A Demand Driven Re-fleeting Approach for Aircraft Assignment Under Uncertainty

Xiaomei Zhu

Thesis Submitted to the Faculty of
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
In
Industrial and Systems Engineering

Ebru K. Bish, Co-Chair
Hanif D. Sherali, Co-Chair
Antonio A. Trani

July 23, 2001
Blacksburg, Virginia

Keywords: Airline Operations Research, Demand Driven Re-fleeting, Fleet Assignment, Revenue Management, Mixed-Integer Programming

Copyright 2001, Xiaomei Zhu

A Demand Driven Re-fleeting Approach for Aircraft Assignment Under Uncertainty

Xiaomei Zhu

(ABSTRACT)

The current airline practice is to assign aircraft capacity to scheduled flights well in advance of departure. At such an early stage in this process, the high uncertainty of demand poses a major impediment for airlines to best match the airplane capacities with the final demand. However, the accuracy of the demand forecast improves markedly over time, and revisions to the initial fleet assignment become naturally pertinent when the observed demand considerably differs from the assigned aircraft capacity. The Demand Driven Re-fleeting (DDR) approach proposed in this thesis offers a dynamic re-assignment of aircraft capacity to the flight network, as and when improved demand forecasts become available, so as to maximize the total revenue.

Because of the need to preserve the initial crew schedule, this re-assignment approach is limited within a single family of aircraft and to the flights assigned to this particular family. This restriction significantly reduces the problem size. As a result, it becomes computationally tractable to include path level demand information into the DDR model, although the problem size can then get very large because of the numerous combinations of composing paths from legs. As an extension, models considering path-class level differences, day-of-week demand variations, and re-capture effects are also presented.

The DDR model for a single family with path level demand considerations is formulated as a mixed-integer programming problem. The model's polyhedral structure is studied

to explore ways for tightening its representation and for deriving certain classes of valid inequalities. Various approaches for implementing such reformulation techniques are investigated and tested. The best of these procedures for solving large-scale challenging instances of the problem turns out to be an integrated approach that uses certain selected model augmentations and valid inequalities generated via a suitable separation routine and a partial convex hull construction process. Using this strategy in concert with properly selected CPLEX options reduces the CPU time by an average factor of 7.48 over an initial model for a test-bed of problems each having 200 flights in total. Prompted by this integrated heuristic approach, a procedure for finding solutions within a prescribed limit of optimality is suggested. To demonstrate the effectiveness of these developed methodologies, we also solved two large-scale practical-sized networks that respectively involve 800 and 1060 flights, and 18196 and 33105 paths in total, with 300 and 396 flights belonging to the designated family. These problems were typically solved within 6 hours on a SUN Ultra 1 Workstation having 260 MB RAM and a clock-speed of 167 MHz, with one exception that required 14 hours of CPU time. This level of computational effort is acceptable considering that such models are solved at a planning stage in the decision process.

Acknowledgments

I first wish to thank my parents and my sister for their love and encouragement throughout my life and especially during these two years when I am thousands of miles away from home. I dedicate this thesis to them for the positive view of life they continuously passed to me.

My deepest appreciation goes to my advisors Dr. Ebru K. Bish and Dr. Hanif D. Sherali. I thank Dr. Bish for her encouragement and understandness. She has been more than just a great advisor who inspired and guided me in completing this thesis with great enthusiasm and patience. She has also been a good friend who cheered me up when I was depressed and showed me directions when I was bewildered. I am grateful for Dr. Sherali's valuable guidance, consistent attention, and amazing teaching. His perfect personality is commented by every one who has worked with him. It is a sincere pleasure to be his advisee.

I am also grateful to my thesis committee member Dr. Antonio A. Trani, for his advise and expertise in the airline operations research. Special thanks are due to Mr. Douglas R. Bish and Ms. Connie Chen at United Airlines. They helped me understand the whole system so that I could construct a realistic data structure, and patiently provided data again and again to assist the computational experiments in this thesis.

I also appreciate Dr. Patrick Koelling for providing a fast computer so that I could save time on waiting for the computational results. I thank Barbara Fraticelli for her help

on how to use AMPL and Seong-Jong Hong for his help on performing the computational experiments.

I thank all my friends that I have acquired in these two years for making my life in Virginia Tech so enjoyable. I also thank the Faculty and the Staff in the Grado Department of Industrial and Systems Engineering. I especially appreciate the help from Ms. Lovedia Cole and Ms. Kathy Buchanan.

In the end, I would like to thank the simopt04 computer for giving me company in the many sleepless nights...

Contents

Abstract **ii**

Acknowledgments **iv**

1 Introduction **1**

1.1 Introduction 1

1.2 Terminology 2

1.3 Background and Motivation 3

1.4 Model Overview 5

1.5 Problem Statement 6

1.5.1 Objective 6

1.5.2 Constraints 8

1.6 Research Contributions 8

2 Literature Review **10**

2.1 Fleet Assignment Problem 10

2.2 Integrated Approaches to FAP and other Scheduling Processes 15

2.3 Re-fleeting Problem 17

2.4 Stochastic Optimization Approaches 21

2.4.1 Meta-Heuristics 22

2.4.2 Robust Optimization 23

3 Model Formulation **26**

3.1 Outline 26

3.2 Background: Network Structure 27

3.3 Notation 31

3.4 A Leg-Based Fleet Assignment Model 33

3.5 A Path-Based Revenue Management System 35

3.6 The Proposed DDR Models 39

3.6.1 The Basic DDR Model 39

3.6.2 Path-Class Level Based DDR Model 41

3.6.3 Integrated Path-based DDR Model with DOW or DOM Variations . 42

3.6.4 DDR with Recapture (DDR-R) 44

4 Model Structure Analysis **47**

4.1 Tightening Representation of and Valid Inequalities for Model DDR 47

4.1.1 Tightening Representation of (8e). 48

4.1.2 Generalized Coefficient Reduction Using Generalized Upper Bound (GUB) Surrogate Constraints 52

4.1.3 Replacing (8g) with a Tighter Set of Valid Inequalities 56

4.2 Generalized Class of Facetial Valid Inequalities 59

4.3 Direct Construction of conv$\{Z_l\}$ 69

5 Numerical Experiments **73**

5.1 Experimental Design 74

5.2 Preliminary Experiments 76

5.2.1 A Priori Model Improvement 77

5.2.2 Convex Hull Construction 81

5.2.3 Generation of Valid Inequalities 83

5.2.4 Further Experiments on Set 2 Problems 84

5.2.5 Node Selection and Branching Priorities and CPLEX Cuts 87

5.3 Solution Approaches 89

5.3.1 Summary of the Proposed Heuristic Approach 90

5.3.2 The Proposed Optimal Approach 91

5.3.3 Experimental Results Obtained from Real-sized Problems 92

6 Summary, Conclusions and Future Research Directions **95**

6.1 Summary and Conclusions 95

6.2 Future Research Directions 98

Appendix **100**

References **103**

Vita **108**

List of Tables

1 Set notation: 31

2 Parameter notation: 32

3 Decision variable notation: 32

4 Average CPU Time Under Different CPLEX Cutting Strategies 78

5 Average CPU Time and Number of Branch-and-Bound Nodes with Propositions 1 and 3. 80

6 Numerical Results for Partial Applications of Proposition 3. 80

7 Average CPU Time for the Q-variable Application for Critical Flights . . . 82

8 Average CPU Time for the Separation Problem Application 83

9 Numerical Results from Different Approaches when Applied to the Set 2 Problems. 85

10 Average CPU Times for Two Node Selection Strategies. 87

11 Average CPU Times for Different Cuts Options. 88

12 CPU Times for Three 400-leg Problems. 89

13 CPU Times for Real Sized Problems. 93

List of Figures

1 Model Overview. 7

2 Illustration of Assignment Relating to Maintenance. 13

3 Integration of Simulation and Optimization in the D^3 Model. 19

4 Time-space Network Structure with Maintenance Arcs. 28

5 An Illustrative Example for the DDR Problem. 50

Chapter 1

Introduction

1.1 Introduction

It is a well accepted general concept in any industry that a better coordination of supply and demand management can yield higher revenues. In the airline industry, *supply management* refers to the assignment of aircraft types (each having a different capacity) and crews (each having different qualifications) to flight legs. Thus, it deals with assigning the "right resources to the right flights at the right time". Similarly, *demand management* (also known as yield or revenue management) refers to the ability to sell the "right tickets to the right customers at the right time" (American Airlines, 1987). Thus, a coordination of supply and demand management requires the integration of airline scheduling and revenue management decisions, which, however, are usually made in several distinct steps. According to Berge and Hopperstad (1993), airlines experience an average of 65% utilization of aircraft capacity, while, at the same time, losing many customers due to insufficient capacity. The high uncertainty of demand, coupled with the separation of demand forecasting, revenue management, pricing, and capacity assignment decisions, greatly hinders an airline's ability to realize more revenue. This situation could be improved by an integrated supply and demand management technique. In this research, we examine an enhancement of the Fleet Assignment Problem (FAP), based on the consideration of demand management. Specifically, we

present the concept of a Demand Driven Re-fleeting (DDR) approach, which updates fleet assignment decisions as more accurate demand forecasts become available over time.

This chapter is organized as follows. In Section 1.2, we present the terminology that will be used throughout this thesis. In Section 1.3, we briefly introduce the research problem and discuss the motivation for this research problem. Section 1.4 provides an overview of our modeling perspective, and presents the scope of this thesis. This is followed by a detailed description of the problem in Section 1.5. Finally, Section 1.6 presents the research contributions from this study.

