# Airline Scheduling and Air Traffic Control: Incorporating Uncertainty and Passenger and Airline Preferences

by

Chiwei Yan

B.S., Tsinghua University (2012)

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Operations Research
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
September 2017

© Massachusetts Institute of Technology 2017. All rights reserved.

Author . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Sloan School of Management
August 11, 2017

Certified by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Cynthia Barnhart
Chancellor and Ford Professor of Civil and Environmental Engineering
Thesis Supervisor

Certified by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Vikrant Vaze
Assistant Professor of Engineering
Thayer School of Engineering, Dartmouth College
Thesis Supervisor

Accepted by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Patrick Jaillet
Dugald C. Jackson Professor, Department of Electrical Engineering and Computer Science
Co-Director, Operations Research Center

# Airline Scheduling and Air Traffic Control: Incorporating Uncertainty and Passenger and Airline Preferences

by

Chiwei Yan

Submitted to the Sloan School of Management
on August 11, 2017, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Operations Research

## Abstract

The global airline industry is a multi-stakeholder stochastic system whose performance is the outcome of complex interactions between its multiple decisions-makers under a high degree of uncertainty. Inadequate understanding of uncertainty and stakeholder preferences leads to adverse effects including airline losses, delays and disruptions. This thesis studies a set of topics in airline scheduling and air traffic control to mitigate some of these issues.

The first part of the thesis focuses on building aircraft schedules that are robust against delays. We develop a robust optimization approach for building aircraft routes. The goal is to mitigate propagated delays, which are defined as the delays caused by the late arrival of aircraft from earlier flights and are the top cause of flight delays in the United States air transportation system. The key feature of our model is that it allows us to handle correlation in flight delays explicitly that existing approaches cannot handle efficiently. We propose an efficient decomposition algorithm to solve the robust model and present the results of numerical experiments, based on data from a major U.S. airline, to demonstrate its effectiveness compared to existing approaches.

The second part of the thesis focuses on improving the planning of air traffic flow management (ATFM) programs by incorporating airline preferences into the decision-making process. We develop a voting mechanism to gather airline preferences of candidate ATFM designs. A unique feature of this mechanism is that the candidates are drawn from a domain with infinite cardinality described by polyhedral sets. We conduct a detailed case study based on actual schedule data at San Francisco International Airport to assess its benefits in planning of ground delay programs.

Finally, we study an integrated airline network planning model which incorporates passenger choice behavior. We model passenger demand using a multinomial logit choice model and integrate it into a fleet assignment and schedule design model. To tackle the formidable computational challenge associated with solving this model, we develop a reformulation, decomposition and approximation scheme. Using data from a major U.S. airline, we prove that the proposed approach brings significant profit improvements over existing methods.

Thesis Supervisor: Cynthia Barnhart
Title: Chancellor and Ford Professor of Civil and Environmental Engineering

Thesis Supervisor: Vikrant Vaze
Title: Assistant Professor of Engineering
Thayer School of Engineering, Dartmouth College

# Acknowledgments

First and foremost, I am deeply indebted to my advisors, Professors Cynthia Barnhart and Vikrant Vaze. Their expertise, guidance, and advice have contributed greatly to this thesis, and I would not have come this far without their support. Cindy: I still remember the time I first met you when I visited MIT in summer 2011. It has been a great joy to learn from you and experience your enthusiasm, remarkable dedication and unbounded energy. I am deeply grateful for your constant encouragement and tremendous support during the past five years. You have been giving me the widest freedom to pursue my own research direction but at the same time providing me the most valuable advice and support. Your kindness and caring nature have made my journey at MIT such a pleasant and cherishable one. Vikrant: I am indebted to our numerous long discussions on research, professional development and personal life. You are such a supportive advisor that provides continuous feedback, tries your best to benefit my work and is willing to share your own experience to help me succeed. Both of you set up a very high standard, and I shall strive to emulate that as I embark on the next chapter of my professional career.

I would also like to thank my committee members, Professors Patrick Jaillet and James Orlin, for providing critical and constructive feedback on this research. Patrick: thank you for your support since my first year at MIT as a MST student and the transition to ORC. It was your 15.081 class that ignited my passion for doing a Ph.D. in OR. Jim: it was an amazing experience to serve two times as your TA for 15.053 and I learned a lot from your professionalism, passion for teaching and dedication to students. I shall carry it with me for any future teaching opportunities I may encounter.

Additionally, this thesis has benefited from discussions with a number of people. Chapter 2 is a result of collaboration with Jerry Kung during our course project in 15.094 taught by Professor Dimitris Bertsimas. Thanks also go to AGIFORS Anna Valicek Award committee for recognizing our work and giving us the opportunity to share it with the industry practitioners. Chapter 3 is a result of a collaboration with

Professor Michael Ball and Prem Swaroop from the University of Maryland during our Service Level Expectation (SLE) project. I would also like to thank the other SLE project colleagues including Professor Mark Hansen, Yi Liu and Lei Kang from the University of California, Berkeley. Chapter 5 is inspired from my internship at Sabre. I have benefited from discussions with Richard Ratliff, Sergey Shebalov, Xiaodong Luo, Weijun Ding, Xiaoqing Sun and Ross Darrow. Special thanks to Xiaodong Luo for providing me the testing instances.

I am very grateful to Maria Marangiello, Shannon McCord, Laura Rose and Andrew Carvalho for their administrative assistance. This thesis has been financially supported by the Robert E. Thurber Fellowship from the Department of Civil and Environmental Engineering at MIT, Federal Aviation Administration through NEXTOR II and the UPS Doctoral Fellowship from MIT Center for Transportation and Logistics.

Many thanks go to my friends at ORC, MST and MIT. I have been extremely lucky to be surrounded by a group of peers who are individually brilliant, yet exceptionally supportive of each other. In particular, thank you to Chong Yang Goh, Charles Thraves, Velibor Misic, Jerry Kung, Louis Chen, Michael Beeler, Miles Lubin, Siong Thye Goh, Rajan Udwani, Alex Weinstein, Nikita Korolko, Hai Wang, Rong Yuan, He Wang, Maokai Lin, Peng Shi, Virot Chiraphadhanakul, Alex Jacquillat, He Sun, Runmin Xu, Tianli Zhou, Chao Zhang, Li Jin, Linsen Chong, Zhan Zhao and Haizheng Zhang.

I am also indebted to all teachers in my life, without whom I would not be where I am today. Especially, my appreciation goes to Professor Hai Jiang from Tsinghua University for stimulating my interest in doing research and supporting me in coming to MIT.

Last but not least, I want to thank my family for their love and confidence in my ability to succeed. Thanks especially to my wife Yiwen Zhu for always being there to support me and for reminding me that there is more to life than research. It is to her and my family that I dedicate this thesis.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Motivation

Over the last 60 years, operations research has played a critical role in the airline industry (Barnhart et al., 2003). Classic models have been developed in almost all facets of the industry ranging from scheduling, routing and crew assignment on the airline operations side, to revenue management on the airline business and marketing side, to planning and management of airport systems and air traffic flow management on the aviation infrastructure side. Yet, airlines around the globe continue to face various major challenges:

1. Airlines continue to suffer from significant flight delays. In year 2016, almost one out of five flights operated by major U.S. carriers arrived more than 15 minutes late (Bureau of Transportation Statistics, 2016a). At major airports in China, which is one of the fastest growing air transportation markets, only around half of the flights typically arrive on-time (BBC, 2016). According to the Bureau of Transportation Statistics (2016a), the top three causes in order of flight delays in the U.S. are (1) late arrival of aircraft, (2) national aviation system delays largely due to air traffic flow imbalances, and (3) air carrier delays such as those due to issues associated with aircraft maintenance, crew availability, aircraft cleaning, baggage loading, etc. Flight delays are costly.

Ball et al. (2010) estimated that the total cost to the economy of all US air transportation delays in 2007 was $32.9 billion.

2. Airlines often operate under thin and volatile profit margins. The Economist (2014) reported a less than 1% industry average profit in 2012, though there has been a recent surge to 4.1% profits in 2016 according to IATA (2017). Airline profit is impacted by many factors including fuel price, travel demand and competition. On the other hand, aircraft ownership and lease entail large fixed costs, and airlines often need to continue making large lease payments or loan repayments regardless of their business conditions. These characteristics of the industry are often considered to be the reasons for its frequent and large losses.

In this thesis, we address some of these challenges by proposing new airline planning models and operating paradigms that improve upon existing methods:

1. Classic airline scheduling models and most models currently implemented in the industry assume data to be deterministic and that flights will depart and arrive on time as planned. This usually creates schedules that are prone to delays and disruptions. We study how to model *uncertainty* and produce *robust* schedules efficiently and effectively;

2. Existing models for planning air traffic flow management programs are centralized, that is, the central authority makes most of the design decisions. We propose an approach to solicit and incorporate *airline preferences* when planning air traffic flow management programs so that airlines' business objectives are better supported and system-wide delay costs are reduced.

3. Most of the current airline network planning models such as fleet assignment and schedule design make simplified assumptions on the structure of passenger demand. We present models and computational approaches to incorporate *passenger preferences* into classic airline network planning models. Through better

modeling of demand for airline fare products and efficient solution approaches, significant profit improvement is obtained.

We summarize the detailed contributions and the structure of the thesis in the following section.

## 1.2 Contribution and Thesis Outline

In Chapter 2, we study an aircraft routing problem to minimize *propagated delay* – the delay caused by the late arrival of aircraft from upstream flights and the main cause of flight delays in the United States. An aircraft route is a sequence of flight legs consecutively flown by the same aircraft. Aircraft routes impact flight delay propagation because, if two consecutive flights with very short connection time between them are flown by the same aircraft, then even a small delay to the first flight will delay (that is, propagate to) the second flight. We propose an aircraft routing model and solution approach based on *robust optimization* that produces aircraft routes less susceptible to delay propagation. The model is data-driven and builds routes that protect against future delays by recognizing the stochasticity of historical flight delays.

To solve the robust model efficiently, we develop a reformulation and a solution algorithm based on simultaneous row and column generation. The key differentiating feature of our model from existing approaches is that we are able to explicitly model and efficiently handle *correlation* in flight leg delays. Note that strong correlation exists in flight delays due to weather and air traffic control. Using both historical delay data and simulated data based on the flight schedule of a major U.S. airline, we evaluate the performance of our model and benchmark against a state-of-the-research stochastic approach. In most cases, we observe that our model outperforms the existing approach by lowering the mean, reducing volatility, and mitigating extreme values of total propagated delay.

In Chapters 3 and 4, we consider the air traffic flow management (ATFM) problem. In particular, we focus on reducing system-wide delay costs through better planning of various ATFM programs. In contrast to existing models focusing on optimizing certain system performance measures from a centralized and system planner's point of view, our proposed paradigms tackle this problem in a *decentralized* way. This approach asks airlines to provide inputs for the design of ATFM and then translate these inputs into a consensus decision. A decentralized approach has a number of benefits including better support of airlines' business objectives, which in turn, leads to reduced system-wide delay costs. It is also expected to produce more equitable ATFM programs in distributing delay costs among involved airlines by explicitly considering each airline's preferences, which could be extremely diverse in practice.

To achieve this goal, in Chapter 3, we first focus on theoretical work that develops a voting mechanism to solicit airlines' preferences and translate them into a consensus ATFM program design. Our proposed voting mechanism is based on a recently developed voting theory called *Majority Judgment* (Balinski and Laraki, 2010). The unique feature of our problem that existing voting applications do not have is that airlines vote over candidate ATFM program designs that are described through a set of performance metrics or design parameters with continuous values from a domain with infinite cardinality. We thus develop theoretical extensions to apply Majority Judgment over candidate spaces described by polyhedral sets. In Chapter 4, we present a follow-up case study to assess the benefits of applying our decentralized approach to the design of ground delay programs (a very common variety of ATFM programs) based on actual schedule data at San Francisco International Airport. The case study confirms that the ground delay program (GDP) designed by the proposed decentralized approach successfully reduces system-wide delay costs and distributes those costs more equitably among airlines compared to the GDP designed by the centralized approach.

Chapter 5 focuses on improving airline profit through better network planning. In particular, we look at how to incorporate passenger demand substitution behaviors efficiently into traditional airline network planning models. We model passenger demand using a multinomial logit discrete choice model and integrate it with a schedule design and fleet assignment model that simultaneously assigns fleet types to flights and selects from optional flights to fly. This integrated model is notoriously hard to solve for real-world instances. In order to solve it efficiently and effectively, we develop a reformulation, decomposition and approximation scheme for the model. Using data from two major U.S. airlines, we present computational experiments that confirm the effectiveness of our solution approach and demonstrate significant profit improvement over comparable methods.

Finally, we conclude this thesis in Chapter 6.

# Chapter 2

# Robust Aircraft Routing

## 2.1 Introduction

Airline delays are prevalent and costly. In 2016, almost one out of every five flight legs operated by major U.S. carriers arrived more than 15 minutes late; of these delayed flights, one-third were a result of propagated delay: the late arrival of an aircraft causing a late departure (Bureau of Transportation Statistics, 2016a). One of the key drivers for these propagated delays is that airlines have in the past employed optimization models in an attempt to maximize profit; this has often led to the creation of tight schedules in an attempt to increase aircraft utilization and decrease crew salaries for time on the ground (Klabjan et al., 2001). As a consequence, delays propagate very rapidly throughout the network. This in turn leads to significant costs for an airline, such as additional pay for the crew, customer dissatisfaction, and further decreased utilization of aircraft due to flight cancellation.

To mitigate the discrepancy between what is planned and what actually happens, the research community has developed robust planning methods to pro-actively consider such delays and disruptions. These methods create schedules with the objective of constructing plans that have one of two kinds of robustness built in: (1) ease of repair once disrupted or (2) diminished propensity of delay disturbances.

Within the first category, Kang (2004) partition flight schedules into independent layers so that delay or disruption in one layer will not affect the others. Each layer is assigned priority based on its expected revenue. Flights in higher layers are protected most, whereas those in lower layers are more likely to experience delay or cancellation. This approach significantly reduces delay and disruption costs. Rosenberger et al. (2004) develop a string-based fleet assignment model that embeds many short cancellation cycles and limits the number of aircraft that can serve each hub. Such characteristics prevent the cancellation of a series of flights and isolate the disruption to a particular hub. Smith and Johnson (2006) introduce the concept of “station purity”, which limits the number of fleet types that can serve each airport. This creates additional aircraft swapping opportunities. Shebalov and Klabjan (2006) propose a robust crew pairing model that maximizes the number of crews that can potentially be swapped in operations. Yen and Birge (2006) develop a two-stage stochastic integer program with recourse for the crew scheduling problem. Their model produces a robust crew schedule by incorporating disruptions in the evaluation of crew schedules during the long-range planning phase. Recently, Froyland et al. (2013) present a recoverable robust aircraft routing model as a two-stage stochastic program. They explicitly model the aircraft routing decision in the first stage, as well as a full set of recovery operations including flight cancellation, delay, and aircraft swapping in the second stage recourse problem.

Our line of research falls into the second category of built-in robustness, where we aim to devise an airline schedule that is less susceptible to delay disturbance. One existing idea in this line of work has been to reduce delay propagation by re-timing the flight schedule. Ahmadbeygi et al. (2010) use a propagation tree to minimize propagated delay so that slack in the schedule can be re-allocated to where it is required most. In this research, we do not study the re-timing idea. Instead, the focus of this chapter is on building robustness into the aircraft routing problem to protect against propagated delay. Aircraft routing is a critical airline planning phase in which the goal is to create a sequence of flight legs to be operated by individual aircraft such that each flight is covered in exactly one routing, and all aircraft are properly maintained. Typical maintenance requirements include, for example, aircraft inspection for every 100 hours of flight time at certain maintenance-compatible stations (Federal Aviation

Administration, 2014b) and so on. The aircraft routing problem is solved separately for each fleet type. As Lan et al. (2006) indicate, this problem can usually be cast as a feasibility problem, thus providing flexibility to achieve desired robustness by designing an appropriate objective function. To illustrate the impact of aircraft routing on delay propagation, suppose two connecting fights are close to each other with respect to timing (the arrival time of the first flight is very close to the departure time of the second flight). In this case, it is not desirable to fly both using a single aircraft, because even a small amount of delay from the first flight will end up propagating to the second flight.

The idea of reducing propagated delay through clever routing is not new. Some of the pioneers include Lan et al. (2006), who develop a stochastic integer program to minimize the total expected propagated delay along aircraft routes where all the flight leg delays are fitted as independent log-normal distributions using historical data. To solve the program, they design an approximation algorithm based on column generation. To the best of our knowledge, no efficient solution approach that can solve this model to optimality exists, even when flight leg delays are assumed to be independent random variables. This is perhaps due to the nonlinearity in the computation of propagated delay. Weide et al. (2010) propose an iterative approach for a robust and integrated aircraft routing and crew pairing problem to mitigate delay propagation. Dunbar et al. (2012) propose an improved solution approach to solve a deterministic integrated aircraft routing and crew pairing problem to minimize total propagated delay, given that flight leg delays are known to be constant. This work was recently extended to incorporate stochastic flight leg delay information (Dunbar et al., 2014). They use historical flight leg delay data to construct random scenarios and develop two algorithms to solve the stochastic program. Borndörfer et al. (2010) develop an alternative robust aircraft routing formulation whose objective is to minimize the sum of the probability of when propagated delay is positive over all flights, assuming that flight leg delays follow certain independent probability distributions. They demonstrate the effectiveness of their model in reducing the amount of propagated delay through computational experiments, but in theory it is unclear how this objective function is related to the actual amount of propagated delay.

In this chapter, we depart from previous methods (Lan et al., 2006; Dunbar et al., 2014) that use *stochastic optimization* to minimize the expectation of propagated delay. We instead utilize *robust optimization* to minimize the maximal possible propagated delay. In contrast to the previous literature, which models flight leg delays as probability distributions, we utilize uncertainty sets to capture the stochasticity of flight leg delays. The major motivation of applying robust optimization to the aircraft routing problem comes from the ability to handle correlation in flight leg delays. Flights arriving to the same airport at similar times are usually delayed due to various air traffic management initiatives or inclement weather. Thus, incorporating delay correlation into robust aircraft planning could potentially yield additional benefits. As mentioned earlier, no efficient methods capable of dealing with delay correlation data have been explored in the robust aircraft routing literature.

On the other hand, robust optimization has been successfully applied to a number of application areas where the problems are combinatorially hard and involve correlated and complex uncertainty (Bertsimas et al., 2011). Although a major criticism of the robust optimization-based approach is the conservativeness of optimizing against the worst case as opposed to the expectation, under many circumstances, it may still perform better than stochastic optimization in expectation due to proper utilization of correlation data. This is mainly due to the added tractability that robust optimization enjoys, which allows it to handle complex uncertainty. Furthermore, additional protection is built into robust optimization, especially when realized data differs significantly from historical data (Bertsimas and Thiele, 2006; Bertsimas et al., 2013).

Applying robust optimization to the aircraft routing problem is fairly new. Until now, only one study (Marla et al., 2017) has considered this approach. They compare the performance of three different classes of models: 1) chance-constrained programming, 2) robust optimization, and 3) stochastic optimization (Lan et al., 2006). In their consideration of robust optimization, they model flight leg delays using a budget uncertainty set developed by Bertsimas and Sim (2004). Although their conclusions do not favor the robust optimization approach, we believe that by modeling uncertainty sets differently and solving the resultant robust problem cleverly, we may still be able to improve upon existing methods. Furthermore, we aim to provide comprehensive benchmarks to better evaluate the value of this approach.

The remainder of this chapter is organized as follows. In Section 2.2, we present our robust formulation of the robust aircraft routing problem in detail. We also discuss how to model the uncertainty set that captures flight leg delays. Our uncertainty set is a polyhedral set based on the Central Limit Theorem and incorporates correlation structure in flight leg delays. In Section 2.3, we develop an exact decomposition solution approach for our robust model based on column-and-row generation. We also discuss detailed computational approaches to solving the resulting separation and pricing sub-problems. In Section 2.4, we conduct extensive computational experiments to evaluate the benefits of our robust model and benchmark it against a state-of-the-research model. Yan and Kung (2016) is based on this chapter.

## 2.2 The Robust Aircraft Routing Problem

In this section, we describe our formulation for the robust aircraft routing problem using the framework of robust optimization. We first outline our mathematical formulation and then discuss how we model and construct uncertainty sets for flight leg delays.

### 2.2.1 Robust Aircraft Routing Formulation

The aircraft routing problem can be formally stated as follows: given a set of aircraft of a specific fleet type and a set of flights that must be operated, determine how the fleet can be routed so that each flight leg is included in exactly one aircraft routing, and all aircraft are properly maintained. Among all feasible assignment of aircraft to routes, we seek the one that incurs the least amount of total propagated delay.

We denote by $\mathcal{R}$ the set of all feasible routes, $\mathcal{F}$ the set of flights, and $N$ the total number of available aircraft. For each $r \in \mathcal{R}$, $F(r)$ represents the sequence of flights along route $r$. All maintenance-feasible routes can be represented by the columns of an $|\mathcal{F}| \times |\mathcal{R}|$ binary matrix $A$, where $A_{i,j} = 1$ if route $j$ visits flight $i$, and $A_{i,j} = 0$ otherwise. Binary decision variables $x_r$ denote whether or not route $r$ is chosen in the optimal solution, with $x_r = 1$ if it is selected, and $x_r = 0$ otherwise. As in Lan et al. (2006) and Dunbar et al. (2012), we consider two kinds of delay in our model:

1. *primary delay*, denoted $d_j$ for each flight leg $j \in \mathcal{F}$, which includes en-route delay, passenger connection delay, ground handling delay, and other delays that are not a function of routing.

2. *propagated delay*, denoted $p_j^r$ for flight leg $j \in F(r)$ on route $r \in \mathcal{R}$, represents the amount of delay propagated to flight $j$ caused by the late arrival of the upstream flight.

Given flight leg primary delay $\mathbf{d} \in R_+^{|\mathcal{F}|}$, the propagated delay $\mathbf{p}^r(\mathbf{d}) \in R_+^{|F(r)|}, r \in \mathcal{R}$ can be calculated iteratively as a function of $\mathbf{d}$. Suppose flights $(i, j)$ are two consecutive flights on route $r$, $\text{slack}_{i,j}$ denotes the slack time for a connection $(i, j)$, which is the difference between the scheduled arrival time of flight $i$ and the scheduled departure time of flight $j$, minus the mean turnaround time for the corresponding aircraft type. The propagated delay to flight $j$ from flight $i$ then follows the expression

$$p_j^r(\mathbf{d}) = \max\{0, p_i^r(\mathbf{d}) + d_i - \text{slack}_{i,j}\}, \quad \forall (i,j) \in r, \forall r \in \mathcal{R}. \tag{2.1}$$

With the above notation, given flight leg delays $\mathbf{d}$, we write the deterministic aircraft routing problem (AR) with the objective of minimizing total propagated delay as the following integer program:

$$\begin{aligned} \textbf{(AR)} \qquad \min_{\mathbf{x}} \quad & \sum_{r \in \mathcal{R}} c_r(\mathbf{d}) x_r = \sum_{r \in \mathcal{R}} \sum_{i \in F(r)} p_i^r(\mathbf{d}) x_r && (2.2) \\ \text{s.t.} \quad & A\mathbf{x} = \mathbf{e} && (2.3) \\ & \sum_{r \in \mathcal{R}} x_r \leq N && (2.4) \\ & \mathbf{x} \in \{0,1\}^{|\mathcal{R}|} \end{aligned}$$

Objective (2.2) is the sum of propagated delay over all flights, given constant flight leg primary delays $\mathbf{d}$. Constraints (2.3) are the flight cover constraints that ensure that each flight leg must be covered by only one aircraft routing. Constraints (2.4) are the fleet count constraints that keep the total number of aircraft used to be less than or equal to the number of available aircraft. To enhance the robustness of this model against all flight leg primary delays lying in a pre-specified uncertainty set $\mathcal{U}$, we consider the following robust aircraft routing (RAR) formulation:

$$
\begin{aligned}
\textbf{(RAR)} \qquad \min_{\mathbf{x}} \ \max_{\mathbf{d}\in\mathcal{U}} \quad & \sum_{r\in\mathcal{R}} c_r(\mathbf{d})x_r = \sum_{r\in\mathcal{R}}\sum_{i\in F(r)} p_i^r(\mathbf{d})x_r && (2.5)\\
\text{s.t.} \quad & A\mathbf{x} = \mathbf{e} && (2.6)\\
& \sum_{r\in\mathcal{R}} x_r \le N && (2.7)\\
& \mathbf{x} \in \{0,1\}^{|\mathcal{R}|}
\end{aligned}
$$

Objective (2.5) is to minimize the maximum possible total propagated delay when individual primary flight delays $\mathbf{d}$ are drawn from the uncertainty set $\mathcal{U}$.

### 2.2.2 Modeling Uncertainty Set $\mathcal{U}$ for Flight Leg Primary Delays d

In contrast to existing literature (Lan et al., 2006; Borndörfer et al., 2010), in which it is assumed that primary delays on flight legs $\mathbf{d}$ are independent for the purposes of tractability, we believe that flight legs have a clear dependence structure that should be accounted for in order to improve solution quality.

As an example, consider two different flights arriving to the same airport within the same hour. Both of these flights will be affected by the same weather conditions – one of the main drivers of primary delay – and may be delayed by similar amounts of time. Furthermore, various air traffic management initiatives are designed to delay flights which share the same characteristics. For instance, ground delay programs (GDPs) assign delays to flights entering into the same destination airport, and airspace flow programs (AFPs) control flights flying through the same flow constrained area (FCA). Moreover, different flow management programs often have complex interactions with each other (Barnhart et al., 2012). Thus, correlation of primary delay between different flights is a natural phenomenon that should be considered when generating aircraft routes.

With this insight in mind, we allow covariance data to play a central role in our uncertainty sets. Using historical schedule data from the Airline Service Quality Performance (ASQP) database, we first compute the primary delay for each flight leg by subtracting the propagated delay from total arrival delay based on the algorithm provided in Lan et al. (2006). We then calculate the sample means $\hat{\mu} \in R_{+}^{|\mathcal{F}|}$ of primary delays for each flight leg and create a sample covariance matrix $\hat{\Sigma} \succeq 0, \hat{\Sigma} \in R^{|\mathcal{F}| \times |\mathcal{F}|}$. Using this data, we create the following uncertainty sets for flight leg delays $\mathbf{d}$, where $C = \hat{\Sigma}^{-\frac{1}{2}}$, and $\hat{\sigma}_f$ denotes the standard deviation of the primary delay for flight $f \in \mathcal{F}$:

$$\mathcal{U}_p := \left\{ \mathbf{d} \in \mathbb{R}_{+}^{|\mathcal{F}|} \mid \|\mathbf{C}(\mathbf{d} - \hat{\mu})\|_p \leq \sqrt{|\mathcal{F}|} \cdot \Gamma; \left| \frac{d_f - \hat{\mu}_f}{\hat{\sigma}_f} \right| \leq \Gamma, \forall f \in \mathcal{F} \right\}, \tag{2.8}$$

where $p \in [0, +\infty]$, $\Gamma \in [0, +\infty)$ are two exogenous parameters. The first constraint in the uncertainty set is inspired by the Central Limit Theorem and is commonly used in the robust optimization literature. To see this, when primary leg delays $\mathbf{d}$ are uncorrelated, $C = \text{diag}(\frac{1}{\sigma_f})_{f \in \mathcal{F}}$, the first constraint is reduced to $\|\text{diag}(\frac{1}{\sigma_f})_{f \in \mathcal{F}}(\mathbf{d} - \mu)\|_p \leq \sqrt{|\mathcal{F}|} \cdot \Gamma$. In the case $p = 2$, this uncertainty set is an ellipsoid centered around $\mu$, where the length of each semi-principal axis is the value of the standard deviation of primary delays for the corresponding particular flight. When the primary delays are correlated, the uncertainty set would correspond to a stretching and rotation of the ellipsoid. For example, when primary delays for two flights are positively correlated, the uncertainty set would capture simultaneous large and small delays for pairs of correlated flights and would lose some regions where one flight delay is large, and the other is small, and vice versa. We also add box constraints for each individual primary delay $d_f$ to control the degree to which each individual primary delay can vary from its historical mean. The parameter $\Gamma$, commonly referred to as the budget of uncertainty, controls the protection level we want: the larger the $\Gamma$, the more conservative we are in the sense that we protect against delays that are allowed to lie in a larger set. For example, for this Central Limit Theorem set (2.8), $\Gamma = 3$ usually gives us a high level of protection when the data follows a normal distribution, because almost all of the probability mass lies in the range of $[\mu - 3\sigma, \mu + 3\sigma]$ for normal random variables. We refer interested readers to Bertsimas et al. (2004) for more details regarding this uncertainty set.

We choose $p = 1$ ($L_1$ norm) and introduce auxiliary variables $y_i, i \in \{1, \cdots, |\mathcal{F}|\}$ so that $\mathcal{U}$ can be equivalently formulated as a polyhedral set:

$$\begin{aligned} \mathcal{U} := \Bigg\{ \mathbf{d} \in \mathbb{R}_{+}^{|\mathcal{F}|} \mid \exists y \in R^{|\mathcal{F}|} \text{ s.t. } & \sum_{i=1}^{|\mathcal{F}|} y_i \leq \sqrt{|\mathcal{F}|} \cdot \Gamma; \\ & \pm c_i^T(\mathbf{d} - \hat{\mu}) \leq y_i, \forall i \in \{1, \cdots, |\mathcal{F}|\}; \\ & \left| \frac{d_f - \hat{\mu}_f}{\hat{\sigma}_f} \right| \leq \Gamma, \forall f \in \mathcal{F} \Bigg\}, \end{aligned} \tag{2.9}$$

where $c_i^T$ is the $i$th row of matrix $C$.

## 2.3 Solution Approach

In this section, we describe our solution approach via a column-and-row generation framework for the robust aircraft routing model (RAR) introduced in Section 2.2.1. We also discuss our computational approaches for solving the separation problem and the pricing problem, respectively.

### 2.3.1 Column-and-Row Generation Framework

As in the deterministic aircraft routing problem (2.2)-(2.4), *RAR* contains a huge number of potential decision variables (aircraft routes). Because it is impractical to enumerate all feasible routes explicitly, branch-and-price (Barnhart et al., 1998b) is often used to solve such problems. Unfortunately, this means that the traditional dualization approach for obtaining the robust counterpart (RC) introduced by Ben-Tal and Nemirovski (1999) cannot be applied in this setting, because we do not have the full set of decision variables included in the model before starting the solution process. However, we can still apply an iterative cutting-plane method (Bertsimas et al., 2016) to repeatedly solve *RAR* with a finite subset of the constraints. At each iteration, we check whether any violated constraints can be generated, adding them if they exist, and re-solving until there are no more violated constraints. In order to use the cutting-plane method, we consider the equivalent epigraph reformulation (epi-RAR) for the robust aircraft routing model (RAR),

$$
\begin{aligned}
\textbf{(epi-RAR)} \qquad \min_{\mathbf{x},z} \quad & z \\
\text{s.t.} \quad & \sum_{r\in\mathcal{R}} \sum_{i\in F(r)} p_i^r(\mathbf{d}) x_r \le z, \quad \forall \mathbf{d}\in\mathcal{U} && (2.10)\\
& A\mathbf{x} = \mathbf{e} && (2.11)\\
& \sum_{r\in\mathcal{R}} x_r \le N && (2.12)\\
& \mathbf{x}\in\{0,1\}^{|\mathcal{R}|}
\end{aligned}
$$

Note that *epi-RAR* is a mixed integer linear program with an infinite number of rows (constraint (2.10)) and a huge number of columns. To address this, we utilize efficient decomposition methods on both rows and columns. We refer to constraints (2.10) as *robustifying constraints* because they help to protect against all possible flight primary delays in the uncertainty set.

### The Separation Problem

For the relaxed problem *epi-RAR* with only a subset of robustifying constraints (2.10), we let $\mathbf{x}^*$ denote the optimal solution, and we let $z^*$ denote the optimal objective value. We denote by $\mathcal{R}_{\mathbf{x}^*}$ the feasible aircraft routes that are specified by $\mathbf{x}^*$ (i.e., $\mathcal{R}_{\mathbf{x}^*} = \{r\in\mathcal{R} \mid x_r^* = 1\}$). The following separation problem (SEP) checks whether any violated constraints in (2.10) must be added into the model:

**(SEP)**

$$z(\mathcal{R}_{\mathbf{x}^*}) = \max_{\mathbf{p},\mathbf{d}} \quad \sum_{r\in\mathcal{R}_{\mathbf{x}^*}} \sum_{i\in F(r)} p_i^r \tag{2.13}$$

$$\text{s.t.} \quad p_j^r = \max\{0, p_i^r + d_i - \text{slack}_{i,j}\}, \forall r \in \mathcal{R}_{\mathbf{x}^*}, \forall (i,j) \in r \tag{2.14}$$

$$\mathbf{d} \in \mathcal{U}$$

Given a specific aircraft routing, *SEP* optimizes the flight leg primary delays $\mathbf{d} \in \mathcal{U}$ to maximize the total propagated delay stated in objective (2.13). We note here that *SEP* cannot be solved by separating into parallel formulations, one for each route. This is because primary delays for flights from different routes could be jointly constrained in uncertainty set $\mathcal{U}$.

Constraint (2.14) is nonlinear in nature, but we can linearize it through a big-M re-formulation as follows, denoted as *SEP-bigM*:

**(SEP-bigM)**

$$z(\mathcal{R}_{\mathbf{x}^*}) = \max_{\mathbf{p},\mathbf{d}} \quad \sum_{r\in\mathcal{R}_{\mathbf{x}^*}} \sum_{i\in F(r)} p_i^r \tag{2.15}$$

$$\text{s.t.} \quad p_j^r \le p_i^r + d_i - \text{slack}_{i,j} + M_j^1 I_j, \quad \forall r \in \mathcal{R}_{\mathbf{x}^*} \ \forall (i,j) \in r \tag{2.16}$$

$$p_j^r \le 0 + M_j^2(1 - I_j), \quad \forall r \in \mathcal{R}_{\mathbf{x}^*}, \forall (i,j) \in r \tag{2.17}$$

$$\mathbf{d} \in \mathcal{U}$$

$$I \in \{0,1\}^{|\mathcal{F}|}$$

where $M_j^1, M_j^2$ are sufficiently large constants (big-$M$s), and $I$ are auxiliary indicator variables that are explained in more detail below.

Since $\mathcal{U}$ is a polyhedral set in our case, *SEP-bigM* is a mixed integer linear program. Along with the objective, constraints (2.16) and (2.17) together restrict that $p_j^r = \max\{0, p_i^r + d_i - \text{slack}_{i,j}\}, (i,j) \in r$, which is the definition of propagated delay. To see this, if auxiliary variable $I_j = 1$, constraints (2.16) become ineffective, and constraints (2.17) indicate $p_j^r = 0$ due to the maximization of the objective function; similarly, if $I_j = 0$, constraints (2.16) ensure $p_j^r = p_i^r + d_i - \text{slack}_{i,j}$. Thus at the optimal solution, $I_j$ will automatically pick the right value to ensure $p_j^r = \max\{0, p_i^r + d_i - \text{slack}_{i,j}\}$. We solve *SEP-bigM* with route $\mathcal{R}_{\mathbf{x}^*}$ and denote the optimal flight leg primary delay allocation by $\mathbf{d}^*$. If $z(\mathcal{R}_{\mathbf{x}^*}) \leq z^*$, then no violated constraint need be added; if $z(\mathcal{R}_{\mathbf{x}^*}) > z^*$, we add the violated constraint $\sum_{r\in\mathcal{R}} \sum_{i\in F(r)} p_i^r(\mathbf{d}^*)x_r \leq z$ to the relaxed master problem and re-solve.

Solving *SEP-bigM* is not an easy task due to the combinatorial structure of mixed integer programs. However, tightening the big-$M$ constants for each constraint can make the formulation substantially stronger than if big-$M$s are assigned arbitrarily large values. We present a method below to determine tightened but sufficiently large big-$M$s.

**Proposition 2.1.** *Let $d_j^* = \max_{\mathbf{d}\in\mathcal{U}} d_j, \forall j \in \{1, \cdots, |\mathcal{F}|\}$. $p_j^r(\mathbf{d}^*), \forall j \in \{1, \cdots, |\mathcal{F}|\}$ is the corresponding propagated delays under flight leg delays $\mathbf{d}^* = \{d_1^*, \cdots, d_{|\mathcal{F}|}^*\}$. Then $M_j^1 = slack_{i,j}$, $M_j^2 = p_j^r(\mathbf{d}^*)$ are valid values of big-Ms for SEP-bigM.*

*Proof.* In the optimal solution, indicator variable $I_j$ is 1 if $p_i^r + d_i - \text{slack}_{i,j} \leq 0$, which makes $p_j^r = \max\{0, p_i^r + d_i - \text{slack}_{i,j}\} = 0$. Since constraint (2.17) becomes $p_j^r \leq 0$ under $I_j = 1$, in order to let $p_j^r = 0$ in the optimal solution, the right hand side of constraint (2.16) should be greater than or equal to 0, i.e. $p_i^r + d_i - \text{slack}_{i,j} + M_j^1 \geq 0$. Since $p_i^r, d_i \geq 0$, setting $M_j^1 = \text{slack}_{i,j}$ is valid. On the other hand, in the optimal solution, $I_j$ is 0 if $p_i^r + d_i - \text{slack}_{i,j} \geq 0$, thus ensuring $p_j^r = \max\{0, p_i^r + d_i - \text{slack}_{i,j}\} = p_i^r + d_i - \text{slack}_{i,j}$. Since constraint (2.16) becomes $p_j^r \leq p_i^r + d_i - \text{slack}_{i,j}$ under $I_j = 0$, to make $p_j^r = p_i^r + d_i - \text{slack}_{i,j}$ in the optimal solution, the right hand side of constraint (2.17) should satisfy $M_j^2 \geq p_i^r + d_i - \text{slack}_{i,j}$. Note that by construction, $d_i \leq d_i^*, \forall i \in \{1, \cdots, |\mathcal{F}|\}, \forall \mathbf{d} \in \mathcal{U}$, thus $p_i^r + d_i - \text{slack}_{i,j} \leq \max\{0, p_i^r + d_i - \text{slack}_{i,j}\} = p_j^r(\mathbf{d}) \leq p_j^r(\mathbf{d}^*), \forall r \in \mathcal{R}_{\mathbf{x}^*}, \forall (i,j) \in r$ because $p_j^r(\mathbf{d})$ is a monotonically increasing function. This result shows that $M_j^2 = p_j^r(\mathbf{d}^*)$ is valid. □

Computing tightened big-$M$ according to Proposition 2.1 requires solving $|\mathcal{F}|$ separate optimization problems. The complexity of these optimization problems depends on how we construct uncertainty set $\mathcal{U}$. Under the set (2.9) we use, these optimization problems become linear programs, which can be solved very efficiently. With tightened big-$M$, this formulation works fairly efficiently when we test it for moderately large industry size instances in Section 2.4. When using it to solve very large instances, careful construction of uncertainty set $\mathcal{U}$ can greatly speed up *SEP-bigM*. For instance, for the uncertainty set described in (2.8), we could group flights according to their destination airport or pathway and estimate the covariance matrix for each group individually by assuming that primary delays between two flights in two different groups are independent. In that case, $C = \hat{\Sigma}^{-\frac{1}{2}}$ becomes sparse, making *SEP-bigM* easy to solve. Along the same lines, directly estimating a sparse inverse covariance matrix from data is also useful in increasing tractability for large instances (Friedman et al., 2008; Hsieh et al., 2011).

### The Pricing Problem

Each time a new robustifying constraint is added to the relaxed master problem, the resulting program needs to be solved using column generation. Let $G = (\mathcal{N}, \mathcal{A})$ be a directed acyclic graph with a single source node $s$ and a single terminal node $t$. The node set $\mathcal{N}$ represents the set of flights and arc set $\mathcal{A}$ corresponds to feasible connections between flights. For simplicity and testing purposes, the source node and sink node are dummy nodes that link to all flight nodes. In practice, various operational restrictions could be added. For example, the source node and sink node can only be linked to flights departing from and arriving to maintenance-compatible airports, respectively. Maintenance connection arcs could be added to represent maintenance opportunities (Barnhart et al., 1998a). Each flight node $i$ possesses a weight $-\mu_i$ corresponding to the negative dual price of the $i^{\text{th}}$ flight covering constraint (2.11). We denote by $r$ the dual price of the fleet count constraint (2.12). Without loss of generality, suppose there are $k$ constraints in the robustifying constraint set (2.10). Let $\mathbf{d^1}, \mathbf{d^2}, \cdots, \mathbf{d^k}$ be the corresponding flight primary delays vectors and $s_1, s_2, \cdots, s_k$ be the dual prices for robustifying constraints 1 to $k$. Thus for the RAR pricing problem, we wish to find an $s-t$ path $\pi^* = \{s, n_1, n_2, \cdots, t\}$ that minimizes reduced cost

$$\pi^* = \underset{\pi \text{ is an } s-t \text{ path}}{\arg\min} \; rc_\pi = \left\{ \left( -r - \sum_{i \in \pi : i \neq s,t} \mu_i \right) - \sum_{j=1}^{k} \left( \sum_{i \in \pi : i \neq s,t} s_j p_i^\pi(\mathbf{d^j}) \right) \right\}. \quad (2.18)$$

If reduced cost $rc_{\pi^*} < 0$, the minimizing route $\pi^*$ forms a new column of $A$ and is assigned a cost of $\sum_{i \in \pi^* : i \neq s,t} p_i^{\pi^*}(\mathbf{d^j})$ for robustifying constraints $j = 1, 2, \cdots, k$.

### Solving the Pricing Problem

We discuss here in detail the revised label setting algorithm to solve (2.18), which dynamically calculates the propagated delays using (2.1) and the reduced cost of the path. As Dunbar et al. (2012) point out, because the propagated delay $p_i^\pi(\cdot)$ is not a simple sum of delays along the path from $s$ to $i$, problem (2.18) cannot easily be cast as a minimum cost network flow problem. The label setting algorithm we develop here is modified from Dunbar et al. (2012).

Let $\pi$ be an $s-t$ path in $G$ (an ordered collection of nodes $\{s, n_1, \cdots, n_q, t\}$ in $\mathcal{N}$ with $(s, n_1), (n_q, t) \in \mathcal{A}$ and $(n_l, n_{l+1}) \in \mathcal{A}$ for all $l = 1, \cdots, q-1$). For $n \in \pi$, let $\pi(n)$ denote the ordered collection of nodes in the path $\pi$ truncated so that the final node in the list is $n$. Define truncated reduced cost $rc_{\pi(n)} = \left( -r - \sum_{i \in \pi(n) : i \neq s,t} \mu_i \right) - \sum_{j=1}^{k} \left( \sum_{i \in \pi(n) : i \neq s,t} s_j p_i^{\pi(n)}(\mathbf{d^j}) \right)$. Denote by $p_n^\pi(\cdot)$ the propagated delay at node $n$, computed along path $\pi(n)$ using (2.1). Due to the nonlinear nature of the propagated delay formula (2.1), our labels must track both the accumulated reduced cost $rc_{\pi(n)}$ at node $n$ along path $\pi$ and the propagated delay $p_n^\pi(\cdot)$. This motivates the following dominance conditions for paths destined for the same node.

**Definition 2.1** (Path Dominance Condition)**.** *Let paths $\pi(n)$, $\eta(n)$ be two different paths destined for the same node $n$. We say that $\pi(n)$ dominates $\eta(n)$ if $rc_{\pi(n)} \leq rc_{\eta(n)}$, $p_n^{\pi(n)}(\mathbf{d^j}) \leq p_n^{\eta(n)}(\mathbf{d^j}), \forall j \in \{1, \cdots, k\}$ and $(rc_{\pi(n)}, p_n^{\pi(n)}(\mathbf{d^j})) \neq (rc_{\eta(n)}, p_n^{\eta(n)}(\mathbf{d^j}))$ such that $s_j < 0$, where $k$ is the number of robustifying constraints (2.10) at the current iteration.*

**Lemma 2.1.** *Let $m \in \mathcal{N}$ such that $(n, m) \in \mathcal{A}$. If $\pi(n)$ dominates $\eta(n)$, then $\{\pi(n), m\}$ dominates $\{\eta(n), m\}$.*

*Proof.* From (2.1) we have $\forall j \in \{1, \cdots, k\}$ such that $s_j < 0$,

$$p_m^{\{\pi(n),m\}}(\mathbf{d^j}) = \max\{0, p_n^{\{\pi(n),m\}}(\mathbf{d^j}) + d_n^j - \text{slack}_{n,m}\}, \tag{2.19}$$

$$p_m^{\{\eta(n),m\}}(\mathbf{d^j}) = \max\{0, p_n^{\{\eta(n),m\}}(\mathbf{d^j}) + d_n^j - \text{slack}_{n,m}\}. \tag{2.20}$$

By Definition 2.1, (2.19) and (2.20),

$$p_n^{\{\pi(n),m\}}(\mathbf{d^j}) = p_n^{\pi(n)}(\mathbf{d^j}) \le p_n^{\eta(n)}(\mathbf{d^j}) = p_n^{\{\eta(n),m\}}(\mathbf{d^j})$$

Thus we have,

$$p_m^{\{\pi(n),m\}}(\mathbf{d^j}) \le p_m^{\{\eta(n),m\}}(\mathbf{d^j}),\ \forall 1 \le j \le k, s_j < 0 \tag{2.21}$$

For the reduced cost, linear programming duality tells us that dual prices $s_j \le 0,\ \forall 1 \le j \le k$ for all robustifying constraints (2.10), so we have:

$$\begin{aligned} rc_{\{\pi(n),m\}} &= rc_{\{\pi(n)\}} - \mu_m - \sum_{j=1}^{k} s_j p_m^{\{\pi(n),m\}}(\mathbf{d^j}) \\ &= rc_{\{\pi(n)\}} - \mu_m - \sum_{1 \le j \le k : s_j < 0} s_j p_m^{\{\pi(n),m\}}(\mathbf{d^j}), \\ rc_{\{\eta(n),m\}} &= rc_{\{\eta(n)\}} - \mu_m - \sum_{j=1}^{k} s_j p_m^{\{\eta(n),m\}}(\mathbf{d^j}) \\ &= rc_{\{\eta(n)\}} - \mu_m - \sum_{1 \le j \le k : s_j < 0} s_j p_m^{\{\eta(n),m\}}(\mathbf{d^j}). \end{aligned}$$

Since $rc_{\{\pi(n)\}} \le rc_{\{\eta(n)\}}$ and (2.21), we have $rc_{\{\pi(n),m\}} \le rc_{\{\eta(n),m\}}$. □

By induction, Lemma 2.1 suggests that if $\varpi$ is a path from $m$ to terminal node $t$, and $(n, m) \in \mathcal{A}$, we have $rc_{\{\pi(n),\varpi\}} \le rc_{\{\eta(n),\varpi\}}$. Thus equipped with this definition of path dominance, we can directly apply the label setting algorithm of Dunbar et al.

(2012) (Algorithm 1) to solve (2.18) where at each node, we only create labels for those paths that are not dominated by any other paths at that node.

**Algorithm 1** (Dunbar et al., 2012) Label Setting Algorithm for Pricing New Columns

```
Initialize:
    Set $I_s = \{s\}$ and $I_i = \emptyset$ for all $i \in \mathcal{N} \backslash \{s\}$.
    Set $M_i = \emptyset$ for each $i \in \mathcal{N}$.
loop
    if $\bigcup_{i \in \mathcal{N}} (I_i \backslash M_i) = \emptyset$ then
        return $\arg\min_{\pi(t) \in I_t} rc_{\pi(t)}$
    else
        choose $i \in \mathcal{N}$ and $\pi(i) \in I_i \backslash M_i$ so that $rc_{\pi(i)}$ is minimal
        for $(i, j) \in \mathcal{A}$ do
            if path $\{\pi(i), j\}$ is not dominated by $\eta(j)$ for any $\eta(j) \in I_j$ then
                set $I_j = I_j \bigcup \{\pi(i), j\}$
            end if
        end for
        set $M_i = M_i \bigcup \{\pi(i)\}$
    end if
end loop
```

This label setting algorithm is fairly efficient on the instances we test in Section 2.4. One major reason for this is that some of the robustifying constraints (2.10) are not binding in the optimal solution of the relaxed master problem. Thus, some of the dual prices corresponding to those robustifying constraints are zero. This makes the pricing algorithm, or more precisely the dominance condition, easier than it appears to be.

### 2.3.2 Overall Algorithm

We summarize the column-and-row generation algorithm that we use to solve *epi-RAR* in Algorithm 2. The algorithm iteratively adds robustifying constraints and new columns by solving the corresponding separation and pricing sub-problems until the upper bound provided by the separation problem matches with the optimal value of the current relaxed *epi-RAR* problem.

**Algorithm 2** Simultaneous Column-and-Row Generation

```
Given:
    Relaxed epi-RAR with only constraints (2.11) and (2.12)
Initialize:
    Add first robustifying constraint $\sum_{r\in\mathcal{R}}\sum_{i\in F(r)} p_i^r(\mathbf{d}^0)x_r \leq z$ with an
    arbitrary $\mathbf{d}^0 \in \mathcal{U}$.
loop
    Solve relaxed epi-RAR using branch-and-price, get optimal routes $\mathcal{R}_{\mathbf{x}^*}$ and
    optimal value $z^*$.
    Given $\mathcal{R}_{\mathbf{x}^*}$, solve SEP-bigM, get optimal flight primary delays $\mathbf{d}^*$ and optimal
    value $z(\mathcal{R}_{\mathbf{x}^*})$.
    if $z(\mathcal{R}_{\mathbf{x}^*}) > z^*$ then
        Add robustifying constraint $\sum_{r\in\mathcal{R}}\sum_{i\in F(r)} p_i^r(\mathbf{d}^*)x_r \leq z$ to relaxed epi-RAR.
    else
        return $\mathcal{R}_{\mathbf{x}^*}$
    end if
end loop
```

## 2.4 Computational Experiments

In this section, we compare the performance of our robust model against that of the state-of-the-research approach (Dunbar et al., 2014) on both historical delay data and simulated delay data. All computer programs were written in C++. The branch-and-price scheme was implemented by calling SCIP 3.1.1 (Achterberg, 2009) using CPLEX 12.6.0 as the linear programming solver. The mixed integer program for the separation problem is solved using CPLEX 12.6.0. All computational tests are performed on a laptop equipped with an Intel Core i7-3520M CPU running at 2.90 GHz and 8 GB memory. For solving some instances in our implementation, we observe that the branch-and-price solution process finds provably good solution very quickly but fails to prove optimality for a long time. We thus limit the runtime of the branch-and-price process to 120 seconds. If SCIP failed to prove optimality within 120 seconds, the best dual bound was used to report optimality gap of the solution. There was no runtime limitation for solving the separation problem.

### 2.4.1 Benchmark Model: Local Approach in Dunbar et al. (2014)

Based on the models and algorithms developed in Dunbar et al. (2012), Dunbar et al. (2014) further incorporate stochastic delay information to minimize expected total propagated delay. They model delay stochasticity by constructing a set of random scenarios $\Omega$, where each scenario $\omega \in \Omega$ corresponds to primary delay values $\mathbf{d}^{\omega}$ for each flight. Each random scenario is realized with equal probability. They develop two algorithms to tackle this problem: 1) the exact approach and 2) the local approach. Since their approaches were originally designed for integrated aircraft routing and crew scheduling, we simplify here to only aircraft routing. The exact approach enumerates all the feasible aircraft routes $\mathcal{R}$. For each feasible routing $r \in \mathcal{R}$, they calculate its expected total propagated delay $\mathbb{E}_{\mathbf{d}}\left[c_r(\mathbf{d})\right] = \sum_{\omega\in\Omega} \frac{1}{|\Omega|}\left(\sum_{i\in F(r)} p_i^r(\mathbf{d}^{\omega})\right)$. They then directly solve the deterministic model in (2.2)-(2.4) with all decision variables generated. They change the objective function to be $\sum_{r\in\mathcal{R}} \mathbb{E}_{\mathbf{d}}\left[c_r(\mathbf{d})\right] x_r$, which represents the sum of expected total propogated delay over all selected routings. Although this is an exact solution approach, it is not practical for industry-sized problems due to its lack of an efficient column generation process. As a result, we consider the local approach as our benchmark for comparison. The local approach is an approximation approach that incorporates stochastic delay information from the scenarios within the label-setting algorithm used in the pricing problem. At each step of the label-setting algorithm, it calculates the average delay propagation arriving at each flight node over all scenarios $\omega \in \Omega$. We denote $\hat{p}_j^r$ as the average propagated delay at node $j$ along path $r$, and $\hat{p}_j^r = \frac{1}{|\Omega|}\sum_{\omega\in\Omega} \max\left\{0, \hat{p}_i^r + d_i^{\omega} - \text{slack}_{i,j}\right\}$. With this notation, we introduce the path dominance condition used in the local approach in Dunbar et al. (2014).

**Definition 2.2** (Path Dominance Condition in Dunbar et al. (2014))**.** *Let $\pi(n), \eta(n)$ be two different paths destined for the same node $n$. We say that $\pi(n)$ dominates $\eta(n)$ if $rc_{\pi(n)} \leq rc_{\eta(n)}$ and $\hat{p}_n^{\pi(n)} \leq \hat{p}_n^{\eta(n)}$.*

The optimal solution of the local approach can be computed by column generation using the pricing algorithm (Algorithm 1) with path dominance condition as Definition 2.2.

### 2.4.2 Case Studies Based on Historical Delay Data

To evaluate the effectiveness of our robust model, we apply Algorithm 2 to create routes for two of the largest fleet types operated by a major US airline in the year 2007. The characteristics of the underlying networks are listed in Table 2.1, and all the flights selected are operated on a daily basis over the testing period. Because in practice the model will be built using historical data and then applied to future operations, all routings were created using ASQP data containing historical flight primary delays for the 31 days of July 2007 (the training set) and then evaluated out-of-sample on the 31 days of August 2007 (the testing set).

| Network | Number of flight legs | Number of aircraft |
|---|---|---|
| $N_1$ | 106 | 24 |
| $N_2$ | 117 | 23 |

Table 2.1: Characteristics of Two Routing Problems

For both sets of data, we benchmark the routes of our robust model against two other routes:

1. The original route in the historical schedule data as the baseline (B):

   This is the airline's actual route for the selected flights.

2. The local approach provided in Dunbar et al. (2014) (DFW) as in Section 2.4.1:

   The flight primary delays for the 31 days in July 2007 constitute 31 random delay scenarios. Each random scenario is realized with probability $\frac{1}{31}$.

We evaluate the performance of these three approaches along three criteria: 1) the average total propagated delay, 2) the volatility of total propagated delay (standard deviation), and 3) worst-case total propagated delay (maximum value). Figures 2-1 and 2-2 compare the performance of the robust approach over a range of values of the budget of uncertainty $\Gamma \in \{0.2, 0.4, \cdots, 2.8, 3.0\}$ to the DFW method on the training data from July 2007. We compute the ratios: $100 \cdot (\mathrm{B} - \mathrm{DFW})/\mathrm{B}$ and $100 \cdot (\mathrm{B} - \mathrm{RAR})/\mathrm{B}$ in each performance evaluation criterion to measure the improvement over the baseline (B). The long dashed lines represent the relative reduction in percentage in corresponding performance criterion for approach DFW over the baseline case (B), and the solid lines represent the relative reduction in percentage in corresponding performance criterion for approach RAR over the baseline (B) under various values of uncertainty budget $\Gamma$. Notice that for the robust approach, as $\Gamma$ increases, the performance along the measures of standard deviation and maximum value of propagated delay improves almost monotonically. This is because the routings are created on training data, and larger values of $\Gamma$ protect against precisely the maximum total propagated delay.

