network, even when multiple parent flights propagate flights to the same child node. Consider the following example in Table 1.

Table 1: Multiple Parent Flights Result in Additional Delay

Flight #	Root Delay	Depart. Time	Block Time	Pilot Connect	Aircraft Connect
752	30	055	135	823	641
214	50	124	062	Off	823
⋮	⋮	⋮	⋮	⋮	⋮
823	20	240	090	024	024

Our algorithm starts by simulating Flight #752, which incurs a root delay of 30 minutes. This means that the connecting flights (Flight #823 and #641) would have the earliest possibility of departing at time: 255 (55 (Departure) + 30 (Delay) + 135 (Block) + 35 (Turn Time) = 255). We can see that Flight #823 was supposed to leave at time 240, so at this point, we have incurred a propagated delay of 15 minutes ($255-240$).

Suppose, we now simulate Flight #214, which also incurs a root delay, however, it is a 50 minute delay. This results in Flight #823's earliest departure time as 271. (124 (Departure) + 50 (Delay) + 62 (Block) + 35 (Turn Time) = 271). We know Flight #823 has already been delayed until 255 from Flight #752; however, due to Flight #214, the departure time of Flight #823 has to be pushed back further, to 271 in this case.

This example shows that when multiple parent flights interact, the departure time of a subsequent flight may be impacted by (n)either/both parents.

4.3 Model Implementation

The simulation algorithm is implemented according to the algorithm and diagram provided in Figure 4. We use the C++ programming language for its ability to easily store and create linked lists (flight schedules) and speed to implement the actual simulation.

If we consider flight schedules with their connecting flights to be represented as out-trees, we determine that the depth of these trees rarely exceed four levels in our given data set. With this information, we are able to run a large number of replications in a matter of seconds; more information about the actual computational results is presented in our Results Section.

4.4 Model Verification

As part of the simulation process, we verified our model to ensure that it is indeed measuring our objective function of total propagated network delay. Our verification process consisted of both white-box and black-box testing, which will be discussed in this section.

As a first attempt at the verification process, our simulation code was tested at the functional level. Using a generated set of input values for which the output had been pre-computed, the results from function calls were verified.

To complete functional verification, we created several complex flight schedule scenarios to ensure that our model functions correctly under all situations.

4.5 Model Validation

To validate the results of our simulation, we followed advice presented in Sargent (2007). As suggested, we use a technique referred to as "comparing to other models." As part of another project, we had access to expected values of propagated delay given a pre-computed delay occurrence. These data values were in the form of: Given the fact that Flight 1 suffered a 60 minute delay, there exists a total of 35 minutes of propagated delay in the network.

While this model only considered one delay at a time, it allowed us to compare the expected values to our simulation. Matching up these values allowed us to further conclude that our simulation was indeed functioning correctly given our set of assumptions.

5 COMPUTATIONAL EXPERIMENTS

AhmadBeygi et al. (2008) presents a linear program to reallocate extra minutes (slack time) that exist between flights. Theoretically this algorithm creates a flight schedule that intrinsically is able to deal with delays by reallocating slack time to those flight connections that are most likely to experience delay. Using our simulation algorithm, we are able to verify the results of the presented model and conclude it is indeed a better schedule due to a reduced amount of propagated delay.

The linear program provides for solutions at different layers of delay propagation. The Single-Layer Model (SLM) restricts itself to only consider the propagation of delay from a parent flight to immediate possible connec-

tions, while the Multi-Layer Model (MLM) considers all layers of propagation. Using origin-based probabilities, the program computes a new schedule which should be less sensitive to delays occurring in the flight network.

Both the Single-Layer and Multi-Layer Models presented in AhmadBeygi et al. (2008) use a surrogate objective function to approximate the amount of delay propagated by the flights in the network. In the case where two different inbound flights both affect a common connecting flight, this model provides an over-estimate of the amount of propagated delay that is incurred by this new schedule. Conversely, in the case where two connecting flights both incur a root delay, this model provides an under-estimate of their impact on subsequent downstream flights. Our simulation model provides us with a means to assess the significance of these errors.

Using an implementation of this linear program, we were able to generate several schedules based on two original input schedules (data set 1 and data set 2). For each of the respective data sets we generated four new schedules representing various levels of flexibility. We use our simulation algorithm to determine the robustness of such a generated schedule versus the original flight schedule. The results from this simulation are presented in Table 2, table which shows 95% confidence intervals for the amount of reduction experienced in the Single-Layer and Multi-Layer Scheduling model when compared to the original schedule. Each confidence interval was generated from 1000 replications for each data set/duty restriction/layer type combination.

Table 2: 95% Confidence Intervals for Delay Reduction

Data Set	Duty Rest.	Single-Layer Model Schedule	Multi-Layer Model Schedule
1	0	(3.2%, 5.4%)	(4.4%, 6.6%)
1	5	(21.8%, 23.9%)	(23.4%, 25.5%)
1	10	(35.6%, 37.5%)	(37.5%, 39.2%)
1	15	(46.0%, 47.6%)	(47.8%, 49.3%)
2	0	(2.6%, 5.0%)	(3.5%, 5.9%)
2	5	(22.7%, 24.8%)	(24.3%, 26.3%)
2	10	(37.0%, 38.8%)	(38.9%, 40.6%)
2	15	(47.9%, 49.5%)	(50.1%, 51.7%)

As evident from the above-seen values, we are able to conclude that schedules produced by Single-Layer and Multi-Layer Scheduling models are more robust than their respective original schedules in terms of their ability to absorb randomly occurring network delay.