1.2 Terminology

We first define some important terms that will be used throughout this thesis.

Aircraft Type - Certain model of aircraft, such as the MD 767-300. All aircraft of the same type have the same cockpit configuration, crew rating (crew qualifications requirement), maintenance requirements, capacity, etc.

Aircraft Family - A set of aircraft types, each with the same cockpit configuration and crew rating. Thus, the same crew can fly any aircraft type of the same family. An example of an aircraft family would be the Boeing 757/767 family, which consists of seven aircraft types, such as MD 767-300 and MQ 757-200, with capacity ranges between 182-244. In addition to different total capacities, the split between first-class capacity and coach capacity is also different for different aircraft types.

Tail - An individual aircraft.

Spill - The phenomenon that when demand is higher than the capacity assigned to the flight, excess customers will be lost due to capacity limitations.

Spoil - The phenomenon that when demand is lower than the capacity assigned to the flight, some seats will be unused due to low demand.

Leg/Flight Leg - A leg is a segment connecting two stops of a flight at a specific departure time, i.e., a leg spans the journey from the time an aircraft takes off until it lands.

Path - A path refers to a flight leg (or a set of flight legs) between a specific origin and destination, and at a specific departure time. Thus, there can be multiple paths between each origin-destination pair. For example, one of the paths between Roanoke and Denver consists of the Roanoke-Chicago flight that leaves at 10:30 am, followed by the Chicago-Denver flight that leaves at 2:30pm.

Through Flight - Two or more flight legs that are desirable to be flown by the same aircraft. Through flights are attractive to those customers flying multiple legs between their origin and destination. Even though the aircraft stops at an intermediate destination, the benefit to the customers is that they can stay on the same aircraft until they reach their final destination.

O/D - The origin and destination of a path.

Fare-class - A class corresponds to a particular fare restriction. For example, a Y fare is the unrestricted fare (i.e., after purchase, the departure day can be changed with no penalty), whereas a W fare is more restricted (i.e., the departure day can be changed only by incurring a penalty, and the ticket should be purchased at least two weeks in advance of flight departure).

Path-class - Naturally, a path-class refers to a particular path and fare-class combination.

Turn-time - The minimum time an aircraft needs between its landing time and the next take-off time. This includes the time for some minor inspections, preparation of the aircraft for its next trip, and its movement to the runway. Turn-time is aircraft and airport dependent, and is typically 30 - 40 minutes for domestic flights.

In the following section, we discuss the problem definition and the motivation that led us to study this problem.

1.3 Background and Motivation

One of the major airline operational problems is the schedule planning process, which usually consists of several sequential subproblems - the schedule design problem, the fleet assignment

problem, the crew scheduling problem, and the aircraft routing problem (Teodorovic, 1988). *Fleet assignment* refers to the assignment of aircraft types to flight legs, whereas *crew scheduling* refers to the assignment of each crew to those flight legs that it is qualified to fly. Finally, *aircraft routing* (*or rotation*) refers to the scheduling of the individual aircraft to routes, while satisfying the given fleet assignments. According to certain union contracts, crew scheduling needs to be completed well in advance of flight departures so that the crew can be notified early enough. This forces initial fleet assignments to be completed even earlier (usually a couple of months before flight departures) in order to provide the fleet information to the crew scheduling stage.

Demand is highly variable in the airline industry (Berge and Hopperstad, 1993). However, the accuracy of the demand forecast improves markedly over time. Consequently, as more information is gathered about demand, updates to the initial fleet assignment become appealing to implement in order to avoid customer *spill* or *spoil* in case when the observed demand is unexpectedly high or low as compared with the aircraft capacity assigned to that flight leg. Because of the need to preserve the initial crew schedule, however, a re-assignment of all fleets is prohibitive. This leads to the concept of *limiting fleet re-assignments to aircraft within the same family*, where the latter, as defined above, consists of different aircraft types, each having a different capacity, but with the same cockpit configuration. Hence a re-assignment of aircraft within one family will only change the aircraft originally assigned to a crew to one having a different capacity, but yet serviceable by the same crew. In what follows, we refer to the *re-assignment of aircraft within one family based on updated demand forecast* as the *Demand Driven Re-fleeting* (*DDR*) problem. This problem is the focus of the present thesis.

Revenue can further be improved at a later stage when more flexibility is added through dynamically swapping aircraft. Although only a limited number of exchanges are possible between a small number of aircraft due to the proximity to departure times, these swaps can make use of a more accurate demand forecast, and thus, can be promising for profitability.

In what follows, we present an overview of our model, and then provide additional

details regarding the DDR problem, its relationships with other scheduling problems that arise within the overall situational context, and research contributions in this thesis.

1.4 Model Overview

Clearly, there is high dependency between FAM (the Fleet Assignment Model), DDR, and swapping. The inputs to the FAM are the flight network, i.e., legs with O/D and departure times, as well as different *types* of available aircraft. The output of the FAM is the assignment of different aircraft types to legs. This assignment is next fed into the crew scheduling model to determine a crew schedule that is feasible with respect to the given fleet assignments. Recall that during the process of solving the DDR, we need to preserve the crew schedule; that is, each DDR problem can re-assign fleet types to flights within a particular family. Thus the DDR is confined to consider flight legs that are assigned to this family by FAM. While doing this, it can revise the given maintenance schedules significantly, as long as the stated maintenance requirements are satisfied. Observe that this reduction from all the families to only one family, and from all flight legs to the legs assigned to this particular family alone, reduces the problem size considerably (by approximately 70-80% on the average). Consequently, a more detailed analysis including the consideration of path demand information and demand randomness is possible in the DDR problem. On the other hand, even though the initial fleeting constrains the DDR solution (i.e., each flight leg in the re-fleeting can only be assigned to an aircraft of the same family as its initial aircraft), the revised fleeting can still differ significantly from the initial fleeting, by virtue of a re-assignment of aircraft types based on demand-capacity requirements. As a result, DDR will, probably, be performed only once before departure, and the time to perform DDR will be at least two or three weeks in advance of departure to avoid major disturbances to the planning process. Hereafter, we use *DDR assignment* to refer to the assignment obtained by the DDR model.

Naturally, demand forecast keeps improving even after the DDR is solved. To further utilize this information, *limited* swaps between different types of the same family can be

performed later on, with an attempt to preserve the revised fleeting to the greatest extent (mainly due to the desire or need to preserve airport services' schedules), and so as to take advantage of a more detailed demand forecast (specific for each day). The Demand Driven Dispatch (D^3) proposed by Berge and Hopperstad (1993), which will be reviewed in detail in the next chapter, is an example of this daily swapping process. In this swapping process, the DDR assignment replaces the initial assignment as an input to the daily fleeting application, and the swaps are selected such that soon after the swaps, the revised fleeting solution is still intact. This assures that the swaps can be potentially performed on a daily basis, based on the DDR assignment.

We envision the overall fleeting process to be somewhat similar to the one given in Figure 1. The focus of this thesis will be on the DDR problem. Analyzing the details of these two latter problems, along with the dependencies between all the three problems, is a long-term research goal and a future research direction.

In the next section, we provide further details on the DDR problem.

1.5 Problem Statement

1.5.1 Objective

The objective of the Demand Driven Re-fleeting (DDR) problem is to revise the initial fleeting assignments as and when better demand forecast becomes available, so as to maximize the total revenue.

As mentioned above, the FAM assigns each flight leg to an aircraft type. Given this assignment, we want to find a new assignment for each family so as to best "match" aircraft capacities with demands, with the intent that the spill and spoil, or lost customers and empty seats, are minimized, and the profit is maximized.

Clearly, a solution to the DDR problem needs to satisfy all the constraints that are present in the initial fleeting, which will be described in Section 1.5.2. As mentioned above, there are, however, two main differences between the initial fleet assignment and the DDR

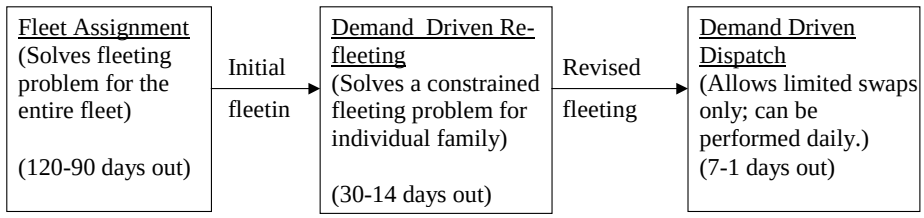


Figure 1: Model Overview.

problem: (1) The DDR problem is solved for each aircraft family, instead of the whole fleet. Thus, its size is considerably smaller than that for the fleet assignment problem. (2) At the time of solving the DDR, a more accurate and detailed demand forecast (at the path-class level) is available. Thus, a solution approach to the DDR problem should take advantage of the smaller problem size, and the improved and more detailed demand forecast.

Below, we introduce the primary constraints that need to be satisfied by the DDR problem.

1.5.2 Constraints

Similar to the fleet assignment problem, the DDR problem needs to satisfy the following sets of constraints. Recall that the DDR problem will be solved for each aircraft family, that is, all flight legs initially assigned to an aircraft type of a family in the initial fleet assignment still need to be assigned to an aircraft type within the same family.