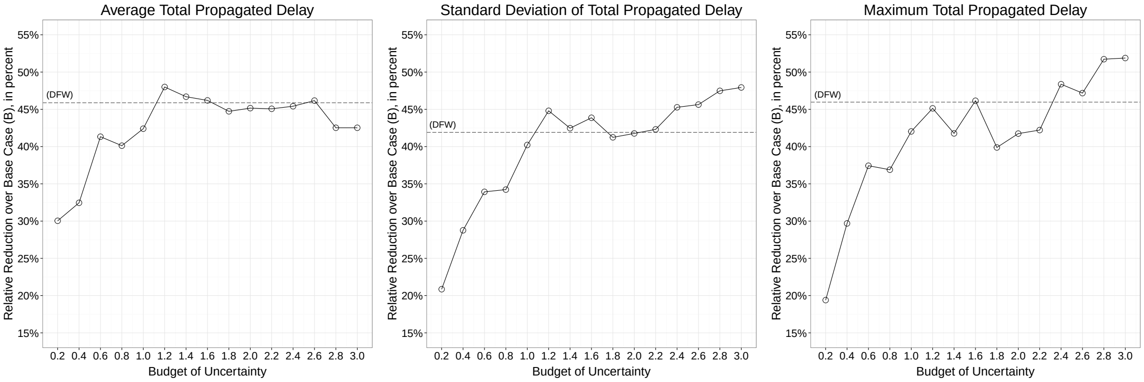


Figure 2-1: Relative Improvements of the Algorithms RAR over B and DFW over B on Network $N_1$ (July 2007)

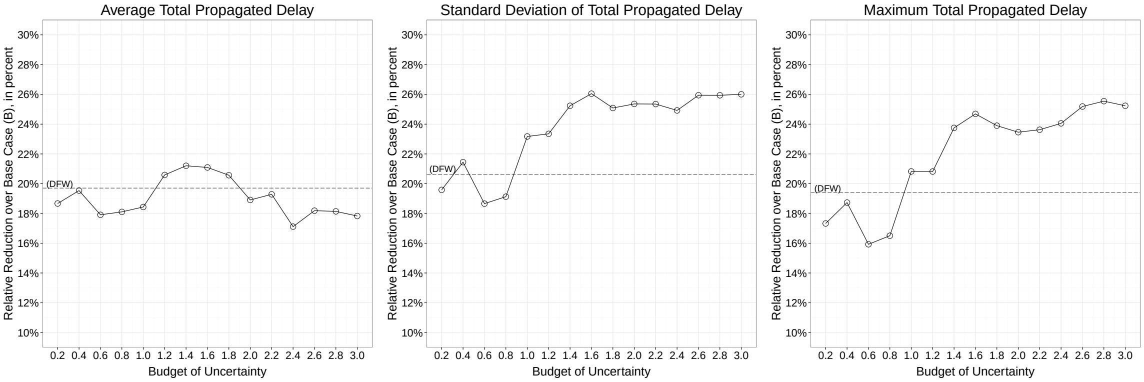


Figure 2-2: Relative Improvements of the Algorithms RAR over B and DFW over B on Network $N_2$ (July 2007)

When the optimal routes informed by the robust model formulated using the training data (July 2007) are applied to testing data (August 2007) as in Figures 2-3 and 2-4, we no longer see this monotonicity trend. In practice, we would select one particular value of $\Gamma$ to create future routings. Guidelines for selecting the budget of uncertainty in previous literature (Bertsimas et al., 2011, 2017) usually focus on probabilistic guarantees, i.e. the uncertainty set can be tuned so that constraints are robustly feasible with at least some probability. However, uncertainty of primary delays in our problem does not affect route feasibility, it only changes the amount of total propagated delay. Thus, we decide to choose $\Gamma$ based on the performance of propagated delay on the training set. We prioritize average total propagated delay as our main performance goal, thus based on Figures 2-1 and 2-2, we consider $\Gamma = 1.2$ to be appropriate for network $N_1$ and $\Gamma = 1.4$ to be appropriate for network $N_2$ because the robust model yields the lowest average total propagated delay under these two $\Gamma$ in the training set. Table 2.2 presents summary statistics for all propagated delay performance criteria under the testing set for routes produced by each approach under both networks $N_1$ and $N_2$. Rows with values of $\Gamma$ that are selected based on training set performance are in boldface. Note that generally, the best way to set the parameter $\Gamma$ would be: (1) divide our historical dataset into training and validation components; (2) select the value of $\Gamma$ that yields the best performance on the validation set (cross-validation can also be applied); and (3) once that value of $\Gamma$ is set, we would then use it to generate future aircraft routings. For the sake of simplicity and also due to a dearth of data, we only use the training set to select the value of $\Gamma$ in this work.

Both approaches improve upon the baseline routing considerably in terms of average, standard deviation, and maximum of total propagated delay. For testing data on network $N_1$, when $\Gamma = 1.2$, RAR performs better than DFW in all of the performance criteria of total propagated delay that we consider (8.2% larger decrease in average total propagated delay, 12.1% larger decrease in standard deviation, and 10.1% larger decrease in maximum total propagated delay, as compared to baseline). For testing data on network $N_2$, when $\Gamma = 1.4$, RAR outperforms DFW according to the standard deviation (2.2% larger decrease from baseline) and maximum value

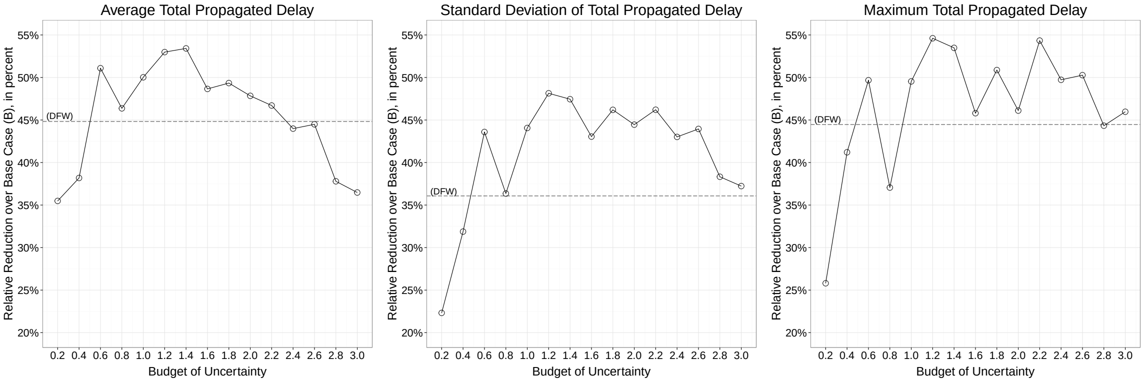


Figure 2-3: Relative Improvements of the Algorithms RAR over B and DFW over B on Network $N_1$ (August 2007)

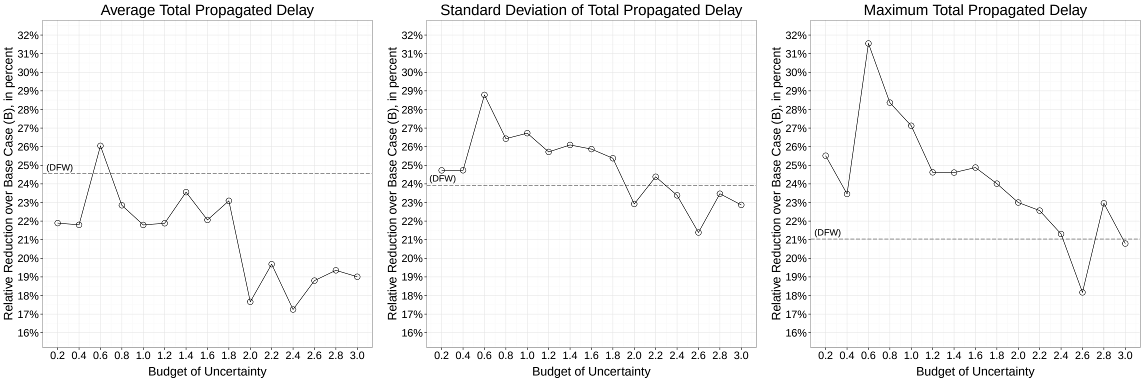


Figure 2-4: Relative Improvements of the Algorithms RAR over B and DFW over B on Network $N_2$ (August 2007)

of total propagated delay (3.6% larger decrease from baseline), but loses slightly in terms of average total propagated delay (1.0% smaller decrease from baseline).

In addition to this specific value that we have inspected, for both flight networks under testing data, RAR actually outperforms DFW in reducing volatility and extreme delay under a wide range of $\Gamma$ values (for network $N_1$, $\Gamma \in [1.0, 2.6]$; for network $N_2$, $\Gamma \in [0.2, 1.8]$). For very small ($\leq 0.4$) or very large ($\geq 2.6$) $\Gamma$ values, RAR loses superiority in at least two evaluation criteria in one of the networks. This is because the uncertainty set $\mathcal{U}$ constructed by these $\Gamma$ are either too conservative (for large $\Gamma$ values) or too restrictive (for small $\Gamma$ values) in representing potential primary delay. In other words, such $\Gamma$ values do not accurately capture real world data. When $\Gamma \in [1.0, 2.6]$, in network $N_1$, compared to DFW, RAR can reduce on average 9.1% more of the standard deviation and on average 6.1% more of the maximum delay value from the baseline. When $\Gamma \in [0.2, 1.8]$, in network $N_2$, RAR can reduce on average 2.1% more of the standard deviation and on average 5.0% more of the maximum delay value compared to DFW from the baseline. The superiority in reducing extreme delay values of RAR intuitively makes sense because of the min-max objective in RAR formulation. The reduction in volatility is a surprising by-product of reduced extreme delay values, even though we do not explicitly include it in the objective for RAR. Most surprisingly, for network $N_1$, RAR also has better performance in reducing average total propagated delay in the training set. When $\Gamma \in [1.0, 2.6]$, on average 3.8% more of the average total propagated delay can be reduced. In network $N_2$, RAR mostly underperforms DFW in reducing average delay value, but not by too much (on average only $-1.8\%$ when $\Gamma \in [0.2, 1.8]$), especially in light of the gains made in reducing volatile and extreme values.

| Approach | AVG. Total Propagated Delay (minute) | STD. Total Propagated Delay (minute) | MAX Total Propagated Delay (minute) | Total CPU Time (second) | *SEP-bigM* CPU Time (second) | Number of Robustifying Cuts Added | Optimality Gap (%) |
|---|---|---|---|---|---|---|---|
| | | | Flight Network $N_1$, 106 flights / 24 aircraft | | | | |
| B | 1,042.6 | 1,409.8 | 5,856 | – | – | – | |
| DFW | 575.3 | 901.2 | 3,252 | 217.04 | – | – | 0.00 |
| RAR ($\Gamma = 0.2$) | 672.6 | 1,095.2 | 4,345 | 161.70 | 3.56 | 3 | 3.79 |
| RAR ($\Gamma = 0.4$) | 644.5 | 1,023.7 | 3,874 | 679.01 | 49.65 | 6 | 1.06 |
| RAR ($\Gamma = 0.6$) | 509.5 | 795.3 | 2,947 | 572.90 | 53.75 | 5 | 2.33 |
| RAR ($\Gamma = 0.8$) | 559.2 | 897.3 | 3,686 | 223.63 | 61.30 | 3 | 2.41 |
| RAR ($\Gamma = 1.0$) | 521.1 | 788.6 | 2,955 | 403.16 | 109.15 | 5 | 0.39 |
| **RAR ($\boldsymbol{\Gamma = 1.2}$)** | **490.2** | **731.1** | **2,658** | **602.34** | **185.94** | **7** | **0.00** |
| RAR ($\Gamma = 1.4$) | 485.7 | 740.9 | 2,724 | 1,041.99 | 491.78 | 5 | 0.93 |
| RAR ($\Gamma = 1.6$) | 535.3 | 802.9 | 3,174 | 935.79 | 505.36 | 5 | 0.24 |
| RAR ($\Gamma = 1.8$) | 528.2 | 758.5 | 2,877 | 2,004.52 | 1,064.69 | 8 | 0.62 |
| RAR ($\Gamma = 2.0$) | 543.8 | 783.1 | 3,156 | 779.50 | 337.32 | 4 | 0.43 |
| RAR ($\Gamma = 2.2$) | 555.7 | 758.4 | 2,674 | 1,482.82 | 898.21 | 5 | 0.45 |
| RAR ($\Gamma = 2.4$) | 584.0 | 803.4 | 2,944 | 4,123.59 | 3,146.46 | 8 | 1.49 |
| RAR ($\Gamma = 2.6$) | 578.7 | 790.2 | 2,912 | 3,315.71 | 2,453.99 | 7 | 2.00 |
| RAR ($\Gamma = 2.8$) | 648.6 | 869.5 | 3,260 | 1,758.19 | 918.81 | 6 | 1.75 |
| RAR ($\Gamma = 3.0$) | 662.3 | 884.9 | 3,164 | 1,593.94 | 1,102.16 | 6 | 1.32 |
| | | | Flight Network $N_2$, 117 flights / 23 aircraft | | | | |
| B | 1,131.9 | 1,651.4 | 7,392 | – | – | – | |
| DFW | 854.0 | 1,256.7 | 5,837 | 140.11 | – | – | 0.00 |
| RAR ($\Gamma = 0.2$) | 884.1 | 1,243.1 | 5,506 | 28.43 | 2.38 | 3 | 0.00 |
| RAR ($\Gamma = 0.4$) | 885.2 | 1,243.0 | 5,658 | 23.68 | 3.97 | 2 | 0.00 |
| RAR ($\Gamma = 0.6$) | 837.1 | 1,176.1 | 5,060 | 420.06 | 18.50 | 5 | 0.01 |
| RAR ($\Gamma = 0.8$) | 873.3 | 1,214.9 | 5,295 | 50.73 | 5.86 | 2 | 0.00 |
| RAR ($\Gamma = 1.0$) | 885.2 | 1,210.1 | 5,387 | 320.87 | 22.26 | 4 | 0.09 |
| RAR ($\Gamma = 1.2$) | 884.2 | 1,226.7 | 5,572 | 77.06 | 12.93 | 3 | 0.00 |
| **RAR ($\boldsymbol{\Gamma = 1.4}$)** | **865.3** | **1,220.5** | **5,573** | **209.96** | **17.89** | **4** | **0.00** |
| RAR ($\Gamma = 1.6$) | 882.2 | 1,224.2 | 5,553 | 335.99 | 22.93 | 4 | 0.00 |
| RAR ($\Gamma = 1.8$) | 870.5 | 1,232.3 | 5,617 | 321.51 | 11.92 | 4 | 0.09 |
| RAR ($\Gamma = 2.0$) | 932.0 | 1,272.9 | 5,692 | 489.12 | 12.49 | 4 | 0.05 |
| RAR ($\Gamma = 2.2$) | 909.1 | 1,248.8 | 5,724 | 420.46 | 10.27 | 4 | 0.09 |
| RAR ($\Gamma = 2.4$) | 936.7 | 1,265.2 | 5,817 | 446.1 | 10.50 | 4 | 0.06 |
| RAR ($\Gamma = 2.6$) | 919.1 | 1,298.2 | 6,049 | 95.82 | 9.73 | 2 | 0.00 |
| RAR ($\Gamma = 2.8$) | 912.8 | 1,263.8 | 5,695 | 94.95 | 8.71 | 2 | 0.00 |
| RAR ($\Gamma = 3.0$) | 916.7 | 1,273.7 | 5,855 | 100.52 | 5.68 | 2 | 0.00 |

Table 2.2: Comparative Algorithmic Performance on Two Test Networks

As for computational tractability, the efficiency of the RAR model depends on the tractability of both the separation problem and the branch-and-price problem. Table

2.2 reports the total CPU time, CPU time for the separation problem, number of robustifying cuts needed until Algorithm 2 converges, and the optimality gap for the final robust solution. Since we impose 120 seconds runtime limit on the branch-and-price solution process, the best objective value we get from each branch-and-price process cannot serve as a valid lower bound for the optimal value of RAR if SCIP fails to prove optimality. However, the best dual bound at the end of the solution process can still provide a valid lower bound of RAR. We denote UB as the lowest optimal objective value of all the separation problems when Algorithm 2 converges. UB serves as an upper bound for the optimal value of RAR. Similarly, we denote LB as the highest dual bound value of all the branch-and-price processes solved when Algorithm 2 terminates. LB is a valid lower bound for the optimal value of RAR. The optimality gap is then defined as $\frac{\text{UB}-\text{LB}}{\text{UB}} \times 100\%$. Within the runtime limitation, for network $N_1$, RAR only solves one instance to optimality ($\Gamma = 1.2$), however, for many instances, provably good solutions (with optimality gap less than 1%) can be obtained; for network $N_2$, RAR solves 9 out of 15 instances to optimalily, and all the unsolved instances have very high quality solutions (with optimality gap less than 0.1%). We observe that when the separation problem and branch-and-price process are both easy to solve (e.g., the 9 fully solved instances in network $N_2$), the RAR model is more efficiently solved compared to DFW. However, when one of the problems is relatively hard to solve, RAR is less efficient to solve. Computational time reduction for the separation problem could be achieved by performing matrix sparsifying tricks suggested in Section 2.3.1. Tailored branching rules (Barnhart et al., 1998a) could be utilized to speed up the branch-and-price solution process. We also point out here that the solution time of RAR does not depend on the size of the historical delay data for training. No matter how many days of historical flight delay we use, we can transform all of them into a single uncertainty set of the same size. On the other hand, the solution time of DFW grows approximately linearly with the number of days involved in the training set because the algorithm needs to average over all training days for each flight leg during the column generation process. This suggests that RAR might require less computational effort when our model uses more historical delay data as part of its training set.

In this case study, careful modeling of the uncertainty set $\mathcal{U}$ allows us to reduce the volatility of solution performance. In some cases, average performance also benefits when compared to the existing state-of-the-research stochastic optimization approach. We suspect that these benefits come from the following: (1) the robust approach provides additional robustness when the realized primary delays are slightly different from the estimated distribution in the training data set; and (2) the robust model provides a tractable and exact approach to deal with correlated primary delays, and incorporating delay correlation improves the robustness of resultant routings.

### 2.4.3 Case Studies Based on Simulated Delay Data

To provide more insight into the relative performance of RAR over DFW, we conduct an additional round of computational experiments where the flight primary delay data are generated from simulated probability distributions instead of historical data. We seek to quantify the robustness of both methods with respect to changes in probability distributions. These case studies are motivated by the idea that primary flight leg delay rarely corresponds to a specific distribution, but rather a composite of several types of delays, each with differing individual distributions that may vary throughout different times of the day, month, and year (Tu et al., 2008). Because of this distribution ambiguity and variation over time, it is an important feature of robust operations planning methods to be able to protect against ambiguity of future delays when trained on historical data.

Previous literature suggests several classes of candidate probability distributions that are commonly used to model flight leg delay (Mueller and Chatterji, 2002; Tu et al., 2008; Bai, 2006). Out of these, we pick three representative ones: normal (truncated), gamma, and log-normal. The computational experiment is set up as follows. For the training data set, we use historical flight delay data in July 2007 (the training set we use in Section 2.4.2) to compute the observed mean and variance of each flight leg and the Spearman's rank correlation coefficient matrix of all flight legs. We pick Spearman's rho instead of the widely-used Pearson's rho because Spearman's rho preserves order when nonlinear transformations are applied to random variables. This is convenient for generating random variables with different distributions, but the same correlation structure (MathWorks, 2014). We then generate the set of training data with 1,000 samples for each flight leg such that the marginal distributions of flight legs follow a truncated normal distribution with the same mean and variance calculated from the July 2007 data. We also fix the Spearman's rank correlation coefficient matrix of the simulated data to be the same as the one calculated from July 2007 data. For testing data sets, we create three different groups of testing sets where the marginal distributions of flight leg delay are distributed as truncated normal, gamma, and log-normal respectively. For each group, we create testing sets in the following way:

- *Deviation in mean*

  We keep the standard deviation and Spearman's rank correlation coefficient matrix of the testing set the same as the training set. We then generate testing data by setting the mean to be 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2 times the mean of the training set.

- *Deviation in standard deviation*

  We keep the mean and Spearman's rank correlation coefficient matrix of the testing set the same as the training set. We then generate testing data by setting the standard deviation of the testing set to be 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2 times the standard deviation of the training set.

- *Deviation in correlation structure*

  We keep the mean and standard deviation of the testing set the same as the training set. We then generate testing data with different Spearman's rank correlation coefficient matrix. We define the multiplier of correlation $\alpha \in (-\infty, 1]$ as a parameter to control how the correlation structure deviates from the training set. $\alpha = 0$ means there is no change in correlation structure. As $\alpha$ increases to 1, the primary delays become more independent, with full independence at $\alpha = 1$. As $\alpha$ decreases to $-\infty$, the primary delay data becomes more correlated, with perfect correlation at $\alpha = -\infty$. We pick $\alpha = \ln(3/4), \ln(5/6), \ln(11/12), 0, \ln(1+(e-1)/12), \ln(1+(e-1)/6), \ln(1+(e-1)/4)$ to generate testing data with various correlation structures. The detailed meaning of $\alpha$ and the correlation perturbing procedures are provided in Appendix A.2.

In total, we have 19 testing sets for each distribution and each flight network. We apply the above generation procedure to both networks $N_1$ and $N_2$ that we used in Section 2.4.2. We use the training set to create routings using both RAR and DFW and then test the performance of the generated routings under the testing sets created using the procedure above. Note that in DFW, there are 1000 random scenarios, each realized with probability 1/1000. Similar to Section 2.4.2, for both networks, we select uncertainty budget $\Gamma$ that has the best performance in reducing average total propagated delay in the training set. For network $N_1$, $\Gamma = 1.2$; for network $N_2$, $\Gamma = 1.4$. Since we do not have a baseline routing anymore, we calculate the relative performance ratio of RAR over DFW as $100 \cdot (\text{DFW} - \text{RAR})/\text{DFW}$ for each performance criteria. The results are summarized in Tables 2.3, 2.4 and 2.5, where each table corresponds to a specific testing data delay distribution. Complete results with all values of $\Gamma \in \{0.2, 0.4, \cdots, 2.8, 3.0\}$ are presented in Appendix A.1.

Overall, in all 114 testing sets, RAR outperforms DFW in reducing average value in 58% (66/114) of the cases, reducing standard deviation in 99% (113/114) of the cases, and reducing extreme value in 99% (113/114) of the cases.

When testing data takes the same distributional form as the training data (truncated normal, Table 2.3), for both networks, RAR consistently outperforms DFW in reducing standard deviation and extreme value, especially when testing data has a larger mean or standard deviation than the training set. The only exception is the case when standard deviation multiplier is 0.5 in network $N_2$. On the other hand, in most cases, DFW outperforms RAR in reducing average propagated delay, especially when the testing data has a smaller mean or variance compared to the training data and when the tesing data has a different correlation structure. Overall, similar to what we observe in the computational results of network $N_2$ in Section 2.4.2, the improvement in reducing volatility and extreme value comes at the cost of a diminished performance with regard to decreasing the average value. However, this cost is usually small in comparison to the reductions in volatility and extreme value that we observe.

When the testing set is distributed as gamma or log-normal (i.e., different from the training set, Tables 2.4 and 2.5), in most of the cases when only mean and standard deviation are varying, RAR results in a route that has substantially smaller standard deviation, extreme value and even average value. As before, the increase in average total propagated delay is relatively small. When the correlation structure deviates from the testing data, we see that the impact of such deviation on RAR's performance is much larger than the impact of deviation in mean or standard deviation. Under almost all correlation multipliers, all three performance criteria (especially for the extreme value reduction criterion) are inferior to the cases when correlation structure is unchanged. Deviation in correlation changes the shape of the uncertainty set instead of the volume, which might cause this substantial downgrade in RAR's performance. Also, changing correlation structure might have less impact on the performance of DFW because the approach does not explicitly use correlation data. Though RAR is less effective in these correlation-varying cases, it still outperforms DFW in at least two criteria. Moreover, the small inferiority in one criterion, if exists, is usually offset by the superiority of the other two criteria.

In summary, compared to the existing stochastic optimization approach, the RAR model provides additional benefit when realized flight leg delay differs in distribution from historical data.

| Multiplier of mean | Multiplier of std | Multiplier of correlation | RPR of mean reduction (%) | RPR of std reduction (%) | RPR of extreme value reduction (%) |
|---|---|---|---|---|---|
| Flight Network $N_1$, 106 flights and 24 aircraft, best $\Gamma = 1.2$ | | | | | |
| 0.50 | 1.00 | 0.00 | -0.5 | 4.1 | 0.4 |
| 0.75 | 1.00 | 0.00 | -0.3 | 4.3 | 0.1 |
| 1.00 | 1.00 | 0.00 | -0.1 | 4.5 | 1.3 |
| 1.25 | 1.00 | 0.00 | 0.2 | 4.7 | 1.9 |
| 1.50 | 1.00 | 0.00 | 0.5 | 4.9 | 2.8 |
| 1.75 | 1.00 | 0.00 | 0.9 | 5.1 | 3.2 |
| 2.00 | 1.00 | 0.00 | 1.2 | 5.2 | 3.4 |
| 1.00 | 0.50 | 0.00 | -5.2 | 3.4 | 2.8 |
| 1.00 | 0.75 | 0.00 | -1.6 | 4.1 | 0.3 |
| 1.00 | 1.25 | 0.00 | 1.0 | 4.9 | 3.8 |
| 1.00 | 1.50 | 0.00 | 1.9 | 5.3 | 2.5 |
| 1.00 | 1.75 | 0.00 | 2.5 | 5.7 | 2.6 |
| 1.00 | 2.00 | 0.00 | 3.0 | 5.9 | 2.5 |
| 1.00 | 1.00 | $\ln(3/4)$ | 0.7 | 5.7 | 6.7 |
| 1.00 | 1.00 | $\ln(5/6)$ | 0.4 | 5.2 | 6.7 |
| 1.00 | 1.00 | $\ln(11/12)$ | 0.0 | 4.0 | 1.7 |
| 1.00 | 1.00 | $\ln(1+(e-1)/12)$ | -0.3 | 4.0 | 5.4 |
| 1.00 | 1.00 | $\ln(1+(e-1)/6)$ | -0.3 | 4.3 | 2.9 |
| 1.00 | 1.00 | $\ln(1+(e-1)/4)$ | -0.2 | 4.7 | 10.2 |
| Flight Network $N_2$, 117 flights and 23 aircraft, best $\Gamma = 1.4$ | | | | | |
| 0.50 | 1.00 | 0.00 | -2.4 | 1.2 | 1.1 |
| 0.75 | 1.00 | 0.00 | -2.0 | 1.5 | 1.2 |
| 1.00 | 1.00 | 0.00 | -1.6 | 1.8 | 1.0 |
| 1.25 | 1.00 | 0.00 | -1.2 | 2.1 | 0.9 |
| 1.50 | 1.00 | 0.00 | -0.9 | 2.4 | 1.0 |
| 1.75 | 1.00 | 0.00 | -0.6 | 2.6 | 1.3 |
| 2.00 | 1.00 | 0.00 | -0.3 | 2.8 | 1.3 |
| 1.00 | 0.50 | 0.00 | -5.5 | -1.1 | 1.7 |
| 1.00 | 0.75 | 0.00 | -3.0 | 1.1 | 1.1 |
| 1.00 | 1.25 | 0.00 | -1.2 | 2.2 | 1.4 |
| 1.00 | 1.50 | 0.00 | -0.1 | 2.4 | 1.6 |
| 1.00 | 1.75 | 0.00 | 0.3 | 2.5 | 1.1 |
| 1.00 | 2.00 | 0.00 | 0.6 | 2.6 | 1.0 |
| 1.00 | 1.00 | $\ln(3/4)$ | -2.1 | 1.4 | 1.3 |
| 1.00 | 1.00 | $\ln(5/6)$ | -2.3 | 1.0 | 3.5 |
| 1.00 | 1.00 | $\ln(11/12)$ | -2.1 | 1.2 | 1.2 |
| 1.00 | 1.00 | $\ln(1+(e-1)/12)$ | -1.9 | 0.9 | 2.3 |
| 1.00 | 1.00 | $\ln(1+(e-1)/6)$ | -1.7 | 1.7 | 0.7 |
| 1.00 | 1.00 | $\ln(1+(e-1)/4)$ | -2.3 | 0.8 | 2.4 |

Table 2.3: Relative Performance Ratio (RPR) under Different Mean / Standard Deviation / Correlation Multiplier of Testing Data Following Truncated Normal Distribution

| Multiplier of mean | Multiplier of std | Multiplier of correlation | RPR of mean reduction (%) | RPR of std reduction (%) | RPR of extreme value reduction (%) |
|---|---|---|---|---|---|
| Flight Network $N_1$, 106 flights and 24 aircraft, best $\Gamma = 1.2$ | | | | | |
| 0.50 | 1.00 | 0.00 | 3.9 | 9.2 | 20.4 |
| 0.75 | 1.00 | 0.00 | 3.2 | 8.4 | 18.4 |
| 1.00 | 1.00 | 0.00 | 2.8 | 7.9 | 17.0 |
| 1.25 | 1.00 | 0.00 | 2.5 | 7.6 | 16.0 |
| 1.50 | 1.00 | 0.00 | 2.3 | 7.3 | 15.1 |
| 1.75 | 1.00 | 0.00 | 2.1 | 7.2 | 13.8 |
| 2.00 | 1.00 | 0.00 | 2.2 | 7.0 | 10.7 |
| 1.00 | 0.50 | 0.00 | -2.4 | 5.9 | 13.5 |
| 1.00 | 0.75 | 0.00 | 1.3 | 7.4 | 16.0 |
| 1.00 | 1.25 | 0.00 | 3.6 | 8.2 | 17.9 |
| 1.00 | 1.50 | 0.00 | 3.9 | 8.3 | 18.4 |
| 1.00 | 1.75 | 0.00 | 4.0 | 8.3 | 18.8 |
| 1.00 | 2.00 | 0.00 | 4.1 | 8.4 | 19.2 |
| 1.00 | 1.00 | $\ln(3/4)$ | 3.5 | 7.3 | 9.1 |
| 1.00 | 1.00 | $\ln(5/6)$ | 4.0 | 7.9 | 9.4 |
| 1.00 | 1.00 | $\ln(11/12)$ | 1.4 | 4.0 | 8.2 |
| 1.00 | 1.00 | $\ln(1+(e-1)/12)$ | 1.6 | 6.6 | 8.0 |
| 1.00 | 1.00 | $\ln(1+(e-1)/6)$ | 3.1 | 7.0 | 1.2 |
| 1.00 | 1.00 | $\ln(1+(e-1)/4)$ | 3.2 | 8.5 | 10.8 |
| Flight Network $N_2$, 117 flights and 23 aircraft, best $\Gamma = 1.4$ | | | | | |
| 0.50 | 1.00 | 0.00 | 2.2 | 9.4 | 27.9 |
| 0.75 | 1.00 | 0.00 | 1.2 | 4.7 | 15.6 |
| 1.00 | 1.00 | 0.00 | 0.6 | 3.9 | 11.0 |
| 1.25 | 1.00 | 0.00 | 0.2 | 3.5 | 9.9 |
| 1.50 | 1.00 | 0.00 | 0.0 | 3.3 | 9.0 |
| 1.75 | 1.00 | 0.00 | 0.0 | 3.3 | 8.4 |
| 2.00 | 1.00 | 0.00 | 0.0 | 3.4 | 7.9 |
| 1.00 | 0.50 | 0.00 | -3.6 | 0.8 | 5.8 |
| 1.00 | 0.75 | 0.00 | -0.9 | 2.9 | 9.4 |
| 1.00 | 1.25 | 0.00 | 1.5 | 4.6 | 14.9 |
| 1.00 | 1.50 | 0.00 | 2.1 | 5.2 | 15.7 |
| 1.00 | 1.75 | 0.00 | 2.7 | 5.6 | 15.8 |
| 1.00 | 2.00 | 0.00 | 3.1 | 6.0 | 15.8 |
| 1.00 | 1.00 | $\ln(3/4)$ | -0.5 | 2.4 | 1.0 |
| 1.00 | 1.00 | $\ln(5/6)$ | -1.0 | 2.2 | 6.6 |
| 1.00 | 1.00 | $\ln(11/12)$ | -0.3 | 2.8 | 2.9 |
| 1.00 | 1.00 | $\ln(1+(e-1)/12)$ | -0.4 | 1.7 | 1.2 |
| 1.00 | 1.00 | $\ln(1+(e-1)/6)$ | -0.3 | 3.0 | 1.4 |
| 1.00 | 1.00 | $\ln(1+(e-1)/4)$ | -0.5 | 1.8 | 2.3 |

Table 2.4: Relative Performance Ratio (RPR) under Different Mean / Standard Deviation / Correlation Multiplier of Testing Data Following Gamma Distribution

| Multiplier of mean | Multiplier of std | Multiplier of correlation | RPR of mean reduction (%) | RPR of std reduction (%) | RPR of extreme value reduction (%) |
|---|---|---|---|---|---|
| Flight Network $N_1$, 106 flights and 24 aircraft, best $\Gamma = 1.2$ | | | | | |
| 0.50 | 1.00 | 0.00 | 3.8 | 11.1 | 25.8 |
| 0.75 | 1.00 | 0.00 | 2.8 | 10.0 | 24.3 |
| 1.00 | 1.00 | 0.00 | 2.3 | 9.5 | 23.1 |
| 1.25 | 1.00 | 0.00 | 2.0 | 9.1 | 22.1 |
| 1.50 | 1.00 | 0.00 | 1.8 | 8.8 | 21.2 |
| 1.75 | 1.00 | 0.00 | 1.7 | 8.5 | 20.4 |
| 2.00 | 1.00 | 0.00 | 1.7 | 8.2 | 19.5 |
| 1.00 | 0.50 | 0.00 | -2.0 | 9.5 | 21.2 |
| 1.00 | 0.75 | 0.00 | 1.0 | 9.4 | 22.5 |
| 1.00 | 1.25 | 0.00 | 3.0 | 9.5 | 23.7 |
| 1.00 | 1.50 | 0.00 | 3.4 | 9.5 | 24.1 |
| 1.00 | 1.75 | 0.00 | 3.7 | 9.5 | 24.5 |
| 1.00 | 2.00 | 0.00 | 3.9 | 9.6 | 24.9 |
| 1.00 | 1.00 | $\ln(3/4)$ | 1.8 | 5.3 | 8.2 |
| 1.00 | 1.00 | $\ln(5/6)$ | 2.6 | 5.6 | 5.8 |
| 1.00 | 1.00 | $\ln(11/12)$ | -1.0 | 1.5 | 6.9 |
| 1.00 | 1.00 | $\ln(1+(e-1)/12)$ | 0.2 | 5.1 | 5.6 |
| 1.00 | 1.00 | $\ln(1+(e-1)/6)$ | 1.4 | 3.7 | -4.6 |
| 1.00 | 1.00 | $\ln(1+(e-1)/4)$ | 2.2 | 7.2 | 8.9 |
| Flight Network $N_2$, 117 flights and 23 aircraft, best $\Gamma = 1.4$ | | | | | |
| 0.50 | 1.00 | 0.00 | 1.5 | 8.5 | 15.3 |
| 0.75 | 1.00 | 0.00 | 0.6 | 7.3 | 15.4 |
| 1.00 | 1.00 | 0.00 | 0.0 | 6.3 | 15.3 |
| 1.25 | 1.00 | 0.00 | -0.4 | 5.6 | 15.0 |
| 1.50 | 1.00 | 0.00 | -0.4 | 5.2 | 14.7 |
| 1.75 | 1.00 | 0.00 | -0.4 | 5.0 | 14.5 |
| 2.00 | 1.00 | 0.00 | -0.3 | 4.8 | 14.2 |
| 1.00 | 0.50 | 0.00 | -3.6 | 3.3 | 12.8 |
| 1.00 | 0.75 | 0.00 | -1.3 | 5.3 | 14.7 |
| 1.00 | 1.25 | 0.00 | 0.8 | 6.9 | 15.5 |
| 1.00 | 1.50 | 0.00 | 1.4 | 7.4 | 15.6 |
| 1.00 | 1.75 | 0.00 | 1.8 | 7.7 | 15.6 |
| 1.00 | 2.00 | 0.00 | 2.2 | 8.0 | 15.6 |
| 1.00 | 1.00 | $\ln(3/4)$ | -1.6 | 2.3 | 0.8 |
| 1.00 | 1.00 | $\ln(5/6)$ | -2.3 | 2.0 | 6.5 |
| 1.00 | 1.00 | $\ln(11/12)$ | -1.1 | 2.7 | 2.5 |
| 1.00 | 1.00 | $\ln(1+(e-1)/12)$ | -1.5 | 1.6 | 1.2 |
| 1.00 | 1.00 | $\ln(1+(e-1)/6)$ | -1.4 | 2.6 | 0.0 |
| 1.00 | 1.00 | $\ln(1+(e-1)/4)$ | -1.5 | 1.8 | 4.7 |

Table 2.5: Relative Performance Ratio (RPR) under Different Mean / Standard Deviation / Correlation Multiplier of Testing Data Following Log-normal Distribution

# Chapter 3

# Applying Majority Judgment over a Polyhedral Candidate Space

Different from Chapter 2 where we aim at mitigating impact of flight delay by improving airline operations, in Chapters 3 and 4, we explore how to reduce delay cost by soliciting and incorporating airline preferences into planning air traffic flow management (ATFM) programs. In this chapter, we focus on developing a voting mechanism to gather airline preferences over different ATFM program designs. However, please note that our development here is not confined to the ATFM context. Instead, we present a generalized voting mechanism that is applicable to settings where candidates are drawn from a domain with infinite cardinality described as polyhedral sets. In the following Chapter 4, we refocus on ATFM context and present a detailed case study to quantify the benefits of applying the developed voting mechanism to ground delay program design.

## 3.1 Introduction

Voting is concerned with aggregating evaluations over a multitude of voters, in ways that the final outcome has appeal to a large cross-section of the decision-makers. Over centuries, investigators seeking a fool-proof voting system have been riddled by a result - famously known in social choice theory as Arrow's Impossibility Theorem

(Arrow, 1951). It states: "when voters have three or more distinct alternatives, no voting system can convert the ranked preferences of individuals into a community-wide (complete and transitive) ranking while also meeting a certain set of criteria, namely: unrestricted domain, non-dictatorship, Pareto efficiency, and independence of irrelevant alternatives". Majority Judgment (MJ) is a recently proposed procedure (Balinski and Laraki, 2010, 2014) which involves grading - instead of preference ranking - of each candidate. With this richer preference elicitation, MJ bypasses Arrow's Impossibility Theorem. The method also enjoys enhanced strategy-proofness due to its median-seeking criterion, and hence, its authors claim it to be "a better alternative to all other known voting and ranking methods, in theory and in practice."

MJ has already been practiced in spirit in many contests and juries around the world, as well as in a few political elections. It takes as input the "grades" given by the voters, and produces a *majority grade* for each candidate as an output. The majority grade can be used to select a winner (called a *majority winner*) and/or compute rank-orderings (called *majority ranking*) as well. If voters have equal weights, then the majority grade of a candidate is the highest grade such that an absolute majority of the voters grade that candidate at least as highly as that grade. If we order voters' grades in a descending order, in the case of an odd number of voters, it is the median of the grades; if there are even number of voters, then it is the lower of the two middlemost grades. On the other hand, if voters have different weights, then the majority grade is the highest grade such that a subset of voters with an absolute majority of the total weight (the sum of their weights should be strictly greater than one half of the total weight) grade that candidate at least as highly as that grade.

| Candidates: | $C_1$ | $C_2$ | $C_3$ |
|---|---|---|---|
| Best Grade: | 5 (Excellent) | 3 (Good) | 4 (Very Good) |
| . | 4 (Very Good) | 3 (Good) | 4 (Very Good) |
| . | 4 (Very Good) | 3 (Good) | 4 (Very Good) |
| Majority Grade: | 4 (Very Good) | 1 (Reject) | 3 (Good) |
| . | 2 (Passable) | 1 (Reject) | 1 (Reject) |
| Worst Grade: | 2 (Passable) | 1 (Reject) | 1 (Reject) |

Table 3.1: MJ Example

We illustrate MJ using the example in Table 3.1. Suppose there are six voters with equal weights, voting on three candidates: $C_1$, $C_2$, $C_3$. They assign one of these five grades to each candidate: 5 (Excellent), 4 (Very Good), 3 (Good), 2 (Passable), 1 (Reject). The grades thus obtained for each candidate by voting are then sorted from best to worst, as given in Table 1. The majority grade for each candidate (marked "Majority Grade") is the fourth grade from the top, because an absolute majority of (four out of six) voters would give at least that grade to the candidate. The majority ranking for the example is: $C_1 \succ C_3 \succ C_2$, according to the order of the majority grades. $C_1$ is the majority winner. The method involves additional procedures for breaking ties when ranking candidates and when deciding majority winner. We refer interested readers to Balinski and Laraki (2010) for more details. MJ requires a common language accepted by all the voters for grading the candidates. Grades may be either discrete (as above) or continuous. A continuous grading language could be: $[0, 1]$, where 0 is commonly understood as "unacceptable", and 1 as the "most favorable".

The computation and ranking of majority grades require a complete evaluation of all candidates from all voters. We study important social choice problems where such evaluation is impossible to execute due to the huge (or even infinite) number of possible candidates. Our work is motivated by a problem in ATFM where a consensus input is required from a set of competing airlines in order to design and implement certain ATFM programs (see Swaroop and Ball (2012); Evans et al. (2016); Ball et al. (2014b, 2017) for a description of this application and some preliminary ideas on the approach discussed in this chapter). The required input is a numerical vector out of a polyhedral feasible region that specifies to an air navigation service provider how certain tradeoffs among performance criteria should be made. A second more widely-known application involves the use of group decision-making mechanisms to find a capital budget allocation. There are many capital budgeting application contexts, particularly in the public sector, where some sort of consensus must be reached among multiple decision makers in determining the allocation of a capital budget. In a basic problem statement, there are $n$ projects and a total available budget, $C$. The desired outcome is a feasible budget allocation: $m_i$ for each project $i = 1, ..., n$. The set of feasible allocations is then defined by $P = \{(m_1, \cdots, m_n) : \ \sum_{i=1}^{n} m_i \leq C, \ m_i \geq 0, \ \forall i = 1, \cdots, n\}$. Viewing each feasible solution as an MJ candidate, the number of candidates is infinite and uncountable, making the direct application of MJ impossible. In this chapter, we develop mechanisms and computational approaches to apply MJ within this context.

### 3.1.1 Relevant Research

Extensive studies have been conducted on the subjects of voting and elections. The most widely-used methods include Approval Voting, Point Summing, Borda Count, among others. As already compared in Balinski and Laraki (2010, 2014), MJ outperforms these methods in both theory and practice in several criteria such as bypassing impossibility theorems, rate of successful manipulation and expressiveness of the votes. Thus, although these other methods also have the potential to be extended to handle candidate spaces with infinite size, we are not going to explore them.

Aside from voting systems which only determine a single winner, there also are group-ranking approaches that take ranking input from voters and output preference rankings of all candidates. Kemeny and Snell (1962) define the group-ranking problem when each voter's input is an ordinal preference ranking. The group ranking is the one that minimizes the deviation from individual voters' rankings based on some distance measure. The analytic hierarchy process (AHP) developed by Saaty (1977) for multicriteria decision making has also been used in group-rankings that requires intensity ranking (a pairwise comparison that provides the magnitude of the degree of preference) input from voters. AHP turns an intensity ranking matrix into a vector of weights for all candidates. Since its invention, it has been successfully applied to numerous areas to evaluate, rank or select candidates. However, like most of the group-ranking methods including Kemeny and Snell (1962), AHP builds upon the assumption that a full-ranking list for all candidates is provided by each voter. This prevents us from directly applying this approach. There is perhaps also a philosophical difference in our approach using MJ and the approaches using AHP. The AHP literature typically assumes that participants do not have precise knowledge of their own preferences. Preferences are quantified through an iterative process using pairwise comparisons. In our MJ-based approach, we address applications where voters can precisely estimate their grade functions, which map all candidates into numerical grades. For example, in our capital budgeting application, it is possible to relate every feasible capital allocation to a numerical grade that measures voter's preference, such as expected returns on investment; and in our air traffic flow management application, it is possible (at least conceptually) to relate an airline's grade function to that airline's expected financial performance under every candidate vector in question.

Hochbaum and Levin (2006) propose generalized optimization approaches to produce group rankings where voters' inputs are also intensity rankings. Different from previous approaches, their approaches are amenable to partial preference lists: each voter provides a partial list that evaluates and compares only a subset of all the candidates. The partial preference lists are motivated by the fact that different voters may have different areas of expertise in reviewing candidates, and voters may not have the capability to review all the candidates in time, the latter of which is also the case in our problem. We want to point out here that although the feature in Hochbaum and Levin (2006) of incorporating partial lists is relevant to the context we are considering, it is difficult to apply this approach to our case where the candidate space is modeled as a polyhedron. Especially when the dimension of the polyhedron is high, coming up with good candidates for voting seems non-trivial. Also, in most of the existing group-ranking approaches, strategic behaviors are not considered. However, in the applications that we consider, it is also the case that the voters are competitors so, in many cases, there would be few or no incentives for the voters to cooperate or to seek a consensus. Building our work on top of MJ helps to restrict strategic manipulations.

Finally, our work is related to the computational social choice literature. In particular, Xia (2011) investigates "combinatorial voting" where the candidates have a multi-attribute structure. There is a set of attributes and a candidate is identified by the values these attributes take. Combinatorial voting faces similar challenges as our problem – the number of candidates is exponentially large. Xia (2011) thus develops new preference eliciting and aggregation methods to solicit preferences and compute winner efficiently. However, these methods cannot be directly applied to our context because they rely on the requirement that each attribute in the combinatorial domain takes a value from a finite set while in our case it could take any continuous value from an interval.

### 3.1.2 Contributions

We now summarize the contributions of this chapter. First, we provide structure to the problem of applying MJ over a polyhedral candidate space and define an optimization model that is useful for approaching this problem. Specifically, in Section 3.2, after describing the general problem setting, we define an optimization model that can find the majority winner over a polyhedral candidate space. This model assumes "perfect information" in the sense that the voters' grade functions over the entire polyhedral candidate space are inputs to the model. While such information would not be available in a practical setting, this model is very useful in subsequent analysis in this chapter and should be useful in further study of this problem. Second, in Section 3.3, we define two specific mechanisms for the problem of interest assuming voters will grade or act truthfully. The first iteratively generates candidates and submits each of these to voters for grading. The second requires a single input from each voter, namely, the most preferred candidate. A consensus candidate is then generated based on this information using techniques from robust optimization. These two methods have differences in information requirements and also differences in the quality of solutions produced and the overall process execution times. In Section 3.4, we investigate the strategic behaviors of the two developed mechanisms. In particular, we develop computational tools to evaluate the likelihood that gains are possible from strategic manipulation. Finally, in Section 3.5, we describe two specific applications for the problem of interest and experimentally demonstrate both the effectiveness and strategy-proofness of the mechanisms defined in Section 3.3 for these applications.

A paper based on this chapter can be found in Yan et al. (2017a).

## 3.2 General Problem Setting

In this section, we discuss the general problem setting. Section 3.2.1 defines the basic problem structure and gives underlying assumptions. Section 3.2.2 defines an important optimization model that is fundamental to subsequent analysis.

### 3.2.1 Problem Setting and Assumptions

The general context for the problem we address involves a group of voters, $N$, and a central planner who jointly seek to select a candidate. These voters are not necessarily cooperative nor do they necessarily have common goals. The form of the candidate we seek is a numerical vector $\mathbf{m} \in \mathbb{R}^n$ that is within a candidate space $P$, which is modeled as a polyhedron. $n$ is the dimension of $P$. We assume that each voter $i \in N$ has a grade function $g_i(\cdot)$ that maps each candidate $\mathbf{m} \in P$ to a real number (common language across all voters) that represents the value of $\mathbf{m}$ to voter $i$. We assume that $g_i(\cdot)$ is known to each voter $i$, but is unknown to the other voters and the central planner. Each voter $i \in N$ also has a weight $w_i \in \mathbb{R}^+$. We address the problem of designing a mechanism in which the central planner exchanges information with the voters and produces the desired candidate $\mathbf{m}$. We apply MJ in this setting, i.e., we aim to ensure that the final selected candidate $\mathbf{m}$ has the highest majority grade among all candidates in $P$. We now describe some assumptions regarding the structure of the candidate space $P$ and the grade functions:

**Assumption 3.1.** *The candidate space $P$ is a bounded polyhedron in the positive orthant.*

The polyhedral assumption induces convexity of the candidate space, which is useful to facilitate defining grade functions over $P$, as we will see in Assumption 3.2. We require $P$ to be bounded to model the context where resources are limited. We restrict the candidate space to be in the positive orthant because each component of our candidate usually means certain amount of resource allocation or a specific performance metric.

**Assumption 3.2.** *Each voter's grade function $g_i(\mathbf{m})$ is continuous, concave and component-wise non-decreasing in $\mathbf{m}$, i.e., all partial derivatives of $g_i(\mathbf{m})$ are non-negative. The common grading language allows for continuous grades in $[0, 1]$, where a higher grade implies better acceptability by a voter. Grade 1 is always achievable and corresponds to the voter's most preferred candidate:* $\arg\max_{\mathbf{m}\in P} g_i(\mathbf{m})$.

The assumption that the grade should be in $[0, 1]$ could be ensured by simple rescaling if the original value function did not have this property. Note that the assumption that the best candidate achieves grade 1, rather than some value within $[0, 1]$, is for simplicity and can indeed be relaxed. We will further discuss this issue in Section 3.3.2 where this particular assumption is used. Continuity would be a reasonable assumption for many applications: very small changes in a candidate's component values should not induce jumps in grades. The non-decreasing assumption implies that higher values of the individual components of a candidate $\mathbf{m}$ are as good as or better than lower values. The concavity assumption expresses the diminishing returns property.

### 3.2.2 Computing the Majority Winner with Perfect Information

In this section we describe an optimization model whose solution is the majority winner, i.e. the candidate in $P$ that has the highest majority grade. This model and its solution, however, do not provide a practical solution to our problem since this model requires perfect knowledge of each voter's grade function. Nevertheless, this model will be useful as a key building block for two proposed mechanisms in Section 3.3 to address the imperfect information case and also in experimental evaluations of proposed mechanisms.

We start the analysis with an example of three voters grading one single candidate (Table 3.2a). The total weight of the three voters sums up to 100. Based on the weight information, we define *majority set* as the following,

**Definition 3.1** (Majority Set)**.** *A majority set $B$ is a subset of voters which satisfies* *the following property:*

$$\sum_{i \in B} w_i > \frac{\sum_{i \in N} w_i}{2},$$

*i.e. the sum of weights of all the voters in $B$ exceeds one half of the total weight. We further define the set of all possible majority sets as $\mathcal{B}$.*

| **Voter** | **Weight** | **Grade** |
|---|---|---|
| 1 | 23 | 1.00 |
| 2 | 46 | 0.90 |
| 3 | 31 | 0.85 |

(a) Voters' grades on one candidate

| **Majority Set** | **Voter with Minimal Grade** | **Minimum Grade** |
|---|---|---|
| (1,2) | 2 | 0.90 |
| (1,3) | 3 | 0.85 |
| (2,3) | 3 | 0.85 |
| (1,2,3) | 3 | 0.85 |

(b) All possible majority sets and their minimum grades

Table 3.2: MJ Illustration

In the example from Table 3.2a, we can easily see that the set of all possible majority sets $\mathcal{B}$ is $\{(1,2),(2,3),(1,3),(1,2,3)\}$. In Table 3.2b, we list all possible majority sets in the first column. For each majority set, we specify the voter which gives the minimum grade in the majority set (column 2) and the corresponding minimum grade (column 3). We would like to show that the maximum value of column 3, 0.90 in this case, is indeed the majority grade of this candidate. We formalize this result in the following lemma.

**Lemma 3.1.** *The majority grade $g_{MJ}(\mathbf{m})$ of a candidate $\mathbf{m}$ can be equivalently calculated as follows:*

$$g_{MJ}(\mathbf{m}) = \max_{B \in \mathcal{B}} \left\{ \min_{i \in B} g_i(\mathbf{m}) \right\} \tag{3.1}$$

*Proof.* This result follows directly from the definition of the majority grade: it is the highest grade such that there exists a majority set where all voters in the majority set grade that candidate at least as highly as that grade. The inner minimization in Eq. (3.1) is based on the fact that $\min_{i \in B} g_i(\mathbf{m})$ is the highest grade such that every voter in majority set $B$ will give a grade greater than or equal to it. The outer maximization in Eq. (3.1) then aims to maximize that grade by searching over all possible majority sets. □

Lemma 3.1 above inspires us to develop a mathematical program to search over all possible majority sets $\mathcal{B}$ in order to find the majority winner. We define a vector of continuous variables, $\mathbf{m}$, as the candidate vector that we pick. Also, we define binary variables $x_i = 1, \forall i \in N$ if voter $i$ is chosen to be in the majority set. Similarly, we define binary variables $y_i = 1, \forall i \in N$ if voter $i$ assigns the lowest grade to $\mathbf{m}$ among all voters in the majority set indicated by $\mathbf{x}$. Continuous variables $v_i, \forall i \in N$ represent the set of grades given by voter $i$ to candidate $\mathbf{m}$. Finally, auxiliary continuous variable $z$ is equal to the majority grade of candidate $\mathbf{m}$ as we will show next in Theorem 3.1. Theorem 3.1 introduces a mathematical program that computes the majority winner over the entire candidate space $P$.

**Theorem 3.1.** *Given each voter's grade function $g_i(\cdot)$, weight $w_i$, and feasible polyhedron $P$, the candidate with the highest majority grade $g^*_{MJ}$ is the optimal solution $\mathbf{m}^*$ of the following mathematical program:*

$$g^*_{MJ} = \max_{\mathbf{x},\mathbf{y},\mathbf{v},\mathbf{m},z} \quad z \tag{3.2}$$

$$s.t. \quad \sum_{i \in N} w_i x_i \geq \frac{\sum_{i \in N} w_i}{2} + \epsilon_0 \tag{3.3}$$

$$\sum_{i \in N} y_i = 1 \tag{3.4}$$

$$x_i \geq y_i, \qquad \forall i \in N \tag{3.5}$$

$$\mathbf{m} \in P \tag{3.6}$$

$$v_i = g_i(\mathbf{m}), \qquad \forall i \in N \tag{3.7}$$

$$v_j \leq G^{max}(1 - x_i) + G^{max}(1 - y_j) + v_i, \quad \forall i \in N, j \in N : i \neq j \tag{3.8}$$

$$z \leq v_i + G^{max}(1 - y_i), \qquad \forall i \in N \tag{3.9}$$

$$x_i, y_i \in \{0, 1\}, \qquad \forall i \in N \tag{3.10}$$

*where $G^{max}$ is the maximum possible grade (under our assumptions $G^{max} = 1$) and $\epsilon_0$*

*can be any positive number smaller than*

$$\min_{\left\{\mathbf{x}\in\{0,1\}^n:\ \sum_{i=1}^n w_i x_i > \frac{\sum_{i=1}^n w_i}{2}\right\}} \left( \sum_{i=1}^n w_i x_i - \frac{\sum_{i=1}^n w_i}{2} \right)$$

*which is the smallest positive difference between* $\sum_{i=1}^n w_i x_i$ *and* $\frac{\sum_{i=1}^n w_i}{2}$ *for all* $\mathbf{x} \in \{0,1\}^n$.