Even though our simulation algorithm is recursive in nature, it still performs well for a large number of replications. Simulating a schedule with more than 500 flights for 1000 replications takes less than 5 seconds to run on a 2.2GHz Intel machine. It is our conjecture that our ap-

proach will remain tractable even for larger flight networks with deeper propagation trees.

In addition, in an effort to reduce the variance and the overall number of replications required, we use common random numbers when simulating both the original and enhanced flight schedules. This practice allows us to construct a paired-t test for which we check whether the confidence interval of the difference between two schedules includes zero. If it is *not* included, we can conclude that with some alpha-confidence that one schedule is indeed more robust than another.

6 CONCLUSIONS AND FUTURE WORK

The presented algorithm successfully simulates a flight schedule, incorporating the relationships between inbound and outbound flight connections, as well as the cyclic nature of the daily repeating schedule. Our recursive approach addresses these challenges while maintaining tractability. This enables us to provide an accurate estimate of the total propagated delay minutes, a measure that can be used to draw conclusions about the robustness of a given flight schedule and, in particular, its ability to deal with common, everyday delays.

6.1 Future Work

As presented, our algorithm works well in estimating the total minutes of propagated delay given a particular flight schedule; however it does not take into consideration the use of recovery operations. While it is often difficult to provide accurate costs for each of the possible recovery operation and the fact that such operation decisions are usually made ad-hoc as Rosenberger et. al (2000) notes, it may be worth exploring the use of scenarios. Such an extension would then serve to explore the comparison between two schedules, subjecting them to the same delay with an identical set of possible recovery operations. This would provide a more accurate depiction of actual flight schedules, since recovery operations are frequently implemented in practice.

In addition, it may be worth incorporating passenger connections as a measure of robustness. While it may be the case that one schedule may be significantly better than another when it comes to minimizing the total amount of propagated delay, this may not minimize the number of passengers missing a connection.

ACKNOWLEDGEMENTS

We gratefully acknowledge undergraduate assistance during the early development of this simulation. These undergraduates include: Kristina Behrens, Tina Dimoski, and Michael Krautmann. In addition, we would like to

acknowledge Jonathan Cohn and Caitlin Brown for their invaluable editorial comments.

REFERENCES

- AhmadBeygi, S., A. Cohn, and M. Lapp. 2008. Decreasing Airline Delay Propagation By Re-Allocating Scheduled Slack. Working paper.
- Air Transport Association. Costs of Delays. 2008. Available via <http://www.airlines.org/economics/specialtopics/ATC+Delay+Cost.htm> [accessed February 25, 2008].
- Barlow, J., T. Valdivieso, A. Risal, V. Shah, S. Parmekar, R. Perumal, S. Davies, and J. Peterson. 1997. Airline flight reservation system simulator for optimizing revenues. United States Patent 5652867.
- Belobaba, P. P., and J. Wilson. 1997. Impacts of yield management in competitive airline markets. *Journal of Air Transport Management*: 3-9.
- Lee, L. H., H. Huang, C. Lee, E. P. Chew, W. Jaruphongs, Y. Y. Yong, Z. Liang, C. H. Leong, Y. P. Tan, K. Namburi, E. Johnson, and J. Banks. 2003. Discrete Event Simulation Model for Airline Operations: SIMAIR. In *Proceedings of the 2003 Winter Simulation Conference*, ed. S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, 1656-1662.
- Rosenberger, J., A. Schaefer, D. Goldsman, E. L. Johnson, A. J. Kleywegt, and G. L. Nemhauser. 2000. SIMAIR: A Stochastic Model of Airline Operations. In *Proceedings of the 2000 Winter Simulation Conference*, ed. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, 1118-1122.
- Sargent, R. G. 2007. Verification and Validation of Simulation Models. In *Proceedings of the 2007 Winter Simulation Conference*, ed. S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 124-137.
- U.S. Department of Transportation, Bureau of Transportation Statistics. 2008. Airline On-Time Statistics and Delay Causes. Available via http://www.transtats.bts.gov/OT_Delay/OT_DelayCause1.asp?pn=1 [accessed February 22, 2008].

AUTHOR BIOGRAPHIES

SHERVIN AHMADBEYGI is a Ph.D. candidate in the Industrial and Operations Engineering Department at the University of Michigan. His main focus is using operations research methods to model and solve complex, large-scale transportation and logistics problems. He is specifically interested in studying the behavior of these systems under uncertainty.

AMY COHN is an Assistant Professor in the Industrial and Operations Engineering Department at the University of Michigan. She holds the Ph.D. in Operations Research from MIT. Her primary research interest is in applied discrete optimization problems, particularly in the airline industry.

MARCIAL LAPP is a graduate student in the Industrial and Operations Engineering Department at the University of Michigan. His research interests lie in modeling and solving large-scale optimization problems focused on the transportation and logistics industries. He holds a Masters and Bachelors degree in Computer Science from the University of Michigan.

OMER TSIMHONI is an Adjunct Assistant Professor in the Industrial and Operations Engineering Department at the University of Michigan, where he teaches simulation to undergraduate and graduate students. He is an Assistant Research Scientist at the University of Michigan Transportation Research Institute (UMTRI), where he studies the human factors of driving. His research interest is in the application of simulation and optimization tools to cognitive and physical modeling of human performance.