- **Supply Management Constraints:** Constraints that ensure the feasibility of assigning aircraft types to legs. These are also the basic constraints used in the FAM, and include:

  Cover Constraints: Each flight leg originally assigned to a family needs to be assigned to an aircraft type within that same family; that is, each flight leg needs to be *covered* by an aircraft type from this family.

  Balance Constraints: The flow in and out of each airport needs to be balanced.

  Aircraft Availability Constraints: The number of available aircraft of each type is given and cannot be exceeded.

- **Demand Management Constraints:** These are constraints that ensure that projected sales do not exceed either capacities or demands.

- **Other Constraints:** These include constraints that integrate supply management and demand management decisions, as well as aircraft maintenance constraints. The details of maintenance constraints are described in Appendix 1.

1.6 Research Contributions

In this thesis, we propose an overall framework for supply management in airlines that takes advantage of flexibilities in the system capacity. This framework includes three sequential stages - the fleet assignment stage, the DDR stage, and the daily swapping stage. As the flight departure time draws near, these stages utilize more accurate forecast information to

better match the fleet capacities with demands so as to pursue better profits. The focus of this thesis is the DDR component of this framework.

There has been an increasing trend in recent years to integrate the different scheduling processes, but the literature on integrating the scheduling processes with revenue management is still limited. The proposed Demand Driven Re-fleeting model realizes this integration, and thus partially integrates the supply and demand management strategies in the airline industry.

In the fleet assignment as well as the re-fleeting literature, demands are generally assumed to be leg-independent. However, we will show later that demands are highly dependent between legs that share the same path. On the other hand, path-level demands more accurately reflect the real system. To our knowledge, this is the first effort that includes path-level demands into fleet assignment related models. Effective solution approaches are designed for such a DDR model, and are tested on practical sized problems to demonstrate its applicability in the airline industry.

However, this thesis does not attempt to answer the questions on the interrelation of the three stages in the framework, such as how do the initial fleet assignment, the DDR problem, and the subsequent swapping process interact, and what is the impact of the initial fleet assignment on the DDR and on the swaps. These extensions remain to be very interesting future research ideas.

Chapter 2

Literature Review

As mentioned in Chapter 1, the re-fleeting problem is highly constrained by the solution to the fleet assignment problem (FAP). Thus, in what follows, we first provide an overview of the literature on FAP in Section 2.1. Then, in Section 2.2, we discuss literature on different approaches for integrating FAP with other scheduling processes, and in Section 2.3, we present a review of the literature on the re-fleeting and swapping problems. Finally, in Section 2.4, we discuss literature on different solution approaches for stochastic optimization problems.

2.1 Fleet Assignment Problem

The Fleet Assignment Problem (FAP) is a major component of the airline scheduling process, and thus, has been studied at considerable length by researchers and practitioners. Most researchers use the same every day model (i.e., the same flight schedule is used every day of the planning period) for the FAP. They formulate the FAP as an integer problem (IP) and propose a variety of approaches to obtain a good solution in reasonable computing time (see, for instance, Abara 1989, Hane *et al.* 1995, Clarke *et al.* 1996). For general approaches in solving IP problems, we refer readers to Sherali and Driscoll 2000 for an extensive review.

Most IP formulations employ a time-space network (Gopalan and Talluri 1998, Clarke 1996, Talluri 1996), in which each airport at each event (arrival or departure time of a flight to/from that airport) is represented as a node. There are three types of arcs in the network: *ground arcs* representing aircraft staying at the same station for a given period of time, *flight arcs* representing each flight leg, and *overnight arcs* (or *wrap-around arcs*) used to connect the last event of the day with the first event of the following day. This "wrapping up" ensures aircraft balance at each airport, in addition to the same daily schedule.

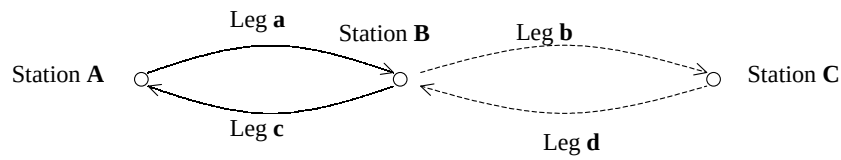
In time-space networks, one limitation is that the aircraft types are not differentiated when on the ground. The assumption is that all aircraft types have the same capability and are able to fly all flight legs. Rushmeier and Kontogiorgs (1997) formulated a multicommodity network structure to accommodate different aircraft type capabilities. In their network, each node is an event representing the start of an operation (f, e), i.e., a flight f operated by aircraft type e. Each *flight activity arc* represents a type e that is used to operate flight leg f and is also available for connection to flight leg g, and each *sit activity arc* represents a type e that is used to operate flight leg f and is also available not only to the immediate successor flight leg g, but also to all later flights. Compared with time-space networks, the sit activity arcs are similar to the ground arcs when no flight legs are prohibited for any types. On the other hand, this new network structure can easily grow unmanageably since the choice of flight connections is large. The authors suggested limiting the number of connections to a small size to keep the problem tractable.

Most IP models include the following three sets of constraints (Gopalan and Talluri, 1998): (a) *Cover Constraints*, which demand that each leg be assigned to one aircraft type. (b) *Balance Constraints*, which require that the flow in and out of an airport be balanced (i.e., the total number of aircraft flying in and currently at that station should equal the total number of aircraft that are either flying out or staying at the station). (c) *Availability Constraints* (*Count Constraints*), which require that the number of aircraft of each type that are assigned to flights cannot exceed the total available number. Hereafter, we will refer to these three sets of constraints as the *basic FAP constraints*. In the objective function,

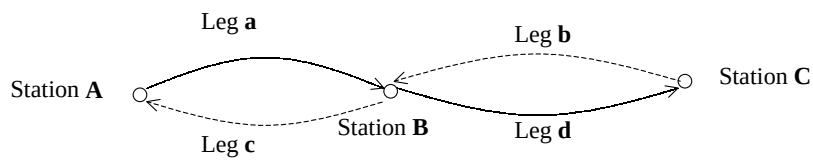
most models consider the cost of assigning a certain aircraft type to a flight leg. This cost term consists of two components: *the operating cost* and the *spill cost* (Abara 1989, Hane *et al.* 1995, Clarke *et al.* 1996). Although the operating cost is relatively predictable for each aircraft type and flight leg, the spill cost depends on the highly uncertain demand, and thus, can be very difficult to estimate (Yu and Yang, 1998). Thus, the demand variability and distribution are not explicitly considered in these integer programming formulations.

In addition, most formulations of this type consider a relaxation of the complete fleeting problem and do not include aircraft maintenance constraints. From the FAP output, each leg is assigned to one aircraft type in the family. However, a seemingly feasible assignment can be infeasible in practice when maintenance is considered. For example, as shown in Figure 2, in Example (a), legs **a** and **c** are assigned to aircraft type 1, and legs **b** and **d** are assigned to aircraft type 2. Aircraft type 1 might reside overnight at Station **A** and aircraft type 2 at Station **C**. Possible maintenance can be performed overnightly at these two respective stations if Station **A** is qualified to maintain type 1 and Station **C** is qualified to maintain type 2. Alternatively, in Example (b) of Figure 2, legs **a** and **d** are assigned to one type while legs **b** and **c** assigned to another. As a result, type 1 and 2 will take turns to reside overnight at the two stations **A** and **C** every other day. Both examples are feasible in terms of assignment without maintenance. However, the assignment in Example (b) is infeasible unless maintenance is always performed every other day, or both stations are qualified to maintain both types.

One of the few papers on the FAP that includes maintenance constraints is by Clarke *et al.* (1996), who propose two approaches. In the first approach, in addition to the basic FAP constraints, they include an additional constraint requiring no less than a given percentage of aircraft to be available at each maintenance station: this is a very simple approximation to the maintenance requirements. The second approach considers the maintenance requirements in greater detail, and utilizes a list of scheduled maintenance activities, which includes, for each maintenance activity, the maintenance station, aircraft type, earliest starting time and latest completion time for maintenance, and its expected duration. Thus,



Example (a) Type 1 is assigned to legs **a** and **c**, and Type 2 is assigned to legs **b** and **d**.



Example (b) One type is assigned to legs **a** and **d**, and the other type is assigned to legs **b** and **c**. The two types switch assignments every day.

Figure 2: Illustration of Assignment Relating to Maintenance.

there is a *time-window* corresponding to each maintenance requirement.

Recall that FAA requires four types of maintenance checks. Among them, the **C** and **D** checks take longer than 24 hours and are not considered in this model: aircraft scheduled to undergo these checks are set as "unavailable" in the model. Clarke *et al.*, then, classify the other two maintenance requirements into two groups based on the expected duration: the *short maintenance requirements* and the *long maintenance requirements*, which correspond to the **A** checks and the **B** checks, respectively. Based on this list, they create *maintenance arcs*, which depart from and arrive at the same station, and which are parallel to the

ground arcs. The duration of each long maintenance arc is, generally, longer than half of its time-window length. This ensures that no aircraft can take two long maintenance arcs - corresponding to the same requirement - in a row. This, however, is not the case for short maintenance, due to its duration. Thus, extra effort is made to prevent double-counting, which might happen when each maintenance duration is shorter than half of the time-window length, and thus, the same aircraft can take more than one short maintenance arc - of the same requirement - one after the other. To avoid this situation, each flight arc that can be followed by a short maintenance is duplicated: one is the real flight arc, the other is the flight arc *extended* by the duration of the short maintenance requirement. The cover constraints are next transformed to ensure that for each flight leg, either its real flight arc or its extended flight arc is covered by each aircraft type. This, however, increases the number of integer variables significantly. A relaxation is proposed, which includes a mix of maintenance arcs and these extended flight arcs.