*Proof.* Constraint (3.3) indicates that the sum of weights in the selected majority set should exceed one half of the total weight. Constraint (3.4) is to ensure that we select one voter as the one who assigns the minimum grade to $\mathbf{m}$ among all voters in the selected majority set. There could be multiple voters who give a grade equal to this minimum grade, in which case any one of them can be chosen. Constraints (3.5) make sure that if voter $i$ is selected to be the one to give the minimum grade, then it should definitely be included in the majority set. Constraint (3.6) restricts $\mathbf{m}$ to be selected from the feasible polyhedron $P$. Constraints (3.7) state that $v_i$ is the grade that voter $i$ gives to candidate $\mathbf{m}$ according to its grade function $g_i(\cdot)$. Constraints (3.8) ensure that if voter $j$ is selected to be the one to give the minimum grade to candidate $m$ (i.e., if $y_j = 1$), then its grade $v_j$ should indeed be less than or equal to other grades in the selected majority set. To see this, note that if $y_j = 1$ and $x_i = 1$, which means voter $i$ is included in the majority set and voter $j$ is selected to be the one who gives the minimum grade, then constraints (3.8) become $v_j \leq v_i$; otherwise, constraints (3.8) are redundant. Finally, constraints (3.9) along with objective (3.2) ensure that auxiliary variable $z$ equals the minimum grade of $\mathbf{m}$ in the selected majority set. To see that, note that if $y_i = 1$, then constraints (3.9) are reduced to $z \leq v_i$; and if $y_i = 0$, then constraints (3.9) are redundant. Now we can see that the mathematical program described above is the same as $\max_{\mathbf{m}\in P, B\in\mathcal{B}} \{\min_{i\in B} g_i(\mathbf{m})\}$. Finally, from Lemma 3.1, we know that this is equivalent to $\max_{\mathbf{m}\in P} g_{\mathrm{MJ}}(\mathbf{m})$. □

Note that in theory multiple candidates in $P$ could have the same highest majority grade. In this situation certain tie-breaking rules, as developed by Balinski and Laraki (2010) in the original MJ theory, need to be imposed. However, we don't explicitly model tie-breaking rules in the formulation (3.2) - (3.10) because ties happen relatively rarely in the contexts of interest since we allow voters to grade continuously in $[0,1]$ according to certain grade functions over an uncountably infinite candidate space.

Although Theorem 3.1 allows us to formulate the problem of finding a majority winner over a polyhedral candidate set, the resultant formulation is not computationally approachable because of the non-convexity introduced by constraints (3.7). Since the grade functions $g_i(\cdot), \forall i \in N$ are concave, these equality constraints make the formulation a mixed-integer non-convex program. Fortunately, Corollary 3.1 below shows that constraints (3.7) can be relaxed without loss of optimality so that the resultant formulation becomes a mixed-integer convex program where effective and exact algorithms exist (Byrd et al., 2006; Bonami et al., 2008; Achterberg, 2009; Lubin et al., 2016). Corollary 3.1 also exhibits a simplification of the formulation. Proof of Corollary 3.1 can be found in Appendix B.5.1.

**Corollary 3.1.** *Constraints* (3.5) *are redundant and constraints* (3.7) *can be equivalently relaxed to $v_i \leq g_i(\mathbf{m}), \forall i \in N$. Thus the following mixed-integer convex programming formulation MJ-OPT is equivalent to the formulation described in* (3.2)-(3.10).

$$
\begin{aligned}
\text{(MJ-OPT)} \quad \max \quad & z \\
s.t. \quad & (3.3) - (3.4), (3.6) \\
& v_i \leq g_i(\mathbf{m}), \quad \forall i \in N \qquad (3.11) \\
& (3.8) - (3.10)
\end{aligned}
$$

## 3.3 Mechanism Design and Underlying Models

Section 3.2.2 enables the computation of a majority winner if a central planner has knowledge of every voter's grade function. However, as discussed earlier, this is typically private information known only to the voter. In Sections 3.3.1 and 3.3.2 we develop two mechanisms that employ Theorem 3.1 but do not require explicit knowledge of grade functions. Note that in this section, we assume that voters submit their inputs truthfully according to their grade functions, i.e., there's no strategic behavior. We will discuss strategic behavior in Section 3.4.

### 3.3.1 Mechanism I: An Iterative Approach

Our first method is an iterative approach. To start, the voters are provided an initial set of candidates and are asked to grade these candidates. The central planner then statistically estimates the grade functions based on each voter's submitted grades. We denote the estimated grade functions as $\hat{g}_i(\mathbf{m}),\ \forall i \in N$. We don't make any assumptions on the central planner's knowledge of the functional form of each voter's true grade function, apart from knowing that it is a concave and component-wise non-decreasing function. So the central planner cannot directly estimate the parameters. Instead, we use a non-parametric regression method called "convex regression" (Kuosmanen, 2008; Lim and Glynn, 2012) to fit the best concave and component-wise non-decreasing function to estimate each voter's true grade function. Interestingly this best fit function turns out to be a piecewise-linear function with number of pieces equal to the number of graded candidates. Suppose the central planner provides $k$ candidates to the voters to grade. For each voter $i$, the estimation problem takes two types of inputs: (1) the set of candidates, and (2) voter $i$'s grade associated with each of these candidates. The estimation problem outputs the best-fit piecewise-linear function's coefficients $(\mathbf{c}_i^j, d_i^j)$ for each piece $j \in \{1, \cdots, k\}$. Thus, the form of the best-fit piecewise-linear concave function for voter $i$ is $\hat{g}_i(\mathbf{m}) = \min_{j\in\{1,\cdots,k\}} \left\{(\mathbf{c}_i^j)^T\mathbf{m} + d_i^j\right\}$. The detailed formulation is presented in Appendix B.1.

The central planner then generates new candidates based on the estimated grade functions $\hat{g}_i(\mathbf{m}),\ \forall i \in N$. The generated candidates should be of good quality, in the sense that they should have relatively high majority grades. Thus we propose model $\widehat{\text{MJ}}$-OPT as a building block for this iterative approach. Model $\widehat{\text{MJ}}$-OPT, which is presented next, produces the majority winner with the estimated grade functions. The only difference between this formulation and formulation MJ-OPT is that we replace $x_i \le g_i(\mathbf{m}),\ \forall i \in N$ in constraints (3.7) with $x_i \le \hat{g}_i(\mathbf{m}),\ \forall i \in N$.

Note that $x_i \leq \hat{g}_i(\mathbf{m}),\ \forall i \in N$ is equivalent to constraints (3.12) since $\hat{g}_i(\mathbf{m}) = \min_{j\in\{1,\cdots,k\}} \left\{(\mathbf{c}_i^j)^T\mathbf{m} + d_i^j\right\}$.

$$
\begin{aligned}
(\widehat{\text{MJ-OPT}})\quad \max\quad & z \\
\text{s.t.}\quad & (3.3)-(3.4),(3.6) \\
& v_i \leq (\mathbf{c}_i^j)^T\mathbf{m} + d_i^j,\ \ \forall i \in N,\ j \in \{1,\cdots,k\} \qquad (3.12) \\
& (3.8)-(3.10)
\end{aligned}
$$

Based on the discussion above, we develop an algorithm to iteratively generate new candidates, refine the estimation of voter grade functions and thus approach the true majority winner over the entire polyhedral space. The overall flow of the algorithm is summarized in Algorithm 3.

**Algorithm 3** Mechanism I: An Iterative Approach

```
Initialize:
    Select a set of initial candidates, $S$, from the efficient frontier of feasible
    space $P$.
    num_iter = 1
while num_iter $\leq n_{\max}$ do
    Obtain each voter's grade for all newly added candidates in set $S$.
    Estimate each voter's grade function by running the estimation problem using
    the obtained grades of all candidates in set $S$.
    Generate a new candidate $\mathbf{m}_{\text{new}}$ by solving $\widehat{\text{MJ-OPT}}$ .
    if num_iter $= n_{\max}$ then
        Add $\mathbf{m}_{\text{new}}$ to set $S$.
    else
        if $||\mathbf{m}_{\text{new}} - \mathbf{m}||_2 \geq \epsilon,\ \forall \mathbf{m} \in S$ then
            Add $\mathbf{m}_{\text{new}}$ to set $S$.
        else
            Randomly perturb $\mathbf{m}_{\text{new}}$ to get another candidate $\mathbf{m}'_{\text{new}}$ on the efficient
            frontier of $P$.
            Add $\mathbf{m}'_{\text{new}}$ to set $S$.
        end if
    end if
    num_iter = num_iter + 1
end while
return the candidate with the highest majority grade in set $S$.
```

Let $\bar{P}$ be the *efficient frontier* of the feasible candidate space $P$, where efficient frontier is defined as the subset of the candidate space such that no candidate in $P$ dominates any candidates in $\bar{P}$ in terms of the component values. In other words, there does not exist a candidate $\mathbf{m} \in P$ such that the following holds: $m_i \geq m'_i$, $\forall i \in \{1, \cdots, n\}$ and $\exists i \in \{1, \cdots, n\}$ such that $m_i > m'_i$, for some $\mathbf{m'} \in \bar{P}$. We can restrict ourselves to the candidates on the efficient frontier because any candidates not on the efficient frontier will always be dominated by some candidates on the efficient frontier in terms of majority grade. Thus, Algorithm 3 starts with a set of initial candidates on the efficient frontier. The algorithm then enters into a loop to (1) first obtain the grades of the candidates that haven't been graded yet, then (2) estimate each voter's grade function by running the convex regression model, then (3) generate a new candidate $\mathbf{m}_{\text{new}}$ by solving $\widehat{\text{MJ}}$-OPT, and finally (4) add an appropriate new candidate based on $\mathbf{m}_{\text{new}}$ to the set $S$. Step (4) here is a bit more involved than others: If it is the last iteration, we simply add $\mathbf{m}_{\text{new}}$ to $S$. If not, we first check whether the candidate obtained from solving $\widehat{\text{MJ}}$-OPT is distinct enough from all existing candidates in the set $S$ by measuring their euclidean distance to see whether it is at least equal to a certain threshold $\epsilon$. If so, we add it to the set $S$; otherwise, we randomly perturb $\mathbf{m}_{\text{new}}$ to get another candidate on the efficient frontier, and add the perturbed candidate to $S$. The details of the random perturbation method that we use are outlined in Appendix B.3. The perturbation prevents the algorithm from generating similar or even identical candidates, which deteriorate its performance. The algorithm terminates after a certain pre-specified number of iterations $n_{\text{max}}$ and returns the candidate with the highest majority grade among the candidates in set $S$. Theorem 3.2 below reveals the convergence properties of Algorithm 3 under a mild assumption of the random perturbation method. Its proof can be found in Appendix B.5.2.

**Theorem 3.2.** *Let $\bar{P}$ be the efficient frontier of the candidate space $P$ and $\mathbf{m'}$ be the candidate we get after randomly perturbing $\mathbf{m}$. If $\forall \mathbf{m} \in \bar{P}$ and for each subset $A \subset \bar{P}$, we have $Pr(\mathbf{m'} \in A) > 0$, then Algorithm 3 converges to the true majority winner almost surely as the total number of iterations $n_{max} \to \infty$.*

### 3.3.2 Mechanism II: A Single-Shot Robust Approach

The multi-round estimation approach developed in Section 3.3.1 relies on asking voters to evaluate and grade multiple candidates. In practice, such evaluations might be expensive, time-consuming or even somewhat ambiguous to voters. Alternatively, in many contexts, it might be easier for voters to come up with their most preferred candidate. For instance, in the capital budgeting example we discussed in Section 3.1, a voter may find it easier to generate its most preferred capital allocation rather than evaluating several alternative allocations. However, we note that the strategyproofness properties for MJ may not hold for this mechanism because we change the input from grades to candidates.

In this section we propose a second mechanism that requires voters to only submit their most preferred candidate, $\mathbf{m}_i^* = \arg\max_{\mathbf{m}\in P} g_i(\mathbf{m}),\ \forall i \in N$. This is a candidate that maximizes that voter's own grade function. The approach is single-shot (or non-iterative) in that it requires only one round of voter inputs. To better understand this approach, we first define the set of valid grade functions $G_i(\cdot), \forall i \in N$ as follows.

**Definition 3.2** (set of valid grade functions: $G_i(\cdot),\ \forall i \in N$)**.** *For each voter $i$ with an associated most preferred vector, $\mathbf{m}_i^*$, we define a* set of valid grade functions $G_i(\cdot)$ *such that for any $g_i(\cdot) \in G_i(\cdot)$, $g_i(\cdot)$ satisfies Assumption 3.2 and $g_i(\mathbf{m_i^*}) = 1$ (i.e., $g_i(\cdot)$ is maximized at $\mathbf{m}_i^*$).*

We call this approach a *single-shot robust approach* because we are looking for the candidate $\mathbf{m}^{\text{rob}}$ which has the best worst-case guarantee in terms of majority grade when the voters' grade functions are allowed to vary within the set of valid grade functions. In other words, $\mathbf{m}^{\text{rob}}$ is the optimal solution of the following problem:

$$\mathbf{m}^{\text{rob}} = \arg\max_{\mathbf{m}\in P} \left\{ \min_{g_1\in G_1,\cdots,g_{|N|}\in G_{|N|}} g_{\text{MJ}} \left\{ g_1(\mathbf{m}), \cdots, g_{|N|}(\mathbf{m}) \right\} \right\}, \qquad (3.13)$$

where $g_{\text{MJ}} \left\{ g_1(\mathbf{m}), \cdots, g_{|N|}(\mathbf{m}) \right\}$ is the function that calculates the majority grade of candidate $\mathbf{m}$ given each voter's grade function $g_i(\cdot)$. Eq. (3.13) is a robust optimization problem where the uncertainty set is defined as the set of valid grade functions. A similar idea has also been utilized in Armbruster and Delage (2015) to address decision making under uncertainty when decision maker's utility function is not completely known.

In order to solve the problem defined in (3.13), we first define the lower bound grade function as follows.

**Definition 3.3** (lower bound grade function: $\tilde{g}_i(\cdot),\ \forall i \in N$)**.** *For each voter $i$, we define its lower bound grade function $\tilde{g}_i(\cdot) \in G_i(\cdot)$ such that $\forall g_i(\cdot) \in G_i(\cdot)$, we have $\tilde{g}_i(\mathbf{m}) \leq g_i(\mathbf{m}),\ \forall \mathbf{m} \in P$.*

Essentially, the lower bound grade function $\tilde{g}_i(\cdot)$ is the one that takes the point-wise minimum value among all functions in $G_i(\cdot)$. We show in Lemma 3.2 below how the inner problem in Eq. (3.13) can be reformulated as a closed-form formula.

**Lemma 3.2.**

$$\min_{g_1 \in G_1, \cdots, g_{|N|} \in G_{|N|}} g_{\text{MJ}} \left\{ g_1(\mathbf{m}), \cdots, g_{|N|}(\mathbf{m}) \right\} = g_{\text{MJ}} \left\{ \tilde{g}_1(\mathbf{m}), \cdots, \tilde{g}_{|N|}(\mathbf{m}) \right\},\ \forall \mathbf{m} \in P$$

.

*Proof.* Suppose there exists an optimal solution $\left\{ g'_1, \cdots, g'_{|N|} \right\}$ of the inner minimization problem in (3.13). We know that $g'_1(\mathbf{m}) \geq \tilde{g}_1(\mathbf{m}), \cdots, g'_{|N|}(\mathbf{m}) \geq \tilde{g}_{|N|}(\mathbf{m}),\ \forall \mathbf{m} \in P$ by the definition of lower bound grade functions (Definition 3.3). By Lemma 3.1 we know that $\forall \mathbf{m} \in P$ we have,

$$\begin{aligned} g_{\text{MJ}} \left\{ g'_1(\mathbf{m}), \cdots, g'_{|N|}(\mathbf{m}) \right\} &= \max_{B \in \mathcal{B}} \left\{ \min_{i \in B} g'_i(\mathbf{m}) \right\} \\ &\geq \max_{B \in \mathcal{B}} \left\{ \min_{i \in B} \tilde{g}_i(\mathbf{m}) \right\} \\ &= g_{\text{MJ}} \left\{ \tilde{g}_1(\mathbf{m}), \cdots, \tilde{g}_{|N|}(\mathbf{m}) \right\}. \end{aligned}$$

This proves that $\left\{ \tilde{g}_1, \cdots, \tilde{g}_{|N|} \right\}$ is also an optimal solution. □

With Lemma 3.2, the robust optimization problem in Eq. (3.13) is equivalent to

$$\mathbf{m}^{\text{rob}} = \arg\max_{\mathbf{m}\in P} \; g_{\text{MJ}} \left\{ \tilde{g}_1(\mathbf{m}), \cdots, \tilde{g}_{|N|}(\mathbf{m}) \right\}$$

which can be solved using the formulation provided in Theorem 3.1 with constraints (3.7) replaced by $v_i \leq \tilde{g}_i(\mathbf{m})$, $\forall i \in N$. In Section 3.3.2, we show that under an additional assumption, the lower bound grade functions $\tilde{g}_i(\mathbf{m})$, $\forall i \in N$ actually take the form of piecewise-linear concave functions and can be constructed via closed-form formulae. Then, in Section 3.3.2, we describe the mathematical formulation to obtain $\mathbf{m}^{\text{rob}}$ .

### Constructing Lower Bound Grade Functions $\tilde{g}_i(\cdot), \forall i \in N$

In order to construct lower bound grade functions $\tilde{g}_i(\cdot), \forall i \in N$, we impose an assumption on the feasible candidate space in addition to Assumption 3.1.

**Assumption 3.3.** *In addition to satisfying Assumption 3.1, the candidate space $P$ also contains the origin point.*

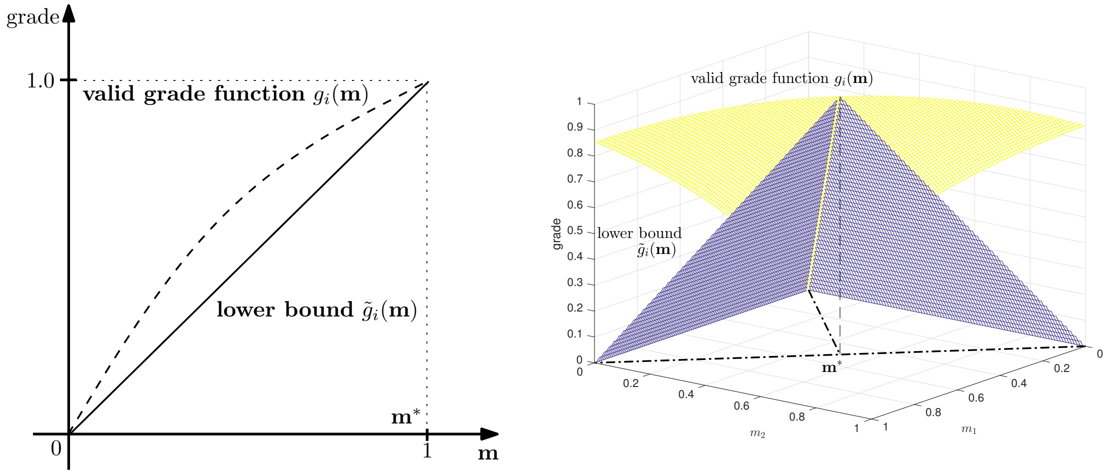
We show that with Assumption 3.3, the lower bound grade functions $\tilde{g}_i(\cdot)$, $\forall i \in N$ are piecewise-linear concave functions, as we show in Theorem 3.3 below. The detailed proof can be found in Appendix B.5.3.

**Theorem 3.3.** *Under Assumptions 3.1, 3.2 and 3.3, the lower bound grade functions are the following piecewise-linear functions:*

$$\tilde{g}_i(\mathbf{m}) = \min_{j \in \{1,\cdots,n\}:[\mathbf{m}_i^*]_j > 0} \left( \frac{1}{[\mathbf{m}_i^*]_j} \right) m_j, \quad \forall i \in N,$$

*where $[\mathbf{m}_i^*]_j$ denotes the $j^{th}$ element of voter $i$'s most preferred candidate $\mathbf{m}_i^*$.*

Figure 3-1 illustrates two examples of the lower bound grade functions constructed by Theorem 3.3. Figure 3-1a is a one-dimensional case, where the feasible candidate space is $[0, 1]$ and $m^* = 1$ is the most preferred candidate. The dashed curve in Figure 3-1a represents a valid grade function $g_i(m)$, while the linear function represented by the solid line is the lower bound grade function $\tilde{g}_i(m) = m$. Similarly, Figure 3-1b represents a two-dimensional case, where the feasible region is $\{\mathbf{m} : m_1 + m_2 \leq 1, m_1 \geq 0, m_2 \geq 0\}$ and the most preferred candidate is $\mathbf{m}^* = (1/2, 1/2)$. The yellow curved surface represents a valid grade function $g_i(\mathbf{m})$ while the purple piecewise-linear surface consisting of parts of two planes represents the lower bound grade function $\tilde{g}_i(\mathbf{m}) = \min\{2m_1, 2m_2\}$.



(a) $n = 1$

(b) $n = 2$

Figure 3-1: Lower Bound Grade Function Illustration

### Optimization Model for Computing $\mathbf{m}^{\text{rob}}$

Using the results in Section 3.3.2, we describe a formulation which computes $\mathbf{m}^{\text{rob}}$. We prove in Theorem 3.3 that $\tilde{g}_i(\mathbf{m}) = \min_{j \in \{1,\cdots,n\}:[\mathbf{m}_i^*]_j > 0} (1/[\mathbf{m}_i^*]_j)\, m_j$. Therefore, $\mathbf{m}^{\text{rob}}$ can be computed as the optimal solution of the following mixed integer linear programming model, denoted as $\text{MJ}_{\text{rob}}$-OPT. Note that constraints (3.14) are equivalent to $x_i \leq \tilde{g}_i(\mathbf{m})$, $\forall i \in N$.

$$
\begin{aligned}
(\mathrm{MJ}_{\mathrm{rob}}\text{-OPT}) \quad \max \quad & z \\
\text{s.t.} \quad & (3.3)-(3.4),(3.6) \\
& v_i \leq \left(\frac{1}{[\mathbf{m}_i^*]_j}\right) m_j,\ \forall i \in N,\ j \in \{1,\cdots,n\} : [\mathbf{m}_i^*]_j > 0 \qquad (3.14) \\
& (3.8)-(3.10)
\end{aligned}
$$

Note that if for voter $i$, the grade of its most preferred candidate is some value $a_i \in [0,1)$ instead of being equal to 1, which violates part of Assumption 3.2, we can simply change constraints (3.14) to (3.15) and the rest of the theory would follow as before.

$$
v_i \leq \left(\frac{a_i}{[\mathbf{m}_i^*]_j}\right) m_j,\ \forall i \in N,\ j \in \{1,\cdots,n\} : [\mathbf{m}_i^*]_j > 0 \tag{3.15}
$$

However, in this case with varying maximum grades, the input of the single-shot robust mechanism would need to change. Specifically, the central planner would need to solicit from each voter its best candidate and its associated grade.

## 3.4 Potential for Strategic Behaviors

In this section, we investigate the potential for strategic behaviors under the two mechanisms developed in Section 3.3. The famous impossibility theorem from Gibbard (1973) and Satterthwaite (1975) already shows that there is no fully strategy-proof voting system that is non-defective (unanimous and nondictatorial) when there are at least three candidates. MJ is no exception although, under a setting with finite number of candidates, it is proved to be the voting system with minimal possibilities of strategic manipulation according to Balinski and Laraki (2014). Thus the focus of this section is to develop methods to (empirically) evaluate how often these strategic manipulation opportunities exist under these two mechanisms. The method we develop here is based on the notion of Nash Equilibrium where we define a successful manipulation as a unilateral deviation from the truthful input that will benefit the manipulator. It should be noted that this analysis identifies *opportunities* for beneficial strategic grading. The ability of a voter to identify when such an opportunity exists and how to execute the proper manipulation could be limited and such calculations might potentially be difficult to perform. In particular, our analysis assumes that the voters have knowledge of all grading functions. In practice, significant measures are likely to be taken by each voter to keep that voter's grading function private. Thus, the results presented here should be viewed as "upper bounds" on the degree of strategic behavior on the part of voters.

### 3.4.1 Mechanism I: An Iterative Approach

The input from voters under the iterative approach is the set of grades that they provide for the candidates to be graded. Note that there is a difference in voter input in the first iteration of the mechanism and the later iterations. In the first iteration, the voters submit grades for all the initial candidates in the set. However, in later iterations, they only provide grade for the newly generated candidate of that iteration. So it is likely for the voters to have more opportunities for strategic behaviors in the first iteration than during later ones. We assume that voters are myopic, i.e. they will only grade strategically within an individual iteration and will not be able to accurately predict the impact of the grading in the current iteration on grade function estimation and candidate generation in future iterations. In other words, we assume that voters grade without the knowledge of the total number of iterations $n_{\max}$. They treat the current iteration to be the final one, and assume that the majority winner of the current iteration will be selected as the final winner. Balinski and Laraki (2014) establish several theoretical results on the resistance to grading manipulation under MJ, which show that MJ minimizes the "probability of cheating" as per a certain definition of "cheating". Swaroop (2013) provides detailed necessary and sufficient conditions on the grade of a specific candidate being manipulatable to benefit the manipulator. Simulation experiments show that cases where these conditions are met are relatively rare. Our work is one-step further than the work by Swaroop (2013): we assess whether a voter will be benefit from simultaneously manipulating grades of *one or more* candidates.

In the first iteration, suppose there are $k$ candidates $\mathbf{m}^1, \cdots, \mathbf{m}^k$ to be graded. Let $v_{i,j}, i \in N, j \in \{1, \cdots, k\}$ be the grades submitted by voter $i$ for candidate $\mathbf{m}^j$. We further denote $\mathbf{v}_i = [v_{i,j}]_{j \in \{1, \cdots, k\}}$ as the vector of grades submitted by voter $i$ for all $k$ candidates and $\mathbf{v}_{-i} = [\mathbf{v_{i'}}]_{i' \in N, i' \neq i}$ as the vector formed by concatenating the grade vectors of all voters except voter $i$. With a similar definition, $v_{i,j}^{\text{true}} = g_i(\mathbf{m}^j)$ is the truthful grade of voter $i$ for candidate $\mathbf{m}^j$ according to its grade function. Then, $\mathbf{v}_i^{\text{true}}$ and $\mathbf{v}_{-i}^{\text{true}}$ are the vectors of truthful grades of all $k$ candidates by voter $i$ and by all voters other than $i$, respectively. Let $p_i(\mathbf{v}_i; \mathbf{v}_{-i}^{\text{true}})$ be the payoff that voter $i$ gets (measured according to its true grade for the majority winner) if voter $i$ submits grade vectors $\mathbf{v}_i$ and all other voters grade truthfully as $\mathbf{v}_{-i}^{\text{true}}$. We define a voter $i$ to have the potential for cheating in the first iteration if

$$\frac{\max_{\mathbf{0} \leq \mathbf{v}_i \leq \mathbf{1}} \; p_i(\mathbf{v}_i; \mathbf{v}_{-i}^{\text{true}})}{p_i(\mathbf{v}_i^{\text{true}}; \mathbf{v}_{-i}^{\text{true}})} \geq 1 + \epsilon, \tag{3.16}$$

which means that unilateral deviation from truthful grading could result in a higher payoff (by at least $\epsilon \times 100\%$) for voter $i$ compared to truthful grading.

In the second or later iterations, voters grade only the newly generated candidate. To illustrate this, suppose $l$ vectors have been graded in all previous iterations, and we denote by $\mathbf{m}^{l+1}$ the newly generated candidate of the current iteration. Let $v_{i,l+1}$ be the grade that voter $i$ gives to the newly generated candidate. Similarly, let $v_{i,l+1}^{\text{true}} = g_i(\mathbf{m}^{l+1})$ be voter $i$'s truthful grade for the newly generated candidate. Let $\mathbf{v}_{-i,l+1}^{\text{true}} = [v_{i',l+1}^{\text{true}}]_{i' \in N : i' \neq i}$. Note that there are $l$ candidates which are already graded in all previous iterations. Then, $\mathbf{v}^{\text{true}} = [v_{i,j}^{\text{true}}]_{i \in N, j = \{1, \cdots, l\}}$ is the vector of truthful grades for all graded candidates in previous iterations from all voters. Let $p_i(v_{i,l+1}; \mathbf{v}_{-i,l+1}^{\text{true}}, \mathbf{v}^{\text{true}})$ be the payoff that voter $i$ gets if voter $i$ submits $v_{i,l+1}$ as the grade for the newly generated candidate assuming truthful grading by all voters in all previous iterations and also assuming truthful grading by all voters other than $i$ for the new candidate in the current iteration. We define a voter $i$ to have the potential for cheating in this iteration if

$$\frac{\max_{0 \leq v_{i,l+1} \leq 1} \; p_i(v_{i,l+1}; \mathbf{v}^{\text{true}}_{-i,l+1}, \mathbf{v}^{\text{true}})}{p_i(v^{\text{true}}_{i,l+1}; \mathbf{v}^{\text{true}}_{-i,l+1}, \mathbf{v}^{\text{true}})} \geq 1 + \epsilon, \tag{3.17}$$

which means that unilateral deviation from truthful grading for the newly generated candidate could result in a higher payoff (by at least $\epsilon \times 100\%$) for voter $i$ compared to truthful grading.

We develop a mixed-integer linear program to solve the maximization problem in the numerators of the left hand sides of inequalities (3.16) and (3.17). The formulation is rather involved, so we defer the details to Appendix B.6.

### 3.4.2 Mechanism II: A Single-Shot Robust Approach

In the single-shot robust approach, voters submit their most preferred candidates instead of grading candidates. To the best of our knowledge, there is no previous literature which has investigated the issue of strategic manipulation for this type of mechanism. Similar to Section 3.4.1, we take a computational approach here. Let $\mathbf{m}_i$ be the most preferred candidate submitted by voter $i$ and let $\mathbf{m}^{\text{true}}_i$ be voter $i$'s true most preferred candidate. We further define $\mathbf{m}^{\text{true}}_{-i} = [\mathbf{m}^{\text{true}}_{i'}]_{i' \in N, i' \neq i}$. Let $q_i(\mathbf{m}_i; \mathbf{m}^{\text{true}}_{-i})$ be the payoff that voter $i$ gets if voter $i$ submits $\mathbf{m}_i$ as its most preferred vector and if all other voters submit their most preferred vectors truthfully. We then define a voter $i$ to have the potential for cheating under the single-shot robust mechanism if

$$\frac{\max_{\mathbf{m}_i \in P} \; q_i(\mathbf{m}_i; \mathbf{m}^{\text{true}}_{-i})}{q_i(\mathbf{m}^{\text{true}}_i; \mathbf{m}^{\text{true}}_{-i})} \geq 1 + \epsilon, \tag{3.18}$$

which means that unilateral deviation from truthful submission of most preferred vector could result in a higher payoff (by at least $\epsilon \times 100\%$) for voter $i$ compared to truthful submission of most preferred vector.

Evaluating $q_i(\mathbf{m}_i; \mathbf{m}^{\text{true}}_{-i})$ itself requires solving the mixed-integer optimization problem MJ$_{\text{rob}}$-OPT. Thus the problem $\max_{\mathbf{m}_i \in P} q_i(\mathbf{m}_i; \mathbf{m}^{\text{true}}_{-i})$ is a bilevel program whose lower-level problem is a mixed-integer linear program. Due to the intractability of finding exact optimal solution for this type of problems, we solve it heuristically by treating $q_i(\mathbf{m}_i; \mathbf{m}_{-i}^{\text{true}})$ as a black-box function and optimize it using off-the-shelf black-box optimization solver (Johnson, 2011).

## 3.5 Case Studies and Computational Experiments

To demonstrate the effectiveness of our approaches, we conducted various computational experiments. In this section, we describe them and present the results. Specifically, we compare the performances of the following four approaches:

1. *Random Generation*:

   This is a baseline approach to find the majority winner. We randomly generate $k$ candidates along the efficient frontier of the feasible region (detailed method is provided in Appendix B.2), and perform one round of candidate grading. The candidate with the highest majority grade among these $k$ candidates is selected as the winner. For different values of $k$, we report the average performance of the majority winner by averaging over the results of 1,000 rounds of simulations.

2. *Iterative Approach Starting with Voters' Best Candidates*:

   This is the iterative approach we discussed in Section 3.3.1 and Algorithm 3. We initialize Algorithm 3 with a set of candidates which is made up of each voter's most preferred candidate. We report the performance of the majority winner among the candidates in the set $S$ at each iteration.

3. *Iterative Approach Starting with Evenly Distributed Candidates*:

   The only difference between this approach and the previous one is that we initialize this approach with a number of evenly distributed candidates along the efficient frontier. The number of initial candidates is set to be equal to the number of voters so that we can have a fair comparison with the iterative approach starting with voters' best candidates.

4. *Single-Shot Robust Approach*:

   This is the robust approach we described in Section 3.3.2.

We benchmark all solutions to $\mathbf{m}^{\text{exact}}$, which is the true majority winner (the candidate that has the highest majority grade based on voters' true grade functions) over the entire polyhedral feasible space, which is obtained by solving formulation MJ described in (3.2) - (3.10). We denote by $v_{\text{exact}}$ the true majority grade of $\mathbf{m}_{\text{exact}}$. In order to better benchmark the solutions, we define a *naïve solution*, $\mathbf{m}_{\text{naïve}}$, to be a randomly generated candidate on the efficient frontier. The naïve solution generation procedure is provided in Appendix B.2. We randomly sample 1,000 different naïve solutions and average their true majority grades. We denote this average by $v_{\text{naïve}}$. We then use $v_{\text{exact}}$ and $v_{\text{naïve}}$ to report the optimality gap of all solutions which is calculated as $\big((v_{\text{exact}} - v) \,/\, (v_{\text{exact}} - v_{\text{naïve}})\big) * 100\%$, where $v$ is the true majority grade of the solution we want to evaluate. This optimality gap measures the extent to which the solutions produced by different approaches outperform the naïve solution compared with the extent to which the true majority winner outperforms the naïve solution. The lower the value of optimality gap the better is the performance.

For each approach, we also report the percentage of voters who have the opportunity to unilaterally deviate from truthful behavior to increase their own payoff. This is defined as the ratio of the number of voters with such opportunities to the total number of voters. For the two iterative approaches, we report this percentage value separately for each iteration because the first iteration and the remaining ones are different in terms of voter input and the way cheating is defined. For the first iteration, we use the criteria described in Eq. (3.16) (coordinated manipulation of one or more votes); for later iterations, we use the criteria described in Eq. (3.17) (manipulation of a single vote). For the single-shot robust approach, we report according to Eq. (3.18). We report this percentage under two different improvement thresholds in Eq. (3.16) - (3.18): namely, $\epsilon = 0.1\%$ and $\epsilon = 1\%$.

All the mathematical programs were written in Julia using JuMP as the algebraic modeling language (Dunning et al., 2017). All linear and mixed-integer linear programming problems were solved using Gurobi 6.5 (Gurobi Optimization Inc, 2016) and all nonlinear optimization problems were solved using IPOPT (Wächter and Biegler, 2006) and BONMIN (Bonami et al., 2008). The black-box optimization problem to calculate the strategic input of the most preferred candidate as per Eq. (3.18) was solved using NLopt (Johnson, 2011).

We conduct two case studies as discussed in Sections 3.5.1 and 3.5.2 respectively: (1) a collaborative air traffic flow management problem configuring airspace performance for competing airlines; and (2) a capital budgeting problem assigning budget to different projects.

### 3.5.1 A Collaborative Air Traffic Flow Management Problem

Air Traffic Flow Management (ATFM) is the body of systems, processes and mechanisms to ensure that air traffic flows do not exceed airport and airspace capacities; and to seek the most efficient use of airspace system resources. It employs several types of traffic management initiatives, and most notably, ground delay programs (GDPs), which are implemented to control the flow of aircraft into a specific airport by delaying flights destined for that airport at their respective origin airports. A GDP is usually implemented for a period of time when increased spacing between successive landing aircraft is necessary, which can occur during adverse weather conditions. Future visions of ATFM - both in the U.S. and in Europe - support a performance-based approach that employs collaboration between the air navigation service providers and the airlines. A key feature of this outlook is to support the airlines' business objectives in ATFM, subject only to system-level constraints like safety and security. The authors refer interested readers to Ball et al. (2014b); Evans et al. (2016); Ball et al. (2017) and Swaroop and Ball (2012) for a detailed discussion of these future visions.

In this case study, we focus on collaboratively designing a GDP. Following Liu and Hansen (2013), we define two system-wide performance metrics under a GDP: capacity-utilization ($m_1$) and predictability ($m_2$). These two metrics are both expected values where the uncertainty is in weather conditions. Capacity-utilization measures the expected value of the ratio of the actual throughput to the maximum throughput that would be possible with perfect information; predictability measures the expected value of the ratio of the flight delay assuming that the GDP were to end as planned, to the actual flight delay incurred. Either metric lies between 0 and 1.

The metrics are inter-related, requiring trade-offs between them. To see that, note that weather forecasts usually involve uncertainty, so the time when weather conditions improve is usually difficult to predict. Setting a GDP end time to be aggressively early maximizes the capacity-utilization ($m_1$), because inbound flights are not delayed at their origin airports any longer than necessary. Therefore, no matter when weather conditions improve, there are enough flights positioned to land. However, last minute extensions may have to be made to the GDP if the adverse weather continues longer, potentially requiring airborne holding. Thus planning for an early GDP end time would be at the cost of predictability ($m_2$). To maximize predictability ($m_2$), the GDP end time should be set conservatively late, so that the GDP would never be extended. This would be at the expense of capacity-utilization ($m_1$), because capacity might be underutilized if conditions were to improve earlier than the set GDP end time. Therefore, there is a trade-off between these two performance metrics, and infinitely many "moderate" GDP designs can be proposed in the intervening space.

An analytical relationship between capacity-utilization ($m_1$) and predictability ($m_2$) is developed for a single airport in Liu and Hansen (2013) by varying the planned length of GDP. This relationship gives a concave efficient frontier, while the frontier as well as all the interior points comprise the feasible candidate space (the set of all possible GDP designs). The feasible candidate space is a convex set in two dimensions and we develop a piecewise-linear approximation to model it as a polyhedron. Details about this feasible candidate space are provided in Appendix B.7.

We apply our mechanisms to the GDP design problem where competing airlines vote for setting the two performance metrics: capacity-utilization ($m_1$) and predictability ($m_2$). Thursday, May 10th, 2007 is selected as the sample day for which the data was obtained. The San Francisco International Airport (SFO) is selected as the airport where the GDP is to be implemented. Ten domestic airlines participate in the voting. Each voter (airline) has a grade function which is a function of $m_1$ and $m_2$. The grade function specification and generation procedures are described in Appendix B.8.

It is important to note that the grade functions for airlines do have practical meanings in that they represent different airlines' inherent preferences across different GDP designs. For example, Bloem et al. (2012) discover from historical data that delay cost functions are different depending on whether the flight is bound for a hub airport or not, which creates heterogeneity among airlines' responses to air traffic flow management initiatives. Yan et al. (2017b) evaluate airlines' preferences over various GDP designs in a simulation environment using historical data. They find that airlines with more opportunities for recovery operations, such as, canceling flights, delaying flights, or swapping aircraft (e.g. a hub-and-spoke airline's operations at its hub airport with many available aircraft of the same type), tend to prefer GDPs with higher predictability. This is because accurate delay information can help them plan and implement efficient recovery operations ahead of time that can significantly reduce operational costs. On the other hand, airlines with fewer recovery opportunities (e.g. an airline's operations at a non-hub airport with less frequent flights) tend to prefer GDPs with higher capacity, even if they have a lower predictability.

Based on these insights, to support the generation of grading functions for our simulation, we assign different preferences to different airlines toward these two metrics (capacity utilization and predictability) according to factors such as their network structure and flight frequency at SFO. The relative preference information is provided in the last column in Table 3.3. The rows in Table 3.3 are in descending order of the number of operations at SFO. As we can see, United Airlines has the largest number of operations and operates a hub-and-spoke network with one hub at SFO. Therefore, we assign to it a higher preference for predictability ($m_2$) than for capacity-utilization ($m_1$). The next five airlines, namely, American Airlines, US Airways, Alaska Airlines, Delta Airlines, and Northwest Airlines, have a moderate number of operations at SFO and hence are assumed to value capacity-utilization ($m_1$) and predictability ($m_2$) almost equally. Frontier Airlines, JetBlue Airways, AirTran Airways and Hawaiian

Airlines, have very few and infrequent operations. Therefore we assume that they prefer capacity-utilization ($m_1$) more than predictability ($m_2$). These assigned metric preferences are employed in our grade function generation procedures, which are described in Appendix B.8.

We assign each airline a weight based on its total number of operations at SFO. Since the largest airline's operations (United Airlines) represent over half of the total operations, we should not directly use the number of operations as the weights if we want to avoid trivial outcomes. If we use them directly, then regardless of any other airlines' grades for any of the candidates, the majority winner would always be determined by United Airlines all by itself. While it is considered fair for United Airlines to have more influence than other airlines, it would not be acceptable for it to have "dictatorial" powers. Therefore, we provide four different weighting schemes (designated as "pow.1", "pow.2", "pow.3", "pow.4") based on four different transformations of each airline's number of operations, which are shown in Table 3.3. Column 2 lists the total number of operations for each airline. The values shown in columns 3 to 6, titled "pow.1", "pow.2", "pow.3" and "pow.4", represent the weights corresponding to the appropriate power transformations to ensure that the airline with the largest number of total operations has 10%, 20%, 30% and 40%, respectively, of the total weight. Suppose $o_i$ is the number of operations of airline $i$; $o_{\max}$ is the number of operations of the largest airline. Under weighting schemes "pow.1", "pow.2", "pow.3" and "pow.4", we would choose $\alpha$ such that $o_{\max}^{\alpha} / \left(\sum_{i \in N} o_i^{\alpha}\right) = 0.1, 0.2, 0.3,$ and $0.4$, respectively. The weight used for each airline $o_i^{\alpha}$ under each different weighting scheme, is listed as the first number in each column from columns 3 to 6. Numbers in parentheses list that airline's proportion of total weight as a percentage. Note that since we have exactly ten voters here, "pow.1" in this case is equivalent to an equal weighting scheme. The choice of an appropriate weighting scheme is certainly an important problem that is worthy of further study.

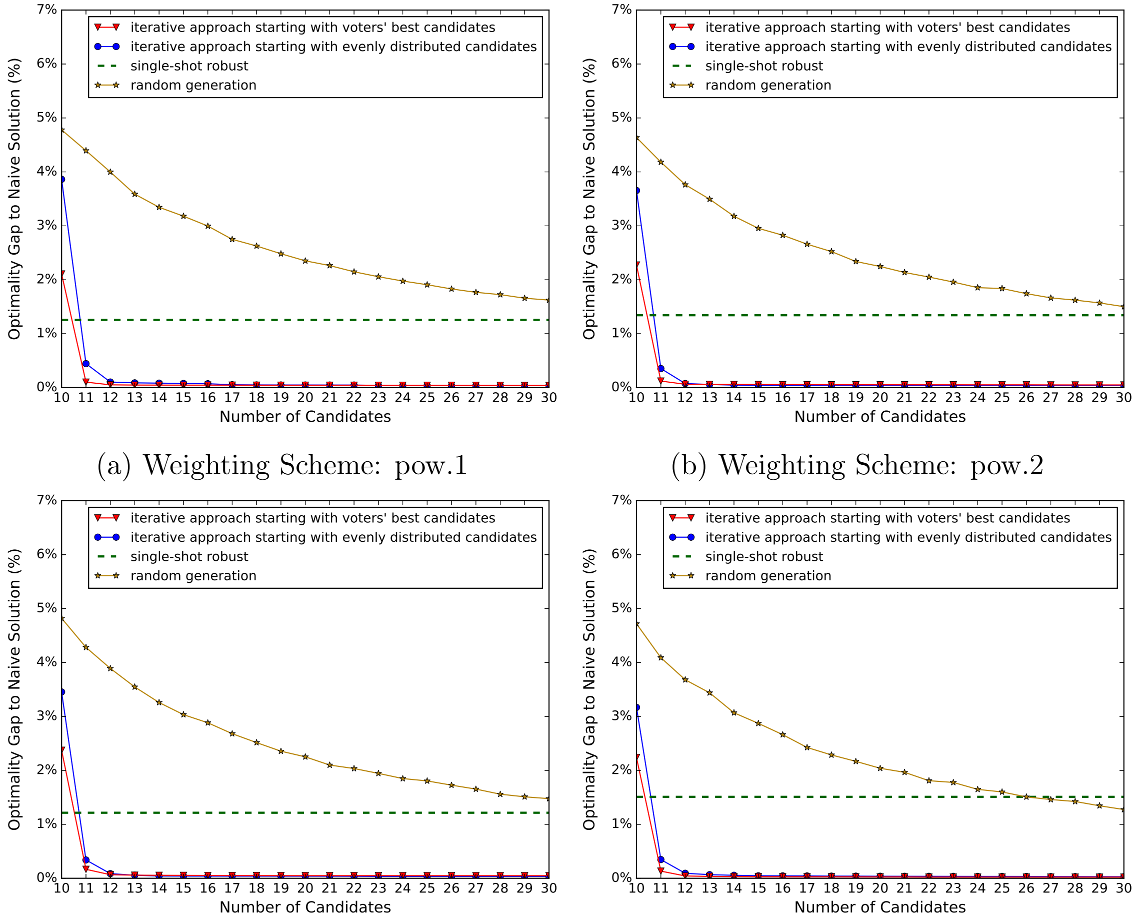
We randomly generate 100 sets of coefficients for the grade functions according to procedures described in Appendix B.8 and report the average optimality gap (given by the formula $((v_{\text{exact}} - v)/(v_{\text{exact}} - v_{\text{naïve}})) * 100\%$ as described earlier in this section) in

| **Airline** | **Num of Ops.** | **pow.1** | **pow.2** | **pow.3** | **pow.4** | **Preference** |
|---|---|---|---|---|---|---|
| United Airlines | 243 | 1.00 (10.0%) | 4.05 (20.0%) | 11.01 (30.0%) | 26.05 (40.0%) | $m_2 \succ m_1$ |
| American Airlines | 41 | 1.00 (10.0%) | 2.57 (12.7%) | 5.06 (13.8%) | 9.06 (13.9%) | $m_1 \approx m_2$ |
| US Airways | 20 | 1.00 (10.0%) | 2.14 (10.6%) | 3.70 (10.1%) | 5.92 (9.1%) | $m_1 \approx m_2$ |
| Alaska Airlines | 18 | 1.00 (10.0%) | 2.09 (10.3%) | 3.53 (9.6%) | 5.56 (8.5%) | $m_1 \approx m_2$ |
| Delta Airlines | 17 | 1.00 (10.0%) | 2.06 (10.2%) | 3.45 (9.4%) | 5.37 (8.2%) | $m_1 \approx m_2$ |
| Northwest Airlines | 12 | 1.00 (10.0%) | 1.88 (9.3%) | 2.96 (8.1%) | 4.37 (6.7%) | $m_1 \approx m_2$ |
| Frontier Airlines | 9 | 1.00 (10.0%) | 1.75 (8.6%) | 2.61 (7.1%) | 3.68 (5.7%) | $m_1 \succ m_2$ |
| JetBlue Airways | 5 | 1.00 (10.0%) | 1.51 (7.5%) | 2.02 (5.5%) | 2.60 (4.0%) | $m_1 \succ m_2$ |
| AirTran Airways | 2 | 1.00 (10.0%) | 1.19 (5.9%) | 1.35 (3.7%) | 1.51 (2.3%) | $m_1 \succ m_2$ |
| Hawaiian Airlines | 1 | 1.00 (10.0%) | 1.00 (4.9%) | 1.00 (2.7%) | 1.00 (1.5%) | $m_1 \succ m_2$ |

Table 3.3: Collaborative ATFM case study setup (05/10/2007 at SFO): United Airlines' operations include those of Mesa Airlines and SkyWest Airlines, American Airlines' operations include those of American Eagle, and Delta Airlines' operations include those of Atlantic Southeast Airlines.

percentage for the aforementioned four different approaches under these four weighting schemes. The results are summarized in Figures 3-2a through 3-2d. Each figure corresponds to a different weighting scheme. For the random generation approach, we report the average optimality gap under $10, 11, \cdots, 30$ candidates. For either of the two iterative approaches, we start with 10 initial candidates and generate one additional candidate in every iteration until we have a total of 30 candidates ($n_{\max} = 20$ in Algorithm 3). For the iterative approach starting with evenly distributed candidates, our initial set of candidates contains 10 evenly distributed candidates on the efficient frontier (see Appendix B.4 for details); while for the iterative approach starting with voters' most preferred candidates, our initial set contains the 10 airlines' most preferred candidates (one preferred candidate per airline). Note that the single-shot robust approach doesn't require any candidate grading; it simply takes the 10 airlines' respective most preferred candidates and outputs one resultant candidate. So the optimality gap remains constant with the number of candidates as the approach does not need or use any additional candidates.

From these results, we see that all of our three approaches perform better than the baseline approach of random generation. Both iterative approaches converge to the true majority winner with zero optimality gap while evaluating at most 20 candidates. The random generation baseline can only decrease the gap to 1-2% even after evaluating 30 candidates. Interestingly, under all weighting schemes, the single-shot robust approach produces an average optimality gap of just over 1%. This value is very competitive compared to the random generation baseline. This demonstrates the effectiveness of the single-shot robust approach when dealing with a low-dimensional candidate space (the dimension of the candidate space in this case study is two). Among all approaches, the iterative approach starting with voters' most preferred candidates has the best performance, as it quickly converges to the true majority winner within one or two iterations under all weighting scheme. However, it also requires the most in terms of input from voters: both the most preferred candidate and the grades for all candidates.



(a) Weighting Scheme: pow.1

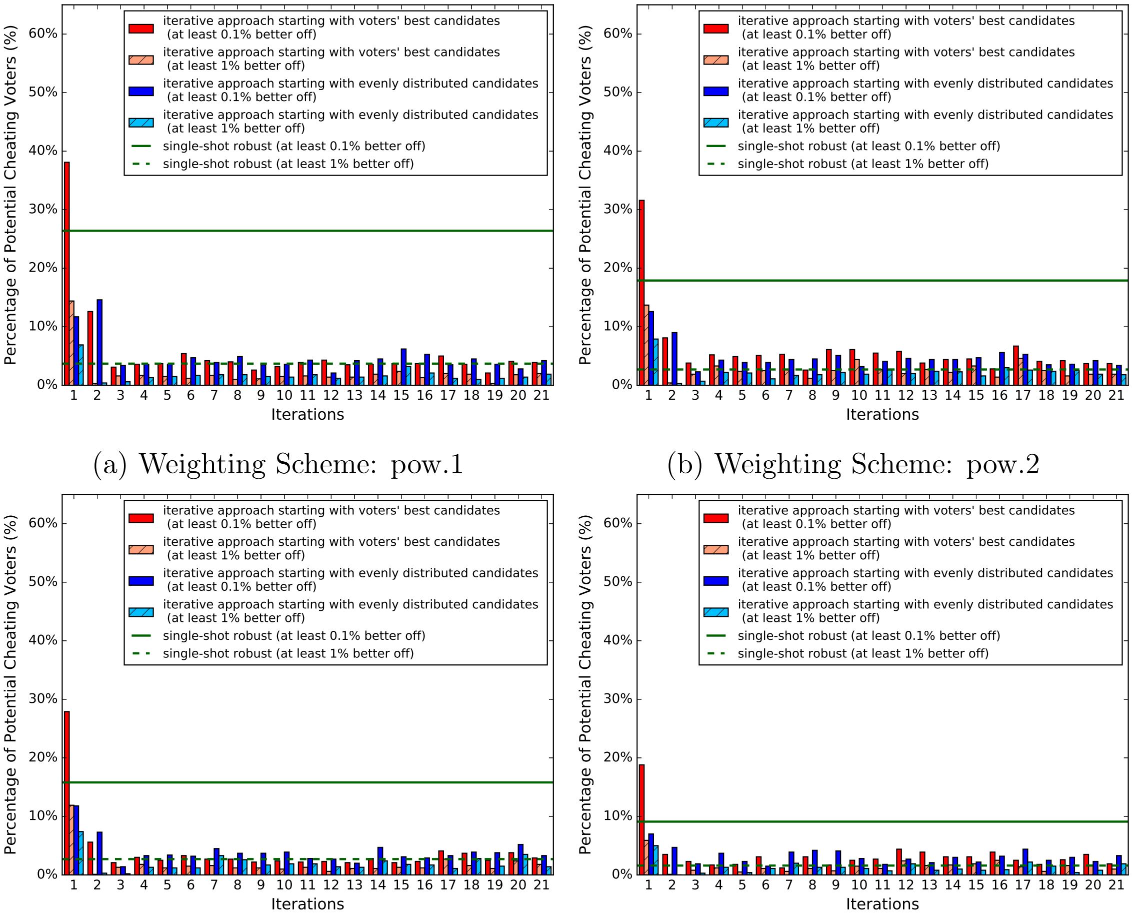
(b) Weighting Scheme: pow.2

(c) Weighting Scheme: pow.3

(d) Weighting Scheme: pow.4

Figure 3-2: Results of the Computational Experiments for the Collaborative Air Traffic Flow Management Case Study

Figures 3-3a through 3-3d demonstrate the percentage of voters with strategic voting (”cheating”) opportunities for the two iterative approaches and for the single-shot robust approach. The percentage is calculated as the average ratio of the number of voters with such opportunities to the total number of voters (10) over all 100 sets of grade function coefficients. Blue bars with and without textures depict the iterative approach starting with voters' most preferred candidates under 0.1% and 1% improvement thresholds respectively. Yellow bars with and without textures depict the iterative approach starting with evenly distributed candidates under 0.1% and 1% improvement thresholds respectively. Solid and dashed red lines represent the single-shot robust approach under 0.1% and 1% improvement thresholds respectively.



(a) Weighting Scheme: pow.1

(b) Weighting Scheme: pow.2

(c) Weighting Scheme: pow.3

(d) Weighting Scheme: pow.4

Figure 3-3: Percentage of Voters with Opportunities for Beneficial Strategic Manipulation for the Collaborative Air Traffic Flow Management Case Study

Overall we observe that as the weights of voters become more differentiated, there is generally a reduction in the percentage of strategic voting opportunities. This is because when weights are equal, each voter has the same power in determining the winner. It is also intuitive to observe that for the two iterative approaches, the likelihood of beneficial strategic voting opportunities exhibits a large drop after the first iteration. The intutitive explanation is that, in the first iteration, voters have a greater opportunity for beneficial manipulation by adjusting the grades of multiple candidates in a coordinated fashion. Comparing the two iterative approaches, we see that the iterative approach starting with voters' best candidates is more prone to opportunistic manipulation in the first iteration. This might be caused by the fact that the candidate set in the first iteration is not as diverse as the set of evenly distributed candidates. Comparing the iterative approach starting with the voters' best candidates and the single-shot robust approach, it is interesting to see that the strategic manipulation opportunities under either improvement threshold are generally comparable across the two approaches. Under the 0.1% improvement threshold, the iterative approach starting with evenly distributed candidates is more resistant to strategic manipulation when compared to the single-shot robust approach (although the difference is quite small). On the other hand, under 1% improvement threshold, their performances are quite similar. Since the iterative approaches have theoretical support for their good strategy-proofness properties, these computational results suggest that, in this case study, the single-shot robust approach has comparably good strategy-proofness properties as well.

### 3.5.2 A Capital Budgeting Problem

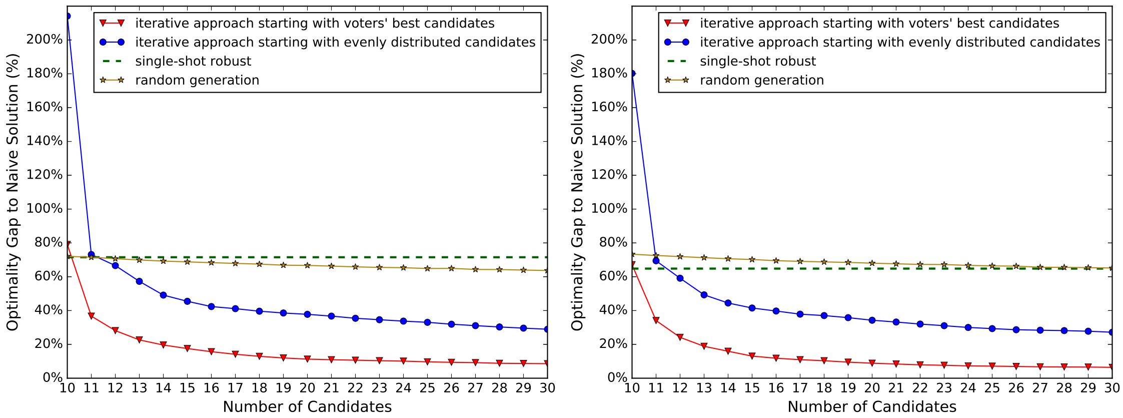
In the second case study, we consider a simple capital budgeting problem, i.e. the one involving allocation of a fixed amount of money ($C$) to $n$ projects. Let $m_1, \cdots, m_n$ denote, respectively, the amounts of money allocated to projects 1 through $n$. Then the feasible candidate space $P$ of all possible allocations is:

$$\left\{(m_1, \cdots, m_n) : \sum_{i=1}^{n} m_i \leq C, m_i \geq 0, \forall i = 1, \cdots, n\right\}.$$

The type of capital budgeting problems that we are interested in are group decision-making problems, i.e. the ones where the final allocation scheme is not determined by a single decision maker but rather by a group. Suppose that the final allocation outcome $\mathbf{m}^* = (m_1^*, \cdots, m_n^*)$ is to be determined by $N$ voters. Each voter has a grade function and a weight. Consistent with Assumption 3.2, the grade functions are concave. The concavity reflects a decreasing rate of return on investments in each project. The detailed specifications and generation methods for the grade functions are discussed in Appendix B.8.

The computational experiments are set up as follows. Without loss of generality, we set the total budget $C$ equal to 1. There are 10 voters. We test two weighting schemes: 1) equal weights, where each voter has the same weight; 2) random weights, where each voter's weight is generated from a uniform distribution between 1 and 10. There are 10 projects to be potentially funded. As before, to measure the performance of the four approaches, we randomly generate 100 sets of grade function coefficients according to procedures described in Appendix B.8. Also, under the random weighting scenario, we randomly generate 100 sets of weights for all voters. For the random generation approach, we report the average optimality gap under $10, 11, \cdots, 30$ candidates. For each of the two iterative approaches we start with 10 candidates, perform 20 iterations ($n_{\max} = 20$ in Algorithm 3), and generate one new candidate in each iteration. For the iterative approach starting with evenly distributed candidates, we start with 10 unit vectors $\mathbf{e}_1, \cdots, \mathbf{e}_{10}$ where $\mathbf{e}_i$ is the vector with all components equal to zero except for the $i^{\text{th}}$ component which is equal to one; while for the iterative approach starting with voters' best candidates, we start with all 10 voters' most preferred candidates, one most preferred candidate per voter.

The average optimality gap for each approach under the two weighting schemes is reported in Figures 3-4a and 3-4b. The average optimality gap for each of the four



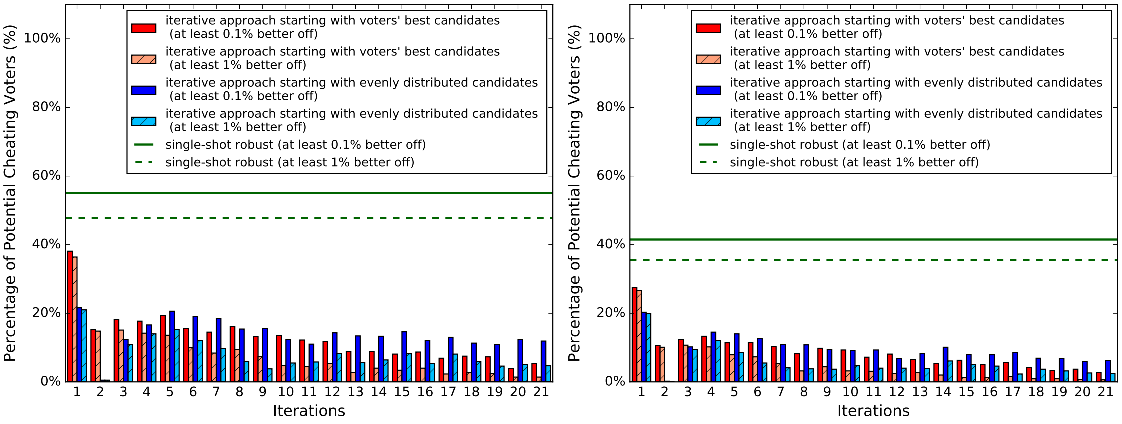
(a) Equal Weights

(b) Random Weights

Figure 3-4: Results of the Computational Experiments for the Capital Budgeting Case Study

approaches is found to be substantially higher than the corresponding values for the collaborative air traffic flow management case study presented in Section 3.5.1. This is possibly due to the significantly higher dimension of the feasible candidate space (increased from 2 to 10), which dramatically increases the cardinality of the candidate space. The optimality gap monotonically decreases for each approach (except for the single-shot robust approach) with an increase in the number of graded candidates. Overall, under both weighting schemes, the iterative approach starting with voters' best candidates is again the best-performing approach. It gradually reduces the average optimality gap to lower than 10% with 30 candidates grading budget. The second-best approach is the iterative approach starting with evenly distributed candidates, which can reduce the gap to lower than 30% with the same 30 candidates grading budget. Both iterative approaches substantially outperform the random generation baseline, which can only provide a solution with an optimality gap of around 65-70% with the 30 candidates grading budget.

The single-shot robust approach, under the random weighting scheme, outperforms the baseline approach in all cases (i.e., for a grading budget of up to 30 candidates). However, under the equal weighting scheme, it outperforms the baseline only when the number of candidates is less than 11. The single-shot robust approach produces solutions with a 71.56% optimality gap under the equal weighting scheme



(a) Equal Weights

(b) Random Weights

Figure 3-5: Percentage of Voters with Opportunities for Beneficial Strategic Manipulation for the Capital Budgeting Case Study

and with a 64.83% optimality gap under the random weighting scheme. Though not as good as its performance presented in Section 3.5.1, the single-shot robust approach does have some merit in this case too, considering its extremely low implementation cost (no candidate grading is required) and relatively good performance against the random generation baseline, especially under the random weighting scheme.

Similar to the results presented Section 3.5.1, the percentages of voters with beneficial strategic grading opportunties are depicted in Figures 3-5a and 3-5b. Here, we find a substantial increase in the frequency of strategic voting opportunities compared to the collaborative air traffic flow management case study possibly due to the increase in the dimension of the candidate space. Similar to the previous case study, we find that the iterative approach starting with voters' best candidates provides more strategic manipulation opportunities compared to the other iterative approach in first iteration. We also see a substantial drop in such opportunities after the first iteration. An important difference compared with the previous case study is that the single-shot robust approach is found to be more prone to strategic manipulations under both improvement thresholds: about a 10%-20% increase compared to the first iteration of both iterative approaches.

# Chapter 4

# Benefits Assessment of Airline-Driven Ground Delay Programs

In this chapter, we assess the benefits of applying the voting mechanism developed in Chapter 3 to the planning of ground delay programs. Unlike Section 3.5.1 where we only look at majority grade as the performance measure of the final outcome. In this chapter, we focus on a comprehensive and concrete assessment in terms of reduction of delay minutes and costs by using such an airline-driven and decentralized approach. We develop a simulation platform and airline recovery decision support models to mimic the interactions between Federal Aviation Administration (FAA) and airlines during a ground delay program. Through this realistic simulation-based testbed, we extract information to quantify benefits to all stakeholders.

## 4.1 Introduction

The ground delay programs (GDPs), first formally described by Odoni (1987), have been a topic of air traffic flow management research for almost three decades. In a GDP, flights bound for congested airports are delayed on the ground (prior to their departure), so as to balance the arrivals at the destination airport with the reduced capacity at that airport. The underlying motivation is that, as long as a delay is unavoidable, it is both safer and less costly for the flight to absorb this delay on the ground rather than in the air. The typical objective of a centralized GDP design problem is to minimize the sum of ground and airborne delay costs, when facing an anticipated demand-capacity imbalance at an airport, by assigning departure delays to flights. Literature about better GDP decision-making can be broadly classified into two streams: the *single airport ground delay problem* (SAGDP) (Richetta and Odoni, 1993, 1994; Ball et al., 2003; Mukherjee and Hansen, 2007) and the *multi-airport ground delay problem* (MAGDP) (Vranas et al., 1994b,a). SAGDP optimizes ground delay assignment assuming a single, isolated airport with reduced capacity, while MAGDP considers network effects such as delay propagation along an aircraft flying through multiple airports. Literature can also be classified into *deterministic* and *stochastic* models depending on whether future capacity information at the airport(s) of interest is deterministic or probabilistic. The third dimension along which we can divide the literature is the nature of the decision-making process: *static* and *dynamic*. The static approaches solve the GDP design problem once and then implement that plan for the rest of the day (Richetta and Odoni, 1993; Vranas et al., 1994a; Ball et al., 2003). On the other hand, the dynamic approach allows updates to the GDP design decisions when more accurate weather and/or capacity information becomes available (Richetta and Odoni, 1994; Vranas et al., 1994b; Mukherjee and Hansen, 2007).

Although the literature can be classified into different categories, they all share a common characteristic: they all present *centralized* methods, wherein the central aviation authorities (e.g., Federal Aviation Administration (FAA), Eurocontrol, etc.) almost completely decide the GDP design. GDP design, however, involves tradeoffs and the central authority often does not have access to all relevant information about each airline's operational characteristics and network strategy while making these decisions in a centralized manner. Therefore, centralized methods are likely to make decisions under insufficient information about airline operations and hence, might result in suboptimal decisions.

Let us delve deeper into some of the factors that should be considered when planning GDPs. A useful example is the decision between *aggressive* and *conservative*

GDP designs, which is necessitated due to the uncertainty around our knowledge of future capacity. Setting GDP duration aggressively short (thus assigning less ground delays to flights) might incur unexpected additional airborne delays due to last-minute GDP extensions if adverse weather continues longer than anticipated. On the other hand, setting GDP duration conservatively long (thus assigning lengthy ground delays to flights) might lead to unused capacity and unnecessary delays. One might argue that this trade-off is accurately captured by the stochastic centralized models with the airborne-versus-ground delay trade-off driving the solution to the appropriate balance. However, the characterization of the tradeoff ignores airline reactions and their recovery operations in response to a GDP.

When the operations of an airline are disrupted, due to a GDP or otherwise, a number of actions are taken to try to bring operations back on track as soon as possible and with as little cost as possible. This is achieved by using a number of alternative recovery actions including intentional flight departure delays, flight cancellations, aircraft reassignments, crew swaps, passenger rebookings, etc. Different airlines have different recovery capabilities due to the differences in their network structures, fleet compositions, and schedule characteristics, to name a few. This variation causes airlines to have different preferences toward alternative GDP designs. For example, an airline operating a frequent shuttle service with low load factors and a single fleet type can often rebook disrupted passengers easily and can also swap aircraft easily. For such airlines, operating under a conservative but more predictable GDP design allows full utilization of the airline's recovery capability to mitigate adverse operational impacts. For an airline with lower schedule frequency and higher load factors, however, recovery is relatively more difficult. For such airlines, aggressive GDP designs may be preferred so that less initial ground delays are assigned and the flights are allowed the best chance to land in time. Centralized models either completely ignore airline recovery actions, or model only simplistic actions, such as delaying of flight departures. However, any accurate performance metric for a GDP design needs to account for the effects of that GDP design on the airlines and passengers impacted by the airline's recovery actions in response to the GDP. Thus the performance of the air transportation system should be measured in terms of the actual operational performance (flight delays, passenger delays, etc.) of all airlines instead of a simple metric such as the sum of ground and airborne delay costs calculated assuming simplified or no recovery actions. We thus argue that GDP planning method should take airlines' preferences into consideration and strive to better support the airlines' business objectives. This approach could improve the air transportation system's performance from the perspectives of both airlines and passengers.

In fact, several recent research studies including Chapter 3 have already been focusing on developing these "airline-driven" GDP decision-making approaches. Evans et al. (2016) evaluate several approaches for collecting preferences from airlines systematically and conclude that voting is the most promising method with respect to airline profitability, system optimality and equity. Swaroop and Ball (2013) and Yan et al. (2017a) adopt a newly developed voting method - Majority Judgment (Balinski and Laraki, 2010) - to the airline-driven GDP design problem and report promising performance based on their analytical and computational studies. In both approaches, the underlying idea is for the FAA to provide an initial list of candidate GDP designs, and then for all involved airlines to grade these candidate designs. Consistent with the Majority Judgment paradigm, the candidate with the highest median grade is declared the winner. The winner candidate is selected for implementation unless it is not deemed to be good enough, in which case new candidate(s) are generated and a new round of grading follows. This iterative process continues until a winner candidate of a round of grading is deemed to be good enough based on pre-determined criteria. We refer interested readers to the FAA funded study by Ball et al. (2014a) for a detailed description of this approach.

Although these studies provide a promising theoretical foundation for airline-driven GDPs, what remains unknown is the extent to which such an airline-driven approach, if implemented, can benefit airlines, FAA, and the air transportation system as a whole. In this chapter, we do not intend to propose new mechanisms to solicit airline preferences and/or to generate output in the form of implementable GDP designs. Instead, we focus on a different question. The goal of this chapter is to present and rigorously quantify the potential benefits of using airline-driven decision-making methods for GDP design compared with existing centralized methods, in a real-world setting. To achieve this goal, we first build an FAA/airline integrated simulation platform in which we can develop and implement various GDP designs, model airline response in terms of recovery operations and evaluate system-wide performance under each design. Airline recovery operations are modeled through an integrated schedule recovery, aircraft recovery and passenger recovery optimization model with all parameters and other inputs obtained from real-world schedule and economic data. Next, through a detailed case study, we provide new insights into the inherent differences across airlines in terms of prioritizing different GDP designs (e.g., aggressive versus conservative) due to the differences in their network structures, fleet compositions, recovery flexibilities and other operational characteristics. Finally, we apply a simplified airline-driven GDP-design approach, proposed in Swaroop and Ball (2013) and Yan et al. (2017a), while explicitly accounting for airline preferences, and compare the resulting system-wide performance with that of the centralized approach for GDP design.

Note that under the current practice in the US, airlines do have channels to discuss their preferences with central authorities before a GDP is issued. The process takes the form of strategic planning teleconferences (SPTs). During SPTs, FAA traffic managers consult with airline/flight operators at both the local and national levels in planning GDPs. While the SPTs perform a very legitimate and vital function in the overall traffic management process, there are several concerns and issues related to SPTs, and more generally, to strategic planning on the day-of-operations. Due to the free-form and unstructured nature of these SPTs, at times, an inordinate amount of time can be devoted to non-critical topics (Ball et al., 2014a). Furthermore, because flight operators are not assigned formal priority weights based on objective measures, often the more persistent and/or the âĂIJloudestâĂİ flight operators have the most influence. In contrast to this ad-hoc approach, we evaluate a more systematic airline-driven approach.

Last, we would like to mention how another stream of literature in the Collaborative Decision-Making (CDM) paradigm (Hoffman et al., 1999) relates to our problem. Under CDM, the FAA allocates arrival landing slots at airports to flights based on the original schedule and the first-come-first-served principle, that is, the flights with earlier scheduled arrival times will receive earlier slots. This step is implemented through the ration-by-schedule (RBS) algorithm (Federal Aviation Administration, 2001). Airlines are then allowed to redistribute their arrival slots among their flights based on their own objectives and considerations such as passenger delays, missed connections, etc. Airlines can also decide to cancel some flights and use these slots for another one of their flights to reduce delays. Empty slots due to cancellations can then be transferred across airlines through the compression algorithm (Hoffman et al., 1999). Although the current CDM tools enhance information exchange between the airlines and the FAA, they do not consider how to collaboratively set GDP designs in the first place. Instead, the GDP design is determined by the central authority (such as the FAA) before the CDM tools can be used by airlines. Unlike these CDM-focused studies, such as Hoffman et al. (1999), our work focuses on this prior step of selecting GDP designs themselves.

The remainder of this chapter is organized as follows. In Section 4.2, we introduce our evaluation framework and describe in detail the major components of our FAA/airline integrated simulation platform, which includes a mixed-integer linear programming (MILP) formulation used for airline recovery optimization. In Section 4.3, we present a numerical case study on a GDP day (May 10th, 2007) at San Francisco International airport (SFO) using the actual schedule data. In this case study, we compare and discuss the GDP design preferences of all the involved airlines. We then compute the optimal GDP designs using both a centralized approach and an airline-driven approach, and compare the system-wide performance under each of these two designs. We present and quantify the benefits of the airline-driven design approach according to various metrics.

A paper based on this chapter can be found in Yan et al. (2017b).

## 4.2 Evaluation Framework

In order to evaluate the system-wide performance under different GDP designs, we develop an FAA/airline integrated simulation platform to facilitate our analysis. The overall modeling framework and evaluation process are depicted in Figure 4-1.

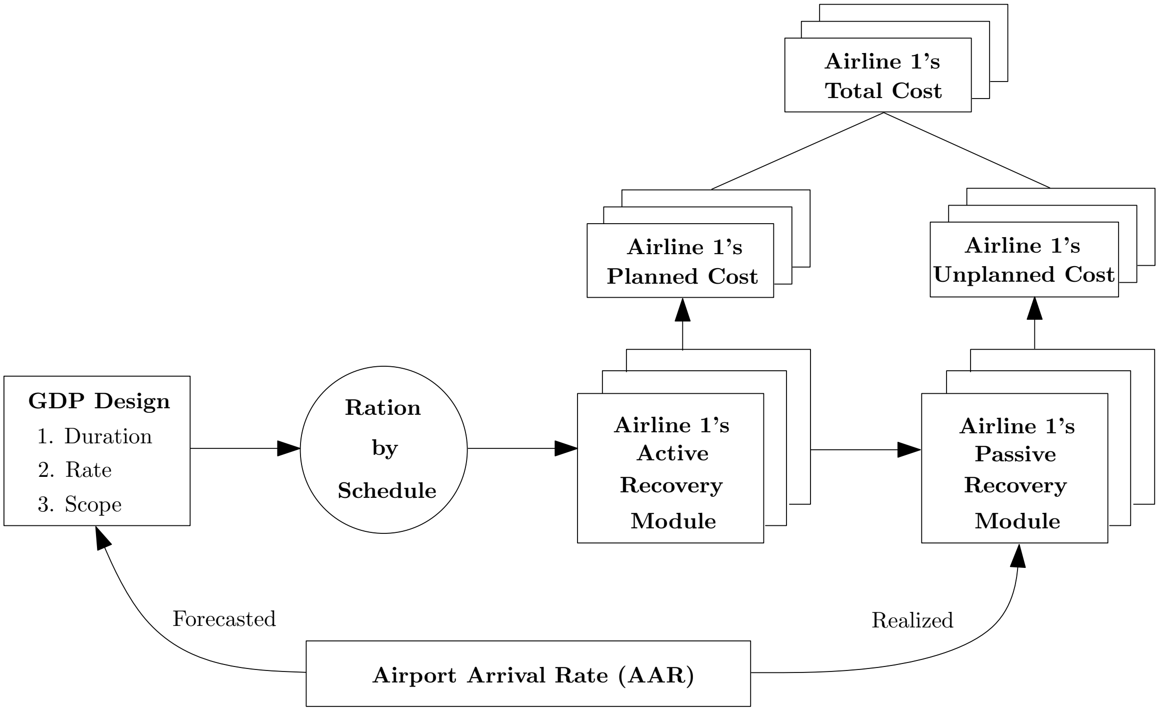

Airline 1's Total Cost
Airline 1's Planned Cost
Airline 1's Unplanned Cost
GDP Design
1. Duration
2. Rate
3. Scope
Ration by Schedule
Airline 1's Active Recovery Module
Airline 1's Passive Recovery Module
Forecasted
Realized
Airport Arrival Rate (AAR)

Figure 4-1: Evaluation Framework

The simulator contains a GDP design module in which we can set appropriate values for different GDP parameters such as planned start time, planned end time, program rate (the arrival rate during the duration of the GDP), and program scope (the set of departure airports affected by the GDP). Once we fix the GDP design, the FAA Ration by Schedule (RBS) module (Federal Aviation Administration, 2001) assigns ground delays to impacted inbound flights. Based on this anticipated delay information, each airline runs its recovery module – the Active Recovery Module in Figure 4-1 – to reduce adverse impacts of delays and disruptions through recovery operations, such as flight cancellations, flight retimings, aircraft swaps, etc. Due to intra-day airport capacity uncertainty, these recovery operations may not get executed as planned. The active recovery module is based on the assumption that the weather forecast at the time of designing the GDP is accurate. However, as the day progresses, the realized weather conditions might be different than initially anticipated, thus rendering some of the planned recovery operations infeasible. For instance, if it turns out that the forecast underestimated the extent of bad weather, then some flights can be expected to have additional airborne delays beyond those forecast at the time of GDP design. This may cause some of the aircraft and passenger connections, which were initially deemed feasible in the recovery plan, to get disrupted. In such cases, airlines may need additional recovery actions to get their schedules back on track. We model this step in the Passive Recovery Module. We call it "passive" because such additional disruptions caused by inaccurate delay information often require urgent fixes, and hence the airlines usually don't have enough time to come up with a sophisticated new recovery plan. Such plans need to be simple in nature, and thus can be less effective compared to the recovery plans developed by the Active Recovery Module. For this reason, we model the Passive Recovery Module as the simple propagation of delay to downstream flights if the aircraft connections in the original recovery plan become infeasible due to delays to an upstream flight. The Passive Recovery Module also re-accommodates the passengers if their itineraries are disrupted by these additional unplanned delays. Note that along with the RBS process, FAA also runs a compression procedure approximately hourly to re-allocate empty slots created by flight cancellations across all airlines based on some predefined rules. However, since the compression procedure is a dynamic process with making empty slots available at different points of time during the GDP duration, we do not model it in the simulator for simplicity.

We refer to the total delay cost calculated by the Active Recovery Module as the "planned cost". This is calculated based on FAA's capacity forecasts at the time of GDP design. We call the additional delay cost calculated by the Passive Recovery Module as the "unplanned cost". This is calculated based on realized capacity values. The sum of the planned and unplanned costs are the total cost incurred by airlines during the GDP. Note that the unplanned cost value might be negative in some cases, such as when the FAA forecasts result in overestimation of the negative impacts of the GDP. In such cases, some flights may depart earlier than their allocated departure time in the GDP design (though no flights can depart sooner than their scheduled departure times). The unplanned cost there represents the amount of cost reduction brought about by an adjustment to the GDP, such as an early cancellation of the GDP because of weather clearing sooner than forecasted. However, even in such cases of cost reduction, unnecessarily costly and irreversible actions may have already been taken to accommodate an anticipated larger and/or longer-term reduction in capacity than what is realized, even though such reductions do not actually materialize. In Section 4.2.1, we describe in detail the modeling approach used within the Active Recovery Module; and in Section 4.2.2, we describe the underlying logic of the Passive Recovery Module.

### 4.2.1 Active Recovery Module

Given the flight departure times assigned by RBS, our integrated *schedule, aircraft and passenger recovery model* (SAPRM) simulates airlines' optimal recovery operations in response to a planned GDP. It integrates schedule recovery, aircraft re-routing and passenger re-accomodation decisions. Schedule recovery decisions may include actions such as flight cancellations and flight re-timings. There are two kinds of re-timing decisions included in our model. First, if a flight is not an inbound flight to the airport where the GDP is to be implemented (simply called the GDP airport from here onward), we call it a *non-controlled flight*. Such a flight can be but need not to be intentionally delayed. Alternatively, if a flight is an inbound flight to the GDP airport, we call it a *controlled flight*. A controlled flight must be re-assigned to an available arrival slot which is later than its scheduled arrival time. Aircraft re-routing decisions mitigate delay by swapping aircraft within the same fleet family, i.e., aircraft produced by the same manufacturer. Due to data unavailability, crew recovery is not explicitly considered in our model. However, we ensure compatibility between the original flight's planned crew and the re-assigned aircraft type for that flight since we only allow swap aircraft within the same fleet family. This reduces the need for crew re-assignments and is expected to reduce the additional crew recovery costs. Passenger recovery decisions are captured through modeling of passenger delays and disruptions. To model the rebooking process for disrupted passengers, we use delay cost estimates for disrupted passengers using a passenger delay calculation algorithm (Bratu, 2003) during the preprocessing step. For simplicity, we assume that all passengers have itineraries with at most one connection. This assumption is not very restrictive as nearly 98% of all passengers in the US domestic air transportation markets have at most one connection (Barnhart et al., 2014).

### Modeling Approach

Instead of a leg-based model, we develop a *string-based model* which utilizes flight strings, a concept introduced by Barnhart et al. (1998a). The model is adapted from the integrated recovery model developed in Petersen et al. (2012) and Bratu and Barnhart (2006). A *flight string* is a sequence of flights to be operated by the same aircraft, where each flight has its assigned arrival and departure times. A flight string simultaneously represents aircraft reroutings and flight retiming decisions. The same sequence of flights might be present in multiple strings, although each such sequence must have a unique set of flight retiming decisions. A string-based model has a number of advantages. Although the number of strings grows significantly with the number of flights, efficient column generation techniques can be employed. Strings are also able to capture network effects and routings that individual flight variables are not capable of capturing. This makes the string-based model a natural fit for evaluation framework in Figure 4-1.

We begin our detailed model description with the following notation:

**Sets:**

$F$: set of flights, indexed by $f$;

$T$: set of aircraft tail numbers, indexed by $t$;

$S(t)$: set of flight strings available to aircraft tail $t \in T$, indexed by $s$;

$R$: set of arrival slots, indexed by $r$;

$R(s)$: set of arrival slots used by string $s \in S(t)$, $t \in T$, indexed by $r$;

$F(s)$: set of flights included in string $s \in S(t)$, $t \in T$, indexed by $f$;

$P$: set of all passenger itineraries, indexed by $p$;

$P^2$: set of passenger itineraries containing two flight legs, indexed by $p$;

$F(p)$: set of all flight legs in itinerary $p \in P$, indexed by $f$;

$\overline{f}(p)$: first flight leg in itinerary $p \in P$;

$\underline{f}(p)$: last flight leg in itinerary $p \in P$;

**Parameters:**

$c_{t,s}$: flight delay cost of assigning aircraft tail $t \in T$ to flight string $s \in S(t)$;

$n_p$: number of passengers on planned itinerary $p \in P$;

$d_p$: disruption cost per passenger on itinerary $p \in P$;

$c_{\text{pax}}$: delay cost per passenger minute;

$t_{\text{minCT}}$: minimum passenger connection time;

$t_p^{\text{STA}}$: scheduled arrival time of itinerary $p \in P$, i.e., scheduled arrival time of $\underline{f}(p)$;

$t_{f,s}^{\text{dep}}$: departure time of flight $f$ in flight string $s$;

$t_{f,s}^{\text{arr}}$: arrival time of flight $f$ in flight string $s$.

**Decision Variables:**

$$z_f = \begin{cases} 1 & \text{if flight } f \in F \text{ is canceled,} \\ 0 & \text{otherwise,} \end{cases}$$

$$x_{t,s} = \begin{cases} 1 & \text{if aircraft tail } t \in T \text{ is assigned to flight string } s \in S(t), \\ 0 & \text{otherwise,} \end{cases}$$

$$\lambda_p = \begin{cases} 1 & \text{if itinerary } p \in P \text{ is disrupted,} \\ 0 & \text{otherwise,} \end{cases}$$

$t_p =$ delay, in minutes, to itinerary $p \in P$.

**Model Formulation:**

The SAPRM formulation is then given as follows:

$$\min \quad \sum_{t \in T} \sum_{s \in S(t)} c_{t,s} x_{t,s} + \sum_{p \in P} (d_p n_p \lambda_p + c_{\text{pax}} n_p t_p) \tag{4.1}$$

$$\text{s.t.} \quad \sum_{t \in T} \sum_{\substack{s \in S(t): \\ F(s) \ni f}} x_{t,s} + z_f = 1, \ \forall f \in F \tag{4.2}$$

$$\sum_{t \in T} \sum_{\substack{s \in S(t): \\ R(s) \ni r}} x_{t,s} \leq 1, \ \forall r \in R \tag{4.3}$$

$$\sum_{s \in S(t)} x_{t,s} \leq 1, \ \forall t \in T \tag{4.4}$$

$$\lambda_p \geq z_f, \ \forall p \in P, \forall f \in F(p) \tag{4.5}$$

$$\sum_{t \in T} \sum_{\substack{s \in S(t): \\ F(s) \ni \underline{f}(p)}} t^{\text{dep}}_{\underline{f}(p),s} x_{t,s} - \sum_{t \in T} \sum_{\substack{s \in S(t): \\ F(s) \ni \overline{f}(p)}} t^{\text{arr}}_{\overline{f}(p),s} x_{t,s} \geq t_{\text{minCT}} - M\lambda_p, \ \forall p \in P^2 \tag{4.6}$$

$$t_p \geq \sum_{t \in T} \sum_{\substack{s \in S(t): \\ F(s) \ni \underline{f}(p)}} t^{\text{arr}}_{\underline{f}(p),s} x_{t,s} - t^{\text{STA}}_p - M\lambda_p, \ \forall p \in P \tag{4.7}$$

$$x_{t,s} \in \{0,1\}, \ \forall t \in T, \forall s \in S(t)$$

$$z_f \in \{0,1\}, \ \forall f \in F$$

$$\lambda_p \in \{0,1\}, t_p \geq 0, \ \forall p \in P$$

The objective function (1) minimizes the sum of string assignment costs (i.e., flight delay costs), the passenger delay costs and passenger disruption costs. Flight assignment constraints (2) require a flight to be either included in exactly one string or canceled. Arrival slot assignment constraints (3) ensure that every arrival slot can be utilized by at most one flight. Similarly, aircraft tail assignment constraints (4) restrict an aircraft tail to be assigned to at most one flight string. On the passenger side, itineraries with insufficient connection time (constraints (6)) or with one or more canceled flight legs (constraints (5)) are classified as disrupted. $M$ in constraints (6) and (7) is a sufficiently large number. Constraints (7) ensure that the itinerary arrival delay, $t_p$, equals the arrival delay of the last flight in itinerary $p$ if itinerary $p$ is not disrupted; and is 0 otherwise.

**Cost Coefficients:**

We use real-world airline economic data to obtain the parameters of our model.

Flight delay cost $c_{t,s}$ is estimated from Form 41 Air Carrier Financial Data (Department of Transportation, 2007). For each airline, we first use Form 41 data to estimate per minute operating cost excluding fuel costs, because we assume no airborne delay will occur during the active recovery phase and fuel spent per minute waiting on the ground is negligibly small compared with that spent flying in the air. Note that this is a reasonable assumption because most of the ground delay is typically absorbed at the gate itself. We then calculate the total ground delay associated with assigning tail $t$ to string $s$. Multiplying it with the per minute cost, we get $c_{t,s}$. Passenger delay cost $c_{\text{pax}}$ is set equal to \$37.6 per hour according to Ball et al. (2010). Passenger itineraries information is generated through algorithms developed by Barnhart et al. (2014) based on schedules and aggregated demand data that is reported publicly. Per passenger disruption cost $d_p$ for each itinerary $p$ is estimated using the passenger delay calculator algorithm developed by Bratu and Barnhart (2005). The basic idea of this algorithm is to estimate the delay cost per passenger on itinerary $p$ by re-accommodating these passengers onto the earliest arriving alternative itinerary with available capacity, assuming that itinerary $p$ is disrupted and all other itineraries are not. $d_p$ serves as an approximation of the true disruption costs because we do not explicitly model passenger flows - for tractability reasons - to get accurate delay minutes. The same approach was adopted in Bratu and Barnhart (2006) and reported to have successful performance. We want to point out that our model specification is customized for each individual airline wherein the parameters specified for each airline are different and are based on the actual economic and operational data of that airline. This helps in capturing inter-airline differences in terms of their inherently different schedules, passenger itineraries, network structures and cost structures.

### Underlying Flight-Connection Network:

For each fleet family, we construct a directed flight-connection network $G = (V, A)$, where the node set $V$ represents the set of flights and the arc set $A$ is the set of potential connections between flights. Each flight leg $f \in F$ is represented by a node. Moreover, for each controlled flight $f$, we also generate its corresponding set of *slot*

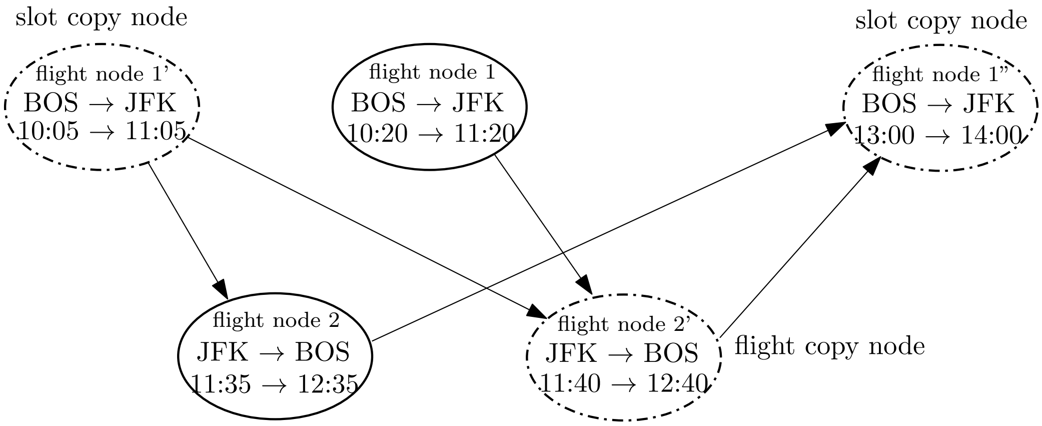


Figure 4-2: Illustration of A Flight-Connection Network

*copy nodes* and add them to our flight-connection network. The timing of each slot copy node equals that of one eligible arrival slot for flight $f$, that is, the slot cannot be earlier than the scheduled arrival time of flight $f$; and it cannot be later than 180 minutes after it or the slot allocated to it under the original GDP design, whichever is later. The complete set of available slots is generated through the RBS algorithm based on the GDP design. For each uncontrolled flight, we generate *flight copy nodes* with departure delays ranging between 0 and 180 minutes in intervals of 5 minutes. A connection arc exists from a node $i$ to another node $j$ if the arrival airport of $i$ is also the departure airport of $j$ and the arrival time of $i$ plus the minimum turn time is earlier than the departure time of $j$. Figure 4-2 illustrates a part of a sample connection network where the GDP airport is JFK. Flight (BOS → JFK), whose scheduled departure and arrival times are 10:00 am and 11:00 am respectively, is an inbound flight impacted by GDP with delayed departure and arrival times of 10:20 am and 11:20 am respectively (represented by flight node 1 in a solid circle). This flight has two slot copy nodes depicted in the figure: node $1'$ and node $1''$, each one corresponding to an eligible arrival slot for flight 1. Note that flight node $1'$ has an earlier arrival time than node 1, but its arrival time (11:05 am) is later than the original scheduled arrival time of 11:00 am. Thus it is an allowable slot copy for flight (BOS → JKF). Flight (JFK → BOS) is an uncontrolled flight which has one 5-minute flight copy node depicted in Figure 4-2 as node $2'$. Figure 4-2 assumes a minimum turn time of 20 minutes.

**Solution Approaches**

Given the large number of flight strings, we initially generate only a subset of the strings in SAPRM. We call this simplified version of the optimization problem (1) through (7) the Restricted Master Problem (RMP). The problem is then iteratively solved using delayed column generation, with new columns generated by solving a *Pricing Subproblem.* As described in Section 2.1.1, the directed network $G = (V, A)$ consists of flight copies as nodes. We extend this network by adding an extra pair of dummy nodes, referred to as the source node $s$ and sink node $t$. In this extended network, a flight string variable corresponds to a directed path from node s to node t. Paths are constructed by computing the reduced cost for every arc $a \in A$. The reduced cost of a path then equals the sum of the reduced costs of individual arcs constituting the path. At each iteration, a path with the most negative reduced cost is generated by solving the pricing subproblem. Thus the pricing subproblem is a shortest $s - t$ path problem where the costs of all the arcs $a \in A$ are calculated using dual variable values of the constraints in the RMP. We refer interested readers to Barnhart et al. (1998a,b) for a detailed treatment of this column generation-based solution process. However, note that our formulation requires one additional constraint to be satisfied for an $s - t$ path to be feasible. Specially, a feasible path cannot visit more than one node corresponding to the same flight. For the example network presented in Figure 4-2, a path such as node 1 $\rightarrow$ node $2'$ $\rightarrow$ node $1''$ is infeasible because flight 1 is visited twice along the path. Incorporation of this constraint requires the solution of a multi-label shortest path problem, thereby increasing the time to solve it. This can have a significant impact on solution time and tractability when the pricing subproblem has to be solved repeatedly. We address this issue by exploiting the structure of our problem. Specially, it is very rare for a feasible string to visit a flight more than once. This is so because for a non-controlled flight, its latest node is no more than 3 hours later than the earliest arrival time, that is, its scheduled arrival time. For a controlled flight, its latest node is usually 3 to 4 hours later than the earliest arrival time in our experiments, which is the earliest arrival slot later than its scheduled arrival time.

It can be more than 3 hours because the delayed arrival time according to the GDP might be more than 3 hours later than its scheduled arrival time. This implies that for a flight to be included twice in a string, an aircraft needs to complete a round trip plus twice the minimum turn time within 3 to 4 hours, a relatively rare occurrence.

Based on this observation, we propose the fast heuristic described in Algorithm 4 below. Because the underlying flight-connection network is acyclic, we speed up computational performance by performing a topological sort of the node and then execute label-correcting operations in the topological order of the nodes. The only difference between Algorithm 4 and the shortest path algorithm for one-to-all shortest paths is as follows. When updating the node cost label, we ensure that the update does not create a path visiting any flight more than once. While Algorithm 4 is not guaranteed to find the most negative reduced cost string every time, our computational experiments almost always produce optimal solutions to the pricing subproblem.

**Algorithm 4** Shortest path with node-visit restriction

```
Flight-connection network $G = (V, A)$
Initialize the cost of each node (except source node) to $+\infty$
Initialize the cost of source node to 0
Create a topological order of all nodes
for each node $n$ in topological order do
    for each node $n'$ such that there exists a directed arc from $n$ to $n'$ do
        if path from $s$ to $n'$ through $n$ does not contain any node which shares the same flight number as $n'$ then
            if cost of $n$ + cost of edge $(n, n')$ < cost of $n'$ then
                cost of $n'$ = cost of $n$ + cost of edge $(n, n')$
                Set predecessor of node $n'$ as $n$
            end if
        end if
    end for
end for
return shortest s-t path
```

To obtain integer solutions for SAPRM, we first solve the corresponding linear programming (LP) relaxation using column generation. We then solve the resultant model as a mixed-integer linear program by imposing necessary integrality constraints while using the same set of columns as generated in the LP solution process. This is a widely-used heuristic approach for solving large-scale integer programing problems without performing column generation at every node of the branch-and-bound tree.

### 4.2.2 Passive Recovery Module

The solution of SAPRM in the active recovery module generates a modified schedule, aircraft routing and passenger re-accommodation plan to minimize delay impacts. However, these recovery operations are generated based on the planned delay information provided by the FAA at the beginning of the GDP. Once more accurate weather information becomes available, the GDP might be (1) extended further if the weather turns out to be worse than anticipated; or (2) cancelled early if the capacity is restored sooner than forecast. In both these situations, the passive recovery module is used to maintain operational feasible under the new delay information. However, because these recovery operations are often executed in real-time, usually there is insufficient time to employ the method used in an active recovery process. Below, we introduce the functioning of the passive recovery module under two scenarios:

1. *When the GDP is extended*:

   In this case, further delays will be incurred. To ensure that the operations still remain feasible, we do not change the aircraft routes decided by the active recovery module, and simply let flight delays propagate along aircraft routes to ensure that the routes can be operated. The passengers incur additional delay and disruptions according to this further-delayed schedule.

2. *When the GDP is cancelled early*:

   In this case, flights might incur less delay than originally anticipated in the active recovery module. We allow flights to depart as soon as possible after the GDP cancellation time as long as the scheduled departure times are earlier than the GDP cancellation time. This reduces some flight delays and some passenger delays.

Note that in practice, it is possible that a GDP is revised more than once as weather conditions evolve. But, for simplicity, we only allow at most one revision in our evaluation framework. Given the often incremental nature of the later GDP revisions, we believe that it is a good approximation of what happens in real-world GDPs.

## 4.3 Case Study

Using the evaluation framework described in Section 4.2, we are able to simulate airlines' responses under different GDP designs. In this section, we first discuss the details of our experimental setup in Section 4.3.1. Then we present two different types of analyses. In Section 4.3.2, we highlight how different airlines differ in prioritizing among alternative GDP designs, and provide insights into the underlying operational factors that explain these differences. In Section 4.3.3, we assess the potential performance benefits of an airline-driven GDP design approach to the United States National Airspace System (NAS) (Swaroop and Ball, 2013; Yan et al., 2017a). We evaluate these benefits with respect to the baseline – a centralized approach for designing GDPs.

### 4.3.1 Experimental Setup

Actual flight schedules for a representative day in the summer of 2007 are obtained for the San Francisco International Airport (SFO). We do not use a more recent schedule because we are not able to get passenger itinerary data for more recent years. We assume that SFO operates at its Visual Flight Rules (VFR) capacity level before a GDP starts; operates at its Instrument Flight Rules (IFR) capacity level during the GDP; and then it returns to its VFR capacity level once the GDP ends. The arrival capacities under the VFR and IFR scenarios are obtained from the Airport Capacity Profiles (Federal Aviation Administration, 2014a). We set up 14 hypothetical GDPs. The planned duration of each GDP, defined as the difference between planned start and end times, is varied from 3 to 9.5 hours in steps of 30 minutes (i.e., 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5, 7, 7.5, 8, 8.5, 9, and 9.5 hours). We model the actual capacity reduction duration by a discrete uniform distribution over 3, 4, 5, 6, 7, 8, 9 and 10 hours, where each of these values have a probability of 1/8. Hence, a GDP with a planned duration of 3 hours is considered to be a highly aggressive design in the sense that with high probability 7/8, the airport capacity is over-estimated by the GDP design. On the other hand, a GDP with a planned duration of 9.5 hours is considered to be highly conservative because with a probability of 7/8, airport capacity is under-estimated by the GDP design. We vary the planned start time of the GDP from 11 am to 5 pm in steps of 1 hour each (i.e., 11 am, 12 pm, 1 pm, 2 pm, 3 pm, 4 pm and 5 pm) to control for any scheduling peculiarities associated with a particular part of the day. Thus, the results of any GDP design to be evaluated are averaged over these 7 different planned start times.

### 4.3.2 Revealing Airlines' Preferences

In this subsection, we highlight how different airlinesâĂŹ have differing preferences in prioritizing between alternative GDP designs. We describe the major operational factors that cause these differences. Table 1 provides an overview of some basic statistics about each airline operating at the SFO airport on our evaluation day. These include number of operations, fleet composition, number of passengers, percentage of connecting passengers, and the average load factors. Figures 4-3 to 4-5 are bar charts describing the total costs (in $) for each airline plotted against the planned GDP duration (in hours) on the x-axis. Each chart consists of 3 data series: 1) total planned cost; 2) total unplanned cost; and 3) total cost in the hypothetical scenario assuming no recovery actions. The total planned cost is the minimal cost produced by the active recovery module based on the FAA's planned GDP delay information. The total unplanned cost is the additional cost (or savings) incurred from the passive recovery module after the actual delay information is revealed. Note that the unplanned cost is an expected value calculated over the various values of actual GDP durations. The total realized cost incurred by an airline is simply the sum of the planned and the unplanned costs, which is depicted as the total height of the bars corresponding to the first two data series in Figures 4-3 to 4-5. The total cost with no recovery is the expected delay cost under a hypothetical scenario where no active recovery actions are undertaken, and instead the delays are simply allowed to propagate. This serves to quantify the cost reduction an airline's recovery operations can achieve, and is depicted in Figures 4-3 to 4-5 as the difference between the total height of the two bars corresponding to the first two data series and that of the third data series.

| **Airlines** | **Impacted Operations** | **Fleet Types** | **Impacted Passengers** | **Connecting Passengers (%)** | **Average Load Factor (%)** |
|---|---|---|---|---|---|
| **United & SkyWest** | 359 | 10 | 24,236 | 32.33% | 75.29% |
| **American & American Eagle** | 70 | 5 | 7,678 | 27.39% | 75.53% |
| **US Airways** | 40 | 4 | 4,007 | 31.57% | 80.43% |
| **Continental & ExpressJet** | 30 | 5 | 3,244 | 20.43% | 70.15% |
| **Delta** | 26 | 4 | 3,750 | 30.29% | 80.72% |
| **Alaska** | 25 | 2 | 2,461 | 9.47% | 75.28% |
| **Northwest** | 23 | 4 | 3,232 | 25.46% | 85.65% |
| **Frontier** | 15 | 2 | 1,351 | 31.68% | 79.35% |
| **JetBlue** | 9 | 1 | 1,180 | 8.05% | 78.46% |
| **AirTran** | 8 | 1 | 973 | 32.58% | 82.32% |

Table 4.1: Airline characteristics, for a summer day in year 2007 at SFO airport

Based on the trends in total realized costs, we categorize the airlines into 3 broad groups: (1) those that prefer an aggressive (shorter) GDP design, (2) those that prefer a moderate (intermediate duration) GDP design, and (3) those that prefer a conservative (longer) GDP design. Note that these categories are not meant to serve as a rigid categorization of these airlines. Instead, they are relevant only for the specific case of the SFO airport, for the day of operations being considered in our experiment, and under the assumptions made in this experiment about the GDP scenarios, capacity distributions, and other parameters. The first category – airlines preferring aggressive GDP designs – includes US Airways, Frontier Airlines, Northwest Airlines, and Continental/ExpressJet. The second category – airlines preferring moderate GDP designs – includes Delta Airlines, American/American Eagle, Alaska Airlines and JetBlue Airways. The third category – airlines preferring conservative GDP designs – includes United/SkyWest and AirTran Airways. Figures 4-3 through 4-5 show that the three categories of airlines correspond to three different broad trends in the variation of total realized costs with planned GDP durations. The total realized cost for the airlines in the first category shows a close to monotonic increase (the total height of the first two data series in the bar charts) as the planned GDP duration increases. Thus, category 1 airlines have a preference for an aggressive, shorter-duration GDP design. The total realized cost for category 2 airlines tend to be comparatively flat and roughly convex, with the point of minimum total realized cost occurring somewhere in the middle (far from either extreme). Thus, they have a preference for a moderate, intermediate-duration GDP design. The total realized cost for the category 3 airlines displays a near-monotonic decreasing trend with planned GDP duration, and thus they have a preference for a conservative, longer-duration GDP design.

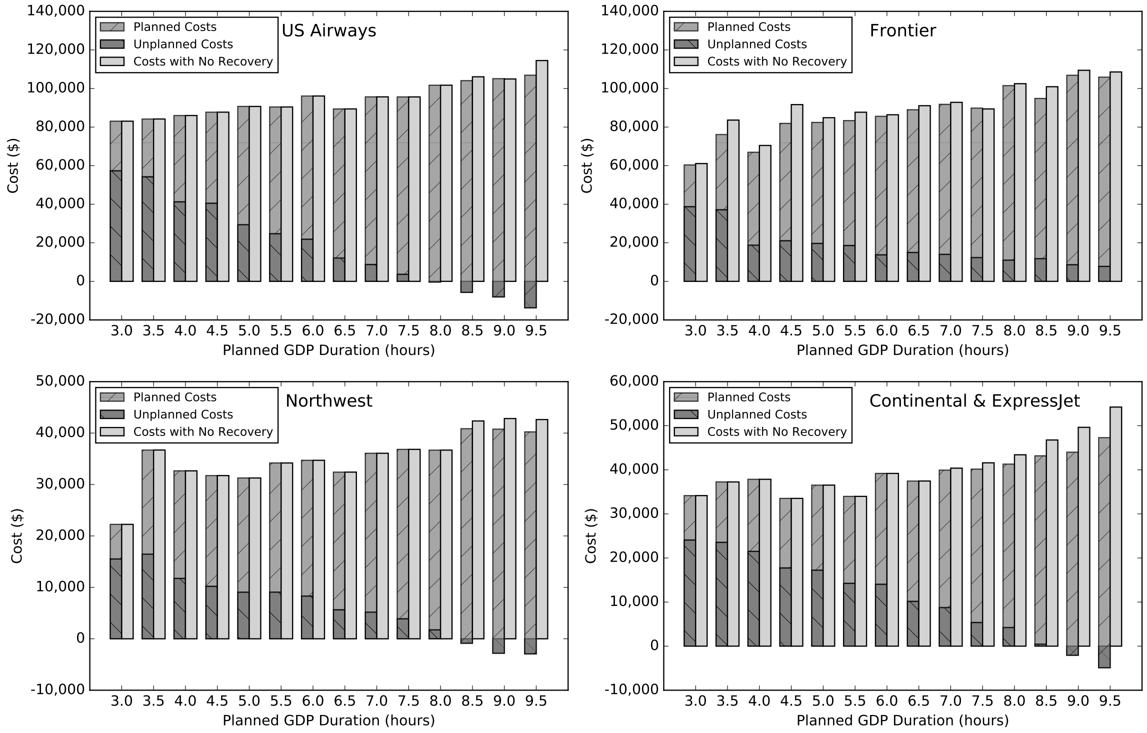


Figure 4-3: Airlines That Prefer Aggressive, Shorter-Duration Designs

These differences between airline categories can be explained in several different ways. One prominent difference becomes visible when we focus on the total cost in the absence of any recovery actions by the airline. This is the third data series in the bar charts. The longer the planned duration of a GDP, the greater is the extent of disruption to the airlineâĂŹs network and hence, the greater is the total realized cost in the absence of any recovery. In fact, Figures 4-3 through 4-5 indicate that

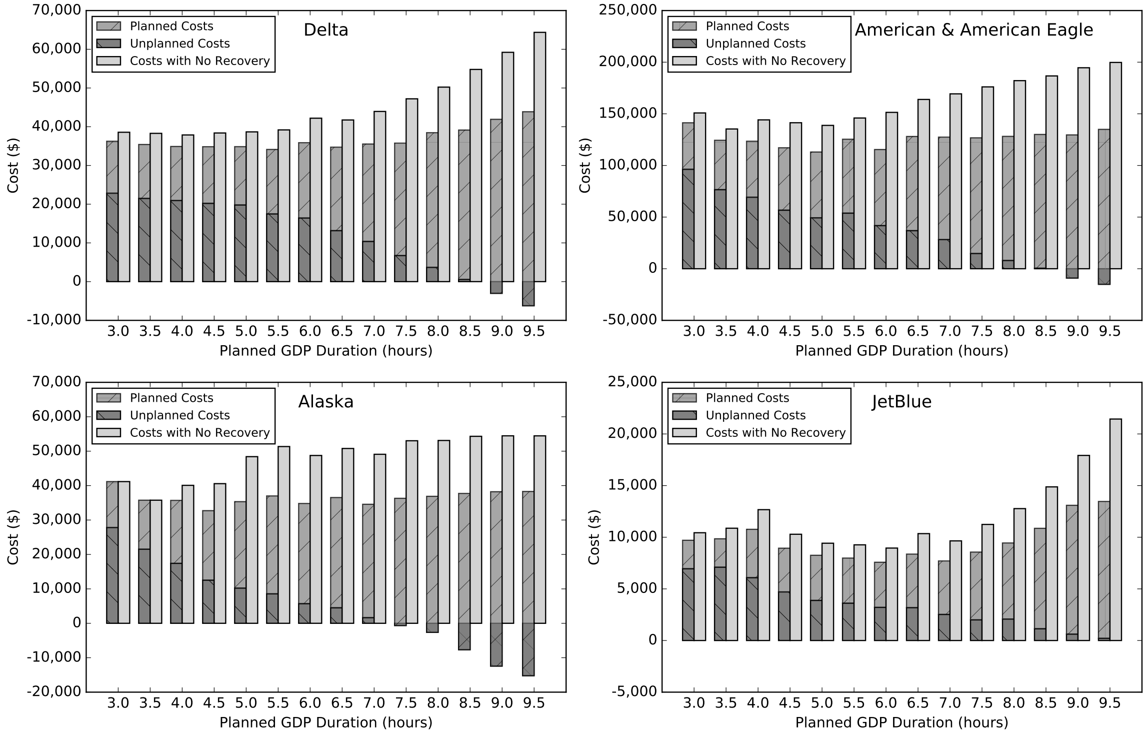


Figure 4-4: Airlines That Prefer Moderate, Medium-Duration Designs

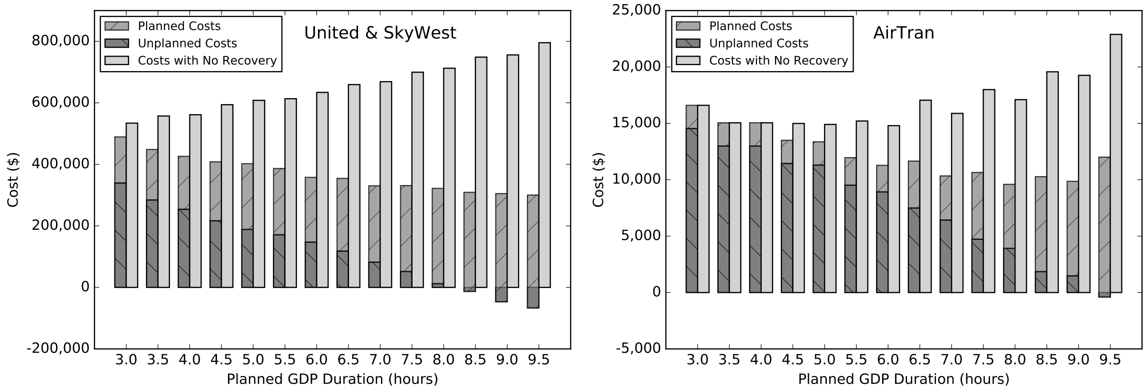


Figure 4-5: Airlines That Prefer Conservative, Long-Duration Designs

the third data series increases almost monotonically with increases in planned GDP duration for all airlines. When a GDP is announced, however, airlines can plan their recovery actions for the operations that are likely to be affected due to the GDP. The longer the planned GDP duration, the more extensive is the set of possible recovery actions that an airline might have to take. Therefore, the difference between the total height of the two bars corresponding to the first two data series and the height of the third data series, indicating the cost reduction realized through the airlineâĂŹs recovery actions, shows a generally increasing trend with increases in the planned GDP duration. Note that some airlines are better than others at taking advantage of advance knowledge of a longer GDP. Such differences are primarily explainable in terms of various operational factors, such as, but not limited to, the number of operations and the number of different fleet types for the airline at the impacted airport. As a result of these differences, some airlines are able to reduce delay costs considerably through effective recovery actions, more so than some other airlines. This phenomenon appears to be an important determinant in classifying airlines into our three categories. We summarize the ratio of total realized cost to total cost without recovery operations in Table 4.2.