Due to the large sizes of these integer problems, some researchers have started investigating the effectiveness of different solution approaches for the FAP. Sosnowska (2000) presents two heuristics for the FAP, both based on randomized local search procedures: Greedy Randomized Adaptive Search Procedure (GRASP) and Simulated Annealing (SA). Sosnowska defines a sequence of flight legs assigned to each aircraft as a *rotation cycle*. Turn-time is included in the constraints. Both GRASP and SA start with randomly generated initial solutions, which consist of a rotation cycle for each aircraft, and perform a search that may lead to a move from the current solution to one of its neighbors. A neighbor is created by a *swap operation*. The swap operation selects a pair of aircraft, determines a common airport in their rotation cycles, and switches the portions of the two rotation cycles following the common airport, if it is feasible to do so. In particular, the balance constraints and time constraints need to be satisfied after the swap, i.e., the first portion of one of the rotation cycles should terminate before the start of the second portion of the other rotation cycle.

In the GRASP algorithm, an inner loop and an outer loop are performed. The inner loop is a *Local Optimization Phase*, which randomly selects a fixed number of pairs of rotation

cycles and performs swap operations. The swap which improves the cost most significantly is stored. This is then passed to the outer loop, the *Construction Phase*, which fixes the best several swaps from this and previous iterations, and reassigns the remaining flights randomly to the other aircraft. Unlike GRASP, which retains only the improving solutions during the neighborhood searches, the SA accepts not only improving solutions but also a proportion of deteriorating ones, with the hope of not getting trapped at a local optimum. The expected demand and the cost of one hour of flight are used to compute the expected profit. There are two main drawbacks of the resulting model. First, demand versus capacity information and demand stochasticity are not included in the initial assignment as well as the swapping steps. Thus, the initial solutions and the potential swaps are generated randomly, and are not based on a better assignment possibility. Second, maintenance constraints are not considered in the model.

2.2 Integrated Approaches to FAP and other Scheduling Processes

As stated in the previous chapter, solving FAP without considering its integration with demand management and the flight network scheduling has some drawbacks, and is not necessarily beneficial to the total revenue management process. Teodorovic (1988) gives a simple example on the integration of FAP and schedule planning. In his example, aircraft are assigned to the network considering flight frequencies, where the operating cost is multiplied by the flight frequency between two stations flown by the same aircraft type to compute the total daily cost. Larger aircraft tend to be assigned to smaller frequency and longer distance flights. Smaller aircraft have lower operating costs, but this can be compensated if the flight frequency is high.

In recent years, instead of the traditional approach of solving different components of the airline scheduling problems separately, researchers have started focusing on an integrated approach for solving the entire scheduling problem.

Given a fleet assignment for a single aircraft type, aircraft of this type can be routed to different selections of flights. Searching for a feasible aircraft routing with the minimal

cost is the objective of aircraft routing problem. Under the traditional approach, a routing problem is solved for individual aircraft types after the FAP. The local optimizations for all the types may not lead to a global optimization for the entire fleet in the airline. In addition, restricted within a single type, infeasibility is more likely to occur as a result of less choices of flights. Barnhart *et al.* (1998) present a string-based MIP model to solve the fleet assignment problem and aircraft routing problem simultaneously for the entire fleet. They define a *string* as a sequence of flow-balance and maintenance feasible flights between maintenance stations. A *augmented string* is a string extended to include the maintenance time at the end of the last station. The model objective is to assign a subset of these augmented strings to the aircraft types so that the cost is minimized under flow and cover constraints. The model is solved using a *branch-and-price* approach, i.e. a LP relaxation is solved at each node of the branch-and-bound tree using column generation. When generating columns, the pricing subproblem is cast as a shortest path problem over a specially constructed network. This network is the time-space network with maintenance arcs carrying maintenance costs, connection arcs carrying flight-related costs for the starting node and maintenance connection arcs carrying maintenance costs and (negative) through revenues. Using the node and islands concept by Hane *et al.* a solution can be achieved in around 5 hours, which is acceptable for a planning model, especially when the maintenance requirements are satisfied.

Cao and Fan (2000) propose a model that integrates the flight network scheduling problem, the fleet assignment problem, and the aircraft routing problem. Their model is based on an initial candidate network and a set of candidate flights. They consider time-dependent demands, and simultaneously determine the flight network, the fleet assignment, and the aircraft routing decisions. They introduce the idea of a Time-related Candidate Flight (TCF), which is a candidate flight associated with a certain time slot. The problem is to assign each aircraft to an appropriate TCF so that the total profit can be maximized. They consider maintenance as a time-window for the aircraft. No TCF can be assigned to this time-window since no flight can be flown by an aircraft under maintenance.

Rexing *et al.* (2000) creatively apply time-windows to the FAP so as to integrate fleet assignment with the configuration of the underlying flight network. Allowing flight departure times to vary within a small time-window relaxes time constraints, and provides more choices of assigning aircraft to flights, and thus might improve the fleet assignment solution. Not surprisingly, the problem size grows along with this added flexibility. The authors propose various approaches to reduce the problem size.

In all these proposed fleet assignment models, revenue is calculated, if at all, using the average demand aggregated over the planning horizon, with little information on demand distributions. Consequently, an improvement to the model could be achieved through the utilization of demand distributions and a more accurate and detailed demand forecast, which is acquired as the aircraft departure time draws near.

Thus, a re-fleeting, based on the added information that these fleet assignment models lack, could provide significant benefits to the airline. The next section provides a brief summary of the literature on re-fleeting related problems.

2.3 Re-fleeting Problem

Research in the re-fleeting area is relatively new, and hence, is very limited. Berge and Hopperstad (1993) are one of the first researchers who proposed this break-through idea of revising fleet assignments over time for each aircraft family. They proposed the concept of *Demand Driven Dispatch* (D^3), which is to dynamically change aircraft type assignments as the flight departure times approach and forecasts improve. They restrict the change of assignment to one aircraft family due to the need to preserve crew schedules.

Berge and Hopperstad formulate the fleet assignment problem as a multi-commodity network flow problem. This, however, consumes prohibitive computational effort. Alternatively, two heuristics are proposed to yield a more efficient performance. The Sequential Minimum Cost Flow Method (SMCF) solves a sequence of two-type assignment problems based on the observation that a fleet assignment problem with only two types can be reduced to a single commodity minimum cost network flow problem. This is done by assigning

all flights to one aircraft type first, and then solving a minimum cost network flow problem by setting the objective function to be the difference in profits when the flight is assigned to the other type. For more than two aircraft types, a minimum cost network flow problem for two types is solved at each iteration, by separating one aircraft type from other types, and aggregating all other types to form the second type. A series of aggregated two-type assignment problems are solved until all the types are assigned. The authors suggest aggregating aircraft types based on a sorted order of their capacities. The second method, the Delta Profit Method (DELPRO), starts with a feasible solution to the fleet assignment, and finds profitable swaps for two aircraft types at a time. This is done by using a longest path algorithm. In their computational experiments, the authors relax the integrality constraints of the multi-commodity network flow problem and solve the corresponding Linear Program (LP). All LP solutions turned out to be integral, thus yielding optimal solutions. These optimal solutions were then compared with the heuristic solutions obtained by DELPRO and SMCF. DELPRO and SMCF respectively attained an accuracy of in excess of 99.9% and 99.99% of optimality. More importantly, these algorithms were found to be sufficiently efficient for the model to be solved frequently (probably daily).

Since the network flow problem is deterministic and no randomness in demand is considered, the revenue corresponding to a given fleet assignment is determined through the use of a simulation model, which generates customer demands based on a forecasted demand distribution. Figure 3 exhibits how the optimization and simulation are integrated in their approach.

As a result of this simulation model, loads, spill and denied boarding can be predicted. Thus, the revenue corresponding to the given fleeting decision is determined considering demand stochasticity, and this information is fed back to the airplane assigner. This reassignment, simulation, and booking predictor and profit estimation process is repeated continuously until the departure time is imminent, and the final assignment is determined. See Figure 3 for this integrated model. The case studies performed showed a 1-5% improvement in operating profit from the application of D^3. This is due to spill avoidance and a

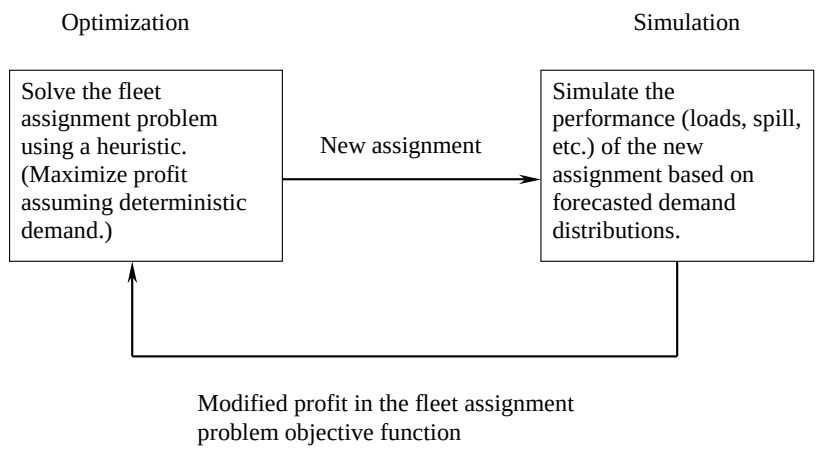


Figure 3: Integration of Simulation and Optimization in the D^3 Model.

reduced number of larger airplanes used.

To our knowledge, there is no major US airline that is implementing the concept of Demand Driven Dispatch to its full extent. However, several airlines are in the process of evaluating the benefits of a D^3 type of approach. An example is Continental Airlines (Pastor, 1999), which has performed limited experiments to test the benefits of the D^3 concept, by considering two types of swaps: 60 DTD (Days to Departure) swaps, and 14 DTD swaps. The former involves different families and thus, might require the crew schedule to be changed, while the latter is confined within one family, but is based on improved

forecasts being available as flight departures near. By considering only the expected revenue gain, occasionally very poor swaps could result because of demand stochasticity. To avoid the negative impact of such unsuccessful swaps, a Monte Carlo Simulation is used to perform a risk analysis so that a distribution of revenue gains for the swap is obtained. With detailed information on the average, variation, minimum, and maximum revenue gains resulting from a swap, together with the percent of trials that yield positive revenue gains, as well as the percent that meet the minimum revenue goal, a better understanding of the potential benefits is achieved. Based on this information, only swaps that meet the minimum revenue goal in 80% of the trials are performed. Experimental results indicate that such a risk analysis improves the actual revenue gain by 8%, and greatly reduces the number of unprofitable swaps performed.

Talluri (1996) improves Berge and Hopperstad's swapping algorithm by reducing its run time. His algorithm guarantees finding a same-day swap opportunity if one exists, which may involve multiple aircraft swappings and multiple stations. However, Talluri's algorithm is limited to two aircraft types only. This algorithm is based on a series of shortest-path algorithms, and can accommodate through flights. Talluri further suggests heuristics to address the issues of different ground times.

Jarrah *et al.* (2000) model the re-fleeting problem as a multicommodity integer network flow problem, with side constraints which, different from Talluri's approach (1996), is applicable to multiple aircraft types and generates a specified number of acceptable solutions. Their basic model is similar to the leg-based FAM, commonly used in literature as discussed in the previous section, with node balancing constraints, flight leg covering constraints, and aircraft availability constraints. They also include constraints in the formulation to place a user-specified limit on the number of fleet type changes. In addition, side constraints for maintenance, crew staffing, and noise restrictions are added for solution feasibility. Around this model, the authors propose five modules for different applications. These modules are primarily targeted at unexpected changes to the FAM, but none of the modules directly

addresses better demand forecasting. Each module resolves the FAM with a slight modification on the constraints. The *Popping Module* allows a user-specified number of aircraft are to be removed from the schedule (e.g. for maintenance). The *Change-of-Gauge Module* is used when a certain number of aircraft is to be reduced for one fleet type and increased for another. This can be a result of miss estimation of aircraft availability in the schedule period. The *Swapping Module* is identical to the FAM except for a limit on the number of allowable swaps for a user-specified fleet types. These first three modules modify the cover constraints of the FAM. The *Utilization Module* changes the total assignment time between two groups of aircraft types. An example application of this module is when the crew staffing level is violated by a particular aircraft type's flight assignments. The *Balancing Module* balances the schedule after some fleet assignments are changed by the users. A pre-processing stage is utilized to detect infeasibilities in the model before the optimization phase. The objective of the model is not to generate one optimal solution, but several alternative near-optimal solutions to allow the user choose the most "appropriate" solution from an operation's perspective. The mixed integer program is solved using a depth-first branch-and-bound approach taking the advantage of function calls in the CPLEX's mathematical optimization subroutine library. When one good solution is achieved, a cut is added to eliminate this solution and ensure that the new solution contains a strict subset of the already obtained solution. Computational experiments for realistic problem sizes suggest that multiple acceptable solutions can be generated in typically less than five minutes as a result of effective preprocessing and solution techniques.

2.4 Stochastic Optimization Approaches

The primary difficulty in re-fleeting, as in most real life problems, is the uncertainty. In DDR, the number of potential customers is highly uncertain. The stochastic aspect of passenger demand considerably hinders an optimum solution to the scheduling problem. Thus, it is highly desirable to make use of an optimization approach that explicitly considers this stochasticity. Below, we provide an overview of the different approaches that can be

used for stochastic optimization. Although these stochastic optimization approaches are not adopted in this thesis, they are good alternatives for conducting future research.

2.4.1 Meta-Heuristics

Meta-heuristic is a general framework for heuristics in solving global optimization problems, notably combinatorial optimization problems. One hurdle for optimization methods is to avoid the solution from being trapped at a local optimum. Some meta-heuristics, such as Simulated Annealing and Tabu Search, have proven to be effective in escaping from local optima.

Simulated annealing (SA) was derived from statistical mechanics as an analogy to the way a crystalline solid performs at the limit of a low temperature when allowed to cool slowly (the annealing process) into a minimum energy crystalline structure (Kirkpatrick *et al.*, 1983).

The SA algorithm avoids being trapped in a local optimum by accepting a fraction of non-improving solutions. It generates neighbor solutions for the current solution. If a neighbor improves the function value, the current solution is then replaced. Otherwise, the neighbor is accepted with a certain probability; often the parameter or function used follows an exponential decay. This process is repeated until the termination criterion is satisfied, and the final solution prescribed is the best local optimum detected in this process.

```
Select initial solution x_0;
Initialize T > 0, temperature counter t = 0;
DO
    FOR n = 1 to N(t) DO
        Generate a new (neighbor) solution x_n;
        δ = f(x_n) − f(x_0);
        IF δ < 0 or random(0,1) < e^(−δ/T) THEN
            x_0 = x_n;
        END IF
```

END FOR

$t = t + 1;$

$T = T(t);$

UNTIL termination criterion is true.

Table 2.4.1 A Generic SA Algorithm.

A simplified process is given in Table 2.4.1 (Eglese, 1990; Smith *et al.*, 1995), where the objective is to minimize function f. The probability of accepting an inferior neighbor solution is set as $e^{-\delta/T}$, where δ is the deterioration in the objective function (an increase for a minimization problem), and T is a control parameter that is comparable with the temperature in physical annealing. A small δ implies that the new solution is not a significant deterioration from the current one and is more likely to be accepted. A high T value also gives a high possibility for the solution to be accepted. The algorithm starts with a relatively large T value to avoid being prematurely trapped in a local optimum. This T value is gradually decreased, following some function $T(t)$. The initial value of T, the function $T(t)$, as well as the number of iterations at each temperature, $N(t)$, are some basic choices for SA algorithm users (Eglese, 1990).

2.4.2 Robust Optimization

Robust Optimization (RO) is an approach that attempts to reduce the impact of noise, errors, and incomplete data on solutions, by considering uncertainty in the formulation, while allowing infeasible solutions, with a penalty. Thus, in addition to *decision variables*, it includes *control variables* so as to reflect the uncertainty, as well as a set of *error variables* to measure the infeasibility. Mulvey and Vanderbei (1995) provide a detailed introduction and some example applications of this method. They formulate the RO model as follows:

$$Minimize \quad \sigma(x, y_1, \ldots, y_s) + \omega\rho(z_1, \ldots, z_s)$$

$$\textit{subject to}$$

$$Ax = b$$

$$B_s x + C_s y_s + z_s = e_s, \quad \forall\ s \in \Omega$$

$$x \geq 0,\ y_s \geq 0, \qquad \forall\ s \in \Omega$$

where $\Omega = \{1, \ldots, S\}$ is a set of scenarios, x represents the primary first-stage decisions, and y_s and z_s respectively denote the control or recourse variables and the error variables corresponding to scenario $s \in \Omega$. Each scenario $s \in \Omega$ is associated with the set $\{B_s, C_s, e_s\}$ of realizations of the coefficients, and the probability of scenario s occurring is p_s, with $\Sigma_{s=1}^{S} p_s = 1$.

The first term in the objective function is called the *solution robustness term.* It takes the value of $\sigma(x, y_s)$ when y equals y_s, with probability p_s, and can be used to model risk. The authors suggest two approaches for modeling risk: the mean/variance models of Markowitz (1991) and the utility models of von Neumann-Morgenstern (1953). An example for the former approach is to have the mean plus a constant times the variance as the model robustness term, where the mean is the expected objective function value over all the scenarios, and similarly, the variance is computed over all the scenarios for the objective function value. The utility models permit users to assign a utility (weight) to the outcome of each scenario. Thus, this model can utilize a concave utility function as the solution robustness term. Compared to the mean/variance models that use the objective probabilities as parameters, the utility models are more subjective and reflect the users' preferences. The mean/variance models require symmetry for the distribution of the scenario, while the utility models are more general.

The second term in the objective function is a feasibility penalty function and is called the *model robustness term.* Unlike Stochastic Linear Programming (SLP), which requires feasibility for all scenarios, RO allows infeasibilities for the various scenarios, but at a penalty.

Clearly, the first term and the second term form a tradeoff between *model robustness* (more emphasis on feasibility and less on optimality) and *solution robustness* (more emphasis

on optimality and less on feasibility). This tradeoff is controlled by the parameter ω, which can be adjusted according to the risk attitude of the decision maker.