For all four category 1 airlines, across all planned GDP durations, the ratio of total realized cost to total cost without recovery does not fall below 87%. The recovery potential for these airlines is low even when faced with long planned GDP durations. For all four of the category 2 airlines, the ratio of total realized cost to total realized cost without recovery is close to 70% for longer GDP durations. This means that these airlines are able to achieve a moderate amount of cost reduction through effective recovery. Finally, for both category 3 airlines, the ratio of total realized cost to total realized cost without recovery is close to (or below) 50% for longer GDP durations. This suggests that these airlines can greatly reduce their realized costs through effective recovery decisions. As a result of these differences, category 1 airlines prefer an aggressive, shorter-duration GDP design, because they do not benefit much from recovery decisions when the GDPâĂŹs are designed to be longer. Moreover, with shorter planned durations, there is a chance that the GDP will end

| **Airlines** | | **GDP Planned Duration (hours)** | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | **3.0** | **3.5** | **4.0** | **4.5** | **5.0** | **5.5** | **6.0** | **6.5** | **7.0** | **7.5** | **8.0** | **8.5** | **9.0** | **9.5** |
| **Categ. 1** | **US Airways** | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0.98 | 1.00 | 0.93 |
| | **Frontier** | 0.99 | 0.91 | 0.95 | 0.89 | 0.97 | 0.95 | 0.99 | 0.98 | 0.99 | 1.00 | 0.99 | 0.94 | 0.98 | 0.98 |
| | **Northwest** | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0.96 | 0.95 | 0.94 |
| | **Continental & ExpressJet** | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0.99 | 0.97 | 0.95 | 0.92 | 0.89 | 0.87 |
| **Categ. 2** | **Delta** | 0.94 | 0.92 | 0.92 | 0.91 | 0.90 | 0.87 | 0.85 | 0.83 | 0.81 | 0.76 | 0.77 | 0.71 | 0.71 | 0.68 |
| | **American & American Eagle** | 0.94 | 0.92 | 0.86 | 0.83 | 0.81 | 0.86 | 0.76 | 0.78 | 0.75 | 0.72 | 0.70 | 0.70 | 0.67 | 0.68 |
| | **Alaska** | 1.00 | 1.00 | 0.89 | 0.81 | 0.73 | 0.72 | 0.71 | 0.72 | 0.70 | 0.68 | 0.69 | 0.69 | 0.70 | 0.70 |
| | **JetBlue** | 0.93 | 0.91 | 0.85 | 0.87 | 0.88 | 0.86 | 0.85 | 0.81 | 0.80 | 0.76 | 0.74 | 0.73 | 0.73 | 0.72 |
| **Categ. 3** | **United & SkyWest** | 0.92 | 0.80 | 0.76 | 0.69 | 0.66 | 0.63 | 0.56 | 0.54 | 0.49 | 0.47 | 0.45 | 0.41 | 0.40 | 0.38 |
| | **AirTran** | 1.00 | 1.00 | 1.00 | 0.90 | 0.90 | 0.79 | 0.76 | 0.68 | 0.65 | 0.59 | 0.56 | 0.52 | 0.51 | 0.52 |

Table 4.2: Ratio of Total Realized Cost to Total Cost without Recovery Operations

as planned and the airline will be positioned to take full advantage of that. Category 3 airlines prefer a conservative, longer-duration GDP design primarily because they can choose from a larger set of recovery alternatives and recover a large proportion of the cost if they plan for a longer GDP duration.

Comparing the values in Table 4.1 for the different airline categories provides insights as to why the airlines have such different recovery potentials. United/SkyWest Airlines is the airline with by far the largest presence at SFO. It has 359 operations, which is approximately 50% more than the number of operations for the remaining nine airlines put together. As a result, they have the greatest recovery potential because there are many more opportunities for swapping aircraft, for swapping slots, and for rebooking passengers onto alternative itineraries. AirTran Airways also has a high recovery potential. Even though it has only eight operations at SFO, all eight operations have the same aircraft type making swapping of aircraft very easy and thus, ensuring flexibility in its recovery plans. At the other extreme, we see that category 1 airlines such as Northwest Airlines and Continental/ExpressJet Airlines have few operations and a relatively larger variety of aircraft types. Their recovery capabilities are the worst, likely due to fewer aircraft swapping opportunities.

Another likely reason for the low recovery potential could be the high values of average load factors for airlines such as Northwest Airlines and US Airways, there by making makes passenger reaccommodation more difficult.

As shown in the first and second data series in Figures 4-3 to 4-5, the planned costs increase almost monotonically and the unplanned costs decrease almost monotonically with the planned GDP durations for all airlines. Variations in the planned cost when compared with the variations in the unplanned costs provide further insights into our airline categorization scheme and the factors leading to the operational differences between airlines. We define the “cost ratio” as the ratio of the difference between the maximum and minimum planned costs and the difference between the maximum and minimum unplanned costs. We summarize this cost ratio for different airlines in Table 4.3.

United/SkyWest and AirTran Airways are the only two airlines that have a cost

| Airlines | | Cost Ratio |
|---|---|---|
| **Category 1** | **US Airways** | 1.34 |
| | **Frontier** | 2.47 |
| | **Northwest** | 1.97 |
| | **Continental & ExpressJet** | 1.45 |
| **Category 2** | **Delta** | 1.26 |
| | **American & American Eagle** | 0.94 |
| | **Alaska** | 0.93 |
| | **JetBlue** | 1.56 |
| **Category 3** | **United & SkyWest** | 0.53 |
| | **AirTran** | 0.69 |

Table 4.3: Cost Ratios of Different Airlines

ratio below 90%. In fact they are both considerably lower than 90%, with 53% for United/SkyWest and 69% for AirTran. This means that for these two airlines, the reduction in unplanned costs is larger than the increase in planned costs as the planned GDP duration increases. As a result, these airlines prefer a longer planned duration to ensure total cost is minimized. Not surprisingly, these are the only two category 3 airlines in our data. At the other extreme, we have airlines such as Frontier Airlines (247%) and Northwest Airlines (197%), which have the highest cost ratios across all airlines. This means that for these two airlines, the reduction in unplanned costs is smaller than the increase in planned costs as the planned GDP duration increases. As a result, these airlines prefer a shorter planned duration to ensure that the total realized cost is minimized. Not surprisingly, these are two of the four category 1 airlines in our data.

### 4.3.3 Airline-Driven GDP Design Approach and Airport-wide Benefits Assessment

We now sum all the airlines' total realized costs to conduct an airport-wide performance analysis under different GDP designs. Table 4.4 summarizes the total realized costs (the sum of planned and unplanned costs as we discussed in Section 4.3.2) for each airline under different GDP designs. The italicized number in each row is the minimum among all the numbers in that row. Table 4.4 also provides an airport-wide total cost, defined as the sum of total realized costs incurred by all airlines operating at the airport as a result of the GDP. The last row in the table lists the centralized objective function value under different designs. Here, the centralized objective function refers to the summation of airborne delay costs and ground delay costs for all flights heading into the GDP-affected airport. This objective function is used extensively in centralized GDP decision-making approaches, for example, Richetta and Odoni (1993), Mukherjee and Hansen (2007), just to name a few. The GDP design with the minimum centralized objective function value serves as our baseline, representing the state-of-the-research design.

We conclude from Table 4.4 that a GDP design with planned duration of 8 hours is the cost-minimizing design from the centralized decision-making perspective (the state-of-the-research design), with the total airport-wide cost of \$825,808. The system optimal design, that is, the one with the lowest airport-wide cost, has a planned duration of 7 hrs and a total cost \$809,297, which is 2.0% less than that of the centralized design. Note that since each minute of airborne delay is usually much more expensive than each minute of ground delay (roughly a 3:1 ratio), the centralized decision-making approach is very conservative, independent of the characteristics of the airlines operating at the airport under consideration. Thus, delay cost reduction could be even larger at airports where the majority of airlines prefer aggressive GDP designs.

To assess the performance of an airline-driven GDP design approach in reducing airport-wide delay costs, in Table 4.5, we use a linear transformation to convert all airlines' total realized costs (summarized in Table 4.4) into grades, with a maximum grade of 100. The weight of a particular airline is calculated based on its number of operations impacted by the GDP. Because the airline with the largest number of impacted operations (United Airlines) has over half of the total operations, we cannot directly use the number of operations as the weights if we want to avoid United Airlines' decisions simply dictating the final outcome of the majority judgment process. Here we fix the largest airline's (United AirlineâĂŹs) proportion of total weight at 40% and use a power-root transformation as follows. Let $o_i$ be the number

| Airlines | | GDP Planned Duration (hours) | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | **3** | **3.5** | **4** | **4.5** | **5** | **5.5** | **6** | **6.5** | **7** | **7.5** | **8** | **8.5** | **9** | **9.5** |
| **Categ. 1** | **US Airways** | *83,058* | 84,186 | 85,994 | 87,735 | 90,695 | 90,418 | 96,115 | 89,400 | 95,663 | 95,637 | 101,711 | 104,089 | 105,107 | 106,905 |
| | **Frontier** | *60,362* | 76,148 | 66,946 | 81,898 | 82,396 | 83,363 | 85,580 | 88,988 | 91,783 | 89,850 | 101,507 | 94,825 | 106,871 | 105,891 |
| | **Northwest** | *22,247* | 36,705 | 32,657 | 31,738 | 31,265 | 34,185 | 34,704 | 32,411 | 36,074 | 36,831 | 36,690 | 40,855 | 40,764 | 40,228 |
| | **Continental & ExpressJet** | 34,152 | 37,247 | 37,844 | *33,511* | 36,526 | 33,968 | 39,176 | 37,459 | 39,935 | 40,162 | 41,300 | 43,174 | 44,005 | 47,296 |
| **Categ. 2** | **Delta** | 36,256 | 35,408 | 34,897 | 34,846 | 34,860 | *34,132* | 35,880 | 34,732 | 35,531 | 35,773 | 38,467 | 39,139 | 41,918 | 43,874 |
| | **American & American Eagle** | 141,340 | 124,389 | 123,490 | 117,142 | *112,998* | 125,420 | 115,407 | 128,040 | 127,462 | 126,762 | 128,174 | 130,014 | 129,585 | 134,946 |
| | **Alaska** | 41,167 | 35,758 | 35,713 | *32,724* | 35,337 | 37,002 | 34,810 | 36,539 | 34,573 | 36,305 | 36,882 | 37,731 | 38,215 | 38,301 |
| | **JetBlue** | 9,705 | 9,849 | 10,766 | 8,939 | 8,252 | 7,983 | *7,577* | 8,367 | 7,707 | 8,563 | 9,446 | 10,863 | 13,090 | 15,468 |
| **Categ. 3** | **United & SkyWest** | 489,250 | 448,340 | 426,198 | 408,230 | 402,122 | 386,515 | 357,885 | 354,516 | 330,232 | 330,824 | 322,038 | 309,187 | 304,852 | *300,218* |
| | **AirTran** | 16,600 | 15,049 | 15,050 | 13,499 | 13,363 | 11,954 | 11,280 | 11,651 | 10,338 | 10,645 | *9,592* | 10,268 | 9,864 | 12,001 |
| **Airport-wise Cost** | | 934,137 | 903,079 | 869,554 | 850,262 | 847,815 | 844,941 | 818,413 | 822,104 | *809,297* | 811,352 | 825,808 | 820,144 | 834,271 | 845,128 |
| **Centralized Objective** | | 244,986 | 235,343 | 226,604 | 221,638 | 214,614 | 210,056 | 202,624 | 201,951 | 196,292 | 189,613 | *188,450* | 195,662 | 209,204 | 220,389 |

Table 4.4: Airline Costs, Airport-wide Costs, and Centralized Objective Function Values

| Airlines | | Impacted Operations | Weights | GDP Planned Duration (hours) | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | **3.0** | **3.5** | **4.0** | **4.5** | **5.0** | **5.5** | **6.0** | **6.5** | **7.0** | **7.5** | **8.0** | **8.5** | **9.0** | **9.5** |
| **Categ. 1** | **US Airways** | 40 | 12.28 | 100 | 99 | 97 | 95 | 92 | 92 | 86 | 93 | 87 | 87 | 82 | 80 | 79 | 78 |
| | **Frontier** | 15 | 6.3 | 100 | 79 | 90 | 74 | 73 | 72 | 71 | 68 | 66 | 67 | 59 | 64 | 56 | 57 |
| | **Northwest** | 23 | 8.43 | 100 | 61 | 68 | 70 | 71 | 65 | 64 | 69 | 62 | 60 | 61 | 54 | 55 | 55 |
| | **Continental & ExpressJet** | 30 | 10.1 | 98 | 90 | 89 | 100 | 92 | 99 | 86 | 89 | 84 | 83 | 81 | 78 | 76 | 71 |
| **Categ. 2** | **Delta** | 26 | 9.16 | 94 | 96 | 98 | 98 | 98 | 100 | 95 | 98 | 96 | 95 | **89** | 87 | 81 | 78 |
| | **American & American Eagle** | 70 | 17.97 | 80 | 91 | 92 | 96 | 100 | 90 | 98 | 88 | 89 | 89 | 88 | **87** | **87** | **84** |
| | **Alaska** | 25 | 8.92 | **79** | 92 | 92 | 100 | 93 | 88 | 94 | 90 | 95 | 90 | 89 | 87 | 86 | 85 |
| | **JetBlue** | 9 | 4.45 | 78 | **77** | **70** | 85 | 92 | 95 | 100 | 91 | 98 | 88 | 80 | 70 | 58 | 49 |
| **Categ. 3** | **United & SkyWest** | 359 | 54.63 | 61 | 67 | 70 | **74** | **75** | **78** | **84** | **85** | **91** | 91 | 93 | 97 | 98 | 100 |
| | **AirTran** | 8 | 4.11 | 58 | 64 | 64 | 71 | 72 | 80 | 85 | 82 | 93 | **90** | 100 | 93 | 97 | 80 |

Table 4.5: Airlines' Grades under Different GDP Designs and the Majority Winner Design

of operations of airline $i$, and let $o_{\max}$ be the number of operations of the largest airline. We choose $\alpha$ such that $\frac{o_{\max}^{\alpha}}{\sum_{i \in N} o_i^{\alpha}} = 0.4$, where $o_i^{\alpha}$ is the weight of airline $i$.

With these weights and grades, the majority grades of each candidate GDP design are given by the bold and underlined numbers in Table 4.5. The majority winner, that is, the design having the highest majority grade (of 91), turns out to be the one with planned duration of 7 hours. Interestingly, this coincides with the system optimal design shown in Table 4.4. Note that the set of GDP design candidates is an important factor determining the performance of our airline-driven approach. Previous work (Yan et al., 2017a; Swaroop and Ball, 2013; Swaroop, 2013) develop an iterative approach – generating new GDP candidate designs if the current winner is not good enough – to enhance the performance of the airline-driven approach. For the sake of concise presentation, the benefit assessment in this chapter considers only a single-round approach. Thus, the benefits presented here could be viewed as "lower bounds" on the true benefits under a complete iterative approach. We refer interested readers to aforementioned literature for a detailed theoretical exposition of the complete airline-driven approach.

Table 4.6 summarizes the extent to which each airline is worse-off under the centralized design and our airline-driven design relative to their most preferred design. The average percentage increase in the airlines' total realized costs under the airline-driven design is 19.06%, almost 6 percentage points smaller than that under the centralized design. The standard deviation of the increments under the airline-driven design is 19.78%, which is also more than 2% smaller than that under the centralized design. These results suggest that by explicitly incorporating airlines' preference variations, the airline-driven approach produces improved and also more equitable GDP designs in terms of distributing the delay costs across airlines.

We conclude this section by taking a look at performance metrics other than costs. In Table 4.7, we calculate the average ground delay in minutes per flight, the average airborne delay in minutes per flight, the average number of disrupted passengers, and the average passenger delay in minutes per passenger under different GDP designs. The airline-driven design compared with the centralized design, reduces

| **Airlines** | **Centralized Design** | **Airline-Driven Design** |
|---|---|---|
| **US Airways** | 22.46% | 15.18% |
| **Frontier** | 68.16% | 52.05% |
| **Northwest** | 64.92% | 62.15% |
| **Continental & ExpressJet** | 23.24% | 19.17% |
| **Delta** | 12.70% | 4.10% |
| **American & American Eagle** | 13.43% | 12.80% |
| **Alaska** | 12.71% | 5.65% |
| **JetBlue** | 24.67% | 1.72% |
| **United & SkyWest** | 7.27% | 10.00% |
| **AirTran** | 0.00% | 7.78% |
| **Mean** | **24.96%** | **19.06%** |
| **Standard Deviation** | **22.00%** | **19.78%** |

Table 4.6: Percentage Increment over Each Airline's Most Preferred Design

average ground delay by 2.21 minutes per flight, average passenger delay by 0.32 minutes per passengers, and the average number of disrupted passengers by 5. This is at the cost of increasing average airborne delay by 0.51 minutes per flight.

In summary, we find in this specific case study at the SFO airport that the system is better off in terms of both reducing costs and enhancing inter-airline equity when operated under the slightly more aggressive approach suggested by the airline-driven design rather than the state-of-the-research (centralized) approaches. These benefits of the airline-driven approach results from incorporating not only the trade-offs between airborne and ground delays, but most importantly, airlines' diverse preferences among GDP designs, because of their different business objectives and operating characteristics.

| **GDP Planned Duration (hours)** | **Avg. Ground Delay (minutes per flight)** | **Avg. Airborne Delay (minutes per flight)** | **No. of Disr. Passengers** | **Avg. Passenger Delay (minutes per passenger)** |
|---|---|---|---|---|
| **3** | 10.40 | 1.87 | 674 | 9.34 |
| **3.5** | 12.62 | 1.92 | 725 | 9.13 |
| **4** | 13.41 | 1.79 | 679 | 9.08 |
| **4.5** | 14.56 | 1.71 | 681 | 8.88 |
| **5** | 15.98 | 1.70 | 685 | 8.81 |
| **5.5** | 17.18 | 1.67 | 674 | 8.91 |
| **6** | 18.21 | 1.65 | 671 | 8.93 |
| **6.5** | 19.72 | 1.59 | 673 | 9.18 |
| **7 (Airline-Driven)** | 20.52 | 1.46 | 668 | 9.34 |
| **7.5** | 21.14 | 1.14 | 678 | 9.44 |
| **8 (Centralized)** | 22.73 | 0.95 | 673 | 9.66 |
| **8.5** | 24.70 | 0.54 | 646 | 10.12 |
| **9** | 25.34 | 0.37 | 667 | 10.34 |
| **9.5** | 26.59 | 0.00 | 662 | 10.81 |

Table 4.7: Overall Airport-wide Performance under Different GDP Designs

# Chapter 5

# Choice-Based Integrated Airline Fleet Assignment and Schedule Design

## 5.1 Introduction

Airlines often operate under thin and volatile profit margins. The Economist (2014) reported a less than 1% industry average profit for the U.S. airlines in 2012, though recently the U.S. airline industry profit margins have surged to 4.1% according to IATA (2017). In the U.S., many mergers have occurred during the past decade. This consolidation trend has been accompanied by efforts by the airlines toward better utilization of capacity, and a reduction in operations and frequencies in unprofitable markets. Indeed, the total number of departures in U.S. domestic markets has reduced more than 15% from 9.8 million in 2006 to 8.3 million in 2016 (Bureau of Transportation Statistics, 2016c), while the average load factor has surged from 79.1% to 84.6% in the same period (Bureau of Transportation Statistics, 2016b). These trends are directly related to two very important schedule planning decisions made by airlines, namely, fleet assignment and flight network/timetable development.

The airline fleet assignment model (FAM) assigns aircraft types to flight legs in an airline's schedule in order to maximize profit contribution subject to various operational constraints. Profit contribution is calculated as the difference between expected revenue and operating cost. Airline operating cost is dependent on both aircraft type characteristics and flight leg characteristics, and it consists of crew costs, fuel costs, landing fees and aircraft maintenance costs (Subramanian et al., 1994). Expected revenue is a function of flight schedules (of the airlines as well as competing airlines), flight seating capacity, passenger demand and the revenue management policy adopted by the airline.

Hane et al. (1995) proposed a *basic fleet assignment model* (BFAM) using a multicommodity flow formulation on a time-space network. BFAM lays the foundation for later development of FAM. However, it assumes that revenue of the entire network can be perfectly prorated over individual flight legs. This assumption is problematic because airlines sell fare products that can contain multiple flight legs. Each fare product is characterized by a given itinerary and a given fare class (e.g. a Y-class ticket from Seattle (SEA) to Boston (BOS) through Dallas/Fort Worth (DFW) on given flight legs) and an itinerary can use capacities on multiple flight legs, which creates network dependencies. Motivated by this, Barnhart et al. (2002) proposed an *itinerary-based fleet assignment model* (IFAM) which explicitly models itinerary demand and its network effects using a passenger mix model (PMM) to evaluate revenue. PMM is similar to, but a simplification of the network revenue management (NRM) problem in which airlines optimally control passenger flows to maximize revenue for given flight seating capacities. In PMM, fare products are typically aggregated by itineraries, itinerary demand is assumed to be deterministic, and passengers are assigned to itineraries. On the other hand, NRM is a stochastic control problem where demand is random and controlled by dynamically accepting or rejecting booking requests for fare products (usually implemented by adjusting a fare product offer set over time). Jacobs et al. (2008) develop an approach which integrates BFAM with NRM. The approach relies on a decomposition similar to Benders' decomposition, where the NRM is solved iteratively as a subproblem using stochastic optimization approaches.

A number of previous papers have studied the integrated airline fleet assignment model with schedule design (SD-FAM) given its potential for increasing profitability by simultaneously optimizing flight schedule and fleet assignment decisions. These models often do not attempt to design a schedule from scratch, instead they make modifications to an existing schedule. Desaulniers et al. (1997) studied an integrated model allowing flight departure times to be altered within a time window and proposed a solution methodology based on branch-and-price. Lohatepanont and Barnhart (2004) presented a model that selects flights to be operated from a base schedule containing some mandatory and some optional flights. Their model is an extension of IFAM (Barnhart et al., 2002). Sherali et al. (2010) conducted polyhedral analyses of an SD-FAM with optional flight selection, and proposed several techniques to enhance its computational performance.

Despite being focused on different stages of the scheduling process, fleet assignment and schedule design face a common modeling challenge involving spill and recapture behavior of passengers. In fleet assignment, when demand exceeds assigned seating capacity, "spilled" passengers might be recaptured on other alternative fare products of the airline or lost to competitors. Similarly, in schedule design, when a certain itinerary is not available because a flight is deleted from the schedule, demand of fare products on substitutable itineraries increases. Such spill and recapture behavior adds extra complexity to revenue modeling in SD-FAM. In fleet assignment, Barnhart et al. (2002) used spill variables and recapture rates defined for pairs of substitutable itineraries to model the substitution behavior when demand exceeds capacity. The recapture rate is estimated using the Quality of Service Index (QSI). In schedule design, Lohatepanont and Barnhart (2004) further introduced demand correction terms to compensate for changes in demand when changes are made to the schedule.

A more recent advancement in modeling passenger spill and recapture behavior has occured in choice modeling and its application to revenue management. Discrete choice models capture such behavior by assuming the demand to be tied to markets and not to individual fare products. Passengers then choose to buy fare products belonging to the same market according to certain probabilities that depend on which fare products are offered. Talluri and van Ryzin (2004) first utilized discrete choice models to describe passenger fare product substitution behavior and provided structural results related to the single-leg revenue management problem. Later, Gallego et al. (2004), Liu and van Ryzin (2008) and Bront et al. (2009) extended these results to an NRM setting, using a choice-based deterministic linear programming (CDLP) model to produce an upper bound on the optimal value and as an effective heuristic for the stochastic control problem.

There are several advantages to using a choice-based NRM approach to model spill and recapture behavior and to evaluate revenue in SD-FAM. First, sophisticated choice models can be estimated systematically and rigorously using the wealth of passenger transaction/shopping data and fare product data (Vulcano et al., 2012; van Ryzin and Vulcano, 2014) available today. Second, choice models can captures changes in passenger preferences as fare product availability changes given that purchase probability depends on the offer set of fare products. Third, modeling revenue using an NRM approach is consistent with airline practice. Choice-based NRM with its promising revenue improvements (confirmed by empirical studies (Vulcano et al., 2010; Dai et al., 2015)) represents an ongoing shift in the industry. Finally, recent advances in choice-based NRM have lead to the development of sales-based equivalent reformulations of CDLP under some widely-used choice models, such as multinomial logit (MNL) (Gallego et al., 2014). These reformulations are compact linear programs, which have the potential to provide computational advantages over existing approaches.

This chapter studies an integration of SD-FAM with choice-based NRM where optional flight selection and fleet assignment decisions are made jointly. We denote the model as CSD-FAM, and focus on the case where demand is modeled by an MNL choice model. Our work is not the first to study such an integrated model. Wang et al. (2014) first proposed a series of airline planning models, including fleet assignment and schedule design, that used choice-based NRM as their revenue evaluation mechanism. Similar to our research, they studied the case where passenger demand follows MNL choice models and leveraged a compact sales-based linear program proposed by Gallego et al. (2014). They benchmarked their integrated FAM against one whose revenue is evaluated using the deterministic linear program (DLP) of the independent demand NRM (Talluri and van Ryzin, 2006), and found significant profit improvement. However, along with benefits, they encountered formidable computational challenges that prevented this approach from being applied to practical-sized instances and from extending it to schedule design decisions. Atasoy et al. (2014) studied an integrated problem involving schedule design, fleet assignment, and pricing. They too considered an MNL model where the price decision is factored into the choice probability. Unfortunately, the resultant formulation is a mixed-integer non-convex program for which only toy instances can be solved.

Faced with this computational challenge, the major contribution of this research is enhancing the computational performance of CSD-FAM. To achieve that, we adapt the existing FAM approximation and reformulation developed by Barnhart et al. (2009) where flights are partitioned into partly separable subnetworks and composite variables are utilized to model fleet assignment decisions. The model optimizes a profit function which is an upper bound of the true profit. This subnetwork-based formulation enjoys computational advantages because of its tighter LP relaxation bound. However, spill and recapture behavior is ignored in their model and adding such consideration leads to stronger network dependencies between flights. Note that flights can be dependent not only because they are providing capacity for the same fare product, but also because the products for which they provide capacity are jointly considered by passengers from a specific market. Such additional dependencies consequently lead to looser profit bounds and inferior solutions. In order to extend this framework for CSD-FAM and enjoy its computational advantages while still obtaining good solutions, we propose a subnetwork-based *mixed* formulation where both composite variables and traditional flight-fleet assignment variables co-exist in the model. We further develop a model to optimize fare proration over flight legs to tighten the profit bound. These modeling enhancements are shown to greatly improve the performance of the subnetwork-based formulation. We summarize our contributions as follows:

1. We provide a tractable reformulation and reliable approximation of the choice-based integrated fleet assignment and schedule design problem that enables it

to be solved for full-scale airline instances, and in doing so, achieves significant profit improvements.

2. Within this reformulation and approximation framework, we develop an innovative fare-split approach to allocate fare over flight legs and achieve significant profit improvements over commonly used heuristics, such as distance-based proration.

Note that aside from the SD-FAM literature, there are also some studies which specifically focus on passenger flow models for airline networks (Soumis and Nagurney, 1993; Dumas and Soumis, 2008). These papers aim to assign passengers accurately to itineraries, given seating capacities, using certain equilibrium models. These methods consider stochastic demand and also passenger spill-and-recapture behavior by assuming recapture rates. However, the incorporation into SD-FAM is not practical because these models are described as a system of nonlinear equations and solved using iterative fixed point algorithms. Moreover, these approaches typically do not consider the impact of NRM on passenger flows in cases where revenue management policies lead to fare products be closed for sale.

The remainder of this chapter is organized as follows. In Section 5.2, we introduce the initial formulation of CSD-FAM and then our proposed subnetwork-based mixed formulation. In Section 5.3, we discuss the solution approach for our formulation, which includes a variable reduction technique (Section 5.3.1), a partition algorithm to create subnetworks (Section 5.3.2), and an optimization problem to optimize fare split over flight legs (Section 5.3.3). In Section 5.4, we describe computational experiments using a full-scale instance from a major US airline to demonstrate the performance of our proposed approach.

## 5.2 Choice-Based Fleet Assignment and Schedule Design under MNL

In this section, we introduce the model of CSD-FAM with MNL choice. We start with an initial version of the model using classic flight-fleet variables, first introduced by Wang et al. (2014), in Section 5.2.1. We then propose a subnetwork-based reformulation adapted from Barnhart et al. (2009) in Section 5.2.2. Different from Barnhart et al. (2009), we allow both types of variable definitions to co-exist in the model: subnetwork-based composite variables and traditional flight-fleet variables.

### 5.2.1 Initial Formulation

Denote $\mathcal{L}$ as the set of all flight legs. It is the union of two non-overlapping sets: $\mathcal{L}_m$ is the set of mandatory flights that must be flown, and $\mathcal{L}_o$ is the set of optional flights that may be flown depending on profitability and operational considerations. $\mathcal{F}$ is the set of all fleet types. $\mathcal{F}(l)$ is the set of compatible fleet types that can be used to operate flight leg $l$. Following the classic model setup in Hane et al. (1995), for each fleet type $f \in \mathcal{F}$, we construct a time-space network containing flights that can be operated by fleet type $f$. Figure 5-1 shows an example time-space network with two airports: LaGuardia (LGA) and Boston (BOS). The horizontal axis represents time, and the vertical axis represents different airports. Let $N_f$ be the set of all nodes in the time-space network of fleet type $f$. Each node $v \in N_f$ corresponds to either an arrival (plus minimum turn time to ensure feasible connection) or a departure of a flight. Note that we only include flights that can be flown by fleet type $f$. There are three types of directed arcs in the network: (1) mandatory flight arcs, represented by solid arrows linking an arrival node and a departure node in two different airports; (2) optional flight arcs, represented by dashed-dotted arrows linking an arrival node and a departure node in two different airports; (3) ground arcs, represented by dashed arrows linking two consecutive nodes at the same airport. Wrap-around arcs are a special type of ground arc that connect the last node to the first node at a specific airport to ensure that the fleet assignment can be operated on a repeated schedule. Note that we only consider daily schedules in this chapter.

Let $v^-$ denote the predecessor node located at the tail of the ground arc entering node $v$. Let $I(v)$ denote the set of incoming flight legs (flight arcs) into node $v$ and $O(v)$ denote the set of outgoing flight legs (flight arcs) from node $v$. We define a count time (dashed vertical line in Figure 5-1) to count the total number of aircraft both in the air and on the ground at that point in time. This point in time can be chosen arbitrarily because conservation of flow throughout the time-space network ensures that the count is the same at each point in time. We define $T_F(f)$ as the set of flight arcs (including both mandatory and optional) in the time-space network $f$ that traverse the count time. By construction, for each node $v \in N_f$ in the time-space network, there is only one incoming ground arc as well as one outgoing ground arc. Thus, any ground arc can be uniquely identified by the node it originates from. We define $T_N(f)$ as the set of all nodes in time-space network of fleet type $f$ whose outgoing ground arc traverses the count time. Let $n_f$ be the number of available aircraft for fleet type $f$. We define binary decision variables $x_{l,f} = 1, \forall l \in \mathcal{L}, \forall f \in \mathcal{F}$ if flight leg $l$ is assigned fleet type $f$ and $x_{l,f} = 0$ otherwise. Let $y_v, \forall v \in N_f, \forall f \in \mathcal{F}$ be the continuous variables that represent the number of aircraft on the ground arc originating from node $v$. Note that because $x_{l,f}$ are binary variables, $y_v$ will always take on integer values.

Let $\mathcal{P}$ be the set of fare products the airline offers. Each fare product is a unique combination of an itinerary and a fare class. We partition passengers into a set of market segments $\mathcal{M}$. Passengers in a particular market segment $m \in \mathcal{M}$ are *only* interested in a specific subset of fare products $\mathcal{P}(m) \subset \mathcal{P}$. We call $\mathcal{P}(m)$ the consideration set for market segment $m$. A market segment can be a specific origin-destination pair, e.g., all flights from Boston to New York City, or it could also be more restrictive, e.g., morning direct flights from New York City to San Francisco that are less than \$600. However, in order to model demand as an MNL choice model, we assume that all consideration sets $\mathcal{P}(m), m \in \mathcal{M}$ are disjoint, i.e. $\mathcal{P}(m_1) \cap \mathcal{P}(m_2) = \emptyset$, $\forall m_1, m_2 \in \mathcal{M}, m_1 \neq m_2$. Overlapping consideration sets make the demand process

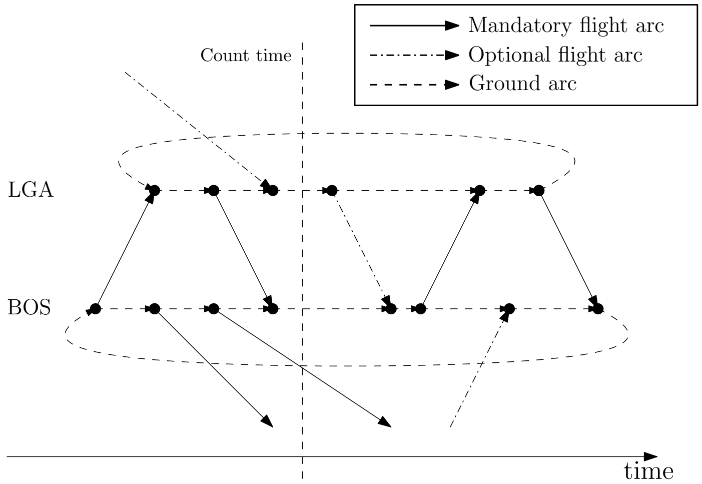


Figure 5-1: A time-space network with two airports for a specific fleet type.

become a mixed MNL model, which is out of the scope of this paper. For each product $p$ in consideration set $\mathcal{P}(m)$, let $v_p$ denote its attractiveness value. For each market segment $m$, we also define $v_m$ as the aggregated attractiveness value of competing airlines' products as well as that of the no-fly alternative in the same market segment. If $\bar{\mathcal{P}}$ is the set of available products, i.e. products that are not closed for sale because of the revenue management system or insufficient capacity, then a passenger from market $m$ will choose fare product $i$ with probability $v_i/(\sum_{p\in\bar{\mathcal{P}}\cap\mathcal{P}(m)} v_p + v_m)$ if $i \in \mathcal{P}(m)$ and zero otherwise, which is consistent with an MNL choice model. Let $\Lambda_m$ be the total demand for market segment $m$. Let $h_p$ be the price of fare product $p$ and $\mathbf{h} = \{h_p\}_{p\in\mathcal{P}}$ be the fare vector. Define indicator variable $\delta_{l,p} = 1$ if fare product $p$ uses flight leg $l$ and zero otherwise. We then define continuous decision variables $s_p, p \in \mathcal{P}$ as the number of units sold of product $p$, and $s_m, m \in \mathcal{M}$ as the number of passengers choosing either the competing airlines' products or the no-fly alternative in market segment $m$.

With these notations, we present the CSD-FAM formulation as follows.

(CSD-FAM)

$$\max \qquad \sum_{p\in\mathcal{P}} h_p s_p - \sum_{l\in\mathcal{L}} \sum_{f\in\mathcal{F}(l)} c_{l,f} x_{l,f} \tag{5.1}$$

$$\text{s.t.} \qquad \sum_{f\in\mathcal{F}(l)} x_{l,f} = 1, \quad \forall l \in \mathcal{L}_m \tag{5.2}$$

$$\sum_{f\in\mathcal{F}(l)} x_{l,f} \leq 1, \quad \forall l \in \mathcal{L}_o \tag{5.3}$$

$$y_{v^-} + \sum_{l\in I(v)} x_{l,f} - y_v - \sum_{l\in O(v)} x_{l,f} = 0, \quad \forall v \in N_f,\ \forall f \in \mathcal{F} \tag{5.4}$$

$$\sum_{v\in T_N(f)} y_v + \sum_{l\in T_F(f)} x_{l,f} \leq n_f, \quad \forall f \in \mathcal{F} \tag{5.5}$$

$$\sum_{p\in\mathcal{P}} \delta_{l,p} s_p \leq \sum_{f\in\mathcal{F}(l)} \mathrm{CAP}_f x_{l,f},\ \forall l \in \mathcal{L} \tag{5.6}$$

$$\sum_{p\in\mathcal{P}(m)} s_p + s_m = \Lambda_m,\ \forall m \in \mathcal{M} \tag{5.7}$$

$$\frac{s_p}{v_p} - \frac{s_m}{v_m} \leq 0,\ \forall p \in \mathcal{P}(m),\ \forall m \in \mathcal{M} \tag{5.8}$$

$$s_m \geq 0, \quad \forall m \in \mathcal{M} \tag{5.9}$$

$$s_p \geq 0, \quad \forall p \in \mathcal{P} \tag{5.10}$$

$$x_{l,f} \in \{0,1\}, \quad \forall f \in \mathcal{F}(l),\ \forall l \in \mathcal{L} \tag{5.11}$$

$$y_v \geq 0, \quad \forall v \in N_f,\ \forall f \in \mathcal{F} \tag{5.12}$$

Objective function (5.1) maximizes the total profit contribution, comprised of the revenue from selling fare products minus the total operating cost, where $c_{l,f}$ is defined as the operating cost of assigning fleet type $f$ to flight $l$. Constraints (5.2) ensure that each mandatory flight is covered by exactly one fleet type. In contrast, constraints (5.3) allow optional flights to remain uncovered, i.e., be removed from the schedule. Constraints (5.4) are the flow balance constraints ensuring that for a certain type of fleet, the number of aircraft into a node equals the number going out. Constraints (5.5) ensure that for each fleet type $f$, the total number of aircraft being used (either in the air or on the ground) is less than or equal to the number of aircraft available. Note that the integrality of $y_v$ variables can be relaxed because integrality is guaranteed by the balance constraints (5.4) and the binary $x_{l,f}$ variables.

Constraints (5.6) - (5.10) are the sales-based linear programming constraints developed by Gallego et al. (2014), representing the fluid approximation of the NRM problem under an MNL choice model. The optimal values are known to provide a tight upper bound on the actual stochastic control problem (Kunnumkal and Talluri, 2016). Constraints (5.6) are the capacity constraints which state that the aggregated ticket sales on a flight leg should not exceed the capacity of its assigned fleet type, where $\mathrm{CAP}_f$ is defined as the seating capacity of fleet type $f$. Constraints (5.7) state that the total sales in market $m$ plus the number of passengers choosing either a product of a competing airline or the no-fly alternative in the same market should equal the total market demand $\Lambda_m$. Finally, constraints (5.8) are scaling constraints to ensure that the sale of each product $p$, that belongs to market $m$, is bounded from above by the demand $((v_p/v_m)s_m)$ for that product, which depends on the attractiveness value of that fare product in a fashion that is consistent with the MNL choice model. The equality corresponds to the case where product $p$ is always available to be booked during the selling horizon. If product $p$ is ever closed for sale due to capacity constraints or revenue management policy, then the strict inequality $s_p < (v_p/v_m)s_m$ may hold. Note that the competing airlines' products are assumed always to be available.

Let us note a couple of important details related to this formulation. First, unlike in Lohatepanont and Barnhart (2004) where two separate modeling concepts – recapture rates and demand correction terms – are used in CSD-FAM, the unified choice-based framework is used to model spill and recapture. This is achieved by tying demand with markets instead of products, and then allocating market demand to products according to their availability and attractiveness values. Second, the compactness of constraints (5.6) - (5.10) is an important feature of MNL choice models. In general, CDLP for choice-based NRM needs $2^{|\mathcal{P}|}$ variables, where each variable corresponds to a particular assortment of products. Because the number of variables is exponential in the number of products $|\mathcal{P}|$, column generation is required.

### 5.2.2 Subnetwork-Based Mixed Formulation

Although the CSD-FAM formulation is compact, solving it presents a formidable computational challenge. Wang et al. (2014) and Di et al. (2016) both conduct computational experiments on the pure FAM version of the problem, i.e., the one without optional flight legs ($\mathcal{L}_m = \mathcal{L}$), and report substantial difficulties in solving it, and this issue is further exacerbated with the incorporation of schedule design decisions.

In order to enhance the tractability of the CSD-FAM formulation, in this section, we adapt an existing work by Barnhart et al. (2009) to propose a subnetwork-based mixed formulation, denoted as S-CSD-FAM. The key idea of Barnhart et al. (2009) is to isolate network effects by partitioning flights into many small subnetworks. As mentioned earlier in Section 5.1, our proposed mixed formulation is a combination of both subnetwork-based variables and traditional flight-fleet assignment variables.

We represent the revenue evaluation elements in the CSD-FAM formulation as a function $r(\mathbf{x};\mathbf{h})$, which calculates the optimal revenue given a fleet assignment decision $\mathbf{x}$ and a product price vector $\mathbf{h}$, as follows:

$$
\begin{aligned}
r(\mathbf{x};\mathbf{h}) = \max_{\mathbf{s}} \quad & \sum_{p\in\mathcal{P}} h_p s_p && (5.13)\\
\text{s.t.} \quad & \sum_{p\in\mathcal{P}} \delta_{l,p} s_p \leq \sum_{f\in\mathcal{F}(l)} \mathrm{CAP}_f x_{l,f},\ \forall l \in \mathcal{L} && (5.14)\\
& \sum_{p\in\mathcal{P}(m)} s_p + s_m = \Lambda_m,\ \forall m \in \mathcal{M} && (5.15)\\
& \frac{s_p}{v_p} - \frac{s_m}{v_m} \leq 0,\ \forall p \in \mathcal{P}(m),\ \forall m \in \mathcal{M} && (5.16)\\
& s_m \geq 0, \quad \forall m \in \mathcal{M} && (5.17)\\
& s_p \geq 0, \quad \forall p \in \mathcal{P} && (5.18)
\end{aligned}
$$

The essence of S-CSD-FAM is to decompose and approximate the value of $r(\mathbf{x};\mathbf{h})$ using revenue calculations for many small subnetworks. We start by partitioning the flight legs $\mathcal{L}$ into $K$ subnetworks: $\mathcal{L}^1, \cdots, \mathcal{L}^K$ such that $\cup_{k=1}^K \mathcal{L}^k = \mathcal{L}$ and $\mathcal{L}^i \cap \mathcal{L}^j =$

$\emptyset,\ i \neq j$. We denote any such partition as $\Pi = \{\mathcal{L}^1, \cdots, \mathcal{L}^K\}$. For each $\mathcal{L}^k$, we denote the local product set $\mathcal{P}^k = \{p \in \mathcal{P} : \exists l \in \mathcal{L}^k \text{ s.t. } \delta_{l,p} = 1\}$, as the set of products that use at least one flight leg in $\mathcal{L}^k$. Similarly, we define the local market set $\mathcal{M}^k = \{m \in \mathcal{M} : \mathcal{P}^k \cap \mathcal{P}(m) \neq \emptyset\}$, as the set of markets that include at least one product in the set $\mathcal{P}^k$. We further define the local fare vectors $\left[\mathbf{h}^1, \cdots, \mathbf{h}^K\right]$ for subnetworks $\mathcal{L}^1, \cdots, \mathcal{L}^K$ where each $\mathbf{h}^k \in \mathbb{R}_+^{|\mathcal{P}|}$, and the local fleet assignment $\mathbf{x}^k$ for each subnetwork $\mathcal{L}^k$ as per the following two definitions.

**Definition 5.1** (Local Fare Vectors)**.** *Fare vectors* $\left[\mathbf{h}^1, \cdots, \mathbf{h}^K\right]$ *for subnetworks* $\mathcal{L}^1, \cdots, \mathcal{L}^K$ *are called local fare vectors if they satisfy the following conditions:*

1. $\sum_{k=1}^K \mathbf{h}^k = \mathbf{h}$ *and* $\mathbf{h}^k \geq \mathbf{0}, \forall k \in \{1, \cdots, K\}$.

2. $h_p^k = 0$ *if* $p \notin \mathcal{P}^k, \forall k \in \{1, \cdots, K\}$.

In other words, local fare vectors are nonnegative vectors that sum up to the original fare vector $\mathbf{h}$. For products that don't belong to the local product set, their local fares are set to zero.

**Definition 5.2** (Local Fleet Assignment)**.** *Given a fleet assignment* $\mathbf{x}$*, the local fleet assignment* $\mathbf{x}^k$ *for subnetwork* $\mathcal{L}^k$ *is defined as:*

$$x_{l,f}^k = \begin{cases} x_{l,f}, & \forall l \in \mathcal{L}^k, f \in \mathcal{F}(l) \\ 1, & \forall l \in \mathcal{L} \setminus \mathcal{L}^k, f = \arg\max_{f' \in \mathcal{F}(l)} CAP_{f'} \\ 0, & \textit{otherwise} \end{cases}$$

$\mathbf{x}^k$ matches fleet assignment $\mathbf{x}$ for flights in $\mathcal{L}^k$, and assigns the compatible (that is, fleet type able to operate flight leg $l$) fleet type with the largest capacity for flights not in $\mathcal{L}^k$.

Denote the extended subnetwork $\mathcal{L}_+^k = \{l \in \mathcal{L} : \exists p \in \mathcal{P}^k \text{ s.t. } \delta_{l,p} = 1\}$, the set of all flights covered by products in local product set $\mathcal{P}^k$. Please note that $\mathcal{L}^k \subseteq \mathcal{L}_+^k$ because local product set $\mathcal{P}^k$ can possibly cover flights not in $\mathcal{L}^k$. With Definitions

5.1 and 5.2, consider the local revenue term $r^k(\mathbf{x}^k, \mathbf{h}^k)$ as follows.

$$
\begin{aligned}
r^k(\mathbf{x}^k; \mathbf{h}^k) = \max_{\mathbf{s}} \quad & \sum_{p \in \mathcal{P}^k} h_p^k s_p && (5.19)\\
\text{s.t.} \quad & \sum_{p \in \mathcal{P}^k} \delta_{l,p} s_p \leq \sum_{f \in \mathcal{F}(l)} \mathrm{CAP}_f x_{l,f}^k, \quad \forall l \in \mathcal{L}_+^k && (5.20)\\
& \sum_{p \in \mathcal{P}(m) \cap \mathcal{P}^k} s_p + s_m \leq \Lambda_m, \quad \forall m \in \mathcal{M}^k && (5.21)\\
& \frac{s_p}{v_p} - \frac{s_m}{v_m} \leq 0, \quad \forall p \in \mathcal{P}(m) \cap \mathcal{P}^k, \ \forall m \in \mathcal{M}^k && (5.22)\\
& s_m \geq 0, \quad \forall m \in \mathcal{M}^k && (5.23)\\
& s_p \geq 0, \quad \forall p \in \mathcal{P}^k && (5.24)
\end{aligned}
$$

We now state Lemma 5.1 below that the sum of the local revenue terms of $K$ subnetworks is an upper bound on the overall true optimal revenue, for any fleet assignment $x$, for any overall fare vector $h$, and for any local fare vectors $\left[\mathbf{h}^1, \cdots, \mathbf{h}^K\right]$.

**Lemma 5.1.** *For any fleet assignment* $\mathbf{x}$ *and its corresponding local fleet assignment* $\mathbf{x}^k$*, and for any local fare vector* $\mathbf{h}^k$*, the following holds:*

$$\sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k) \geq r(\mathbf{x}; \mathbf{h}).$$

*Proof.* For any fleet assignment $\mathbf{x}$, we denote $\mathbf{s}^*$ as the optimal solution of problem $r(\mathbf{x}; \mathbf{h})$. $\mathbf{s}^*$ is also a feasible solution for problems $r^1(\mathbf{x}^1; \mathbf{h}^1), \cdots, r^K(\mathbf{x}^K; \mathbf{h}^K)$. To see that, we first confirm that $\mathbf{s}^*$ satisfies constraints (5.22) - (5.24) because these constraints are all present in the formulation of $r(\mathbf{x}; \mathbf{h})$. From Definition 5.2, we have $\sum_{l \in \mathcal{F}(l)} \mathrm{CAP}_f x_{l,f}^k \geq \sum_{l \in \mathcal{F}(l)} \mathrm{CAP}_f x_{l,f}$. Therefore $\mathbf{s}^*$ also satisfies constraints (5.20). For constraints (5.21), we have

$$\sum_{p \in \mathcal{P}(m) \cap \mathcal{P}^k} s_p^* + s_m^* \leq \sum_{p \in P(m)} s_p^* + s_m^* = \Lambda_m, \ \forall m \in \mathcal{M}^k, \forall k \in \{1, \cdots, K\}$$

The last equality is because $s^*$ is a feasible solution of $r(\mathbf{x};\mathbf{h})$ and satisfies constraints (5.15).

We thus have

$$\begin{aligned}
\sum_{k=1}^{K} r^k(\mathbf{x}^k;\mathbf{h}^k) &\geq \sum_{k=1}^{K}\left(\sum_{p\in\mathcal{P}^k} h_p^k s_p^*\right)\\
&= \sum_{k=1}^{K}\left(\sum_{p\in\mathcal{P}} h_p^k s_p^*\right)\\
&= \sum_{p\in\mathcal{P}}\left(\sum_{k=1}^{K} h_p^k\right) s_p^*\\
&= \sum_{p\in\mathcal{P}} h_p s_p^*\\
&= r(\mathbf{x};\mathbf{h})
\end{aligned}$$

The first inequality holds because of the feasibility of $\mathbf{s}^*$ for each of these $K$ problems. The first equality holds because $h_p^k = 0\ \forall p \in P \setminus P^k$. The second equality is a re-arrangement of terms. The third equality holds because of Definitions 5.1. The last equality is due to the fact that $s^*$ is the optimal solution of problem $r(\mathbf{x};\mathbf{h})$. □

With this decomposition of the revenue calculation, we introduce formulation CSD-FAM(Π), as a function of partition Π, as follows,

(CSD-FAM(Π))

$$\max \quad \sum_{k=1}^{K} r^k(\mathbf{x}^k;\mathbf{h}^k) - \sum_{l\in\mathcal{L}}\sum_{f\in\mathcal{F}(l)} c_{l,f}x_{l,f} \tag{5.25}$$

$$\text{s.t.} \quad \sum_{f\in\mathcal{F}(l)} x_{l,f} = 1, \quad \forall l \in \mathcal{L}_m \tag{5.26}$$

$$\sum_{f\in\mathcal{F}(l)} x_{l,f} \leq 1, \quad \forall l \in \mathcal{L}_o \tag{5.27}$$

$$y_{v^-} + \sum_{l\in I(v)} x_{l,f} - y_v - \sum_{l\in O(v)} x_{l,f} = 0, \quad \forall v \in N_f,\ \forall f \in \mathcal{F} \tag{5.28}$$

$$\sum_{v\in T_N(f)} y_v + \sum_{l\in T_F(f)} x_{l,f} \leq n_f, \quad \forall f \in \mathcal{F} \tag{5.29}$$

$$x_{l,f} \in \{0,1\}, \quad \forall f \in \mathcal{F}(l),\ \forall l \in \mathcal{L} \tag{5.30}$$

$$y_v \geq 0, \quad \forall v \in N_f,\ \forall f \in \mathcal{F} \tag{5.31}$$

Note that the only difference between CSD-FAM and CSD-FAM($\Pi$) is in the objective functions (5.1) and (5.25). CSD-FAM($\Pi$) uses an upper-bound approximation of the true revenue function, as presented in Lemma 5.1. This approximation allows the calculation of the local revenue term to depend only on the fleet assignments for flights within each subnetwork. This motivates a reformulation of CSD-FAM($\Pi$) using composite variables.

For each subnetwork $\mathcal{L}^k$ in $\Pi = \{\mathcal{L}^1, \cdots, \mathcal{L}^K\}$, we define $\mathcal{X}^k$ as the set of all possible local fleet assignments for flights in $\mathcal{L}^k$. As an illustratation, suppose $\mathcal{L}^k = \{l_1, l_2\}$ and $\mathcal{F}(l_1) = \mathcal{F}(l_2) = \{f_1, f_2\}$. In words, consider a subnetwork with only two flights and let each flight have the same set of two compatible fleet types. Moreover, $l_1 \in \mathcal{L}_o$ and $l_2 \in \mathcal{L}_m$. Then $\mathcal{X}^k = \{(f_1, f_1), (f_1, f_2), (f_2, f_1), (f_2, f_2), (\emptyset, f_1), (\emptyset, f_2)\}$, where $\emptyset$ denotes assigning no fleet type to a specific flight, i.e., removing the flight from the schedule. For each local composite fleet assignment $j \in \mathcal{X}^k$, a binary decision variable $w_j$ is defined such that it takes value 1 if the local composite fleet assignment $j$ is adopted for all flights in $\mathcal{L}^k$, and is zero otherwise. These variables are called composite because a value of 1 for a particular variable indicates assignment decisions for multiple flights.

Denote $\tau^j_{l,f} = 1$ if local composite fleet assignment $w_j$ assigns fleet type $f$ to flight leg $l$, and $\tau^j_{l,f} = 0$ otherwise. Let $\text{in}^j_v = \sum_{l \in I(v)} \tau^j_{l,f}$ denote the number of incoming flight arcs to node $v$ ($v \in N_f$) specified by local composite fleet assignment $j$. Similarly, let $\text{out}^j_v = \sum_{l \in O(v)} \tau^j_{l,f}$ be the number of outgoing flight arcs from node $v$ specified by $j$. Also let $\text{count}^j_f = \sum_{l \in T_F(f)} \tau^j_{l,f}$ be the number of flight legs (flight arcs) that traverse the count time and are covered by fleet type $f$, as specified by $j$. $c_j = \sum_{l \in \mathcal{L}} \sum_{f \in \mathcal{F}(l)} c_{l,f} \tau^j_{l,f}$ is the operating cost associated with local composite fleet assignment $j$. For each subnetwork $k$ and for each local composite fleet assignment $j \in \mathcal{X}^k$, we can compute the local revenue term $r(\mathbf{x}^k; \mathbf{h}^k)$ as $r_j$.

we now present formulation S-CSD-FAM($\Pi$) below.

(S-CSD-FAM(Π))

$$\max \quad \sum_{k=1}^{K} \sum_{j \in \mathcal{X}^k} (r_j - c_j) w_j \tag{5.32}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{X}^k} w_j = 1, \ \forall k = \{1, \cdots, K\} \tag{5.33}$$

$$y_{v^-} + \sum_{k=1}^{K} \sum_{j \in \mathcal{X}^k} \text{in}_v^j w_j - y_v - \sum_{k=1}^{K} \sum_{j \in \mathcal{X}^k} \text{out}_v^j w_j = 0, \quad \forall v \in N_f, \ \forall f \in \mathcal{F} \tag{5.34}$$

$$\sum_{v \in T_N(f)} y_v + \sum_{k=1}^{K} \sum_{j \in \mathcal{X}^k} \text{count}_f^j w_j \leq n_f, \quad \forall f \in \mathcal{F} \tag{5.35}$$

$$w_j \in \{0, 1\}, \ \forall j \in \mathcal{X}^k, \ \forall k \in \{1, \cdots, K\} \tag{5.36}$$

$$y_v \geq 0, \quad \forall v \in N_f, \ \forall f \in \mathcal{F} \tag{5.37}$$

Note that we calculate $r_j$ offline for each $j \in \mathcal{X}^k, \ k \in \{1, \cdots, K\}$ by solving (5.19) - (5.24) for each $j \in \mathcal{X}^k$. S-CSD-FAM(Π), with its change of variable, is equivalent to CSD-FAM(Π). Constraints (5.33) are the reformulated cover constraints requiring that for each subnetwork $\mathcal{L}^k$, exactly one local composite fleet assignment is selected. In contrast to constraints (5.26) and (5.27) where the cover constraints for mandatory and optional flights are modeled separately, constraints (5.33) model the cover requirements with a "no assignment option" for optional flights in the definition of the composite variables. Constraints (5.34) and (5.35) are the reformulated flow balance and fleet availability constraints, respectively.