RO is different from sensitivity analysis, as the latter is a post-optimality study, providing only measures, but no preventions or recourse, to the sensitivity of the solution. Compared to SLP, they are both pre-optimality analysis, and SLP is a special case of RO in that SLP considers only the first moment of the objective function value, while RO considers higher moments. This renders the latter approach more passive, but also more expensive to implement. Finally, the authors point out that even sequential computers are unable to handle the tasks for more than a small number of scenarios.

Chapter 3

Model Formulation

3.1 Outline

In this chapter, we present a model formulation of our Demand Driven Re-fleeting (DDR) problem. Recall that the Fleet Assignment Model (FAM) assigns a set of flight legs to each *aircraft family*, where an *aircraft family* refers to a set of *crew compatible aircraft types*, and each *aircraft type* has a different capacity. The DDR model is, then, solved for each family so as to re-assign aircraft types of that family to the involved flight legs. As explained in the previous chapter, this restriction is necessary in order to preserve the initial crew schedule. The motivation for DDR is to take advantage of an improved demand forecast that might be obtained as the flight departure times draw near. Since DDR is solved for each family (instead of the entire fleet), the problem size is greatly reduced. This allows a more detailed analysis, as explained in this chapter.

An airline's *supply management* refers to the airline's ability to manage its *supply* through flight scheduling, fleeting, and crew scheduling decisions, whereas *demand management* refers to the airline's ability to manage its *demand* through revenue management and pricing. The main idea behind the DDR model is to **integrate** an airline's *supply management* and *demand management* decisions, while focusing on fleeting and revenue management. However, it is an important future research direction to extend the concepts

proposed in this project to include other components of an airline's supply and demand management such as flight scheduling, crew scheduling, and pricing.

This chapter is organized as follows. We first present, in Section 3.2, the underlying network structure of our model. In Section 3.3, we introduce the notation that will be used throughout this thesis. In Section 3.4, we review the "leg-based" FAM formulation commonly used in the literature. In Section 3.5, we focus on path-based models. Specifically, we introduce a revenue management model that has been receiving much attention in recent research. In Section 3.6, we present our proposed DDR models which, as mentioned above, aim at *integrating* fleeting and revenue management decisions.

3.2 Background: Network Structure

As mentioned in the previous chapter, FAM is solved in aircraft-type detail, and with "aggregated maintenance constraints", if maintenance is included in the model. Specifically, we consider a particular family of aircraft along with all the flights assigned to that family in the FAM. We set up our network as a time-space network, similar to those commonly used in the literature (see, for instance, Rexing *et al.* (2000)). In what follows, we provide details on this network structure.

In this network, there is a *node* associated with each event (flight arrival or departure at a specific time) and each aircraft type. This allows for aircraft type dependent flight times and turn times in our network representation. If flight times and turn times are not significantly different among the various aircraft types within one family, a composite network can be constructed for the entire family. In order to have feasible aircraft connections, arrival nodes are placed at each flight's ready time, instead of the departure time. There are three types of arcs: *ground arcs*, *flight arcs*, and *overnight arcs*. An example network for a specific aircraft type with four stations (airports) and six legs is given in Figure 4. The x-axis depicts the various stations involved in the family's assignment and the y-axis indicates time. The solid vertical arrows are *ground arcs* representing aircraft staying at the same station for a period of time. The ground arcs are connected by *flight arcs*, which are directed

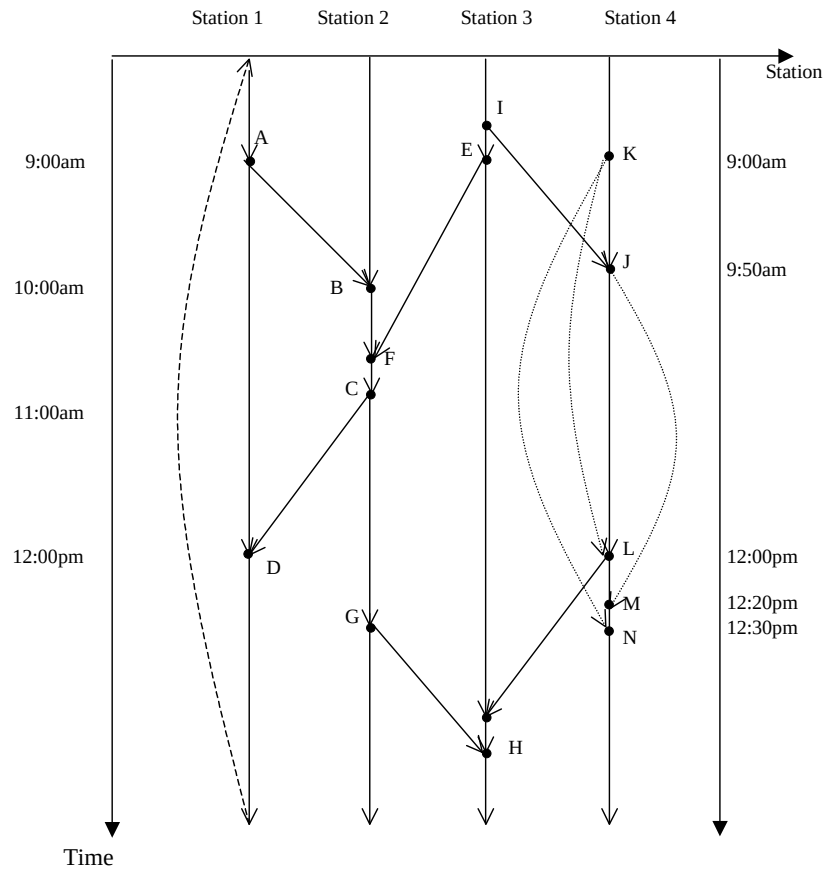


Figure 4: Time-space Network Structure with Maintenance Arcs.

from their origin station to their corresponding destination station. An example of a flight arc is arc AB (see Figure 4), which represents the flight from Station 1 to Station 2 with a departure time of 9:00am and a ready time (arrival time plus turn time) of 10:00am. The *overnight arcs* (or *wrap-around arcs*) are special cases of ground arcs, which guarantee flow balance by connecting the last node (event) of each station to the first node at that station. Hence, they are included in the set of ground arcs in our formulation. For the sake of graph simplicity, in the figure we only show the overnight arc (the long dotted arc) for Station 1.

In addition to these "basic" time-space network arcs, we include *maintenance arcs*, represented by the short dotted arcs in our network, to model the **A** and **B** checks, which are specified by a "maintenance requirement list". As explained previously, each "maintenance requirement" in the list specifies (1) the corresponding maintenance station, (2) the type and (3) the number of aircraft due for that particular maintenance, (4) the maintenance duration, and (5) the maintenance time-window. The maintenance time-window specifies the earliest starting time and the latest completion time during which this specific maintenance activity can be performed. Clearly, the maintenance time-window length must be at least as long as the required maintenance duration. The maintenance arcs are, thus, arcs that "leave" and "return to" the same station within the corresponding time-window, with duration equal to that of the maintenance requirement. For each maintenance requirement, we first add an arc "departing" at the earliest start time, and "returning" after the maintenance duration. The aircraft already on the ground at the earliest start time can take this arc for maintenance. We also add a maintenance arc for each flight that arrives after the foregoing earliest start time, but can complete the required maintenance by the latest completion time. Observe that maintenance arcs for a specific maintenance requirement will be added only to the node of the corresponding aircraft type.

As an example, consider, again, the network given in Figure 4, which corresponds to a specific aircraft type, and suppose that we have two maintenance requirements for this aircraft type at Station 4: Requirement 1 has a time-window of [9:00am, 12:00pm], and

Requirement 2, has a time-window of [9:00am, 12:30pm]. Both requirements have maintenance durations of 2.5 hours. We first add two maintenance arcs, arc KL for Requirement 1 and arc KN for Requirement 2, that start at the earliest time possible (9:00am for both requirements): these are for the aircraft already on the ground. Next, we search for all flights arriving at this station after 9.00am that can undergo maintenance and be completed by the latest completion time of the corresponding maintenance requirement. Observe that flight IJ satisfies this condition for Requirement 2: it arrives at Station 4 at 9:50am, after which it can undergo the 2.5 hour maintenance, and still complete before 12:30pm, i.e., at 12:20pm. Thus, we add another maintenance arc, arc JM for Requirement 2, right after flight arc IJ.

The size of this network can be reduced through the techniques suggested by Jarrah *et al.* (1997). Furthermore, as mentioned above, if all aircraft types have identical flight and turn times, then they can share nodes of the same network.

We now present various mathematical formulations of FAM and DDR based on this network structure. For this purpose, in the next section we introduce the notation that will be used throughout this thesis.

3.3 Notation

Tables 1 - 3 explains the notation used throughout this thesis.

Table 1: Set notation:

K	:	set of aircraft families in the fleet, indexed by k
T_k	:	set of aircraft types in family k, indexed by t
T	:	set of all aircraft types in the fleet, where $T = \cup_{k \in K} T_k$
L_k	:	set of flight legs (arcs) assigned to family k in FAM, indexed by l
L	:	set of all flight legs, where $L = \cup_{k \in K} L_k$
N_t	:	set of nodes in type t's network, $t \in T$; indexed by n
G_t	:	set of ground arcs in type t's network, $t \in T$; indexed by g
OD	:	set of origin-destination pairs, indexed by (o, d)
Π	:	set of all paths, indexed by i
$\Pi(l)$	:	set of paths that pass through leg l, $l \in L$
$\Pi_{o,d}$	:	set of paths with origin o and destination d, $(o, d) \in OD$
C	:	set of classes, indexed by c
CS_t	:	set of arcs passing through a counting time line (e.g., 3am EST) in type t's network, $t \in T$

The role of the sets $CS_t, t \in T$, is to count the number of aircraft of each type by summing the flow on all arcs passing through an arbitrarily chosen time line. A commonly used time line is 3am EST, because only a few (red-eye) flight arcs pass through this line, and all the rest are ground arcs or maintenance arcs.

Table 2: Parameter notation:

c_{lt}	:	cost of covering leg l using aircraft type $t,\ l \in L,\ t \in T$
Cap_t	:	capacity of aircraft type $t,\ t \in T$
μ_i	:	mean demand on path i, $i \in \Pi$
f_i	:	fare price on path i, $i \in \Pi$
(o_i, d_i)	:	origin and destination of path $i,\ i \in \Pi$
A_t	:	number of available aircraft of type t, $t \in T$

$$bf_{ln} = \begin{cases} 1, & \text{if flight } l \text{ begins at node } n \text{ (in aircraft type } t\text{'s network)}, \\ & l \in L,\ n \in N_t,\ t \in T \\ -1, & \text{if flight } l \text{ ends at node } n \text{ (in aircraft type } t\text{'s network)}, \\ & l \in L,\ n \in N_t,\ t \in T \\ 0, & \text{otherwise} \end{cases}$$

$$bg_{gn} = \begin{cases} 1, & \text{if ground arc } g \text{ begins at node } n \text{ (in aircraft type } t\text{'s network)}, \\ & g \in G_t,\ n \in N_t,\ t \in T \\ -1, & \text{if ground arc } g \text{ ends at node } n \text{ (in aircraft type } t\text{'s network)}, \\ & g \in G_t,\ n \in N_t,\ t \in T \\ 0, & \text{otherwise.} \end{cases}$$

In addition, we use μ_X or $E[X]$ to denote the expected value of random variable X.

Table 3: Decision variable notation:

$$x_{lt} = \begin{cases} 1, & \text{if flight leg } l \text{ is flown by aircraft type } t,\ l \in L,\ t \in T \\ 0, & \text{otherwise} \end{cases}$$

y_g	:	number of aircraft (of type t) on ground arc g in type t's network, $g \in G_t,\ t \in T$
q_i	:	number of passengers flown (accepted demand) on path i, $i \in \Pi$

Note that x_{lt} are binary decision variables. We assume that all the types in T_k of family k can be assigned to all the legs in L_k. If, in some special cases, certain type $t \in T_k$ cannot be assigned to leg $l \in L_k$, we can set an upper bound of 0 to x_{lt} so as to prohibit x_{lt} from becoming 1.

Finally, recall that a *leg* refers to the segment connecting two stops of a flight at a specific departure time, whereas a *path* refers to a flight leg (or a set of flight legs) between a specific origin and destination and at a specific departure time. Thus, there can be multiple paths between each origin-destination pair, and each path is uniquely determined by the set of legs it consists of.

In what follows, we first present, in Section 3.4, the basic (i.e., "leg-based") FAM formulation commonly used in the literature. In Section 3.5, we describe a "path-based" Revenue Management System that is recently proposed in the literature. In Section 3.6, we present our "path-based" DDR models.

3.4 A Leg-Based Fleet Assignment Model

We first present the leg-based FAM formulation commonly used in the literature (see, for instance, Rexing *et al.* (2000)). A time-space network is constructed for each aircraft type, as explained in Section 3.2, with the difference being that each type's network now includes all the flight legs that it is qualified to fly (as opposed to flight legs assigned to its corresponding family, as will be done for DDR). However, in order to simplify notation, in what follows we will use the same notation (N_t) to describe the set of nodes in type t's network, $t \in T$.

Most FAM formulations focus on minimizing the total cost of assigning aircraft types

to flight legs.

Model 1 Basic Leg-based FAM (FAM-L):

$$\textit{Minimize} \quad \sum_{l \in L} \sum_{t \in T} c_{lt} x_{lt}$$

subject to

$$\sum_{t \in T} x_{lt} = 1, \quad \forall\ l \in L \tag{1a}$$

$$\sum_{l \in L} bf_{ln}\ x_{lt} + \sum_{g \in G_t} bg_{gn}\ y_g = 0, \quad \forall n \in N_t,\ \forall t \in T \tag{1b}$$

$$\sum_{l \in CS_t} x_{lt} + \sum_{g \in CS_t} y_g \leq A_t, \quad \forall\ t \in T \tag{1c}$$

$$x_{lt} \in \{0, 1\}, \quad \forall\ l \in L,\ \forall t \in T \tag{1d}$$

$$y_g \geq 0, \quad \forall\ g \in G_t,\ \forall t \in T. \tag{1e}$$

The focus of FAM is to find the "same everyday fleeting". Constraints (1a) are the *cover constraints*, requiring each leg to be assigned to exactly one aircraft type. Constraints (1b) are the *balance constraints*, requiring aircraft flow to be conserved at each node. Constraints (1c) are the *count constraints*, which ensure that the number of aircraft of each type used cannot exceed the total number of available aircraft of this type. Finally, Constraints (1d) and (1e) require the assignment variables, $\mathbf{x}$, to be binary, and the ground variables, $\mathbf{y}$, to be nonnegative. Observe that the integrality of variables $\mathbf{y}$ is implied by the integrality of variables $\mathbf{x}$, and hence, need not be explicitly stated in the formulation.

As mentioned in the previous chapter, most models employed in the literature determine the cost, c_{lt}, of assigning type t to leg l based on two components: *the operating cost* and the *spill cost* (Abara (1989), Hane *et al.* (1995), Clarke *et al.* (1996)). Observe that the

expected spill on leg $l,\ l \in L$, resulting from assignment of type $t,\ t \in T$, denoted as $E[s_{lt}]$, can be determined by:

$$E[s_{lt}] = \int_{Cap_t}^{\infty} (D_l - Cap_t)\ h(D_l)\ dD_l, \tag{2}$$

where $h(.)$ denotes the probability density function of leg demand D_l, which is usually modeled as a continuous normal distribution. Of course, it is also possible to maximize the total revenue in this formulation.

Observe that the constraints in Model 1 do not include any *passenger demand* and *aircraft capacity* information. This information can only be considered in the cost term of the objective function. Thus, Model 1 *manages supply* by assigning aircraft type (thus, capacity) to each leg, based on *exogenously determined leg demands.* Consequently, we refer to constraints (1a)–(1e) as the *"supply management"* constraints.

It is possible to extend the basic FAM such that it also includes *aggregate maintenance constraints* at the type level (see Appendix 1 for such an extended formulation).

The next section discusses path-based approaches for airline supply and demand management.

3.5 A Path-Based Revenue Management System

The "leg-based" FAM given in Model 1 considers leg demands as independent and consists only of *supply management* constraints. However, in the United States, about 40% of the passenger trips consist of multiple legs, because of the hub and spoke networks. Clearly, the demand coming from a passenger, who needs to fly multiple legs between her origin and destination, will be dependent on the availability of seats on all legs: if seats on one

leg are unavailable, then this demand will be lost. Thus, (1) leg demands can be highly dependent, and (2) passengers, who demand to fly on a particular leg, are *not identical* in terms of the revenue that they will generate and/or airline resources that they will consume. Therefore, airlines need to utilize techniques so as to *manage their demand* in the most optimal way by deciding on which passengers to accept on which flights, considering dependencies in demands. In addition, airlines can exercise some control over their demand through product (seats on a flight leg) differentiation, which is mostly done based on time of purchase (this is commonly referred to as *revenue management*), as well as their *pricing* decisions; see McGill and van Ryzin (1999) for an extensive review of research on revenue management.

Obviously, an airline's fleeting decisions will greatly constrain the extent to which it can manage its demand. Similarly, an airline has a high level of control over its demand, and therefore, leg demands are not exogenously determined inputs to the fleet assignment model. However, as stated previously, the traditional airline planning process generally makes supply management and demand management decisions sequentially, with little or no interaction between each other. In addition, most research on revenue management has focused on policies that are optimal for a single flight leg. A new stream of research is recently emerging in the revenue management literature, focusing on policies considering path demands. Thus, in the following, we start by considering such a path-based revenue management system. We refer the interested reader to McGill and van Ryzin (1999) for an extensive review of research on revenue management.

Specifically, we discuss a "Displacement Adjusted Virtual Nesting (DAVN)" type of a

revenue management system (see, for instance, Belobaba and Lee (2000)). Such a Revenue Management System considers fleet assignments (the capacity assigned to each flight leg) as given, and determines the path demand accepted by the system so as to maximize total revenue. Thus, it *manages demand*, considering a fixed supply given by the fleeting solution.

In the following, we will not provide all the details of DAVN (see Belobaba and Lee (2000) for details), but present a particular Linear Program (LP), which is an important component of these systems.

Recall that q_i is the *accepted demand* on path i, $i \in \Pi$; these are our *decision variables* (see Section 3.3 for the notation). Given a fleeting solution, we let, for each $l \in L$,

$$CapAss_l \equiv \sum_{t \in T} Cap_t x_{lt}$$

denote the capacity assigned to leg l in the given fleeting solution. Thus, these capacities are *inputs* to the DAVN type of an RM System, which makes use of the following LP, usually solved for each departure day.