Barnhart et al. (2009) study a version of S-CSD-FAM(Π) with only fleet assignment decisions and find that it enjoys a tighter LP relaxation bound, giving the model certain computational benefits. The tightness of this formulation is preserved as we incorporate schedule design decisions. However, the cardinality of $\mathcal{X}^k$ grows exponentially with the size of subnetwork $\mathcal{L}^k$. Therefore, in order to maintain tractability, subnetwork sizes need to be limited. For example, in Barnhart et al. (2009), the size is set to be no more than five. Unfortunately, we find in our preliminary experiments that, under choice-based passenger demand, using such small subnetwork sizes leads to poor solution quality due to the large revenue differences between the approximate objective function (5.32) and the true objective function (5.1). Such large revenue differences are a direct consequence of the stronger network dependencies introduced by passenger spill and recapture. Increasing subnetwork size improves approximation accuracy, but reduces the tractability of the resultant S-CSD-FAM($\Pi$). To obtain the best tradeoff between the two competing features, namely the compactness of the traditional flight-fleet variable representation; and the tightness of the subnetwork-based composite variable representation, we develop a subnetwork-based *mixed* formulation.

We split the subnetwork partition into separate parts $\Pi_c$ and $\Pi_t$. $\Pi_c$ and $\Pi_t$ are mututally exclusive and collectively exhaustive sets of subnetworks in set $\Pi$, where subnetworks in $\Pi_c$ are modeled using composite variables and those in $\Pi_t$ are modeled using traditional flight-fleet assignment variables. Let $\mathcal{L}(\Pi_c) = \cup_{\mathcal{L}' \in \Pi_c} \mathcal{L}'$ be the set of flights represented using the composite variable representation, and similarly $\mathcal{L}(\Pi_t) = \cup_{\mathcal{L}' \in \Pi_t} \mathcal{L}'$ be those represented using flight-fleet assignment variables. The motivation is to use composite variables to represent subnetworks of smaller sizes and to use flight-fleet assignment variable to represent subnetworks of larger sizes. We present our subnetwork-based mixed formulation, denoted as S-CSD-FAM($\Pi_c$,$\Pi_t$) as follows:

(S-CSD-FAM($\Pi_c, \Pi_t$))

$$\max \sum_{\substack{k \in \{1,\cdots,K\}: \\ \mathcal{L}^k \in \Pi_c}} \sum_{j \in \mathcal{X}^k} (r_j - c_j) w_j + \sum_{\substack{k \in \{1,\cdots,K\}: \\ \mathcal{L}^k \in \Pi_t}} r^k(\mathbf{x}^k; \mathbf{h}^k) - \sum_{\substack{k \in \{1,\cdots,K\}: \\ \mathcal{L}^k \in \Pi_t}} \sum_{l \in \mathcal{L}^k} \sum_{f \in \mathcal{F}(l)} c_{l,f} x_{l,f} \tag{5.38}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{X}^k} w_j = 1, \ \forall k \in \{1, ..., K\} : \mathcal{L}^k \in \Pi_c \tag{5.39}$$

$$\sum_{f \in \mathcal{F}(l)} x_{l,f} = 1, \ \forall l \in \mathcal{L}_m \cap \mathcal{L}(\Pi_t) \tag{5.40}$$

$$\sum_{f \in \mathcal{F}(l)} x_{l,f} \leq 1, \ \forall l \in \mathcal{L}_o \cap \mathcal{L}(\Pi_t) \tag{5.41}$$

$$y_{v^-} + \sum_{\substack{k \in \{1,\cdots,K\}: \\ \mathcal{L}^k \in \Pi_c}} \sum_{j \in \mathcal{X}^k} \text{in}_v^j w_j + \sum_{l \in I(v) \cap \mathcal{L}(\Pi_t)} x_{l,f}$$

$$- y_v - \sum_{\substack{k \in \{1,\cdots,K\}: \\ \mathcal{L}^k \in \Pi_c}} \sum_{j \in \mathcal{X}^k} \text{out}_v^j w_j - \sum_{l \in O(v) \cap \mathcal{L}(\Pi_t)} x_{l,f} = 0, \quad \forall v \in N_f, \ \forall f \in \mathcal{F} \tag{5.42}$$

$$\sum_{v \in T_N(f)} y_v + \sum_{\substack{k \in \{1,\cdots,K\}: \\ \mathcal{L}^k \in \Pi_c}} \sum_{j \in \mathcal{X}^k} \text{count}_f^j w_j + \sum_{l \in T_F(f) \cap \mathcal{L}(\Pi_t)} x_{l,f} \leq n_f, \quad \forall f \in \mathcal{F} \tag{5.43}$$

$$w_j \in \{0,1\}, \ \forall j \in \mathcal{X}^k, \ \forall k \in \{1,...,K\} : \mathcal{L}^k \in \Pi_c \tag{5.44}$$

$$x_{l,f} \in \{0,1\}, \ \forall f \in \mathcal{F}(l), \ \forall l \in \mathcal{L}(\Pi_t) \tag{5.45}$$

$$y_v \geq 0, \quad \forall v \in N_f, \ \forall f \in \mathcal{F} \tag{5.46}$$

Objective function (5.38) is the sum of revenue minus operating costs from flights in $\Pi_c$ and $\Pi_t$. Constraints (5.39) are the cover constraints for flights in $\Pi_c$ and constraints (5.40) and (5.41) are the cover constraints for flights in $\Pi_t$. Constraints (5.42) and (5.43) are flow balance and aircraft availability constraints, each containing both types of the variable representations.

## 5.3 Solution Approach of S-CSD-FAM$(\Pi_c, \Pi_t)$

In this section, we provide the details of the entire solution approach of S-CSD-FAM$(\Pi_c, \Pi_t)$. In Section 5.3.1, we start with a simplification of the S-CSD-FAM$(\Pi_c, \Pi_t)$ formulation using the idea of parsimonious enumeration introduced by Barnhart et al. (2009). We then discuss our partition algorithm to get $\Pi_c$ and $\Pi_t$ in Section 5.3.2. Finally, we introduce the fare splitting problem to optimize the local fare vectors $\mathbf{h}^k$ in Section 5.3.3.

### 5.3.1 Parsimonious Enumeration

The cardinality of $\mathcal{X}^k$ for any $\mathcal{L}^k \in \Pi_c$ equals $\left(\prod_{l \in \mathcal{L}^k \cap \mathcal{L}_o} |\mathcal{F}(l)| + 1\right) \left(\prod_{l \in \mathcal{L}^k \cap \mathcal{L}_m} |\mathcal{F}(l)|\right)$ in formulation S-CSD-FAM$(\Pi_c, \Pi_t)$. Barnhart et al. (2009) introduce a parsimonious enumeration to significantly reduce the number of variables associated with each subnetwork. We adopt this method (with a small modification) in S-CSD-FAM$(\Pi_c, \Pi_t)$

as follows.

We define $\overline{\lambda}_l$ as the maximum possible demand for flight leg $l$, which can be thought of as the maximum number of passengers a flight leg can have if there is no capacity limit. It is calculated as the optimal value of the following linear program:

$$
\begin{aligned}
\overline{\lambda}_l = \max \quad & \sum_{p \in \mathcal{P}:\delta_{l,p}=1} s_p \\
\text{s.t.} \quad & \sum_{p \in \mathcal{P}(m)} s_p + s_m = \Lambda_m, \quad \forall m \in \mathcal{M} \\
& \frac{s_p}{v_p} - \frac{s_m}{v_m} \leq 0, \ \forall p \in \mathcal{P}(m), \ \forall m \in \mathcal{M} \\
& s_m \geq 0, \quad \forall m \in \mathcal{M} \\
& s_p \geq 0, \quad \forall p \in \mathcal{P}
\end{aligned}
$$

The linear program maximizes the total sales of products using flight leg $l$ as one of its resources, subject only to demand and MNL consistency constraints with capacity constraints being dropped. This linear program has a closed-form solution given by:

$$
\overline{\lambda}_l = \sum_{m \in \mathcal{M}} \left( \frac{\sum_{p \in \mathcal{P}(m):\delta_{l,p}=1} v_p}{v_m + \sum_{p \in \mathcal{P}(m):\delta_{l,p}=1} v_p} \right) \Lambda_m
$$

The intuition behind this linear program is that to maximize sale on a particular flight, it is optimal to completely close products that are not relevant to that flight, so that demand can be maximally diverted to that particular flight.

Based on $\overline{\lambda}_l$, we further define a *constrained* and *unconstrained* fleet, $\mathcal{F}_c(l)$ and $\mathcal{F}_{uc}(l)$ respectively, for each flight leg $l$. $\mathcal{F}_c(l) = \{f \in \mathcal{F}(l) : \text{CAP}_f < \overline{\lambda}_l\}$ is the set of compatible fleet types whose capacity is smaller than the maximum possible demand $\overline{\lambda}_l$, and similarly $\mathcal{F}_{uc}(l) = \{f \in \mathcal{F}(l) : \text{CAP}_f \geq \overline{\lambda}_l\}$ is the set of compatible fleet types, whose capacity is greater than or equal to $\overline{\lambda}_l$. To illustrate how $\mathcal{F}_c(l)$ and $\mathcal{F}_{uc}(l)$ can be used to reduce the required number of variables, consider a subnetwork $\mathcal{L}^k = \{l_1, l_2\}$ with $l_1, l_2 \in \mathcal{L}_o$. Let there be three fleet types, $f_1, f_2, f_3$, with $\mathcal{F}(l_1) =$

$\mathcal{F}(l_2) = \{f_1, f_2, f_3\}$ and $\mathcal{F}_c(l_1) = \mathcal{F}_c(l_2) = \{f_1\}$. The parsimonious representation of $\mathcal{L}^k$ requires two sets of variables: primary variables and secondary variables, which are shown in Table 5.1 below.

| Primary Variables | Secondary Variables |
|---|---|
| 1. $(l_1, l_2) \leftarrow (f_1, f_1)$ | 10. $l_1 \leftarrow f_2$ |
| 2. $(l_1, l_2) \leftarrow (f_1, \emptyset)$ | 11. $l_1 \leftarrow f_3$ |
| 3. $(l_1, l_2) \leftarrow (f_1, \text{TBD})$ | 12. $l_2 \leftarrow f_2$ |
| 4. $(l_1, l_2) \leftarrow (\emptyset, f_1)$ | 13. $l_2 \leftarrow f_3$ |
| 5. $(l_1, l_2) \leftarrow (\emptyset, \emptyset)$ | |
| 6. $(l_1, l_2) \leftarrow (\emptyset, \text{TBD})$ | |
| 7. $(l_1, l_2) \leftarrow (\text{TBD}, f_1)$ | |
| 8. $(l_1, l_2) \leftarrow (\text{TBD}, \emptyset)$ | |
| 9. $(l_1, l_2) \leftarrow (\text{TBD}, \text{TBD})$ | |

Table 5.1: Parsimonious Representation

As before, $\emptyset$ denotes an optional flight being removed from the schedule. TBD indicates that the fleet assignment is not specified in primary variables but rather in the secondary variables. More specifically, it means that a collection of unconstrained fleet types $\mathcal{F}_{uc}(l)$ is assigned to flight $l$, but the specific one is determined by secondary variables. The secondary variables assign a specific unconstrained fleet type to flight. Note that if $\mathcal{F}_{uc}(l) = \emptyset$, then there is no secondary variable for that fleet type and hence the primary variables cannot have a TBD for that flight leg. The unconstrained fleet can be collectively represented in primary variables because assigning different unconstrained fleet types $f$ to flight leg $l$ yields the same local revenue. For example, two local assignments $(l_1, l_2) \leftarrow (f_1, f_2)$ and $(l_1, l_2) \leftarrow (f_1, f_3)$ have the same local revenue. Note that there are $4 \times 4 = 16$ fleet assignments for flights in $\mathcal{L}^k$ and each of them can be represented by a combination of the primary and secondary variables in Table 5.1, which are 13 in total. For example, $(l_1, l_2) \leftarrow (\emptyset, f_3)$ can be represented by primary variable $(l_1, l_2) \leftarrow (\emptyset, \text{TBD})$ and secondary variable $l_2 \leftarrow f_3$.

We define $\widehat{\mathcal{X}}_1^k$ as the set of primary variables and $\widehat{\mathcal{X}}_2^k$ as the set of secondary variables for subnetwork $\mathcal{L}^k$. Then $\widehat{\mathcal{X}}^k = \widehat{\mathcal{X}}_1^k \cup \widehat{\mathcal{X}}_2^k$ is the set of all parsimonious variables for $\mathcal{L}^k$. We then replace constraints (5.39) with the following two sets of constraints (5.47) and (5.48):

$$\sum_{j \in \widehat{\mathcal{X}}_1^k} w_j = 1, \ \forall k \in \{1, \cdots, K\} \tag{5.47}$$

$$\sum_{j \in \widehat{\mathcal{X}}^k} \tau_{l,f}^j w_j + \sum_{j \in \widehat{\mathcal{X}}_1^k} \sigma_l^j w_j = 1, \ \forall l \in \mathcal{L}^k, \ \forall k \in \{1, \cdots, K\} \tag{5.48}$$

Let us recall the definition of $\tau_{l,f}^j$. $\tau_{l,f}^j = 1$ if local composite fleet assignment $w_j$ assigns fleet type $f$ to flight leg $l$, and $\tau_{l,f}^j = 0$ otherwise. But note that TBD or $\emptyset$ are both not considered to be fleet types in the definition of $\tau_{l,f}^j$. Let $\sigma_l^j = 1$ if optional flight $l$ is removed from the schedule in the definition of local composite fleet assignment variable $w_j$, and $\sigma_l^j = 0$ otherwise. Constraints (5.47) require that exactly one configuration from the primary variables should be selected. Constraints (5.48) state that a flight should be either assigned to a specific fleet type or removed. In order to accurately account for revenue and cost, secondary variables have zero associated revenue, i.e., $r_j = 0, \ \forall j \in \widehat{\mathcal{X}}_2^k, \ \forall k \in \{1, \cdots, K\}$. Primary variables use the same revenue calculation as we did in S-CSD-FAM(II). For a primary variable with some flight leg $l$ indicated as TBD in local composite fleet assignment $j$, we will treat it as being assigned the compatible fleet type with maximum capacity when calculating $r_j$. This is so because assigning any of the unconstrained fleet types makes no difference. Hence we assign the fleet with the maximum possible capacity for simplicity. Calculation of variable costs remains exactly the same for both constrained and unconstrained fleet type variables, and is given by $c_j = \sum_{l \in \mathcal{L}} \sum_{f \in \mathcal{F}(l)} c_{l,f} \tau_{l,f}^j$.

Using parsimonious enumeration, there is a significant reduction in the number of variables when the number of compatible fleet types is large and the proportion of constrained fleet types is small. We refer interested readers to Barnhart et al. (2009) where a detailed evaluation of such reduction is presented.

### 5.3.2 Partition Algorithm for Creating Subnetworks

Creating subnetworks is a key component of the solution process impacting both the solution quality reflected by the accuracy of the objective function approximation, and the computational effort which is affected by the size of the subnetworks. Before introducing the partition algorithm, we are first interested in knowing the conditions under which the approximation is perfect, i.e., $\sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k) = r(\mathbf{x}; \mathbf{h})$. These conditions are characterized in Lemma 5.2 below. We include the proof in Appendix C.1.1.

**Lemma 5.2.** *Define* $\mathcal{P}(\mathbf{h}) = \{p \in \mathcal{P} : h_p > 0\}$*,* $\mathcal{L}(\mathbf{h}) = \{l \in \mathcal{L} : \exists p \in \mathcal{P} \text{ s.t. } h_p > 0, \delta_{l,p} = 1\}$ *and* $\mathcal{M}(\mathbf{h}) = \{m \in \mathcal{M} : \exists p \in \mathcal{P} \text{ s.t. } h_p > 0, p \in \mathcal{P}(m)\}$*. If the following three conditions hold:*

1. $\mathcal{P}(\mathbf{h}^i) \cap \mathcal{P}(\mathbf{h}^j) = \emptyset, \ \forall i, j \in \{1, \cdots, K\} : i \neq j$
2. $\mathcal{L}(\mathbf{h}^i) \cap \mathcal{L}(\mathbf{h}^j) = \emptyset, \ \forall i, j \in \{1, \cdots, K\} : i \neq j$
3. $\mathcal{M}(\mathbf{h}^i) \cap \mathcal{M}(\mathbf{h}^j) = \emptyset, \ \forall i, j \in \{1, \cdots, K\} : i \neq j$

*Then for any fleet assignment* $\mathbf{x}$ *and its local fleet assignment* $\mathbf{x}^k$*, and fare vector* $h$*, we have*

$$\sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k) = r(\mathbf{x}; \sum_{k=1}^{K} \mathbf{h}^k) = r(\mathbf{x}; \mathbf{h})$$

Lemma 5.2 suggests that certain subnetworks and their corresponding local fare vectors are *perfect* in the sense that there is no difference between $\sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k)$ and $r(\mathbf{x}; \mathbf{h})$, and thus CSD-FAM and S-CSD-FAM(Π) are equivalent. We now provide a graphical interpretation of this result. In Figure 5-2, circles represent flights, squares represent fare products and triangles represent markets. An edge exists between a flight node and a fare product node if the fare product uses that flight. Similarly, an edge exists between a fare product node and a market node if the fare product belongs to that market. We call this graph the *dependency graph.* Figure 5-2 represents an example of a small network with five flight legs, six fare products and four markets. Perfect subnetworks and local fare vectors can be found by identifying the maximally connected components of this graph. In this case, we have two connected components: $\mathcal{L}^1 = \{1, 2, 3\}, \mathcal{L}^2 = \{4, 5\}$, and their corresponding local fare vectors $\mathbf{h}^1 = \{300, 450, 560, 280, 0, 0\}, \mathbf{h}^2 = \{0, 0, 0, 0, 310, 380\}$. It is straightforward to check that they satisfy the three conditions listed in Lemma 5.2.

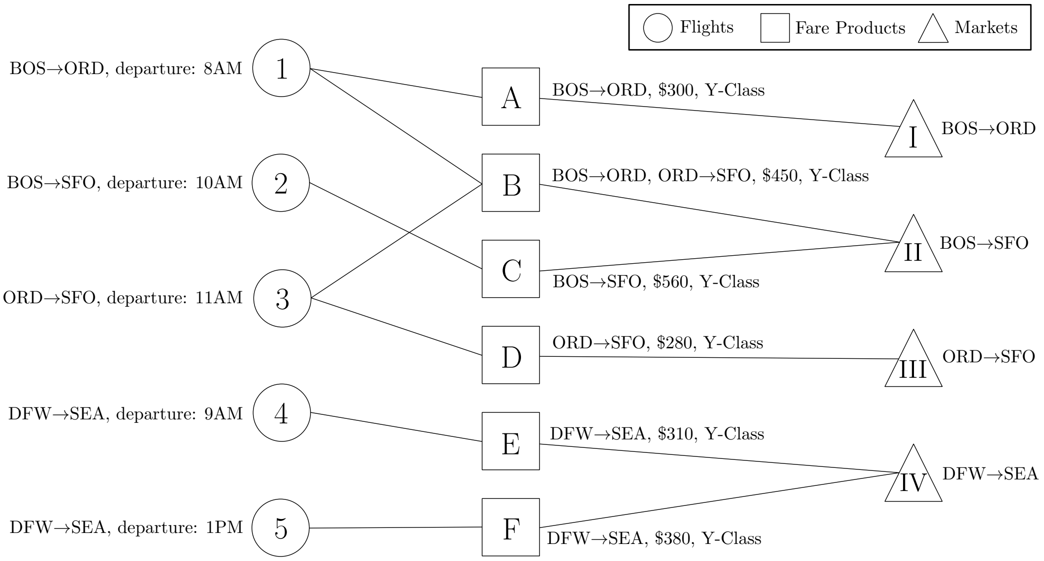


Figure 5-2: Flight, fare product and market dependency network

Unfortunately, these perfect subnetworks created by identifying the connected components usually do not partition the network into sufficiently small subnetworks and thus result in a very limited number of subnetworks. Running S-CSD-FAM with such partitions does not necessarily improve computational performance. However, further fragmentation of the network is likely to yield positive revenue difference $\left(\sum_{k=1}^{K} r^k(\mathbf{x}^k;\mathbf{h}^k) - r(\mathbf{x};\mathbf{h})\right)$ leading to an approximation error. In order to strike a balance between computational efficiency and solution quality, similar to Barnhart et al. (2009), we find an upper bound on the revenue difference in terms of the degree of fragmentation of the network, as shown in Theorem 5.1 below.

**Theorem 5.1.** *For some $\widehat{P} \subset P$, and for some fare vector $h^k$, we define a restricted local fare vector $\widehat{\mathbf{h}}^k$ and its complement $\widetilde{\mathbf{h}}^k$ as follows:*

$$\widehat{h}_p^k = \begin{cases} h_p^k, & p \in \widehat{P} \\ 0, & \text{otherwise} \end{cases}, \qquad \widetilde{h}_p^k = \begin{cases} 0, & p \in \widehat{P} \\ h_p^k, & \text{otherwise} \end{cases}$$

*If the following three conditions hold for $\widehat{\mathbf{h}}^k$:*

1. $\mathcal{P}(\widetilde{\mathbf{h}}^i) \cap \mathcal{P}(\widetilde{\mathbf{h}}^j) = \emptyset,\ \forall i,j \in \{1,\cdots,K\}, i \neq j$

2. $\mathcal{L}(\widetilde{\mathbf{h}}^i) \cap \mathcal{L}(\widetilde{\mathbf{h}}^j) = \emptyset,\ \forall i,j \in \{1,\cdots,K\}, i \neq j$

*3.* $\mathcal{M}(\widetilde{\mathbf{h}}^i) \cap \mathcal{M}(\widetilde{\mathbf{h}}^j) = \emptyset,\ \forall i, j \in \{1, \cdots, K\}, i \neq j$

*Then,*

$$\sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k) - r(\mathbf{x}; \mathbf{h}) \leq \sum_{p \in \widehat{\mathcal{P}}} h_p \overline{s}_p,$$

*where* $\overline{s}_p$ *is the maximum possible sale for product* $p \in \mathcal{P}(m)$*, and is calculated as* $\overline{s}_p = \min\left\{\frac{v_p}{v_p+v_m}\Lambda_m,\ \min_{l \in \mathcal{L}:\delta_{l,p}=1}\{\max_{f \in \mathcal{F}(l)} CAP_f\}\right\}$*. The first term is the maximum possible demand for product* $p$*, and the second term is the maximum possible capacity available to product* $p$*.*

*Proof.* We have,

$$\begin{aligned}
&\sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k) - r(\mathbf{x}; \mathbf{h}) \\
\leq &\sum_{k=1}^{K} r^k(\mathbf{x}^k; \widehat{\mathbf{h}}^k) + \sum_{k=1}^{K} r^k(\mathbf{x}^k; \widetilde{\mathbf{h}}^k) - r(\mathbf{x}; \mathbf{h}) \\
= &\sum_{k=1}^{K} r^k(\mathbf{x}^k; \widehat{\mathbf{h}}^k) + r(\mathbf{x}; \sum_{k=1}^{K} \widetilde{\mathbf{h}}^k) - r(\mathbf{x}; \mathbf{h}) \\
\leq &\sum_{k=1}^{K} r^k(\mathbf{x}^k; \widehat{\mathbf{h}}^k) \\
\leq &\sum_{k=1}^{K} \sum_{p \in \widehat{P}} \widehat{h}_p^k \overline{s}_p = \sum_{p \in \widehat{P}} \left(\sum_{k=1}^{K} h_p^k\right) \overline{s}_p = \sum_{p \in \widehat{P}} h_p \overline{s}_p
\end{aligned}$$

Suppose $\mathbf{s}^*$ is the optimal solution of $r^k(\mathbf{x}^k; \mathbf{h}^k)$, then it is also feasible solution for both $r^k(\mathbf{x}^k; \widehat{\mathbf{h}}^k)$ and $r^k(\mathbf{x}^k; \widetilde{\mathbf{h}}^k)$ because these three problems have the same constraints. This gives $r^k(\mathbf{x}^k; \mathbf{h}^k) = \sum_{p \in \mathcal{P}^k} h_p^k s_p^* = \sum_{p \in \mathcal{P}^k} \widehat{h}_p^k s_p^* + \sum_{p \in \mathcal{P}^k} \widetilde{h}_p^k s_p^* \leq r^k(\mathbf{x}^k; \widehat{\mathbf{h}}^k) + r^k(\mathbf{x}^k; \widetilde{\mathbf{h}}^k)$, which verifies the first inequality. The first equality holds because of the definition of $\widetilde{\mathbf{h}}$ and Lemma 5.2. The second inequality holds because $\sum_{k=1}^{K} \widetilde{\mathbf{h}}^k \leq \mathbf{h}$ and $r(\mathbf{x}; \mathbf{h})$ is a non-decreasing function of $\mathbf{h}$. $r(\mathbf{x}; \mathbf{h})$ is non-decreasing in $\mathbf{h}$ because $r(\mathbf{x}; \mathbf{h})$ is a maximization problem with objective coefficients $\mathbf{h}$ and non-negative decision variables. The third inequality holds because $r^k(\mathbf{x}^k; \widehat{\mathbf{h}}^k) = \sum_{p \in \widehat{P}} \widehat{h}_p^k s_p^* \leq \sum_{p \in \widehat{P}} \widehat{h}_p^k \overline{s}_p$ where $\mathbf{s}^*$ is the optimal solution of $r^k(\mathbf{x}^k; \widehat{\mathbf{h}}^k)$.

The second equality is a re-arrangement of terms and also based on the fact that $\widehat{h}_p^k = h_p^k,\ \forall p \in \widehat{P},\ \forall k \in \{1, \cdots, K\}$. Finally, the third equality is due to Definition 5.1 which states that $\sum_{k=1}^K \mathbf{h}^k = \mathbf{h}$. $\square$

We now provide a graphical interpretation of Theorem 5.1. For any set of flight legs $\mathcal{L}' \subset \mathcal{L}$, products $\mathcal{P}' \subset \mathcal{P}$ and markets $\mathcal{M}' \subset \mathcal{M}$, let $\Pi^{\text{mcc}}(\mathcal{L}', \mathcal{P}', \mathcal{M}')$ denote the partition of flights (subnetworks) created by the maximally connected components of the dependency graph with flight set $\mathcal{L}'$, product set $\mathcal{P}'$ and market set $\mathcal{M}'$. Then Theorem 5.1 essentially states that the difference between the approximate and exact revenues $\left(\sum_{k=1}^K r^k(\mathbf{x}^k; \mathbf{h}^k) - r(\mathbf{x}; \mathbf{h})\right)$ is upper bounded by the total maximum revenue of products in $\widehat{\mathcal{P}}$ such that if we remove $\widehat{P}$ from $P$, we have $\{\mathcal{L}_1, \cdots, \mathcal{L}_K\} = \Pi^{\text{mcc}}(\mathcal{L}, \mathcal{P} \backslash \widehat{\mathcal{P}}, \mathcal{M})$, i.e., the partition $\{\mathcal{L}_1, \cdots, \mathcal{L}_K\}$ can be created by finding the maximally connected components of the dependency graph after removing products in $\widehat{\mathcal{P}}$.

This interpretation motivates us to develop a partition heuristic to fragment the dependency network by gradually removing products in the ascending order of their revenue contributions. The details of the steps of the partitioning process are outlined in Algorithm 5. The algorithm starts with the entire flight, product and market dependency network. It then iteratively removes fare product nodes in ascending order of their maximum revenue $h_p\overline{s}_p$. The coarseness of such removal is controlled by the *revenue_ step* parameter, where the revenue threshold for removing product nodes increases by *revenue_ step* at each iteration. As the network is being fragmented, any flight subnetwork whose size can be represented by no more than *max_ vars* number of parsimonious composite variables will be added to $\Pi_c$, and subnetworks that are too big to be represented by composite variables are contained in $\Pi_t$. The iterative process ends when the ratio of the number of flights represented by traditional flight-fleet variables $|\mathcal{L}(\Pi_t)|$ to the number of total flights $|\mathcal{L}|$ is at most equal to the parameter *max_ ratio*.

**Algorithm 5** Construct Partition $(\Pi_c, \Pi_t)$

```
Initialize:
    $\Pi_c = \emptyset,\ \Pi_t = \Pi^{\mathrm{mcc}}(\mathcal{L}, \mathcal{P}, \mathcal{M})$
for $\mathcal{L}' \in \Pi_t$ do
    if the number of parsimonious variables needed to represent $\mathcal{L}' \leq \mathit{max\_vars}$ then
        $\Pi_t = \Pi_t \setminus \mathcal{L}', \Pi_c = \Pi_c \cup \mathcal{L}'$
    end if
end for
iter $= 1$
while $\frac{|\mathcal{L}(\Pi_t)|}{|\mathcal{L}|} > \mathit{max\_ratio}$ do
    $\Pi_t' = \emptyset$
    for $\mathcal{L}' \in \Pi_t$ do
        $P' = \{p \in \mathcal{P} : \exists l \in \mathcal{L}'$ s.t. $\delta_{l,p} = 1,\ h_p \overline{s}_p > \mathit{revenue\_step} \times$ iter$\}$
        $\mathcal{M}' = \{m \in \mathcal{M} : \exists p \in \mathcal{P}'$ s.t. $p \in \mathcal{P}(m)\}$
        $\Pi' = \Pi^{\mathrm{mcc}}(\mathcal{L}', \mathcal{P}', \mathcal{M}')$
        for $\mathcal{L}'' \in \Pi'$ do
            if number of parsimonious variables to represent $\mathcal{L}'' \leq \mathit{max\_vars}$ then
                $\Pi_c = \Pi_c \cup \mathcal{L}''$
            else
                $\Pi_t' = \Pi_t' \cup \mathcal{L}''$
            end if
        end for
    end for
    $\Pi_t = \Pi_t'$
    iter = iter + 1
end while
return $\Pi_c, \Pi_t$
```

### 5.3.3 Fare Split Problem

Given a partition, the purpose of the fare split problem is to find good local fare vectors $\{\mathbf{h}^1, ..., \mathbf{h}^K\}$ so that the revenue difference $\sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k) - r(\mathbf{x}; \mathbf{h})$ is further minimized for *any* fleet assignment $\mathbf{x}$ and its corresponding local fleet assignment $\mathbf{x}^k$. We thus turn our attention to the following problem:

$$\min_{\{\mathbf{h}^1,...,\mathbf{h}^K\}:\sum_{k=1}^{K}\mathbf{h}^k=\mathbf{h}} \; \max_{\mathbf{x}\in\mathcal{X}} \; \sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k) - r(\mathbf{x}; \mathbf{h}). \tag{5.49}$$

In words, the objective minimizes the worst (i.e. maximum) possible revenue difference across all possible fleet assignments $\mathcal{X}$. Unfortunately, problem (5.49) is computationally intractable. To approximate it, let us consider two particular fleet assignments, $\overline{\mathbf{x}}$ and $\underline{\mathbf{x}}$, defined as follows:

$$\overline{x}_{l,f} = \begin{cases} 1, & f = \arg\max_{f'\in\mathcal{F}(l)} \mathrm{CAP}_{f'}, \forall l \in \mathcal{L} \\ 0, & \text{otherwise} \end{cases}$$

$$\underline{x}_{l,f} = \begin{cases} 1, & f = \arg\min_{f'\in\mathcal{F}(l)} \mathrm{CAP}_{f'}, \forall l \in \mathcal{L} \\ 0, & \text{otherwise} \end{cases}$$

In words, $\overline{\mathbf{x}}$ assigns each flight the fleet type with maximum possible capacity, and $\underline{\mathbf{x}}$ assigns the one with minimum possible capacity. We now focus on the following problem:

$$\min_{\{\mathbf{h}^1,...,\mathbf{h}^K\}:\sum_{k=1}^{K}\mathbf{h}^k=\mathbf{h}} \; \max_{\mathbf{x}\in\{\overline{\mathbf{x}},\underline{\mathbf{x}}\}} \; \sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k) - r(\mathbf{x}; \mathbf{h}), \tag{5.50}$$

where the local fare vectors $\{\mathbf{h}^1, ..., \mathbf{h}^K\}$ are selected to minimize the maximum revenue difference under the two particular fleet assignments in consideration: $\overline{\mathbf{x}}$ and $\underline{\mathbf{x}}$. The intuition behind problem (5.50) is that $\overline{\mathbf{x}}$ and $\underline{\mathbf{x}}$ are two extreme resource settings: one with the most abundant capacity and the other with the most scarce capacity, and finding a balance between these two will likely lead to a fairly good approximation of problem (5.49). Computational experiments confirm its strong performance, as demonstrated in Section 5.4. Also, problem (5.50) can be reformulated as a single linear program using duality theory, the details of which are provided in Appendix C.2.

### 5.3.4 Overall Algorithm

We summarize the overall algorithm in this section. For a typical S-CSD-FAM($\Pi_p, \Pi_t$) run, there are four components that need to be run in order, as follows:

*Step 1.* Run Algorithm 5 to construct a partition such that some of its subnetworks (those in $\Pi_c$) use the composite variable representation, while others (those in $\Pi_t$) use the traditional flight-fleet variable representation.

*Step 2.* Solve the approximation (5.50) of the fare split problem to get local fare vectors $\mathbf{h}^k,\ \forall k \in \{1, \cdots, K\}$.

*Step 3.* Calculate $r_j = r^k(\mathbf{x}^j; \mathbf{h}^k),\ \forall j \in \mathcal{X}^k, \forall k \in \{1, \cdots, K\} : \mathcal{L}^k \in \Pi_c$.

*Step 4.* Solve S-CSD-FAM($\Pi_p, \Pi_c$) described by formulation (5.38) - (5.46).

Step 3 does not require the solution of $\sum_{k \in \{1,\cdots,K\}:\mathcal{L}^k \in \Pi_c} |\mathcal{X}^k|$ linear programs. Instead, a large proportion of the overall computational effort can be saved by utilizing tools from sensitivity analysis for linear programs. Local revenue terms for the same subnetwork only differ in terms of the right hand side of one type of constraints, and thus *100 percent rule* (Bradley et al., 1977) can be applied to determine whether the same basis still remains optimal. Experience from our computational studies shows that only between 20% to 40% of all the linear programming problems needed to be solved. The remaining were solved much faster using the results of the sensitivity analysis.

## 5.4 Computational Experiments

In this section, we describe our computational experiments to demonstrate the effectiveness of our proposed solution approach using S-CSD-FAM($\Pi_c, \Pi_t$). We use a full-scale daily instance from a major US airline in May 2014. The instance consists of 815 domestic flights, 4,290 itineraries leading to 47,190 total fare products (multiple fare products can correspond to the same itinerary), and 819 markets, where each market is characterized as a specific origin-destination pair. All fare products corresponding to the same origin and destination are considered by the passengers in that market. So on average, $47,190/819 \approx 57.6$ products correspond to each market. $v_p$, which is the attractiveness value of each product $p$, $v_m$, which is the total attractiveness value of the products of all competing airlines and the no-fly alternative, and $\Lambda_m$, which is the market demand are estimated using airline booking and product availability data via methods described in Vulcano et al. (2012), which involve sophisticated demand untruncation and choice model estimation. The average market share of the airline under consideration across all its markets is roughly 42%. Seven different fleet types and 187 aircraft are available to operate this network. The seating capacities of available aircraft range from 48 to 165 passengers. In order to fully test the proposed methodology, all flights are set to be optional.

We test our S-CSD-FAM$(\Pi_c, \Pi_t)$ runs with different partitions $(\Pi_c, \Pi_t)$ constructed by the partition algorithm (Algorithm 5). We set *revenue_step* to be 100. Table 5.2 below summarizes the details of the seven runs. S-CSD-FAM-0 corresponds to an *elementary partition* where each individual flight constitutes a separate subnetwork. S-CSD-FAM-1 to S-CSD-FAM-6 are constructed using different values of parameters *max_vars* and *max_ratio* in Algorithm 5. Columns 4 to 10 in Table 5.2 present the number of subnetworks with different sizes in $\Pi_c$, and the total number of subnetworks in $\Pi_c$. Similarly, columns 11 to 13 in Table 5.2 provide details about the subnetworks in $\Pi_t$. Column 11 is the smallest size across all subnetworks, column 12 is the largest size, and column 13 is the total number of subnetworks. Note that there is no information in columns 11 to 13 for S-CSD-FAM-0 to S-CSD-FAM-3 because $\Pi_t = \emptyset$ in their partitions. Also, S-CSD-FAM-6 is essentially the perfect partition $\Pi^{\text{mcc}}(\mathcal{L}, \mathcal{P}, \mathcal{M})$, the partition consisting of subnetworks created by finding maximally connected components of the original dependency graph without removing any fare product nodes, because $max\_ratio = 1$.

| Run | Partiton | | Number of Subnetworks in $\Pi_c$ (of size) | | | | | | | Subnetworks in $\Pi_t$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | max_vars | max_ratio | 1 | 2 | 3 | 4 | 5 | 6 | Total | min_size | max_size | Total |
| S-CSD-FAM-0 | elementary partition | | 815 | 0 | 0 | 0 | 0 | 0 | 815 | – | – | 0 |
| S-CSD-FAM-1 | 1,000 | 0 | 424 | 48 | 94 | 2 | 1 | 0 | 569 | – | – | 0 |
| S-CSD-FAM-2 | 10,000 | 0 | 360 | 45 | 58 | 44 | 3 | 0 | 510 | – | – | 0 |
| S-CSD-FAM-3 | 100,000 | 0 | 304 | 44 | 55 | 20 | 32 | 3 | 458 | – | – | 0 |
| S-CSD-FAM-4 | 1,000 | 1/3 | 232 | 45 | 70 | 2 | 1 | 0 | 350 | 4 | 51 | 29 |
| S-CSD-FAM-5 | 1,000 | 2/3 | 92 | 33 | 34 | 2 | 1 | 0 | 162 | 4 | 450 | 18 |
| S-CSD-FAM-6 | 1,000 | 1 | 26 | 1 | 0 | 0 | 0 | 0 | 27 | 787 | 787 | 1 |

Table 5.2: S-CSD-FAM($\Pi_c, \Pi_t$) Partitions

Table 5.3 describes the computational performance of S-CSD-FAM-0 to S-CSD-FAM-6. Column 4 in the table lists two fare split methods: (1) *dist* indicates that the fare split is calculated using proration by distance, (2) *opt* indicates that the fare split is optimized using problem (5.50) in Section 5.3.3. Proration by distance is a commonly used heuristic in BFAM literature to calculate the profit contribution on a flight leg level. In our case, it refers to an approach that calculates $h_p^k$ as $h_p^k = \left(\sum_{l \in \mathcal{L}^k} \delta_{l,p} d_l\right) / \left(\sum_{l \in \mathcal{L}} \delta_{l,p} d_l\right)$ where $d_l$ is the flying distance of flight leg $l$. Thus, it allocates fare of a fare product to subnetworks in proportion to their distance contributions to the fare product. Note that for the S-CSD-FAM-6 run, both types of fare splits yield the same results because the underlying subnetworks correspond to a perfect partition $\Pi^{\text{mcc}}(\mathcal{L}, \mathcal{P}, \mathcal{M})$.

The computational experiments were conducted on a desktop equipped with an Intel Core i5-2400 3.1 GHz CPU and an 8 GM RAM. All mathematical programs were solved using Gurobi 6.5 with default parameters. We set a 12-hour computation time limit for the branch-and-bound process for solving the S-CSD-FAM($\Pi_c, \Pi_t$) formulation. If an optimal solution is found within 12 hours, we report the solution time in the last column (column 10) of Table 5.3. Otherwise, we report the optimality gap in parentheses as reported by Gurobi at the end of the solution process. Columns 7 to 9 report the computational time (in seconds) of the preprocessing steps, Steps 1 to 3, respectively, from Section 5.3.4. Column 5 reports the daily profit in millions of dollar. The revenue value is evaluated using formulation $r(\mathbf{x}; \mathbf{h})$ described in (5.13) - (5.18). Column 6 lists the number of flights out of 815 optional flights selected for operation in the final solution. Note that S-CSD-FAM-1 to S-CSD-FAM-3 are similar to the pure subnetwork-based formulation introduced by Barnhart et al. (2009).

Overall, we can see that S-CSD-FAM-5 produces the solution with the highest profit. This is achieved by better balancing solution quality and tractability. It outperforms S-CSD-FAM-6 because of its enhanced tractability resulting from a finer partition, even though S-CSD-FAM-6 does not have any revenue approximation error. On the other hand, it outperforms S-CSD-FAM-0 to S-CSD-FAM-4 due to its more accurate revenue approximation resulting from a coarser partition. The large profit difference between S-CSD-FAM-1 to S-CSD-FAM-3 and S-CSD-FAM-5 demonstrates that a pure subnetwork-based formulation (5.32) - (5.37) is not adequate to approximate choice-based revenue, and demonstrates the effectiveness of our approach based on mixed formulations.

A profit boost (as large as \$45,000 daily profit, \$16.4 million annually) can be obtained by using an optimization-based fare split algorithm instead of the commonly used heuristic of proration by distance. The additional computational time required for the fare split optimization was found to be only a few minutes. Calculating the revenue of local composite fleet assignment variables $r_j$ can be time-consuming when the number of composite variables is large. Fortunately, under the subnetwork-based mixed formulation, a relatively small number of composite variables are required for good solution quality. This helps to manage to an acceptable amount of the computational time required for preprocessing under control.

| Run | Partition | | Fare | Profit | Flights | Preproc. Time (sec) | | | Solution |
|---|---|---|---|---|---|---|---|---|---|
| | max_vars | max_ratio | | (\$M/day) | Selected | Partition | Fare | Revenue | Time (sec) |
| S-CSD-FAM-0 | elementary partition | | dist | 5.160 | 805 | 0 | 0 | 18 | 9 |
| | | | opt | 5.167 | 799 | 0 | 19 | 17 | 9 |
| S-CSD-FAM-1 | 1,000 | 0 | dist | 5.413 | 782 | 17 | 0 | 53 | 63 |
| | | | opt | 5.450 | 778 | 17 | 74 | 40 | 40 |
| S-CSD-FAM-2 | 10,000 | 0 | dist | 5.517 | 776 | 16 | 0 | 535 | 205 |
| | | | opt | 5.517 | 772 | 16 | 77 | 386 | 226 |
| S-CSD-FAM-3 | 100,000 | 0 | dist | 5.578 | 766 | 15 | 0 | 18,371 | 795 |
| | | | opt | 5.618 | 751 | 15 | 118 | 11,914 | 1,386 |
| S-CSD-FAM-4 | 1,000 | 1/3 | dist | 5.908 | 732 | 13 | 0 | 34 | (0.63%) |
| | | | opt | 5.941 | 725 | 13 | 108 | 24 | (0.61%) |
| S-CSD-FAM-5 | 1,000 | 2/3 | dist | 6.119 | 679 | 7 | 0 | 16 | (3.14%) |
| | | | opt | 6.164 | 681 | 7 | 148 | 9 | (2.94%) |
| S-CSD-FAM-6 | 1,000 | 1 | – | 6.090 | 642 | 0 | 85 | 1 | (5.79%) |

Table 5.3: Results of the S-CSD-FAM($\Pi_c, \Pi_t$) Computational Runs (with 12 hr CPU Time)

We finally compare the S-CSD-FAM($\Pi_c, \Pi_t$) formulation with two baseline approaches: CSD-FAM and ISD-FAM. We choose S-CSD-FAM-5 under the optimized fare split as our approach. Baseline approach CSD-FAM is the formulation described in (5.1) - (5.12). The other baseline, ISD-FAM, only differs from CSD-FAM in its revenue modeling, which is a simplification based on the deterministic linear program of the independent product demand model, i.e., there is no spill and recapture. The same model is also studied in Sherali et al. (2010). We defer the details of the ISD-FAM formulation to Appendix C.3. Please note that the main purpose of this comparison is to demonstrate the efficiency of our S-CSD-FAM($\Pi_c, \Pi_t$) formulation over a much simpler CSD-FAM formulation. ISD-FAM serves as an additional yardstick demonstrating how a computationally efficient formulation with less comprehensive demand modeling assumptions approximates the choice-based revenue. Recall that there was another SD-FAM model introduced by Lohatepanont and Barnhart (2004) where spill and recapture behavior is modeled using recapture rates and demand correction terms that are informed by QSI. We do not benchmark against that model for three related reasons. First, it is developed under a different set of demand modeling assumptions, making a direct comparison difficult. Also, obtaining "correct" recapture rates for their modeling scheme from a choice-based perspective is not straightforward, if not impossible. Furthermore, evaluation of their model under a choice-based revenue assumption might also be biased.

We summarize the results of this experiment in Table 5.4. We report the performance of each method under two CPU time limits: 5 hours and 12 hours. We present the daily profit and the annual profit improvement over the ISD-FAM solution in millions of dollars. Again all revenue evaluation is conducted with choice-based demand using (5.13) - (5.18). None of the models solves to optimality within 12 hours. So we also report the optimality gap obtained from Gurobi at the end of the solution process. Judging from the optimality gap, we can see that CSD-FAM is much more difficult to solve compared to ISD-FAM, and S-CSD-FAM-5 is more tractable than CSD-FAM, but not as tractable as ISD-FAM. Under the 5 hour limit, the profit of the best CSD-FAM solution found is around 15.3 million dollars less than that for ISD-FAM. S-CSD-FAM-5, however, significantly outperforms ISD-FAM, finding a solution with an annual profit improvement of around 31.4 million dollars. This demonstrates that the tractability issues associated with CSD-FAM deteriorate its performance such that even a simplified method (ISD-FAM), which does not model spill and recapture behaviors, outperforms it. With the 12 hour limit, the CSD-FAM solution outperforms the ISD-FAM solution with an annual increase in profit of 9.1 million dollars, and S-CSD-FAM-5 further improves on the annual profit by another 24.5 million dollars.

| | ISD-FAM | | | CSD-FAM | | | S-CSD-FAM-5, opt | | |
|---|---|---|---|---|---|---|---|---|---|
| Time Limit | Profit ($M/d) | Profit Change ($M/yr) | Gap (%) | Profit ($M/d) | Profit Change ($M/yr) | Gap (%) | Profit ($M/d) | Profit Change ($M/yr) | Gap (%) |
| 5 hr | 6.059 | 0 | 0.12 | 6.017 | (15.330) | 7.74 | 6.145 | 31.390 | 3.52 |
| 12 hr | 6.072 | 0 | 0.05 | 6.097 | 9.125 | 5.68 | 6.164 | 33.580 | 2.94 |

Table 5.4: Comparison of Models

# Chapter 6

# Concluding Remarks and Future Research Directions

In this thesis, we develop and solve models for a set of topics in airline scheduling and air traffic control to address current challenges faced by the airline industry.

1. In Chapter 2, we propose a robust optimization-based formulation for the aircraft routing problem to minimize the maximal possible total propagated delay, assuming primary delays belong to a pre-specified uncertainty set. We provide data-driven methods to construct a flight delay uncertainty set that not only includes the uncertainty of individual flights, but also the correlations among different flights. We then reformulate the robust problem as an integer program. We propose an exact decomposition solution approach under a column-and-row generation framework. Importantly, this solution method can be applied to general robust optimization problems where the nominal problem is solved through a branch-and-price technique.

   Using real-world instances with actual schedule and delay data as well as simulated delay data, we demonstrate the effectiveness and efficiency of our method. We show that by incorporating delay correlation, our robust model outperforms the state-of-the-research stochastic optimization approach by reducing in most cases, all of our propagated-delay based performance criteria: average value, standard deviation and maximum value. In the cases when a deficit in one criterion exists, it is usually offset by gains in the other two criteria.

We attribute these benefits to characteristics of our solution method: (1) the robust approach provides a tractable exact method for dealing with correlated uncertainty, thus providing additional robustness against propagated delay; and (2) the robust approach is less vulnerable to the variations in the input data when compared to stochastic optimization approaches (i.e., the robust approach provides additional robustness when the realized flight delays distributionally differ from historical data).

The idea of applying robust optimization to the airline planning problem is fairly new. There are many possible future research directions in this area. For example, a direct extension of this work could be to apply robust optimization to the integrated aircraft routing and crew pairing problem. It may also be interesting to investigate tailored approaches to modeling flight delay uncertainty sets, so as to achieve greater tractability as well as probabilistic performance guarantees. We believe that robust optimization, with its merits of tractability and attractive solution quality, represents a promising direction for dealing with uncertainty in airline scheduling problems.

2. In Chapter 3, inspired by an application in air traffic flow management (ATFM) to incorporate airline preferences in planning various ATFM programs, we describe a theory, mechanisms and computational approaches to extend the recently developed Majority Judgment (Balinski and Laraki, 2014) electing and ranking method to handle candidate spaces which can be modeled as polyhedral sets. In Chapter 4, we assess the benefits of adopting such a voting-based, decentralized, airline-driven approach for designing ground delay programs (GDPs). Unlike the centralized approaches in the existing literature which develop various optimization models to minimize centralized cost measures, the airline-driven approach systematically incorporates the inherent differences in different airlines' recovery potentials due to the differences in their business objectives

and operating characteristics. Through evaluations using a realistic simulation-based testbed and actual schedule data for San Francisco International Airport, we show that the airline driven approach, compared to the state-of-the-research centralized approaches, reduces airport-wide costs by 2%, and substantially enhances equity in distributing delay costs among airlines.

In addition to the simulation-based benefits assessment described in Chapter 4, we, together with other members in our research team, also conducted human-in-the-loop (HITL) experiments by engaging flight operators from major U.S. airlines as well as FAA air traffic controllers. Feedback from the participants in the experiments was positive and promising. The details of the HITL experiments are provided in Ball et al. (2017). A potential future research direction is to develop better user support tools to help airlines understand and grade candidate GDP designs. Investigating and eliminating potential gaming opportunities during the preference solicitation process is another future research opportunity. We believe that decentralized planning of ATFM programs represents a promising path to enhance system-wide performance. We hope to attract the attention of both practitioners and researchers to this field.

3. Finally, in Chapter 5, we study the integrated airline fleet assignment and schedule design problem incorporating passenger choice behavior. We develop a more tractable reformulation and a reliable approximation scheme for the model. In previous studies, such models suffered from serious computational challenges that prevented them from being implemented on full-scale instances in the industry. Our formulation mixes a subnetwork-based composite variable representation and a traditional flight-fleet variable representation. We also develop a fare split problem to optimize fare proration and achieve significant profit improvement over existing heuristics. Through computational experiments with one full-scale instance from a major U.S. airline, we demonstrate that our proposed method significantly outperforms existing baselines in terms of profit contribution within the same computational budget.

Although our approach is developed based on the assumption that passenger demand is governed by a multinomial logit (MNL) choice model, it can be readily extended to other choice models where a compact deterministic linear programming formulation for network revenue management (NRM) exists. For example, the general attraction model developed in Gallego et al. (2014), and the Markov chain choice model (Feldman and Topaloglu, 2017) both have compact sales-based formulations for NRM. These advanced choice models bypass structural limitations of MNL and enjoy better predictive power. For a general choice model, column generation is required for solving the deterministic linear program for NRM, and thus it will also be required in our solution framework, for which we anticipate additional computational challenges. Developing extensions to more general choice models can be a direct follow-up of our work.

With enhanced tractability, further integration of choice-based demand models with other airline planning processes (such as timetable development including frequency and departure time selection) becomes computationally possible. The enhanced tractability also facilitates stochastic planning models where demand fluctuation is considered (Sherali and Zhu, 2008). With ever-growing collection of passenger shopping data, booking behavior can be accurately estimated using choice models. We believe that choice-based airline planning models represent a promising future research direction to help airlines better understand customer demand, thus making smarter data-driven planning decisions.

# Appendix A

# Additional Experiment Results and Technical Procedures for Chapter 2: Robust Aircraft Routing

## A.1 Detailed Computational Results in Section 4.3

Figures A-1 to A-9 depict the performance in flight network $N_1$, and Figures A-10 to A-18 depict the performance in flight network $N_2$. Specifically, Figures A-1 to A-3, A-10 to A-12 present the relative performance ratio $100 \cdot (\mathrm{DFW} - \mathrm{RAR})/\mathrm{DFW}$ when the mean of the testing data under three different distributions deviate from the training data for two flight networks. Figures A-4 to A-6, A-13 to A-15 show the relative performance ratio $100 \cdot (\mathrm{DFW} - \mathrm{RAR})/\mathrm{DFW}$ when the standard deviation of the testing data under three different distributions deviate from the training data for two flight networks. Figures A-7 to A-9, A-16 to A-18 show the relative performance ratio $100 \cdot (\mathrm{DFW} - \mathrm{RAR})/\mathrm{DFW}$ when the correlation structure of the testing data under three different distributions deviate from the training data for two flight networks. Performance is evaluated in three criteria: (1) average total propagated delay, (2) standard deviation of total propagated delay, (3) maximum total propagated delay. The long dashed line indicates 0% relative performance ratio, which denotes the region where DFW and RAR have the same performance.

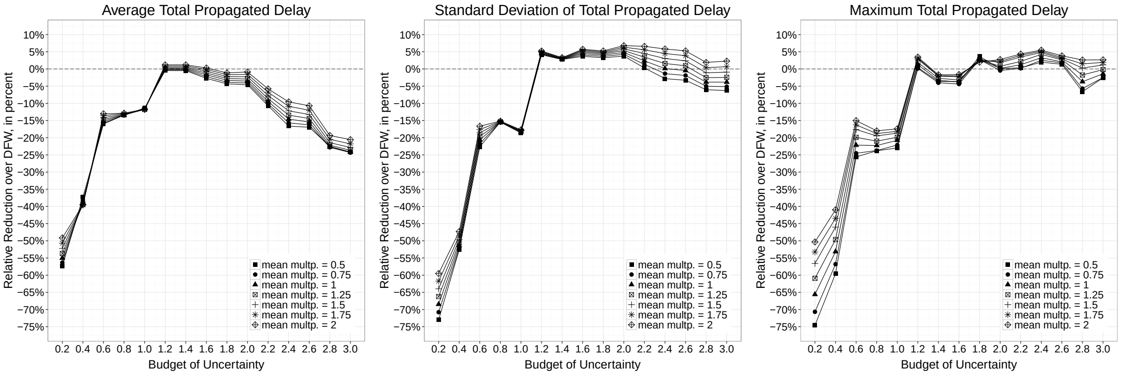


Figure A-1: Impact of Deviation in Mean (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Truncated Normal Distribution / Testing Flight Network: $N_1$)

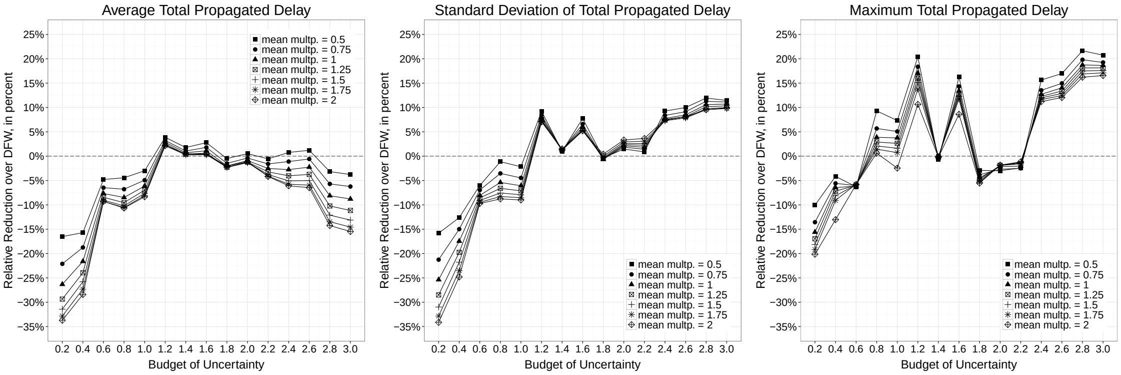


Figure A-2: Impact of Deviation in Mean (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Gamma Distribution / Testing Flight Network: $N_1$)

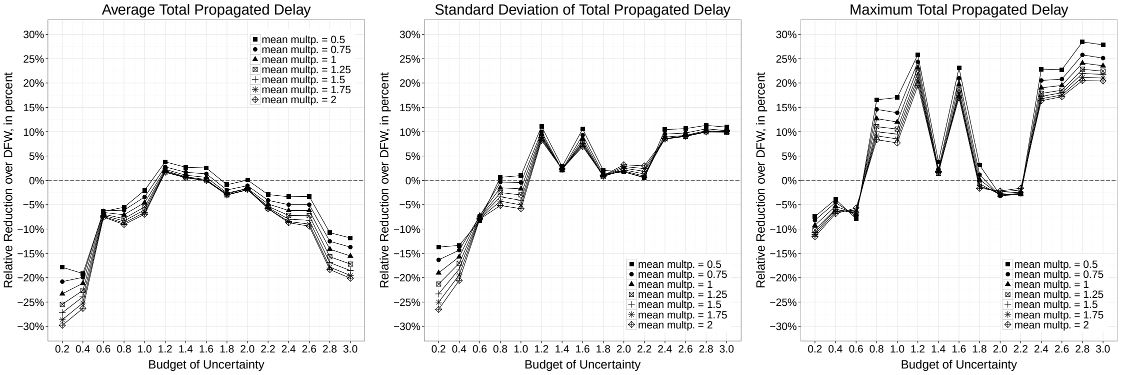


Figure A-3: Impact of Deviation in Mean (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Log-normal Distribution / Testing Flight Network: $N_1$)

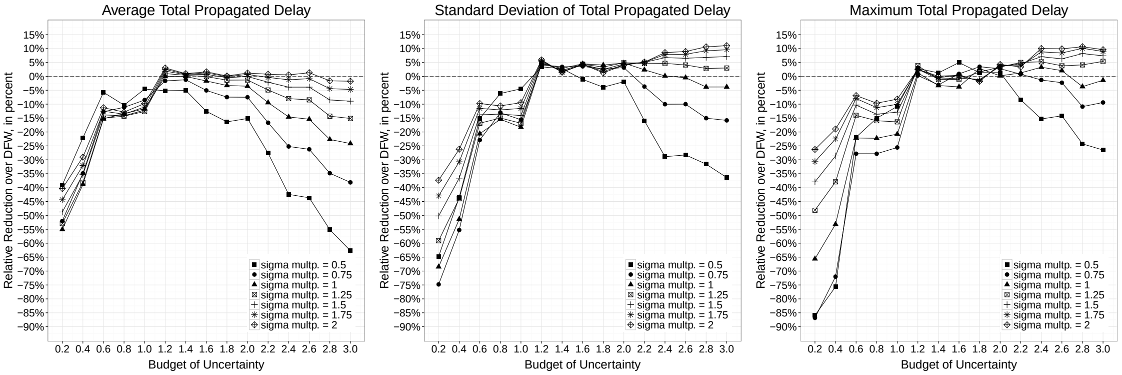


Figure A-4: Impact of Deviation in Standard Deviation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Truncated Normal Distribution / Testing Flight Network: $N_1$)

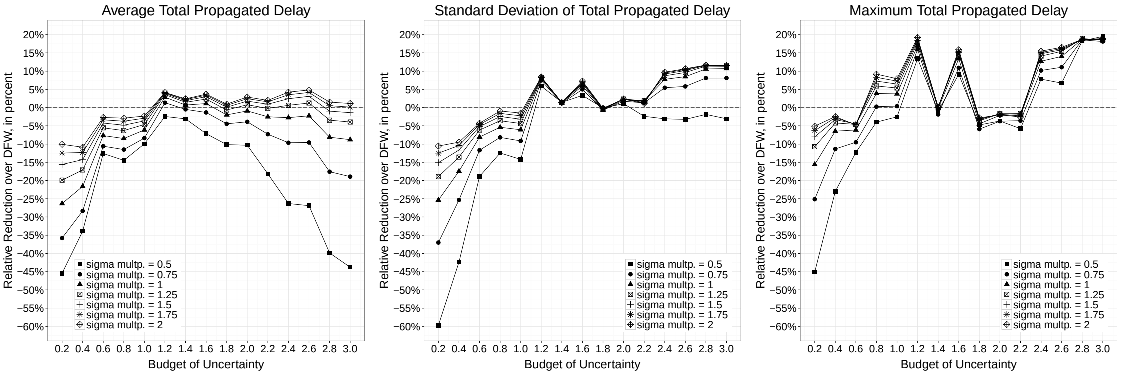


Figure A-5: Impact of Deviation in Standard Deviation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Gamma Distribution / Testing Flight Network: $N_1$)

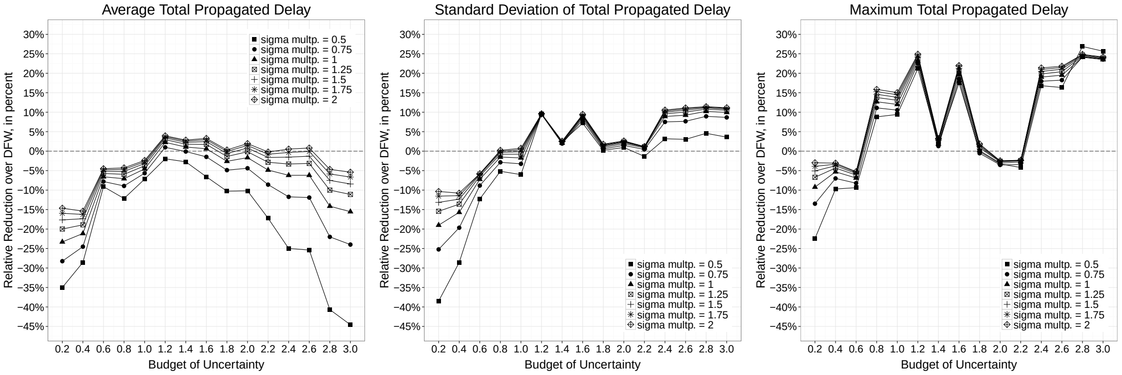


Figure A-6: Impact of Deviation in Standard Deviation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Log-normal Distribution / Testing Flight Network: $N_1$)

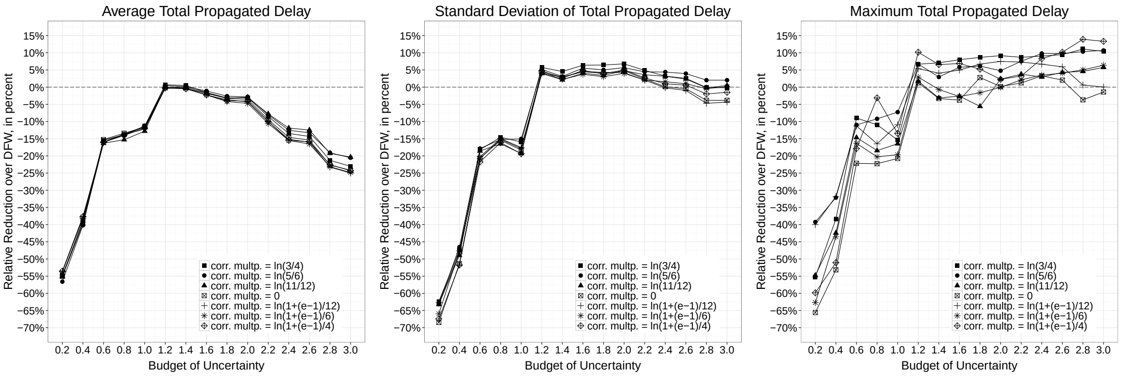


Figure A-7: Impact of Deviation in Correlation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Truncated Normal Distribution / Testing Flight Network: $N_1$)

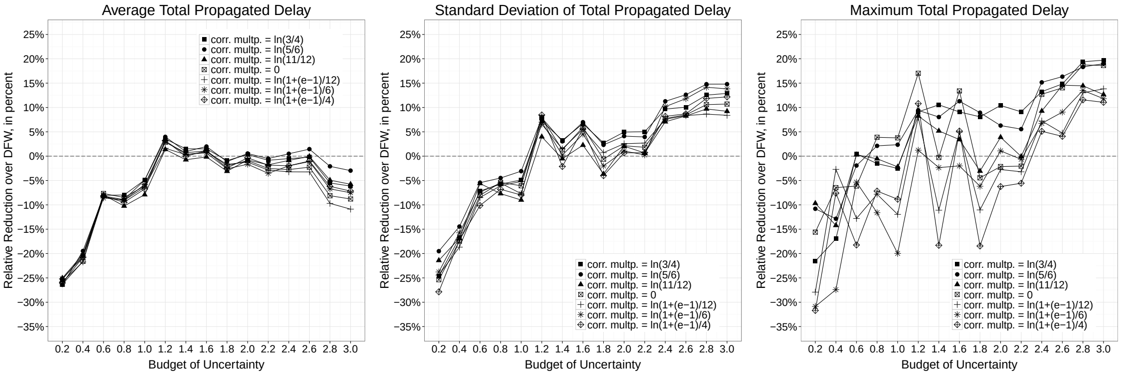


Figure A-8: Impact of Deviation in Correlation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Gamma Distribution / Testing Flight Network: $N_1$)

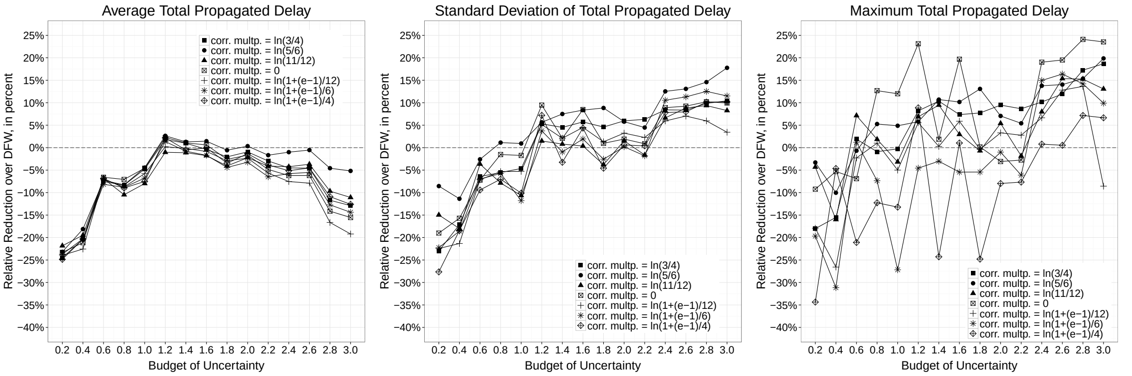


Figure A-9: Impact of Deviation in Correlation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Log-normal Distribution / Testing Flight Network: $N_1$)

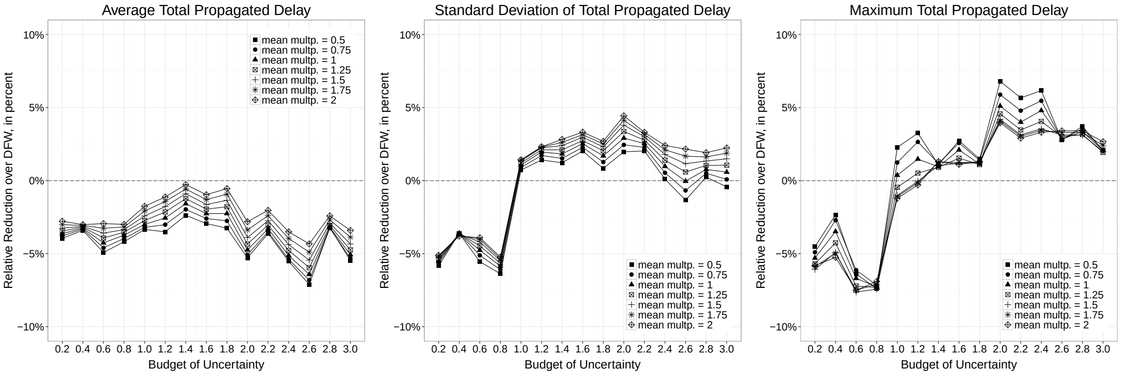


Figure A-10: Impact of Deviation in Mean (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Truncated Normal Distribution / Testing Flight Network: $N_2$)

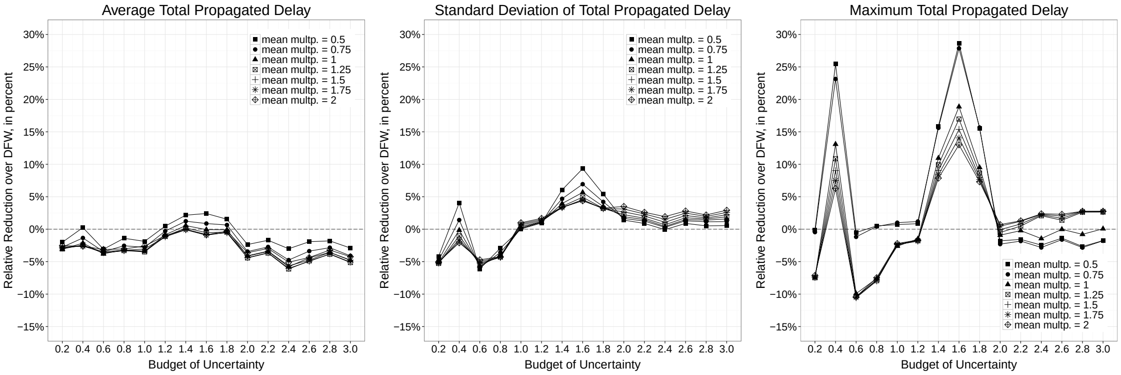


Figure A-11: Impact of Deviation in Mean (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Gamma Distribution / Testing Flight Network: $N_2$)

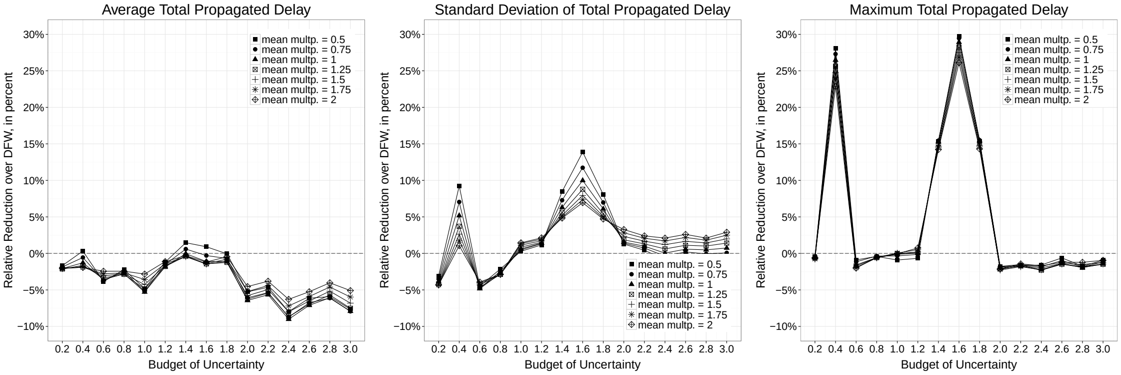


Figure A-12: Impact of Deviation in Mean (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Log-normal Distribution / Testing Flight Network: $N_2$)

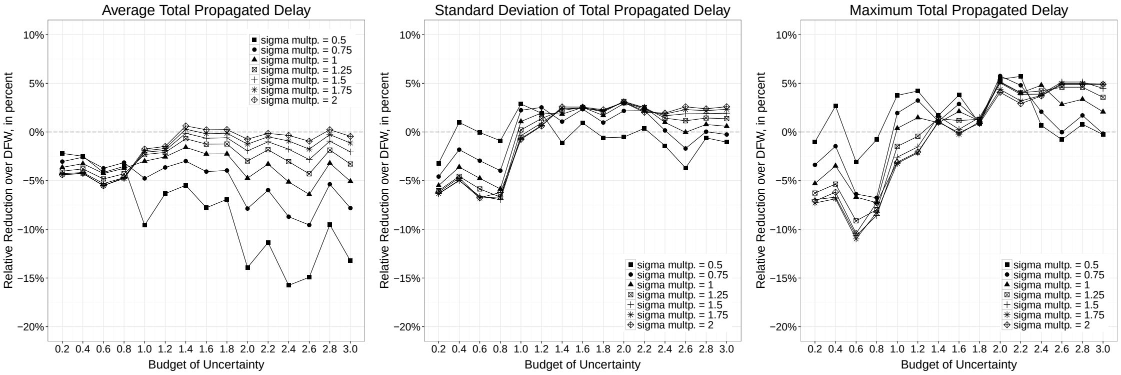


Figure A-13: Impact of Deviation in Standard Deviation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Truncated Normal Distribution / Testing Flight Network: $N_2$)

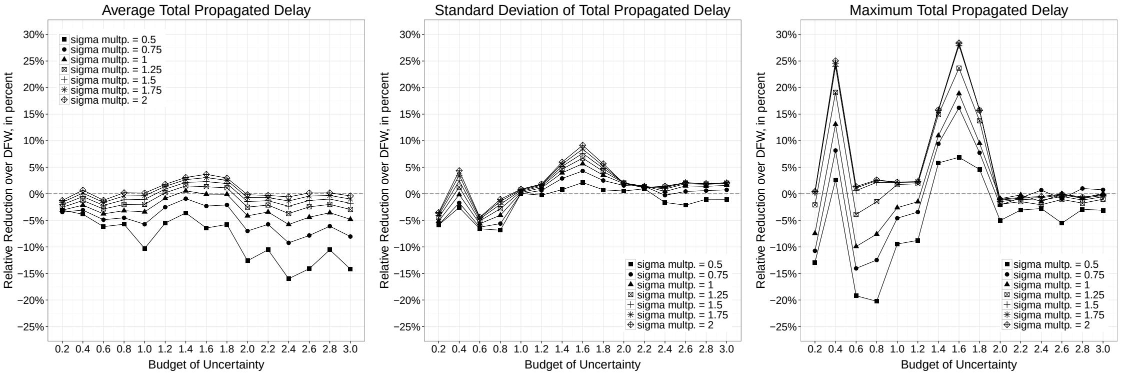


Figure A-14: Impact of Deviation in Standard Deviation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Gamma Distribution / Testing Flight Network: $N_2$)

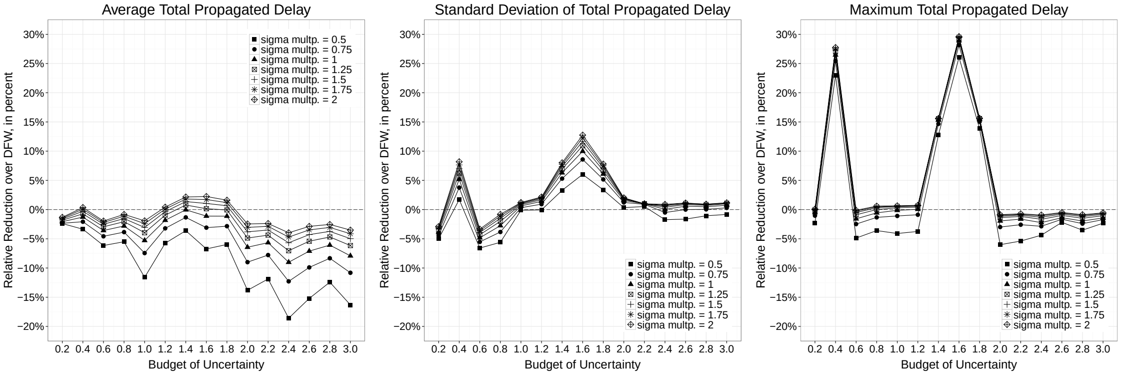


Figure A-15: Impact of Deviation in Standard Deviation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Log-normal Distribution / Testing Flight Network: $N_2$)

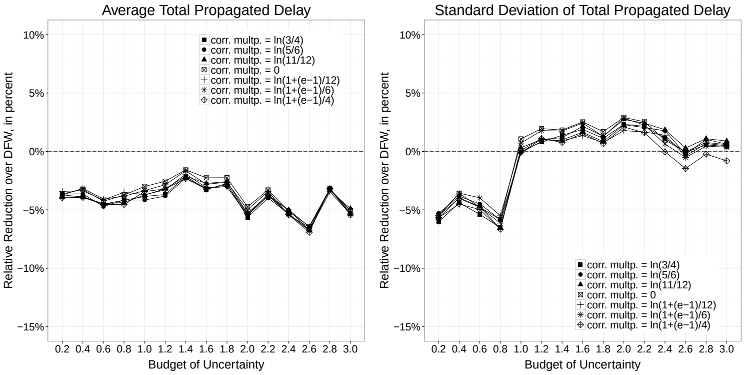


Figure A-16: Impact of Deviation in Correlation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Truncated Normal Distribution / Testing Flight Network: $N_2$)

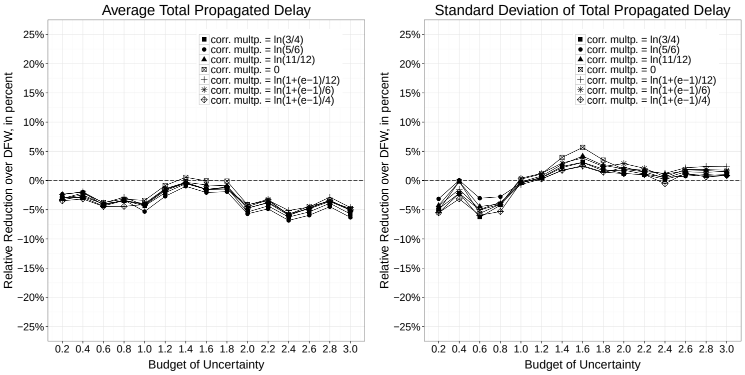


Figure A-17: Impact of Deviation in Correlation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Gamma Distribution / Testing Flight Network: $N_2$)

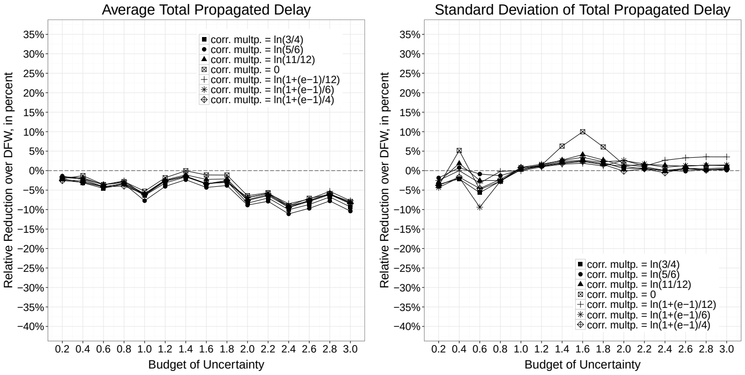


Figure A-18: Impact of Deviation in Correlation (Training Delay Data: Truncated Normal Distribution / Testing Delay Data: Log-normal Distribution / Testing Flight Network: $N_2$)

## A.2 Method to Perturb Spearman's Rank Correlation Coefficient Matrix

The method is inspired by Galeeva et al. (2007) where they provide a way to generate correlation matrices around a base correlation matrix by perturbing its eigenvalues. The detailed procedure we use is as follows,

1. Apply eigenvalue decomposition on the Spearman's rank correlation coefficient matrix $\rho^{\text{train}}$ for the training data set,

   $$\rho_{i,j}^{\text{train}} = \sum_{k,l=1}^{|\mathcal{F}|} V_{i,k}\Lambda_{k,l}V_{l,j},$$

   where $\Lambda_{k,l} = \lambda_k \delta_{k,l}$. $\delta_{k,l} = 1$ if $k = l$; 0, otherwise. $\lambda_1 > \lambda_2 > \cdots > \lambda_{|\mathcal{F}|}$ are the $|\mathcal{F}|$ eigenvalues for $\rho^{\text{train}}$. $V$ is the eigenvectors.

2. Create variables $\sigma_1, \sigma_2, \cdots, \sigma_{|\mathcal{F}|}$ for each eigenvalue, which satisfy the following set of equations

   $$\lambda_1 e^{\sigma_1} = \lambda_2 e^{\sigma_2} = \cdots = \lambda_{|\mathcal{F}|} e^{\sigma_{|\mathcal{F}|}}$$

   There are multiple solutions for the set of equations above. We fix $\sigma_1 = 1$, and the remaining values of $\sigma$ can be calculated accordingly. Since $\lambda_1$ is the largest eigenvalue, it can be seen easily that $\sigma_2, \cdots, \sigma_{|\mathcal{F}|} > 1$.

3. Perturb these eigenvalues with a parameter $\alpha$. The perturbed eigenvalues $\hat{\lambda}_1, \hat{\lambda}_2, \cdots, \hat{\lambda}_{|\mathcal{F}|}$ will be calculated as $\hat{\lambda}_i = \lambda_i e^{\sigma_i \alpha}$, $\forall i = 1, \cdots, |\mathcal{F}|$.

4. Normalize the perturbed eigenvalues to make $\sum_{i=1}^{|\mathcal{F}|} \hat{\lambda}_i = |\mathcal{F}|$.

5. Construct matrix $\rho'$ with the perturbed eigenvalues and the eigenvectors for $\rho_{i,j}^{\text{train}}$,

   $$\rho'_{i,j} = \sum_{k,l=1}^{|\mathcal{F}|} V_{i,k}\hat{\Lambda}_{k,l}V_{l,j},$$

   where $\hat{\Lambda}_{k,l} = \hat{\lambda}_k \delta_{k,l}$.

6. Construct Spearman's rank correlation coefficient matrix by normalizing matrix $\rho'$,

$$\rho_{i,j}^{\text{test}} = \frac{\rho'_{i,j}}{\sqrt{\rho'_{i,i}\rho'_{j,j}}}$$

The normalizing step makes sure that $-1 \leq \rho_{i,j}^{\text{test}} \leq 1$ and $\rho_{i,i}^{\text{test}} = 1$, $\forall i = 1, \cdots, |\mathcal{F}|$.

To understand the meaning of parameter $\alpha \in (-\infty, 1]$:

- if $\alpha = 1$, $\hat{\lambda}_1 = \hat{\lambda}_2 = \cdots = \hat{\lambda}_{|\mathcal{F}|} = 1$, thus $\hat{\Lambda} = I$ and $\rho' = V\hat{\Lambda}V^T = VV^T = VV^{-1} = I$. This leads to $\rho^{\text{test}} = I$, which means the testing data will have independent primary delays.
- if $\alpha = 0$, then $\hat{\lambda}_i = \lambda_i$, $\forall i = 1, \cdots, |\mathcal{F}|$. This means the testing data will have the same Spearman's rank correlation coefficient matrix as the training data.
- for the case $\alpha \to -\infty$, we have

$$\begin{aligned}
\rho_{i,j}^{\text{test}} =& \frac{\rho'_{i,j}}{\sqrt{\rho'_{i,i}\rho'_{j,j}}} \\
=& \frac{\sum_{k,l=1}^{|\mathcal{F}|} V_{i,k}\hat{\Lambda}_{k,l}V_{l,j}}{\sqrt{\sum_{k,l=1}^{|\mathcal{F}|} V_{i,k}\hat{\Lambda}_{k,l}V_{l,i}}\sqrt{\sum_{k,l=1}^{|\mathcal{F}|} V_{j,k}\hat{\Lambda}_{k,l}V_{l,j}}} \\
=& \frac{\sum_{k=1}^{|\mathcal{F}|} V_{i,k}\hat{\lambda}_k e^{\sigma_k \alpha}V_{k,j}}{\sqrt{\sum_{k=1}^{|\mathcal{F}|} V_{i,k}\hat{\lambda}_k e^{\sigma_k \alpha}V_{k,i}}\sqrt{\sum_{k=1}^{|\mathcal{F}|} V_{j,k}\hat{\lambda}_k e^{\sigma_k \alpha}V_{k,j}}}
\end{aligned}$$

  Divide both the numerator and denominator by $e^{\alpha}$, we have

$$\rho_{i,j}^{\text{test}} = \frac{V_{i,1}\hat{\lambda}_1 V_{1,j} + \sum_{k=2}^{|\mathcal{F}|} V_{i,k}\hat{\lambda}_k e^{(\sigma_k - 1)\alpha}V_{k,j}}{\sqrt{V_{i,1}\hat{\lambda}_1 V_{1,i} + \sum_{k=2}^{|\mathcal{F}|} V_{i,k}\hat{\lambda}_k e^{(\sigma_k - 1)\alpha}V_{k,i}}\sqrt{V_{j,1}\hat{\lambda}_1 V_{1,j} + \sum_{k=2}^{|\mathcal{F}|} V_{j,k}\hat{\lambda}_k e^{(\sigma_k - 1)\alpha}V_{k,j}}}$$

  Since $\sigma_2, \cdots, \sigma_{|\mathcal{F}|} > \sigma_1 = 1$, as $\alpha \to -\infty$ we have

$$\lim_{\alpha \to -\infty} \rho_{i,j}^{\text{test}} = \frac{V_{i,1}\hat{\lambda}_1 V_{1,j}}{|V_{i,1}|\hat{\lambda}_1|V_{1,j}|} = \frac{V_{i,1}V_{1,j}}{|V_{i,1}V_{1,j}|} = \begin{cases} 1, & V_{i,1}V_{1,j} > 0 \\ -1, & V_{i,1}V_{1,j} < 0 \end{cases}$$

  This means the testing data will be perfectly correlated. Whether it is positively or negatively correlated depends on the eigenvectors of $\rho^{\mathrm{train}}$.

# Appendix B

# Technical Results and Procedures for Chapter 3: Applying Majority Judgment over a Polyhedral Candidate Space

## B.1 Grade Function Estimation

In the remaining part of this Appendix, we suppress the voter index $i$. Suppose $\mathbf{m}^1, \mathbf{m}^2, \cdots, \mathbf{m}^k$ are $k$ candidates that we ask the voters to grade. $\mathbf{y} = \{y^1, y^2, \cdots, y^k\}$ are the corresponding grades submitted by a specific voter.

We utilize a nonparametric regression method called "convex regression" (concave in our setting) to estimate voters' grade functions. The idea of this estimation method is to come up with an estimator $\hat{g}(\cdot)$ which minimizes of the sum of squares: $\sum_{i=1}^{k}(\hat{g}(\mathbf{m}^i) - y^i)^2$ over functions $\hat{g}$ that are concave and component-wise non-decreasing in $\mathbf{m}$. Let $\mathbf{f} = \{f^1 = \hat{g}(\mathbf{m^1}), f^2 = \hat{g}(\mathbf{m^2}), \cdots, f^k = \hat{g}(\mathbf{m^k})\}$, respectively, be the grades estimated through this best-fit function for candidates $\mathbf{m}^1, \mathbf{m}^2, \cdots, \mathbf{m}^k$ respectively. The formulation for solving the convex regression is presented below

(Lim and Glynn, 2012; Mazumder et al., 2015):

$$\min_{\mathbf{f},\xi} \quad \|\mathbf{f}-\mathbf{y}\|_2^2 \tag{B.1}$$

$$\text{s.t.} \quad f_j \le f_i + \xi_i^T(\mathbf{m}^j - \mathbf{m}^i),\ i \ne j \in \{1,\cdots,k\} \tag{B.2}$$

$$\|\xi_i\|_2^2 \le C,\ i \in \{1,\cdots,k\} \tag{B.3}$$

$$\xi_i \ge \mathbf{0},\ i \in \{1,\cdots,k\} \tag{B.4}$$

Note that $\xi_1,\cdots,\xi_k$ are the subgradients of the best estimated convex function $\hat{g}(\cdot)$ at points $\mathbf{m}^1,\cdots,\mathbf{m}^k$ respectively. Constraints (B.2) represent the concavity requirement regarding the subgradient of a concave function. Constraints (B.3) represent the Lipschitz constraint to bound the $\mathcal{L}_2$ norm of the subgradients $\xi$ from above. These constraints are imposed to prevent overfitting and to ensure certain sublinear convergence rate. We refer interested readers to Mazumder et al. (2015) for details. In our implementation, we chose $C = 10$ which is an upper bound that we empirically found for all the grade functions generated using the method described in Appendix B.8. Finally, constraints (B.4) ensure that the best-fit function is component-wise non-decreasing. After obtaining the optimal solution $\mathbf{f}^*, \xi^*$ of the quadratic program (B.1)-(B.4), the estimated grade function $\hat{g}(\cdot)$ is specified as follows:

$$\begin{aligned}\hat{g}(\mathbf{m}) &= \min_{i\in\{1,\cdots,k\}} \left\{y_i + \left\langle \mathbf{m}-\mathbf{m}^i, \xi_i\right\rangle\right\} \\ &= \min_{i\in\{1,\cdots,k\}} \left\{\xi_i^T\mathbf{m} + \left(y_i - \xi_i^T\mathbf{m}^i\right)\right\}\end{aligned}$$

## B.2 Procedure for Random Generation of Candidates on the Efficient Frontier

We first generate a vector $\mathbf{c} \in [0,1]^n$ where each component $c_i$ is uniformly distributed between 0 and 1. We then solve the following linear program and denote the optimal solutions of this linear program as $\mathbf{m}^*$.

$$\max \quad \sum_{i=1}^{n} m_i \tag{B.5}$$

$$\text{s.t.} \quad c_j m_i = c_i m_j,\ \forall i, j = 1, \cdots, n \tag{B.6}$$

$$\mathbf{m} \in P \tag{B.7}$$

The linear program (B.5) - (B.7) essentially finds a candidate $\mathbf{m}^*$ by moving along the direction $(c_1, c_2, \cdots, c_n)$ as much as it can while staying within the feasible polyhedron $P$. However, the resultant candidate might not be on the efficient frontier. The next linear program further projects $\mathbf{m}^*$ onto the efficient frontier:

$$\max \quad \sum_{i=1}^{n} m_i \tag{B.8}$$

$$\text{s.t.} \quad m_i \geq m_i^*,\ \forall i = 1, \cdots, n \tag{B.9}$$

$$\mathbf{m} \in P \tag{B.10}$$

One can easily check that the optimal solution of this second linear program (B.8) - (B.10) is on the efficient frontier of $P$.

## B.3 Procedure for Random Perturbation of a Candidate on the Efficient Frontier

Let $\mathbf{m}$ be a candidate on the efficient frontier of $P$. For each component of $m_j$, we randomly perturb it to $m'_j = m_j + \epsilon_j$ where $\epsilon_j$ is a random variable. We restrict $m'_j$ such that $m'_j \in [\text{lb}, \text{ub}]$ where $[\text{lb}, \text{ub}]$ is a pre-determined interval, and we sample $m'_j$ with rejection to ensure that this restriction is satisfied. In our implementation in Section 2.4, we pick $[\text{lb}, \text{ub}] = [0, 1]$ and pick $\epsilon_1, \cdots, \epsilon_n$ to be a set of independent identically distributed normal random variables with mean equal to zero and standard deviation of 0.1. After calculating $\mathbf{m}'$ through sampling with rejection, we utilize the method provided in Appendix B.2 to get its projection onto the efficient frontier of

$P$. In other words, we use $\mathbf{m}'$ in place of vector $\mathbf{c}$ and run the two linear programs described by (B.5)-(B.7) and (B.8)-(B.10). The optimal solution of the second linear program (B.8)-(B.10) is the candidate obtained upon random perturbation.

## B.4 Procedure for Generation of Evenly Distributed Candidates on the Efficient Frontier

This Appendix describes the procedure used to generate evenly distributed candidates on the efficient frontier for the collaborative air traffic flow management case study (Section 3.5.1). The method is very similar to the one described in Appendix B.2. Using the sequence of two linear programs described by (B.5)-(B.7) and (B.8)-(B.10), we generate 10 candidates with:

$$\frac{c_1}{c_2} = \{\tan(5.0°), \tan(13.9°), \tan(22.8°), \tan(31.7°), \tan(40.6°),\\ \tan(49.4°), \tan(58.3°), \tan(67.2°), \tan(76.1°), \tan(85.0°)\}.$$

These are evenly distributed directions in the positive orthant ranging between angles of 5% and 85% with the X-axis.

## B.5 Proofs

### B.5.1 Proof of Corollary 3.1

Constraints (3.5) are redundant and constraints (3.7) can be equivalently relaxed to $v_i \leq g_i(\mathbf{m}), \forall i \in N$. Thus the following mixed-integer convex programming formulation MJ-OPT is equivalent to the formulation described in (3.2)-(3.10).

$$\begin{aligned} \text{(MJ-OPT)} \quad \max \quad & z \\ \text{s.t.} \quad & (3.3)-(3.4), (3.6) \\ & v_i \leq g_i(\mathbf{m}), \quad \forall i \in N \end{aligned}$$

$$(3.8) - (3.10)$$

*Proof.* To prove that constraints (3.5) are redundant, we are going to prove that for any optimal solution $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ to the formulation described by (3.2)-(3.4) and (3.6)-(3.10), which violates constraints (3.5), we can always modify the values of $\mathbf{y}$ and transform it into another optimal solution $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ which satisfies constraints (3.5). That is, for any such $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$, we can always find an optimal solution to the formulation described by (3.2)-(3.10) by only changing the $\mathbf{y}$ values. For any such $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ that doesn't satisfy constraints (3.5), there is some voter $i_0$ with $y^*_{i_0} = 1$ and $x^*_{i_0} = 0$. Let $j_0 = \arg\min_{i \in N: x^*_i = 1} v^*_i$. We construct a new solution with $\mathbf{y}^{**}$ such that $y^{**}_{j_0} = 1$ and $y^{**}_j = 0, \forall j \in N \setminus \{j_0\}$. We claim that $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ is also an optimal solution to formulation (3.2)-(3.4) and (3.6)-(3.10) and satisfies $y^{**}_{j_0} = 1$, $x^{**}_{j_0} = 1$. To see that, we only need to prove that $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ is feasible because they have the same objective function values $z^*$. We can verify that constraints (3.3),(3.4),(3.6),(3.7), and (3.10) are satisfied. Constraints (3.8) are satisfied because the way we select $j_0$ ensures that $v^*_{j_0} \leq v^*_i, \forall i \in N : x^*_i = 1$. Constraints (3.9) are also satisfied because of the fact $z^* \leq v^*_{i_0} \leq v^*_{j_0}$. The first inequality holds because $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ satisfies constraints (3.9) and $y^*_{i_0} = 1$. The second inequality holds because $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ satisfies constraints (3.8) and $x^*_{j_0} = 1$, $y^*_{i_0} = 1$. We can repeat this process for all such $i_0$ with $y^*_{i_0} = 1$ and $x^*_{i_0} = 0$ one by one, to arrive at a solution that satisfies constraints (3.5). This proves that constraints (3.5) are redundant.

To prove that equality constraints (3.7) can be relaxed to less-than-or-equal-to constraints, we follow the same proof technique as the one used in the first part of this proof. So we are going to show that for any optimal solution $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ of MJ-OPT which violates constraints (3.7), we can always modify the values of $\mathbf{v}$ and $\mathbf{y}$ to transform it into another optimal solution $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^{**}, \mathbf{m}^*, z^*\}$ of MJ-OPT which satisfies constraints (3.7). We start with an optimal solution of MJ-OPT which doesn't satisfy constraints (3.7), i.e., there exists some $i \in N$ such that $v^*_i < g_i(\mathbf{m}^*)$.

We next divide our discussion into two different cases:

1. If $y_i^* = 0$: Replace $v_i^*$ with $v_i^{**} = g_i(\mathbf{m}^*)$. The new solution is still feasible and thus optimal for MJ-OPT. This is because constraints (3.8) and (3.9) are redundant at index $i$, thus have no restriction on increasing the value of $v_i$.

2. If $y_i^* = 1$: Let $j_0 = \arg\min_{j\in N\setminus\{i\}:x_j^*=1} v_j^*$. Then replace $v_i^*$ with $v_i^{**} = g_i(\mathbf{m}^*)$ and replace $\mathbf{y}^*$ with $\mathbf{y}^{**}$ such that $y_{j_0}^{**} = 1$ and $y_j^{**} = 0, \forall j \in N \setminus \{j_0\}$. Now, we only need to prove that this the new solution is still feasible thus optimal for MJ-OPT. We first claim that $v_{j_0}^* = v_i^*$. To see that, suppose $v_{j_0}^* > v_i^*$ instead (note that $v_i^* \leq v_{j_0}^*$ because of constraints (3.8) and knowing that $y_i^* = 1$, and $x_{j_0}^* = 1$). Replacing $v_i^*$ with $v_i^{'} = \min\{g_i(\mathbf{m}^*), v_{j_0}^*\}$, and adjusting the $y^*$ variables accordingly, the resultant solution is still feasible but with a higher objective value equal to $z^* + (v_i^{'} - v_i^*)$ because $v_i^{'} = \min\{g_i(\mathbf{m}^*), v_{j_0}^*\} > v_i^*$. This reaches a contradiction with the assumption that $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ is an optimal solution. So we have proved that $v_{j_0}^* = v_i^*$. Now consider the new solution $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^{**}, \mathbf{m}^*, z^*\}$ constructed. We can easily verify that it satisfies constraints (3.3), (3.4), (3.5), (3.6), and (3.10). Constraints (3.8) are satisfied because $v_{j_0}^{**} = v_{j_0}^* = v_i^* \leq v_j^*, \forall j \in N : x_j^* = 1$. Constraints (3.9) are also satisfied because $z^* = v_i^* = v_{j_0}^* = v_{j_0}^{**}$, where the first equality holds because of the optimality of $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$.

In summary, for any optimal solution $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ of MJ-OPT, we can transform it to another optimal solution $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^{**}, \mathbf{m}^*, z^*\}$ for MJ-OPT, which satisfies constraints (3.5) and (3.7). This finishes the proof of equivalence between MJ-OPT and formulation described by (3.2)-(3.10). □

### B.5.2 Proof of Theorem 3.2

Let $\bar{P}$ be the efficient frontier of the candidate space $P$ and $\mathbf{m}'$ be the candidate we get after randomly perturbing $\mathbf{m}$. If $\forall \mathbf{m} \in \bar{P}$ and for each subset $A \subset \bar{P}$, we have $Pr(\mathbf{m}' \in A) > 0$, then Algorithm 3 converges to the true majority winner almost surely as the total number of iterations $n_{\max} \to \infty$.

*Proof.* For the rest of this proof, we suppress the voter index $i$ and denote a specific voter's true grade function as $g(\cdot)$ and its estimated grade function after the termination of Algorithm 3 with parameter $n_{\max}$ as $\hat{g}_{n_{\max}}(\cdot)$.

Let $\bar{P}$ be the efficient frontier of candidate space $P$. We are first going to show that $\forall \mathbf{m} \in \bar{P}$, we have $\hat{g}_{n_{\max}}(\mathbf{m}) \to g(\mathbf{m})$ almost surely as the total number of iterations $n_{\max} \to \infty$. We follow the notation of the estimation problem in Appendix B.1. Since here we assume that voters grade *exactly* according to their grade functions (no grading errors or strategic behaviors), we can see that the optimal objective function value of formulation (B.1) - (B.4) is always zero. This is because $\mathbf{f} = \mathbf{y}$ is always feasible thus optimal for formulation (B.1) - (B.4). This means that for any graded candidate $\mathbf{m}^k$, $\hat{g}_{n_{\max}}(\mathbf{m}^k) = g(\mathbf{m}^k)$. Since the random perturbation method ensures that any $\mathbf{m} \in \bar{P}$ has a positive probability of being sampled, we thus have $\hat{g}_{n_{\max}}(\mathbf{m}) \to g(\mathbf{m})$ as $n_{\max} \to \infty, \forall \mathbf{m} \in \bar{P}$ almost surely.

Denote $\hat{g}_{\infty}(\cdot)$ as the limit function of $\hat{g}_{n_{\max}}(\cdot)$ as $n_{\max} \to \infty$. We are going to show that the true majority grades of the optimal candidates from MJ-OPT using $g(\cdot)$ and $\hat{g}_{\infty}(\cdot)$ as the grade functions for each specific voter are the same, i.e. the optimal candidate from MJ-OPT using $\hat{g}_{\infty}(\cdot)$ is the true majority winner. Firstly, we can see that there always exists an optimal solution of MJ-OPT using $\hat{g}_{\infty}(\cdot)$ which is on the efficient frontier $\bar{P}$. This is because $\hat{g}_{\infty}$ is a component-wise non-decreasing function as we discussed in Appendix B.1, and as a result, for any solution which is not on the efficient frontier, we can always find a candidate on the efficient frontier which has higher or equal majority grade. We already proved that $\hat{g}_{n_{\max}}(\mathbf{m}) \to g(\mathbf{m})$ as $n_{\max} \to \infty, \forall \mathbf{m} \in \bar{P}$ almost surely. Therefore, for any optimal solution $\mathbf{m}^* \in \bar{P}$ of MJ-OPT using $\hat{g}_{\infty}$, it must also be an optimal solution of MJ-OPT using $g(\cdot)$.

Since the optimal solution of MJ-OPT using $\hat{g}_{n_{\max}}(\cdot)$ will always be asked to be graded in the last iteration, we thus prove that Algorithm 3 will converge to the true majority winner almost surely. □

### B.5.3 Proof of Theorem 3.3

Under Assumptions 3.1, 3.2 and 3.3, the lower bound grade functions are the following piecewise-linear functions:

$$\tilde{g}_i(\mathbf{m}) = \min_{j \in \{1,\cdots,n\}:[\mathbf{m}_i^*]_j > 0} \left(\frac{1}{[\mathbf{m}_i^*]_j}\right) m_j, \quad \forall i \in N,$$

where $[\mathbf{m}_i^*]_j$ denotes the $j^{\text{th}}$ element of voter $i$'s most preferred candidate $\mathbf{m}_i^*$.

*Proof.* For the rest of the proof, we suppress the voter index $i$ and denote a specific voter's most preferred candidate by $\mathbf{m}^*$ and the claimed lower bound grade function as $\tilde{g}(\mathbf{m}) = \min_{j \in \{1,\cdots,n\}:m_j^* > 0} \left(1/m_j^*\right) m_j$. We are first going to prove that $\tilde{g}(\cdot)$ belongs to the set of valid grade functions: (1) $\tilde{g}(\mathbf{m}^*) = \min_{j \in \{1,\cdots,n\}:m_j^* > 0} \left(1/m_j^*\right) m_j^* = 1$, (2) $\tilde{g}(\cdot)$ is continuous and concave because it is the minimum function of several linear functions, (3) $\tilde{g}(\cdot)$ is component-wise non-decreasing because all these linear functions have positive coefficients, (4) $\tilde{g}(\mathbf{m}) \geq 0, \forall \mathbf{m} \in P$ because $P$ is in the positive orthant, and finally (5) $\tilde{g}(\mathbf{m}) \leq 1, \forall \mathbf{m} \in P$ because $\mathbf{m}^*$ is on the efficient frontier, and hence $\forall \mathbf{m} \in P$ there exists $i \in \{1,\cdots,n\}$ such that $m_i \leq m_i^*$.

Now we only need to prove that $\tilde{g}(\mathbf{m}) = \min_{j \in \{1,\cdots,n\}:m_j^* > 0} \left(1/m_j^*\right) m_j$ is the lower bound function according to Definition 3.3. We prove the theorem by contradiction. Suppose that there exists a grade function $g'(\cdot)$ belonging to the set of valid grade functions according to Definition 3.2, and there exists $\mathbf{m^1} \in P$ such that $g'(\mathbf{m^1}) < \tilde{g}(\mathbf{m^1})$. We denote

$$i = \arg\min_{j \in \{1,\cdots,n\}:m_j^* > 0} \left(\frac{1}{m_j^*}\right) m_j^1. \tag{B.11}$$

Now consider $\mathbf{m^2} = \left(m_i^1/m_i^*\right) \mathbf{m}^*$. We claim that $m_j^1 \geq m_j^2,\ j \in \{1,\cdots,n\}$. To see that, we have

$$\frac{m_j^1}{m_j^2} = \frac{m_j^1}{m_j^*}\frac{m_i^*}{m_i^1} \geq 1,\ \forall j \in \{1,\cdots,n\} : m_j^* > 0 \tag{B.12}$$

The first equality in (B.12) holds because of the way we construct $\mathbf{m}^2$. The inequality in (B.12) holds because of Eq. (B.11), where we have $(1/m_i^*)\, m_i^1 \leq \left(1/m_j^*\right) m_j^1$, $\forall j \in \{1, \cdots, n\} : m_j^* > 0$. On the other hand, $\forall j \in \{1, \cdots, n\} : m_j^* = 0$, we have $m_j^2 = 0$. Thus, we must have $m_j^1 \geq m_j^2 = 0$, $\forall j \in \{1, \cdots, n\} : m_j^* = 0$. We now have $m_j^1 \geq m_j^2$, $\forall j \in \{1, \cdots, n\}$. Since $m_i^1 = m_i^2$, we have

$$\tilde{g}(\mathbf{m}^1) = \tilde{g}(\mathbf{m}^2) = \left(\frac{1}{m_i^*}\right) m_i^1. \tag{B.13}$$

Since grade function $g'(\cdot)$ also satisfies Assumption 3.2, we know that $g'(\mathbf{m})$ is component-wise non-decreasing in $\mathbf{m}$. We thus have $g'(\mathbf{m^1}) \geq g'(\mathbf{m^2})$. This gives us

$$g'(\mathbf{m^2}) \leq g'(\mathbf{m^1}) < \tilde{g}(\mathbf{m^1}) = \tilde{g}(\mathbf{m^2}) \tag{B.14}$$

Since $\mathbf{m^2} = \big((m_i^* - m_i^1)/m_i^*\big)\mathbf{o} + (m_i^1/m_i^*)\mathbf{m}^*$, where $\mathbf{o}$ is the origin point which belongs to $P$ according to Assumption 3.3, and because $g'(\cdot)$ is concave, we have

$$\begin{aligned} g'(\mathbf{m^2}) &\geq \frac{m_i^* - m_i^1}{m_i^*} g'(\mathbf{o}) + \frac{m_i^1}{m_i^*} g'(\mathbf{m}^*) && \text{(B.15)} \\ &\geq \frac{m_i^1}{m_i^*} g'(\mathbf{m}^*) && \text{(B.16)} \\ &= \frac{m_i^1}{m_i^*} && \text{(B.17)} \\ &= \tilde{g}(\mathbf{m^2}) && \text{(B.18)} \end{aligned}$$

Inequality (B.15) holds because of concavity of $g'(\cdot)$. Inequality (B.16) holds because $g'(\mathbf{o}) \geq 0$ and $m_i^* \geq m_i^1$. To see why $m_i^* \geq m_i^1$, note that if $m_i^* < m_i^1$ then it contradicts with $m^*$ being on the efficient frontier. Equality (B.17) holds because $g'(\mathbf{m}^*) = 1$ according to Assumption 3.2. Finally equality (B.18) holds because of Eq. (B.13). We thus reach a contradiction with Eq. (B.14). This completes our proof. □

## B.6 Best Response Problem for Voter Grading

The best response problem for a specific voter $i$ is stated as follows: given a set of candidates and the other voters' grades for each candidate, how to decide voter $i$'s submitted grades so as to maximize its payoff. We develop a mixed-integer linear program to solve this problem.

Let $\mathbf{m}^1, \cdots, \mathbf{m}^k$ be the $k$ candidates to be graded by voter $i$. $y_j$ is a binary variable taking value 1 if vector $\mathbf{m}^j$ is the majority winner, 0 otherwise.

The mathematical formulation orders the voter grades for each candidate in a non-decreasing order from left to right, and designates one particular voter as the *majority grade determining voter* of each candidate. This majority grade determining voter for each candidate is chosen such that all voters to the right including this voter constitute a *minimal majority set*, i.e., if you remove any voter from this set, the rest no longer constitute a majority set. $l_{i,j}, r_{i,j}, e_{i,j}$ are all binary variables where $l_{i,j}$ takes value 1 if voter $i$ is the same as or to the left of the majority grade determining voter for candidate $\mathbf{m}^j$; $r_{i,j}$ takes value 1 if voter $i$ is the same as or to the right of the majority grade determining voter for candidate $\mathbf{m}^j$; and finally, $e_{i,j}$ takes value 1 if voter $i$ is the same as the majority grade determining voter for candidate $\mathbf{m}^j$. Continuous variable $\mu_j$ denotes the majority grade of candidate $\mathbf{m}^j$. $v_{i,j}$ is the grade that voter $i$ submits for candidate $\mathbf{m}^j$. Note that $\forall i' \in N, i' \neq i$, $v_{i',j}$ equals to $g_{i'}(\mathbf{m}^j)$, the truthful grade of voter $i'$ for candidate $\mathbf{m}^j$, because we assume that the other voters grade truthfully. The mixed-integer linear programming formulation is as follows:

$$\max \quad \sum_{j=1}^{k} g_i(\mathbf{m}^j) y_j \tag{B.19}$$

$$\text{s.t.} \quad v_{i,j} \leq \mu_j + (1 - l_{i,j}) G^{\max}, \qquad \forall i \in N, j \in \{1, \cdots, k\} \tag{B.20}$$

$$v_{i,j} \geq \mu_j - (1 - r_{i,j}) G^{\max}, \qquad \forall i \in N, j \in \{1, \cdots, k\} \tag{B.21}$$

$$l_{i,j} + r_{i,j} - e_{i,j} = 1, \qquad \forall i \in N, j \in \{1, \cdots, k\} \tag{B.22}$$

$$\sum_{i \in N} w_i \cdot r_{i,j} \geq \frac{\sum_{i \in N} w_i}{2} + \epsilon_0, \quad j \in \{1, \cdots, k\} \tag{B.23}$$

$$\sum_{i \in N} w_i \cdot l_{i,j} \geq \frac{\sum_{i \in N} w_i}{2}, \qquad j \in \{1, \cdots, k\} \tag{B.24}$$

$$\sum_{i \in N} e_{i,j} = 1, \qquad j \in \{1, \cdots, k\} \tag{B.25}$$

$$\mu_{j'} \leq \mu_j + (1 - y_j) G^{\max}, \qquad j, j' \in \{1, \cdots, k\} : j \neq j' \tag{B.26}$$

$$\sum_{j=1}^{k} y_j = 1 \tag{B.27}$$

$$v_{i',j} = g_{i'}(\mathbf{m}^j), \qquad \forall i' \in N : i' \neq i, j \in \{1, \cdots, k\} \tag{B.28}$$

$$0 \leq v_{i,j}, \mu_j \leq 1, \qquad \forall i \in N, j \in \{1, \cdots, k\} \tag{B.29}$$

$$l_{i,j}, r_{i,j}, e_{i,j}, y_j \in \{0, 1\}, \qquad \forall i \in N, j \in \{1, \cdots, k\} \tag{B.30}$$

where $\epsilon_0$ can be set to any positive number smaller than

$$\min_{\left\{\mathbf{x} \in \{0,1\}^n : \ \sum_{i=1}^{n} w_i x_i > (\sum_{i=1}^{n} w_i)/2\right\}} \left( \sum_{i=1}^{n} w_i x_i - \frac{\sum_{i=1}^{n} w_i}{2} \right)$$

which is the smallest positive difference between $\sum_{i=1}^{n} w_i x_i$ and $(\sum_{i=1}^{n} w_i)/2$.

Objective (B.19) maximizes voter $i$'s payoff as measured by its truthful grade of the majority winner. Constraints (B.20) ensure that if $l_{i,j} = 1$, then the grade of voter $i$ for candidate $\mathbf{m}^j$ is less than or equal to its majority grade $\mu_j$; similarly, constraints (B.21) ensure that if $r_{i,j} = 1$, then the grade of voter $i$ for candidate $\mathbf{m}^j$ is greater than or equal to its majority grade $\mu_j$. Constraints (B.22) state that voter $i$ is the majority grade determining voter for candidate $\mathbf{m}^j$ if and only if $l_{i,j} = r_{i,j} = 1$. Note that, combined with constraints (B.20) and (B.21), this also ensures that $\mu_j = v_{i,j}$ if $l_{i,j} = r_{i,j} = e_{i,j} = 1$. Constraints (B.23) and (B.24) ensure that the majority grade $\mu_j$ of a candidate $\mathbf{m}^j$ is calculated correctly. Constraints (B.23) ensure that the total weight of voters that are the same as or to the left of the majority grade determining voter for candidate $\mathbf{m}^j$ should exceed half of the total weight. Similarly, constraints (B.24) ensure that the total weight of voters that are the same as or to the left of the majority grade determining voter for candidate $\mathbf{m}^j$ should be at least equal to half the total weight. These two rules are imposed according to the discussion in Section 3.1

on the majority grade calculation. Constraints (B.25) ensure that the majority grade for each candidate $\mathbf{m}^j$ is determined by exactly one voter. Constraints (B.26) ensure that if candidate $\mathbf{m}^j$ is the majority winner, then it has the highest majority grade among all candidates. Constraints (B.27) require that there is only one majority winner. Note that we don't consider ties here because continuous grades make ties very unlikely. However, in theory, a tie could happen. In such a case, this formulation will always choose the winner to be the candidate that is most preferred by voter $i$ among all the candidates tied for the highest majority grade. This is not consistent with the tie-breaking rules in MJ. However, for simplicity and due to the rareness of ties, we don't model the tie breaking rules here and leave that as future work. Finally, constraints (B.28) make sure that voters other than $i$ are assumed to grade truthfully according to their true grade functions.

Formulation (B.19) - (B.30) solves the problem of $\max_{\mathbf{0} \leq \mathbf{v}_i \leq \mathbf{1}} \ p_i(\mathbf{v}_i; \mathbf{v}_{-i}^{\text{true}})$ required in Eq. (3.16). For problem $\max_{0 \leq v_{i,l+1} \leq 1} \ p_i(v_{i,l+1}; \mathbf{v}_{-i,l+1}^{\text{true}}, \mathbf{v}^{\text{true}})$ in Eq. (3.17) we can essentially use the same formulation with number of candidates equal to $l+1$ and a new constraint given by:

$$v_{i,j} = g_i(\mathbf{m}^j), \ j \in \{1, \cdots, l\} \tag{B.31}$$

## B.7 Feasible Candidate Space in the Collaborative Air Traffic Flow Management Case Study in Section 3.5.1

We utilize metrics for capacity-utilization and predictability derived for a single airport by Liu and Hansen (2013), assuming a constant scheduled arrival demand rate, $\lambda$. When the GDP is initiated, the airport arrival rate (AAR), defined as the maximum number of arrivals an airport can accept during one hour, is reduced from a known constant high level, $C_H$, which is assumed to be greater than $\lambda$, to a known constant low level, $C_L$, which is lower than $\lambda$. The planned duration of the GDP is $T$, at the end of which the AAR is expected to return to $C_H$. However, because of errors in prediction, the AAR may return to $C_H$ at a different time, $\tau$. When the GDP is initiated, $T$ is set but $\tau$ is unknown, and $\tau$ is assumed to be uniformly distributed between $t_{\min}$ and $t_{\max}$. Conceptually, if $T$ is set close to $t_{\min}$, $\tau$ is likely to be larger than $T$, and the GDP is likely to end late. In this case, capacity will be more highly utilized but there will be less predictability. Alternatively, if $T$ is set close to $t_{\max}$, then $\tau$ is likely to be smaller than $T$, and the GDP is likely to end early. In this case, capacity will be underutilized but the predictability will be high. For this simplified GDP, the only design parameter is $T$, the planned duration of the GDP. Mathematical representations for capacity-utilization and predictability, as developed by Liu and Hansen (2013) are presented below.

Capacity-utilization, $\alpha_c$, is defined as the ratio of realized throughput, from the beginning of the GDP until the time when there is no more delay, to the maximum throughput possible with perfect information. The perfect information case is the one where the airlines are able to take full advantage of the increase in AAR at time $\tau$. The value of $\alpha_c$ varies from 0 to 1, and is shown by Liu and Hansen (2013) to have the expected value shown below, which we use to define our capacity-utilization metric, $m_1$:

$$m_1 = E[\alpha_c] = \frac{t_{\max} - T}{t_{\max} - t_{\min}} + \frac{a/c}{t_{\max} - t_{\min}} \cdot \log \frac{b + cT}{b + ct_{\min}}$$

where $a = \lambda\big((C_H - C_L)/(C_H - \lambda)\big)T, b = C_H\big((C_H - C_L)/(C_H - \lambda)\big)T$ and $c = C_H - C_L$.

Predictability, $\alpha_p$, is defined as the ratio of total flight delay, assuming that the GDP ends at the planned time $T$, to the total realized flight delay, i.e., the delay that is actually incurred given the early or late increase in AAR at $\tau$. Again, $\alpha_p$ varies from 0 to 1, and is shown by Liu and Hansen (2013) to have the expected value shown below, which we use to define our predictability metric, $m_2$:

$$m_2 = E[\alpha_p] = \frac{1}{t_{\max} - t_{\min}} \cdot \left( -\frac{T^2}{t_{\max}} + 2T - t_{\min} \right)$$

For the collaborative air traffic flow management case study at SFO, we use $\lambda =$

40/hour, $C_L = 30$/hour, $C_H = 60$/hour, $t_{\min} = 2$ hours, and $t_{\max} = 6$ hours. Based on our data analysis, these values represent typical GDP parameters at SFO. We generate 101 pairs of $(m_1, m_2)$ by varying $T$ from 2 hours to 6 hours in increments of 0.04 hours. These 101 pairs of points along with the origin point are then passed to a MATLAB-based package – Qhull (Barber et al., 1996) for generating a polyhedron which represents the convex hull of these points. This polyhedron is our feasible candidate space.

## B.8 Specification and Generation of True Grade Functions

In this section, we present the specifications and the process of generation of the underlying true grade functions that we use in the computational experiments in Sections 3.5.1 and 3.5.2. To simplify notation, we will suppress the voter index $i$ in the remaining part of this Appendix. Without loss of generality, the individual components of the candidate vectors, $m_j$, $\forall j \in \{1, \cdots, n\}$ are normalized to have allowable values in $[0, 1]$. Recall that $n$ here is the dimension of the feasible space. In the capital budgeting example (Section 3.5.2), $n = 10$; while in the collaborative air traffic flow management example (Section 3.5.1), $n = 2$.

The grade function $g(\mathbf{m})$ for a specific voter is composed of component-wise value functions $v_j(m_j)$ of each component $m_j$. We define the overall value function $V(\mathbf{m})$, which combines the component-wise value functions as a multiplicative-multilinear function of $v_j(m_j)$'s, and models complementarities and substitutions among the different components. This functional form is based on well-accepted notions developed by economists and marketing researchers in the fields of choice modeling and multi-attribute valuation (Meyer and Johnson, 1995):

$$V(\mathbf{m}) = \sum_{j=1}^{n} r_j v_j(m_j) + \sum_{1 \le j < k \le n} r_{jk} v_j(m_j) v_k(m_k), \tag{B.32}$$

Coefficients $r_j$ here are non-negative. For pairwise interaction coefficients $r_{jk}$, if $r_{jk} >$

0, then it means that components $j$ and $k$ are complements; on the other hand, if $r_{jk} < 0$, then it means that components $j$ and $k$ are substitutes. Finally, the normalization step converts the overall value into a grade, using a simple linear scaling based on the maximum value $V^{\max} = \max_{\mathbf{m}\in P} V(\mathbf{m})$. Thus the grade function for the voter evaluated at candidate $\mathbf{m}$ is specified as:

$$g(\mathbf{m}) = \frac{V(\mathbf{m})}{V^{\max}} \tag{B.33}$$

A quadratic form without an intercept is specified for each component-wise value function, $v_j(m_j) = a_j m_j^2 + b_j m_j$, which was also used by Evans et al. (2016) in the collaborative air traffic flow management context. We require $v_j(m_j)$ to be concave and non-decreasing; and without loss of generality, we also require $v_j(m_j) \in [0,1]$ for all $m_j \in [0,1]$. Therefore, the values of $a_j$ and $b_j$ need to be constrained such that $-1 \leq a_j \leq 0$ and $-2a_j \leq b_j \leq 1 - a_j$. Moreover, to ensure global concavity of the overall grade function $g(\mathbf{m})$, further constraints need to be imposed on the coefficients. We first expand the grade function $g(\mathbf{m})$ according to (B.32) and (B.33) as follows, where $K_j^a, K_j^b, K_{jk}^{aa}, K_{jk}^{ab}, K_{jk}^{ba}, K_{jk}^{bb}$ are some constants that can be derived from $a_j, b_j, r_j, r_{jk}$.

$$g(\mathbf{m}) = \sum_{j=1}^{n} K_j^b m_j + \sum_{j=1}^{n} K_j^a m_j^2 + \sum_{1\leq j<k\leq n} \left( K_{jk}^{bb} m_j m_k + K_{jk}^{ab} m_j^2 m_k + K_{jk}^{ba} m_j m_k^2 + K_{jk}^{aa} m_j^2 m_k^2 \right),$$

where

$$K_j^a = \frac{r_j a_j}{V^{\max}}, K_j^b = \frac{r_j b_j}{V^{\max}}, K_{jk}^{aa} = \frac{r_{jk} a_j a_k}{V^{\max}}, K_{jk}^{ab} = \frac{r_{jk} a_j b_k}{V^{\max}}, K_{jk}^{ba} = \frac{r_{jk} b_j a_k}{V^{\max}}, K_{jk}^{bb} = \frac{r_{jk} b_j b_k}{V^{\max}} \tag{B.34}$$

Note that the component-wise non-decreasing condition on $g(\mathbf{m})$ is equivalent to $\nabla g(m) \geq \mathbf{0}$. Also, the Hessian matrix of a function being negative semi-definite in a given region is a necessary and sufficient condition for the concavity of the function within that region. Let the Hessian matrix of the grade function be:

$$\mathbf{H_g} = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{12} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ g_{1n} & g_{2n} & \cdots & g_{nn} \end{bmatrix}$$

where $g_{jk}$ is the partial derivative of $g(\mathbf{m})$ with respect to $m_j$ and $m_k$. The first order partial derivatives, $g_j$, $j \in (1, 2, \cdots, n)$ are:

$$\begin{aligned} g_j = \frac{\partial g(\mathbf{m})}{\partial m_j} =& K_j^b + 2K_j^a m_j + \sum_{k:k<j} \left( K_{kj}^{bb} m_k + K_{kj}^{ab} m_k^2 + 2K_{kj}^{ba} m_k m_j + 2K_{kj}^{aa} m_k^2 m_j \right) \\ &+ \sum_{k:j<k} \left( K_{jk}^{bb} m_k + 2K_{jk}^{ab} m_k m_j + K_{jk}^{ba} m_k^2 + 2K_{jk}^{aa} m_k^2 m_j \right) \end{aligned}$$

The second-order partial derivatives ($\forall j = 1, \cdots, n; 1 \le j < k \le n$) are:

$$\begin{aligned} g_{jj} = \frac{\partial g_j}{\partial m_j} =& 2 \left( K_j^a + \sum_{k:k<j} \left( K_{kj}^{ba} m_k + K_{kj}^{aa} m_k^2 \right) + \sum_{k:j<k} \left( K_{jk}^{ab} m_k + K_{jk}^{aa} m_k^2 \right) \right) \\ g_{jk} = \frac{\partial g_j}{\partial m_k} =& K_{jk}^{bb} + 2K_{jk}^{ab} m_j + 2K_{jk}^{ba} m_k + 4K_{jk}^{aa} m_k m_j \end{aligned}$$

We then have,

$$\begin{aligned} \mathbf{m^T H_g m} &= \begin{bmatrix} m_1 & m_2 & \cdots & m_n \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{12} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ g_{1n} & g_{2n} & \cdots & g_{nn} \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_n \end{bmatrix} \\ &= \sum_{j=1}^{n} m_j^2 g_{jj} + 2 \left( \sum_{1 \le j < k \le n} m_j m_k g_{jk} \right) \end{aligned}$$

If $\forall \mathbf{m} \in P$ such that $\mathbf{m} \neq \mathbf{0}$, $\mathbf{m^T H_g m} \le 0$, i.e., if $\mathbf{H_g}$ is negative semi-definite over the entire feasible candidate space, then the grade function $g_i(\mathbf{m})$ is concave over $P$.

Now we discuss in detail the process of generating parameters $a_j, b_j, r_j, r_{jk}$ to ensure that the overall grade function is concave and component-wise non-decreasing.

We summarize the overall algorithm for generating one set of voters' grade functions in Algorithm 6 below. As described before, the coefficients $a_j$ and $b_j$ in $v_j(m_j) = a_j m_j^2 + b_j m_j$ should satisfy $a_j \in [-1, 0]$ and $b_j \in [-2a_j, 1 - a_j]$. For sampling, we assume that all parameters, namely $a_j, b_j, r_j$, and $r_{jk}$, are uniformly distributed within their specified ranges. For the purposes of Algorithm 6, we denote the sampling of $r_j \sim \text{unif}(l_j, u_j)$ where the values of $l_j$ and $u_j$ are described as follows. In the capital budgeting case study, we assume that $r_j \sim \text{unif}(2, 6),\ \forall j \in \{1, \cdots, n\}$. However, in the collaborative air traffic flow management case study, $r_j$ values are generated based on metric preferences. For metric preference $m_2 \succ m_1$, $r_1 \sim \text{unif}(2, 4), r_2 \sim \text{unif}(4, 6)$; for metric preference $m_1 \approx m_2$, $r_1 \sim \text{unif}(2, 6), r_2 \sim \text{unif}(2, 6)$; for metric preference $m_1 \succ m_2$, $r_1 \sim \text{unif}(4, 6), r_2 \sim \text{unif}(2, 4)$. Note that $r_j > |r_{jk}|$ to constrain the interaction effects to be smaller than the major effects. We only check a finite set of points (denotes by $P' \subset P$), which are evenly spaced in $P$, for component-wise non-decreasing condition and concavity condition. In the collaborative air traffic flow management case study, we check points in $P' = \{0, 0.1, \cdots, 1\}^2 \cap P$; while in the capital budgeting case study, we check points in $P' = \{0, 0.25, 0.5, 0.75, 1\}^{10} \cap P$.

**Algorithm 6** Generating Voters' True Grade Functions

```
repeat
    for $j = 1 : n$ do
        $a_j \sim \text{unif}(-1, 0)$
        $b_j \sim \text{unif}(-2a_j, 1 - a_j)$
        $r_j \sim \text{unif}(l_j, u_j)$
        for $k = (j + 1) : n$ do
            $r_{jk} \sim \text{unif}(-2, 2)$
        end for
    end for
    concavity condition = true
    non-decreasing condition = true
    for all $\mathbf{m} \in P'$ do
        if $\nabla g(\mathbf{m}) \geq \mathbf{0}$ then
            do nothing
        else
            non-decreasing condition = false
            break the for loop
        end if
        if $\mathbf{m^T H_g m} > 0$ then
            concavity condition = false
            break the for loop
        end if
    end for
until concavity condition and non-decreasing condition are satisfied
calculate $K_j^a, K_j^b, K_{jk}^{aa}, K_{jk}^{ab}, K_{jk}^{ba}, K_{jk}^{bb}$ according to (B.34)
return $K_j^a, K_j^b, K_{jk}^{aa}, K_{jk}^{ab}, K_{jk}^{ba}, K_{jk}^{bb}$
```

# Appendix C

# Technical Results for Chapter 5: Choice-Based Integrated Airline Fleet Assignment and Schedule Design

## C.1 Proofs

### C.1.1 Proof of Lemma 5.2

Define $\mathcal{P}(\mathbf{h}) = \{p \in \mathcal{P} : h_p > 0\}$, $\mathcal{L}(\mathbf{h}) = \{l \in \mathcal{L} : \exists p \in \mathcal{P} \text{ s.t. } h_p > 0, \delta_{l,p} = 1\}$ and $\mathcal{M}(\mathbf{h}) = \{m \in \mathcal{M} : \exists p \in \mathcal{P} \text{ s.t. } h_p > 0, p \in \mathcal{P}(m)\}$. If the following three conditions hold:

1. $\mathcal{P}(\mathbf{h}^i) \cap \mathcal{P}(\mathbf{h}^j) = \emptyset,\ \forall i, j \in \{1, \cdots, K\} : i \neq j$

2. $\mathcal{L}(\mathbf{h}^i) \cap \mathcal{L}(\mathbf{h}^j) = \emptyset,\ \forall i, j \in \{1, \cdots, K\} : i \neq j$

3. $\mathcal{M}(\mathbf{h}^i) \cap \mathcal{M}(\mathbf{h}^j) = \emptyset,\ \forall i, j \in \{1, \cdots, K\} : i \neq j$

Then for any fleet assignment $\mathbf{x}$ and its local fleet assignment $\mathbf{x}^k$, and fare vector $h$, we have

$$\sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k) = r(\mathbf{x}; \sum_{k=1}^{K} \mathbf{h}^k) = r(\mathbf{x}; \mathbf{h})$$

*Proof.* We can further rewrite $r^k(\mathbf{x}^k;\mathbf{h}^k)$ as follows:

$$\begin{aligned}
r^k(\mathbf{x}^k;\mathbf{h}^k) = \max \quad & \sum_{p\in\mathcal{P}(\mathbf{h}^k)} h_p^k s_p \\
\text{s.t.} \quad & \sum_{p\in\mathcal{P}(\mathbf{h}^k)} \delta_{l,p} s_p \le \sum_{f\in\mathcal{F}(l)} \mathrm{CAP}_f x_{l,f},\ \forall l \in \mathcal{L}(\mathbf{h}^k) \\
& \sum_{p\in\mathcal{P}(\mathbf{h}^k)\cap\mathcal{P}(m)} s_p + s_m \le \Lambda_m,\ \forall m \in \mathcal{M}(\mathbf{h}^k) \\
& \frac{s_p}{v_p} - \frac{s_m}{v_m} \le 0,\ \forall m \in \mathcal{M}(\mathbf{h}^k),\ \forall p \in \mathcal{P}(\mathbf{h}^k)\cap\mathcal{P}(m) \\
& s_m, s_p \ge 0,\ \forall m \in \mathcal{M}(\mathbf{h}^k),\ \forall p \in \mathcal{P}(\mathbf{h}^k)
\end{aligned}$$

Because $\mathcal{P}(\mathbf{h}^i)\cap\mathcal{P}(\mathbf{h}^j) = \emptyset$ and $\sum_{k=1}^K \mathbf{h}^k = \mathbf{h}$, we have $h_p^k = h_p, \forall p \in \mathcal{P}(\mathbf{h}^k)$. Because $\mathcal{M}(\mathbf{h}^i)\cap\mathcal{M}(\mathbf{h}^j) = \emptyset$, $\mathcal{P}(\mathbf{h}^k)\cap\mathcal{P}(m) = \mathcal{P}(m), \forall m \in \mathcal{M}(\mathbf{h}^k)$. Finally, by $\mathcal{L}(\mathbf{h}^i)\cap\mathcal{L}(\mathbf{h}^j) = \emptyset$, we have $\sum_{p\in\mathcal{P}(\mathbf{h}^k)} \delta_{l,p} s_p = \sum_{p\in\mathcal{P}} \delta_{l,p} s_p, \forall l \in \mathcal{L}(\mathbf{h}^k)$. With these observations, we can further rewrite $r^k(\mathbf{x}^k;\mathbf{h}^k)$ as:

$$\begin{aligned}
r^k(\mathbf{x}^k;\mathbf{h}^k) = \max \quad & \sum_{p\in\mathcal{P}(\mathbf{h}^k)} h_p s_p \\
\text{s.t.} \quad & \sum_{p\in\mathcal{P}} \delta_{l,p} s_p \le \sum_{f\in\mathcal{F}(l)} \mathrm{CAP}_f x_{l,f},\ \forall l \in \mathcal{L}(\mathbf{h}^k) \\
& \sum_{p\in\mathcal{P}(m)} s_p + s_m \le \Lambda_m,\ \forall m \in \mathcal{M}(\mathbf{h}^k) \\
& \frac{s_p}{v_p} - \frac{s_m}{v_m} \le 0,\ \forall m \in \mathcal{M}(\mathbf{h}^k),\ \forall p \in \mathcal{P}(m) \\
& s_m, s_p \ge 0,\ \forall m \in \mathcal{M}(\mathbf{h}^k),\ \forall p \in \mathcal{P}(\mathbf{h}^k)
\end{aligned}$$

Since $\cup_{k=1}^K \mathcal{L}^k = \mathcal{L}, \cup_{k=1}^K \mathcal{P}^k = \mathcal{P}, \cup_{k=1}^K \mathcal{M}^k = \mathcal{M}$, we realize that $r^k(\mathbf{x}^k;\mathbf{h}^k), \forall k \in \{1,\cdots,K\}$ is a decomposition of $r(\mathbf{x};\mathbf{h})$ that partitions independent constraints, variables, and elements of the objective functions into separate problems (note that (5.15) can be equivalently relaxed to less-than-or-equal-to constraints). This proves that $\sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k) = r(\mathbf{x}; \mathbf{h})$. □

## C.2 Linear Programming Formulation of Problem (5.50)

We want to find a linear programming formulation for problem:

$$\min_{\{\mathbf{h}^1,\ldots,\mathbf{h}^K\}:\sum_{k=1}^K \mathbf{h}^k=\mathbf{h}} \; \max_{\mathbf{x}\in\{\overline{\mathbf{x}},\underline{\mathbf{x}}\}} \; \sum_{k=1}^{K} r^k(\mathbf{x}^k; \mathbf{h}^k) - r(\mathbf{x}; \mathbf{h}).$$

We start by writing out the formulations of $r^k(\overline{\mathbf{x}}^k; \mathbf{h}^k)$ and $r^k(\underline{\mathbf{x}}^k; \mathbf{h}^k)$ where $\overline{\mathbf{x}}^k$ and $\underline{\mathbf{x}}^k$ are local fleet assignment for $\overline{\mathbf{x}}$ and $\underline{\mathbf{x}}$. We then assign dual variables $\overline{\mathbf{r}}^k, \overline{\mathbf{t}}^k, \overline{\mathbf{w}}^k$ and $\underline{\mathbf{r}}^k, \underline{\mathbf{t}}^k, \underline{\mathbf{w}}^k$ for each constraint as illustrated below.

$$\begin{aligned}
r^k(\overline{\mathbf{x}}^k; \mathbf{h}^k) = \max_{\mathbf{s}} \quad & \sum_{p\in\mathcal{P}^k} h_p^k s_p \\
\text{s.t.} \quad & \sum_{p\in\mathcal{P}^k} \delta_{l,p} s_p \le \sum_{f\in\mathcal{F}(l)} \mathrm{CAP}_f \overline{x}_{l,f}^k, \ \forall l \in \mathcal{L}_+^k && (\overline{\mathbf{u}}^k) \\
& \sum_{p\in\mathcal{P}(m)\cap\mathcal{P}^k} s_p + s_m \le \Lambda_m, \ \forall m \in \mathcal{M}^k && (\overline{\mathbf{t}}^k) \\
& \frac{s_p}{v_p} - \frac{s_m}{v_m} \le 0, \ \forall m \in \mathcal{M}^k, \ \forall p \in \mathcal{P}(m)\cap\mathcal{P}^k && (\overline{\mathbf{w}}^k) \\
& s_m, s_p \ge 0, \ \forall m \in \mathcal{M}^k, \ \forall p \in \mathcal{P}^k
\end{aligned}$$

$$\begin{aligned}
r^k(\underline{\mathbf{x}}^k; \mathbf{h}^k) = \max_{\mathbf{s}} \quad & \sum_{p\in\mathcal{P}^k} h_p^k s_p \\
\text{s.t.} \quad & \sum_{p\in\mathcal{P}^k} \delta_{l,p} s_p \le \sum_{f\in\mathcal{F}(l)} \mathrm{CAP}_f \underline{x}_{l,f}^k, \ \forall l \in \mathcal{L}_+^k && (\underline{\mathbf{u}}^k) \\
& \sum_{p\in\mathcal{P}(m)\cap\mathcal{P}^k} s_p + s_m \le \Lambda_m, \ \forall m \in \mathcal{M}^k && (\underline{\mathbf{t}}^k) \\
& \frac{s_p}{v_p} - \frac{s_m}{v_m} \le 0, \ \forall m \in \mathcal{M}^k, \ \forall p \in \mathcal{P}(m)\cap\mathcal{P}^k && (\underline{\mathbf{w}}^k) \\
& s_m, s_p \ge 0, \ \forall m \in \mathcal{M}^k, \ \forall p \in \mathcal{P}^k
\end{aligned}$$

Taking the duals of both formulations, we have

$$\begin{aligned}
r^k(\overline{\mathbf{x}}^k;\mathbf{h}^k) = \min_{\overline{\mathbf{u}}^k,\overline{\mathbf{t}}^k,\overline{\mathbf{w}}^k} \quad & \sum_{l\in\mathcal{L}_+^k}\left(\sum_{f\in\mathcal{F}(l)}\mathrm{CAP}_f\overline{x}_{l,f}^k\right)\overline{u}_l^k + \sum_{m\in\mathcal{M}^k}\Lambda_m\overline{t}_m^k \\
\text{s.t.} \quad & \sum_{l\in\overline{\mathcal{L}}^k}\delta_{l,p}\overline{u}_l^k + \sum_{\substack{m\in\mathcal{M}:\\ p\in\mathcal{P}(m)}}\overline{t}_m^k + (1/v_p)\overline{w}_p^k \geq h_p^k, \quad \forall p\in\mathcal{P}^k \\
& \overline{t}_m^k - \sum_{p\in\mathcal{P}(m)}(1/v_m)\overline{w}_p^k \geq 0, \quad \forall m\in\mathcal{M}^k \\
& \overline{u}_l^k \geq 0, \quad \forall l\in\mathcal{L}_+^k \\
& \overline{t}_m^k \geq 0, \overline{w}_p^k \geq 0, \quad \forall m\in\mathcal{M}^k,\ \forall p\in\mathcal{P}^k
\end{aligned}$$

$$\begin{aligned}
r^k(\underline{\mathbf{x}}^k;\mathbf{h}^k) = \min_{\underline{\mathbf{u}}^k,\underline{\mathbf{t}}^k,\underline{\mathbf{w}}^k} \quad & \sum_{l\in\mathcal{L}_+^k}\left(\sum_{f\in\mathcal{F}(l)}\mathrm{CAP}_f\underline{x}_{l,f}^k\right)\underline{u}_l^k + \sum_{m\in\mathcal{M}^k}\Lambda_m\underline{t}_m^k \\
\text{s.t.} \quad & \sum_{l\in\underline{\mathcal{L}}^k}\delta_{l,p}\underline{u}_l^k + \sum_{\substack{m\in\mathcal{M}:\\ p\in\mathcal{P}(m)}}\underline{t}_m^k + (1/v_p)\underline{w}_p^k \geq h_p^k, \quad \forall p\in\mathcal{P}^k \\
& \underline{t}_m^k - \sum_{p\in\mathcal{P}(m)}(1/v_m)\underline{w}_p^k \geq 0, \quad \forall m\in\mathcal{M}^k \\
& \underline{u}_l^k \geq 0, \quad \forall l\in\mathcal{L}_+^k \\
& \underline{t}_m^k \geq 0, \underline{w}_p^k \geq 0, \quad \forall m\in\mathcal{M}^k,\ \forall p\in\mathcal{P}^k
\end{aligned}$$

Denoting $\overline{r} = r(\overline{\mathbf{x}};\mathbf{h})$ and $\underline{r} = r(\underline{\mathbf{x}};\mathbf{h})$, we have problem (5.50) reformulated as the following linear program:

$$\begin{aligned}
\min_{\substack{\mathbf{h}^k\\ \overline{\mathbf{u}}^k,\overline{\mathbf{t}}^k,\overline{\mathbf{w}}^k,\\ \underline{\mathbf{u}}^k,\underline{\mathbf{t}}^k,\underline{\mathbf{w}}^k}} \quad & z \\
\text{s.t.} \quad & z \geq \sum_{k=1}^{K}\left(\sum_{l\in\mathcal{L}_+^k}\left(\sum_{f\in\mathcal{F}(l)}\mathrm{CAP}_f\overline{x}_{l,f}^k\right)\overline{u}_l^k + \sum_{m\in\mathcal{M}^k}\Lambda_m\overline{t}_m^k\right) - \overline{r}
\end{aligned}$$

$$z \geq \sum_{k=1}^{K}\left(\sum_{l\in\mathcal{L}_{+}^{k}}\left(\sum_{f\in\mathcal{F}(l)} \mathrm{CAP}_{f}\underline{x}_{l,f}^{k}\right)\underline{u}_{l}^{k} + \sum_{m\in\mathcal{M}^{k}} \Lambda_{m}\underline{t}_{m}^{k}\right) - \underline{r}$$

$$\sum_{l\in\overline{\mathcal{L}}^{k}} \delta_{l,p}\overline{u}_{l}^{k} + \sum_{\substack{m\in\mathcal{M}:\\ p\in\mathcal{P}(m)}} \overline{t}_{m}^{k} + (1/v_{p})\overline{w}_{p}^{k} \geq h_{p}^{k}, \ \ \forall p\in\mathcal{P}^{k},\ \forall k\in\{1,\cdots,K\}$$

$$\sum_{l\in\underline{\mathcal{L}}^{k}} \delta_{l,p}\underline{u}_{l}^{k} + \sum_{\substack{m\in\mathcal{M}:\\ p\in\mathcal{P}(m)}} \underline{t}_{m}^{k} + (1/v_{p})\underline{w}_{p}^{k} \geq h_{p}^{k}, \ \ \forall p\in\mathcal{P}^{k},\ \forall k\in\{1,\cdots,K\}$$

$$\overline{t}_{m}^{k} - \sum_{p\in\mathcal{P}(m)} (1/v_{m})\overline{w}_{p}^{k} \geq 0, \ \ \forall m\in\mathcal{M}^{k},\ \forall k\in\{1,\cdots,K\}$$

$$\underline{t}_{m}^{k} - \sum_{p\in\mathcal{P}(m)} (1/v_{m})\underline{w}_{p}^{k} \geq 0, \ \ \forall m\in\mathcal{M}^{k},\ \forall k\in\{1,\cdots,K\}$$

$$\sum_{k:p\in\mathcal{P}^{k}} h_{p}^{k} = h_{p}, \ \ \forall p\in\mathcal{P}$$

$$h_{p}^{k} \geq 0, \ \ \forall p\in\mathcal{P}^{k},\ \forall k\in\{1,\cdots,K\}$$

$$\overline{u}_{l}^{k} \geq 0,\ \underline{u}_{l}^{k} \geq 0, \ \ \forall l\in\mathcal{L}_{+}^{k},\ \forall k\in\{1,\cdots,K\}$$

$$\overline{t}_{m}^{k} \geq 0,\ \underline{t}_{m}^{k} \geq 0,\ \overline{w}_{p}^{k} \geq 0,\ \underline{w}_{p}^{k} \geq 0, \ \ \forall m\in\mathcal{M}^{k},\ \forall p\in\mathcal{P}^{k},\ \forall k\in\{1,\cdots,K\}$$

$z$ is an auxiliary variable to model the maximum of $\left(\sum_{k=1}^{K} r^{k}(\overline{\mathbf{x}}^{k};\mathbf{h}^{k}) - \overline{r}\right)$ and $\left(\sum_{k=1}^{K} r^{k}(\underline{\mathbf{x}}^{k};\mathbf{h}^{k}) - \underline{r}\right)$.

## C.3 ISD-FAM Formulation

We present the formulation of one of the baseline approaches, ISD-FAM, in this section. All notation follows that presented in Section 5.2, except that we introduce $d_p$ as the unconstrained demand of product $p$. Demand in ISD-FAM is product not market specific. In our implementation, we let $d_p = \left(v_p/(v_m + \sum_{p'\in\mathcal{P}(m)} v_{p'})\right)\Lambda_m,\ p\in\mathcal{P}(m)$, i.e., the first-choice demand of product $p$ if all products in market $m$ are available for sale. Objective function (C.1) and constraints (C.2) - (C.6) are the same as those in the CSD-FAM formulation described in (5.1) - (5.12). The only difference is in constraints (C.7) where the sale of product $p$ is upper bounded by $d_p$. There are no market demand constraints (5.7) nor MNL consistency constraints (5.8) because product demands are assumed to be independent across products, and unsatisfied demand is lost and not spilled to similar available products.

(ISD-FAM)

$$\max \quad \sum_{p\in\mathcal{P}} h_p s_p - \sum_{l\in\mathcal{L}} \sum_{f\in\mathcal{F}(l)} c_{l,f} x_{l,f} \tag{C.1}$$

$$\text{s.t.} \quad \sum_{f\in\mathcal{F}(l)} x_{l,f} = 1, \quad \forall l \in \mathcal{L}_m \tag{C.2}$$

$$\sum_{f\in\mathcal{F}(l)} x_{l,f} \leq 1, \quad \forall l \in \mathcal{L}_o \tag{C.3}$$

$$y_{v^-} + \sum_{l\in I(v)} x_{l,f} - y_v - \sum_{l\in O(v)} x_{l,f} = 0, \quad \forall v \in N_f,\ \forall f \in \mathcal{F} \tag{C.4}$$

$$\sum_{v\in T_N(f)} y_v + \sum_{l\in T_F(f)} x_{l,f} \leq n_f, \quad \forall f \in \mathcal{F} \tag{C.5}$$

$$\sum_{p\in\mathcal{P}} \delta_{l,p} s_p \leq \sum_{f\in\mathcal{F}(l)} \mathrm{CAP}_f x_{l,f},\ \forall l \in \mathcal{L} \tag{C.6}$$

$$0 \leq s_p \leq d_p, \quad \forall p \in \mathcal{P} \tag{C.7}$$

$$x_{l,f} \in \{0,1\}, \quad \forall l \in \mathcal{L},\ \forall f \in \mathcal{F}(l) \tag{C.8}$$

$$y_v \geq 0, \quad \forall v \in N_f,\ \forall f \in \mathcal{F} \tag{C.9}$$

# Bibliography

Achterberg, Tobias. 2009. SCIP: Solving constraint integer programs. *Mathematical Programming Computation* **1**(1) 1–41.

Ahmadbeygi, Shervin, Amy Cohn, Marcial Lapp. 2010. Decreasing airline delay propagation by re-allocating scheduled slack. *IIE Transactions* **42**(7) 478–489.

Armbruster, Benjamin, Erick Delage. 2015. Decision making under uncertainty when preference information is incomplete. *Management Science* **61**(1) 111–128.

Arrow, Kenneth. 1951. Individual values and social choice. *New York: Wiley* **24**.

Atasoy, Bilge, Matteo Salani, Michel Bierlaire. 2014. An integrated airline scheduling, fleeting, and pricing model for a monopolized market. *Computer-Aided Civil and Infrastructure Engineering* **29**(2) 76–90.

Bai, Yuqiong. 2006. Analysis of aircraft arrival delay and airport on-time performance. Ph.D. thesis, University of Central Florida Orlando, Florida.

Balinski, Michel L, Rida Laraki. 2010. *Majority judgment: measuring, ranking, and electing*. MIT Press.

Balinski, Michel L, Rida Laraki. 2014. Judge: Don't vote! *Operations Research* **62**(3) 483–511.

Ball, Michael, Cynthia Barnhart, Martin Dresner, Mark Hansen, Kevin Neels, Amedeo Odoni, Everett Peterson, Lance Sherry, Antonio A Trani, Bo Zou. 2010. Total delay impact study: a comprehensive assessment of the costs and impacts of flight delay in the united states. Tech. rep., NEXTOR.

Ball, Michael, Cynthia Barnhart, Antony Evans, Mark Hansen, Lei Kang, Yi Liu, Prem Swaroop, Vikrant Vaze, Chiwei Yan. 2014a. Distributed mechanisms for determining NAS-wide service level expectations: final report. Tech. rep., NEXTOR II.

Ball, Michael, Prem Swaroop, Cynthia Barnhart, Chiwei Yan, Mark Hansen, Lei Kang, Yi Liu, Vikrant Vaze. 2017. Service level expectation setting for air traffic flow management: Practical challenges and benefits assessment. *USA/Europe Air Traffic Management Research & Development Seminar, Seattle, WA*.

Ball, Michael O, Cynthia Barnhart, Mark Hansen, Lei Kang, Yi Liu, Prem Swaroop, Vaze Vaze, Chiwei Yan. 2014b. Distributed mechanisms for determining NAS-wide service level expectations. Tech. rep., NEXTOR II.

Ball, Michael O, Robert Hoffman, Amedeo R Odoni, Ryan Rifkin. 2003. A stochastic integer program with dual network structure and its application to the ground-holding problem. *Operations Research* **51**(1) 167–171.

Barber, C Bradford, David P Dobkin, Hannu Huhdanpaa. 1996. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)* **22**(4) 469–483.

Barnhart, Cynthia, Peter Belobaba, Amedeo R Odoni. 2003. Applications of operations research in the air transport industry. *Transportation science* **37**(4) 368–391.

Barnhart, Cynthia, Dimitris Bertsimas, Constantine Caramanis, Douglas Fearing. 2012. Equitable and efficient coordination in traffic flow management. *Transportation Science* **46**(2) 262–280.

Barnhart, Cynthia, Natashia L Boland, Lloyd W Clarke, Ellis L Johnson, George L Nemhauser, Rajesh G Shenoi. 1998a. Flight string models for aircraft fleeting and routing. *Transportation Science* **32**(3) 208–220.

Barnhart, Cynthia, Amr Farahat, Manoj Lohatepanont. 2009. Airline fleet assignment with enhanced revenue modeling. *Operations Research* **57**(1) 231–244.

Barnhart, Cynthia, Douglas Fearing, Vikrant Vaze. 2014. Modeling passenger travel and delays in the national air transportation system. *Operations Research* **62**(3) 580–601.

Barnhart, Cynthia, Ellis L Johnson, George L Nemhauser, Martin WP Savelsbergh, Pamela H Vance. 1998b. Branch-and-price: Column generation for solving huge integer programs. *Operations Research* **46**(3) 316–329.

Barnhart, Cynthia, Timothy S Kniker, Manoj Lohatepanont. 2002. Itinerary-based airline fleet assignment. *Transportation Science* **36**(2) 199–217.

BBC. 2016. This is why china's airports are a nightmare. URL `http://www.bbc.com/capital/story/20160420-this-is-why-chinas-airports-are-a-nightmare`.

Ben-Tal, Aharon, Arkadi Nemirovski. 1999. Robust solutions of uncertain linear programs. *Operations Research Letters* **25**(1) 1–13.

Bertsimas, Dimitris, David B Brown, Constantine Caramanis. 2011. Theory and applications of robust optimization. *SIAM Review* **53**(3) 464–501.

Bertsimas, Dimitris, Iain Dunning, Miles Lubin. 2016. Reformulation versus cutting-planes for robust optimization. *Computational Management Science* **13**(2) 195.

Bertsimas, Dimitris, Vishal Gupta, Nathan Kallus. 2017. Data-driven robust optimization. *Mathematical Programming* 1–58.

Bertsimas, Dimitris, Eugene Litvinov, Xu Andy Sun, Jinye Zhao, Tongxin Zheng. 2013. Adaptive robust optimization for the security constrained unit commitment problem. *Power Systems, IEEE Transactions on* **28**(1) 52–63.

Bertsimas, Dimitris, Dessislava Pachamanova, Melvyn Sim. 2004. Robust linear optimization under general norms. *Operations Research Letters* **32**(6) 510–516.

Bertsimas, Dimitris, Melvyn Sim. 2004. The price of robustness. *Operations Research* **52**(1) 35–53.

Bertsimas, Dimitris, Aurélie Thiele. 2006. A robust optimization approach to inventory theory. *Operations Research* **54**(1) 150–168.

Bertsimas, Dimitris, John N Tsitsiklis. 1997. *Introduction to linear optimization*, vol. 6. Athena Scientific Belmont, MA.

Bloem, Michael, Haiyun Huang, Nicholas Bambos. 2012. Approximating the likelihood of historical airline actions to evaluate airline delay cost functions. *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*. IEEE, 508–513.

Bonami, Pierre, Lorenz T Biegler, Andrew R Conn, Gérard Cornuéjols, Ignacio E Grossmann, Carl D Laird, Jon Lee, Andrea Lodi, François Margot, Nicolas Sawaya, et al. 2008. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization* **5**(2) 186–204.

Borndörfer, Ralf, Ivan Dovica, Ivo Nowak, Thomas Schickinger. 2010. Robust tail assignment. *Proceedings of the fiftieth annual symposium of AGIFORS*.

Bradley, Stephen, Arnoldo Hax, Thomas Magnanti. 1977. Applied mathematical programming .

Bratu, Stephane. 2003. Airline passenger on-time schedule reliability: Analysis, algorithms and optimization decision models. *Massachusetts Institute of Technology, PhD Thesis* .

Bratu, Stephane, Cynthia Barnhart. 2005. An analysis of passenger delays using flight operations and passenger booking data. *Air Traffic Control Quarterly* **13**(1) 1–28.

Bratu, Stephane, Cynthia Barnhart. 2006. Flight operations recovery: New approaches considering passenger recovery. *Journal of Scheduling* **9**(3) 279–298.

Bront, Juan José Miranda, Isabel Méndez-Díaz, Gustavo Vulcano. 2009. A column generation algorithm for choice-based network revenue management. *Operations Research* **57**(3) 769–784.

Bureau of Transportation Statistics. 2016a. Research and innovative technology. URL `http://www.transtats.bts.gov/ot_delay/OT_DelayCause1.asp?pn=1`.

Bureau of Transportation Statistics. 2016b. TranStats. URL `https://www.transtats.bts.gov/Data_Elements.aspx?Data=5`.

Bureau of Transportation Statistics. 2016c. U.S. air carrier traffic statistics. URL `https://www.transtats.bts.gov/TRAFFIC/`.

Byrd, Richard H, Jorge Nocedal, Richard A Waltz. 2006. Knitro: An integrated package for nonlinear optimization. *Large-scale nonlinear optimization*. Springer, 35–59.

Dai, JG, Weijun Ding, Anton Kleywegt, Xinchang Wang, Yi Zhang. 2015. Choice based revenue management for parallel flights. *Available at SSRN 2404193* .

Department of Transportation. 2007. Air carrier financial reports (form 41 financial data), schedule p52. URL `https://www.transtats.bts.gov/Tables.asp?DB_ID=135`.

Desaulniers, Guy, Jacques Desrosiers, Yvan Dumas, Marius M Solomon, François Soumis. 1997. Daily aircraft routing and scheduling. *Management Science* **43**(6) 841–855.

Di, Mingyang, Diego Klabjan, Sergey Shebalov. 2016. Solving attractiveness-based stochastic fleeting by mapreduce. Working paper.

Dumas, Jonathan, François Soumis. 2008. Passenger flow model for airline networks. *Transportation Science* **42**(2) 197–207.

Dunbar, Michelle, Gary Froyland, Cheng-Lung Wu. 2012. Robust airline schedule planning: Minimizing propagated delay in an integrated routing and crewing framework. *Transportation Science* **46**(2) 204–216.

Dunbar, Michelle, Gary Froyland, Cheng-Lung Wu. 2014. An integrated scenario-based approach for robust aircraft routing, crew pairing and re-timing. *Computers & Operations Research* **45** 68–86.

Dunning, Iain, Joey Huchette, Miles Lubin. 2017. JuMP: A modeling language for mathematical optimization. *SIAM Review* **59**(2) 295–320.

Evans, Antony, Vikrant Vaze, Cynthia Barnhart. 2016. Airline-driven performance-based air traffic management: game theoretic models and multicriteria evaluation. *Transportation Science* **50**(1) 180–203.

Federal Aviation Administration. 2001. What is different about a CDM GDP? Tech. rep. URL `http://cdm.fly.faa.gov/wp-content/list_yo_files_user_folders/cdm_editor/cdm_arch_train_gdpe/GDPS.pdf`.

Federal Aviation Administration. 2014a. Airport capacity profiles. URL `https://www.faa.gov/airports/planning_capacity/profiles/`.

Federal Aviation Administration. 2014b. Aviation maintenance technician handbook - general. URL `https://www.faa.gov/regulations_policies/handbooks_manuals/aircraft/amt_handbook/`.

Feldman, Jacob B, Huseyin Topaloglu. 2017. Revenue management under the markov chain choice model. *Operations Research* forthcoming.

Friedman, Jerome, Trevor Hastie, Robert Tibshirani. 2008. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* **9**(3) 432–441.

Froyland, Gary, Stephen J Maher, Cheng-Lung Wu. 2013. The recoverable robust tail assignment problem. *Transportation Science* **48**(3) 351–372.

Galeeva, Roza, Jiri Hoogland, Alexander Eydeland, Morgan Stanley. 2007. Measuring correlation risk. Tech. rep.

Gallego, G., G. Iyengar, R. Phillips, A. Dubey. 2004. Managing flexible products on a network. Technical Report TR-2004-01. Dept. of Industrial Engineering and Operations Research, Columbia University.

Gallego, Guillermo, Richard Ratliff, Sergey Shebalov. 2014. A general attraction model and sales-based linear program for network revenue management under customer choice. *Operations Research* **63**(1) 212–232.

Gibbard, Allan. 1973. Manipulation of voting schemes: a general result. *Econometrica: journal of the Econometric Society* 587–601.

Gurobi Optimization Inc. 2016. *Gurobi Optimizer Reference Manual*. URL `http://www.gurobi.com/documentation/6.5/`.

Hane, Christopher A, Cynthia Barnhart, Ellis L Johnson, Roy E Marsten, George L Nemhauser, Gabriele Sigismondi. 1995. The fleet assignment problem: Solving a large-scale integer program. *Mathematical Programming* **70**(1) 211–232.

Hochbaum, Dorit S, Asaf Levin. 2006. Methodologies and algorithms for group-rankings decision. *Management Science* **52**(9) 1394–1408.

Hoffman, Robert, William Hall, Michael Ball, Amedeo Odoni, Michael Wambsganss. 1999. Collaborative decision making in air traffic flow management. Tech. rep., NEXTOR.

Hsieh, Cho-Jui, Inderjit S Dhillon, Pradeep K Ravikumar, Mátyás A Sustik. 2011. Sparse inverse covariance matrix estimation using quadratic approximation. *Advances in Neural Information Processing Systems*. 2330–2338.

IATA. 2017. Another strong year for airline profits in 2017. URL `http://www.iata.org/pressroom/pr/Pages/2016-12-08-01.aspx`.

Jacobs, Timothy L, Barry C Smith, Ellis L Johnson. 2008. Incorporating network flow effects into the airline fleet assignment process. *Transportation Science* **42**(4) 514–529.

Johnson, Steven G. 2011. The NLopt nonlinear-optimization package. URL `http://ab-initio.mit.edu/nlopt`.

Kang, Laura Sumi. 2004. Degradable airline scheduling: an approach to improve operational robustness and differentiate service quality. Ph.D. thesis, Massachusetts Institute of Technology.

Kemeny, John G, LJ Snell. 1962. Preference ranking: an axiomatic approach. *Mathematical models in the social sciences* 9–23.

Klabjan, Diego, Andrew J Schaefer, Ellis L Johnson, Anton J Kleywegt, George L Nemhauser. 2001. Robust airline crew scheduling. *Proceedings of TRISTAN IV*. Azores, Portugal.

Kunnumkal, Sumit, Kalyan Talluri. 2016. Technical noteâĂŤa note on relaxations of the choice network revenue management dynamic program. *Operations Research* **64**(1) 158–166.

Kuosmanen, Timo. 2008. Representation theorem for convex nonparametric least squares. *The Econometrics Journal* **11**(2) 308–325.

Lan, Shan, John-Paul Clarke, Cynthia Barnhart. 2006. Planning for robust airline operations: Optimizing aircraft routings and flight departure times to minimize passenger disruptions. *Transportation Science* **40**(1) 15–28.

Lim, Eunji, Peter W Glynn. 2012. Consistency of multidimensional convex regression. *Operations Research* **60**(1) 196–208.

Liu, Qian, Garrett van Ryzin. 2008. On the choice-based linear programming model for network revenue management. *Manufacturing & Service Operations Management* **10**(2) 288–310.

Liu, Yi, Mark Hansen. 2013. Ground delay program decision-making using multiple criteria: a single airport case. *USA/Europe Air Traffic Management Research & Development Seminar, Chicago, IL.*

Lohatepanont, Manoj, Cynthia Barnhart. 2004. Airline schedule planning: Integrated models and algorithms for schedule design and fleet assignment. *Transportation Science* **38**(1) 19–32.

Lubin, Miles, Emre Yamangil, Russell Bent, Juan Pablo Vielma. 2016. Polyhedral approximation in mixed-integer convex optimization. *arXiv preprint arXiv:1607.03566* .

Marla, Lavanya, Vikrant Vaze, Cynthia Barnhart. 2017. Robust optimization: Lessons learned from aircraft routing. *Available at SSRN 2991397* .

MathWorks. 2014. Simulating dependent random variables using copulas. URL `https://www.mathworks.com/help/stats/examples/simulating-dependent-random-variables-using-copulas.html`.

Mazumder, Rahul, Arkopal Choudhury, Garud Iyengar, Bodhisattva Sen. 2015. A computational framework for multivariate convex regression and its variants. *arXiv preprint arXiv:1509.08165* .

Meyer, Robert, Eric J Johnson. 1995. Empirical generalizations in the modeling of consumer choice. *Marketing Science* **14**(3) G180–G189.

Mueller, Eric R, Gano B Chatterji. 2002. Analysis of aircraft arrival and departure delay characteristics. *AIAA's Aircraft Technology, Integration and Operations (ATIO) Technical Forum*. Los Angeles, CA.

Mukherjee, Avijit, Mark Hansen. 2007. A dynamic stochastic model for the single airport ground holding problem. *Transportation Science* **41**(4) 444–456.

Odoni, Amedeo R. 1987. The flow management problem in air traffic control. *Flow control of congested networks*. Springer, 269–288.

Petersen, Jon D, Gustaf Sölveling, John-Paul Clarke, Ellis L Johnson, Sergey Shebalov. 2012. An optimization approach to airline integrated recovery. *Transportation Science* **46**(4) 482–500.

Richetta, Octavio, Amedeo R Odoni. 1993. Solving optimally the static ground-holding policy problem in air traffic control. *Transportation Science* **27**(3) 228–238.

Richetta, Octavio, Amedeo R Odoni. 1994. Dynamic solution to the ground-holding problem in air traffic control. *Transportation Research Part A: Policy and Practice* **28**(3) 167–185.

Rosenberger, Jay M, Ellis L Johnson, George L Nemhauser. 2004. A robust fleet-assignment model with hub isolation and short cycles. *Transportation Science* **38**(3) 357–368.

Saaty, Thomas L. 1977. A scaling method for priorities in hierarchical structures. *Journal of mathematical psychology* **15**(3) 234–281.

Satterthwaite, Mark Allen. 1975. Strategy-proofness and arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of economic theory* **10**(2) 187–217.

Shebalov, Sergey, Diego Klabjan. 2006. Robust airline crew pairing: Move-up crews. *Transportation Science* **40**(3) 300–312.

Sherali, Hanif D, Ki-Hwan Bae, Mohamed Haouari. 2010. Integrated airline schedule design and fleet assignment: Polyhedral analysis and benders' decomposition approach. *INFORMS Journal on Computing* **22**(4) 500–513.

Sherali, Hanif D, Xiaomei Zhu. 2008. Two-stage fleet assignment model considering stochastic passenger demands. *Operations Research* **56**(2) 383–399.

Smith, Barry C, Ellis L Johnson. 2006. Robust airline fleet assignment: Imposing station purity using station decomposition. *Transportation Science* **40**(4) 497–516.

Soumis, François, Anna Nagurney. 1993. A stochastic, multiclass airline network equilibrium model. *Operations research* **41**(4) 721–730.

Subramanian, Radhika, Richard P Scheff, John D Quillinan, D Steve Wiper, Roy E Marsten. 1994. Coldstart: fleet assignment at Delta air lines. *Interfaces* **24**(1) 104–120.

Swaroop, P., M. Ball. 2012. Consensus building mechanism for setting service expectations in air traffic management. *Transportation Research Record: Journal of the Transportation Research Board* **2325** 87–96.

Swaroop, Prem. 2013. Problems and models in strategic air traffic flow management. Ph.D. thesis, University of Maryland, College Park.

Swaroop, Prem, Michael Ball. 2013. Consensus-building mechanism for setting service expectations in air traffic flow management. *Transportation Research Record: Journal of the Transportation Research Board* (2325) 87–96.

Talluri, Kalyan, Garrett van Ryzin. 2004. Revenue management under a general discrete choice model of consumer behavior. *Management Science* **50**(1) 15–33.

Talluri, Kalyan T, Garrett J van Ryzin. 2006. *The theory and practice of revenue management*, vol. 68. Springer Science & Business Media.

The Economist. 2014. Why airlines make such meagre profits. URL `http://www.economist.com/blogs/economist-explains/2014/02/economist-explains-5`.

Tu, Yufeng, Michael O Ball, Wolfgang S Jank. 2008. Estimating flight departure delay distribution – a statistical approach with long-term trend and short-term pattern. *Journal of the American Statistical Association* **103**(481) 112–125.

van Ryzin, Garrett, Gustavo Vulcano. 2014. A market discovery algorithm to estimate a general class of nonparametric choice models. *Management Science* **61**(2) 281–300.

Vranas, Peter B, Dimitris J Bertsimas, Amedeo R Odoni. 1994a. The multi-airport ground-holding problem in air traffic control. *Operations Research* **42**(2) 249–261.

Vranas, Peter BM, Dimitris J Bertsimas, Amedeo R Odoni. 1994b. Dynamic ground-holding policies for a network of airports. *Transportation Science* **28**(4) 275–291.

Vulcano, Gustavo, Garrett van Ryzin, Wassim Chaar. 2010. Choice-based revenue management: An empirical study of estimation and optimization. *Manufacturing & Service Operations Management* **12**(3) 371–392.

Vulcano, Gustavo, Garrett van Ryzin, Richard Ratliff. 2012. Estimating primary demand for substitutable products from sales transaction data. *Operations Research* **60**(2) 313–334.

Wächter, Andreas, Lorenz T Biegler. 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming* **106**(1) 25–57.

Wang, Desmond, Diego Klabjan, Sergey Shebalov. 2014. Attractiveness-based airline network models with embedded spill and recapture. *Journal of Airline and Airport Management* **4**(1) 1–25.

Weide, Oliver, David Ryan, Matthias Ehrgott. 2010. An iterative approach to robust and integrated aircraft routing and crew scheduling. *Computers & Operations Research* **37**(5) 833–844.

Xia, Lirong. 2011. Computational voting theory: game-theoretic and combinatorial aspects. Ph.D. thesis, Duke University.

Yan, Chiwei, Jerry Kung. 2016. Robust aircraft routing. *Transportation Science* forthcoming.

Yan, Chiwei, Prem Swaroop, Michael O Ball, Cynthia Barnhart, Vikrant Vaze. 2017a. Applying majority judgment over a polyhedral candidate space. *Available at SSRN 2746568* .

Yan, Chiwei, Vikrant Vaze, Cynthia Barnhart. 2017b. Airline-driven ground delay programs: A benefits assessment. Working Paper.

Yen, Joyce W, John R Birge. 2006. A stochastic programming approach to the airline crew scheduling problem. *Transportation Science* **40**(1) 3–14.