Model 2 Path-based RM Model (RM-P):

$$\textit{Maximize} \quad \sum_{i \in \Pi} f_i \; q_i$$

subject to

$$\sum_{i \in \Pi(l)} q_i \leq CapAss_l, \quad \forall \; l \in \; L \tag{3a}$$

$$q_i \leq \mu_i, \quad \forall \; i \in \; \Pi \tag{3b}$$

$$q_i \geq 0, \quad \forall \; i \in \; \Pi \tag{3c}$$

Constraints (3a) require that the number of passengers flown on each leg does not violate the capacity of the aircraft type assigned to that leg, and constraints (3b) ensure that the expected demand on each path provides an upper limit on the number of passengers flown on that path. Hence, constraints (3a)–(3c) determine the number of passengers flown on each path i (q_i) by considering the resource consumption that each path passenger requires. Therefore, we will refer to constraints (3a)–(3c) as the *"demand management"* constraints. In reality, the mean demand estimates, μ_i, are not necessarily integers. Thus, by also not requiring q_i to be integral, we are looking for an approximate solution.

To summarize, to our knowledge, most airlines are currently making their fleeting and revenue management decisions in a sequential manner. Although some airlines have started using path level demands in their revenue management systems, most fleet assignment models are still based on leg level demands. In addition, demand forecasts are highly uncertain when fleet assignment decisions are made, but forecasts improve significantly over time. All these statements hold for our industry partner, a major US airline. As a result, a re-fleeting model that would allow the airline to integrate its supply and demand management decisions, while taking advantage of an improved demand forecast obtained as departures near, could greatly benefit the airline.

In the following sections, we draw upon these observations and conclusions to model the DDR problem, and present several variations of it.

3.6 The Proposed DDR Models

In this section, we present our formulation of the DDR that aims at integrating supply and demand management decisions. In Section 3.6.1, we present the basic DDR model that considers an aggregated path-level demands. In Section 3.6.2, we extend the basic model into a DDR model with path-class level demands. In Section 3.6.3, we further extend the model so that DOW(day of week) or DOM(day of month) variation is considered. Finally, in SectionsecDDRR, we introduce the concept of recapture and extend the basid DDR model to include this concept.

As mentioned previously, in this thesis, we will limit the scope of supply management to fleeting and the scope of demand management to revenue management. However, we believe that the future system for an airline is the one that integrates all supply management (e.g., flight scheduling, fleet assignment, crew scheduling) and demand management (e.g., revenue management, pricing) decisions.

3.6.1 The Basic DDR Model

As discussed previously, we propose to solve the DDR separately for each family, so as to preserve the initial crew schedules. This approach also leads to a much smaller problem size than the original fleeting problem, thus allowing a more detailed analysis.

We now set up our basic DDR model that aims at integrating fleeting and revenue management decisions. We solve this DDR problem for each family k, $\forall k \in K$. The formulation for family k is given below.

Model 3 An Integrated Path-based DDR Model (DDR-B) for Family k:

$$\textit{Maximize} \quad \sum_{i \in \Pi} f_i \; q_i - \sum_{l \in L_k} \sum_{t \in T_k} c'_{lt} \; x_{lt}$$

subject to

$$\sum_{t \in T_k} x_{lt} = 1, \quad \forall \; l \in L_k \tag{4a}$$

$$\sum_{l \in L_k} bf_{ln} \; x_{lt} + \sum_{g \in G_t} bg_{gn} \; y_g = 0, \quad \forall n \in N_t, \; \forall t \in T_k \tag{4b}$$

$$\sum_{l \in CS_t} x_{lt} + \sum_{g \in CS_t} y_g \leq A_t, \quad \forall \; t \in T_k \tag{4c}$$

$$\sum_{i \in \Pi(l)} q_i \leq \sum_{t \in T_k} Cap_t x_{lt}, \quad \forall \; l \in \; L_k \tag{4d}$$

$$\sum_{i \in \Pi(l)} q_i \leq CapAss_l, \quad \forall \; l \in \; L - L_k \tag{4e}$$

$$q_i \leq \mu_i, \quad \forall \; i \in \; \Pi \tag{4f}$$

$$x_{lt} \in \{0, 1\}, \quad \forall \; l \in L_k, \; \forall t \in T_k \tag{4g}$$

$$y_g \geq 0, \quad \forall \; g \in G_t, \; \forall t \in T_k \tag{4h}$$

$$q_i \geq 0, \quad \forall \; i \in \; \Pi \tag{4i}$$

In what follows, we refer to this model as the *basic DDR model* for family k and denote it as *DDR-B*. Supply and demand management decisions interact in this model through constraints (4d). Of course, aggregate maintenance constraints can also be considered in this model by the inclusion of constraints (73a)–(73d) in Appendix 1. In the objective function, we use c'_{lt} to denote the cost of covering leg l using aircraft type t, which may (or may not) be different from the cost c_{lt} used previously, such as in Model 1.

Observe that this model only allows the re-assignment of types in family k (set T_k) to legs originally assigned to that family by FAM (set L_k); see constraints (4d) and (4g). Capacity assignments to legs outside the family (set $L - L_k$) are considered *fixed* while

solving the DDR for this family (see constraints (4e)). These *fixed* capacity assignments, together with decision variables for demand accepted on *all* paths ($q_i,\ i \in \Pi$; see constraints (4f) and (4i)), are still included in the formulation so as to consider the *network effect* due to the dependency between all paths and capacity assignments in the network.

Of course, Model 3 can be extended such that re-fleeting is done for all families simultaneously (as long as each leg is re-assigned within its original family). However, this will increase the problem size considerably, and thus, needs further research. Our initial research direction will be to address the problem separately for each family, as is done in Model 3. In this case, the order in which these models for the families are solved will have an impact on the overall decision. Analyzing this issue is another research direction.

3.6.2 Path-Class Level Based DDR Model

It is easy to extend Model 3 to include path-class level details. Assume, for the sake of simplicity in notation, that all path-classes that pass through leg $l,\ l \in L$, can use the same capacity, $CapAss_l$ (i.e., we ignore the difference between coach classes and business/first classes). In what follows, we let $f_{i,c}$, $q_{i,c}$, and $\mu_{i,c}$ denote the fare, number of accepted passengers, and mean demand on path i and class c, respectively, for $i \in \Pi, c \in C$. It is straightforward to extend this model to include first and business classes. Our model objective, as well as demand management constraints (constraints (3a)–(3c)), can thus be transformed as shown in Model 4 below.

Model 4 Path-Class Level Based DDR Model (DDR-C):

$$\textit{Maximize} \quad \sum_{i\in\Pi}\sum_{c\in C} f_{i,c}\ q_{i,c} - \sum_{l\in L_k}\sum_{t\in T_k} c'_{lt}\ x_{lt}$$

subject to

$$\sum_{t\in T_k} x_{lt} = 1, \quad \forall\ l \in L_k \tag{5a}$$

$$\sum_{l\in L_k} bf_{ln}\ x_{lt} + \sum_{g\in G_t} bg_{gn}\ y_g = 0, \quad \forall n \in N_t,\ \forall t \in T_k \tag{5b}$$

$$\sum_{l\in CS_t} x_{lt} + \sum_{g\in CS_t} y_g \leq A_t, \quad \forall\ t \in T_k \tag{5c}$$

$$\sum_{i\in\Pi(l)}\sum_{c\in C} q_{i,c} \leq \sum_{t\in T_k} Cap_t x_{lt}, \quad \forall\ l \in\ L_k \tag{5d}$$

$$\sum_{i\in\Pi(l)}\sum_{c\in C} q_{i,c} \leq CapAss_l, \quad \forall\ l \in\ L - L_k \tag{5e}$$

$$q_{i,c} \leq \mu_{i,c}, \quad \forall\ i \in\ \Pi,\ \forall c \in C \tag{5f}$$

$$q_{i,c} \geq 0, \quad \forall\ i \in\ \Pi,\ \forall c \in C. \tag{5g}$$

Similar to FAM, DDR will generate the "same everyday fleeting". It will be the focus of the Demand Driven Dispatch (DDD) model to change these "everyday assignments" when advantageous (see Chapter 1 for an overview of the overall model).

3.6.3 Integrated Path-based DDR Model with DOW or DOM Variations

In Model 3, we *aggregate* all the demand information into one distribution, and only use an *aggregated* mean, μ_i, for the corresponding path demand over the planning horizon. However, demand distributions and parameters for the same path might differ largely over different days of week (DOW), and of month (DOM). Monday and Friday, for example, are days with high demands on paths passing through business centers, whereas the same paths might experience much lower demands on a Tuesday. In fact, the demand for every day of

the week will, possibly, have a different mean on a large number of paths. In what follows, we let $f_{i,c,w}$, $q_{i,c,w}$, and $\mu_{i,c,w}$ denote the fare, number of accepted passengers, and mean demand on path i, class c, and week day w, for $i \in \Pi, c \in C, w \in W$, where W is the set of days considered. We also let $|W|$ denote the cardinality (size) of set W. Observe that if demand patterns on several days are similar, then these days can still be aggregated and considered as a single day. In this case, we let $|w|$ denote the cardinality of the aggregated day, and give each aggregated day a weight of θ_w, where $\theta_w = \frac{|w|}{|W|}$, and $\sum_{w \in W} \theta_w = 1$. Considering this detail leads to our next DDR model, given as follows.

Model 5 Integrated Path-based DDR Model with DOW or DOM Variations (DDR-W):

$$\textit{Maximize} \quad \sum_{i \in \Pi} \sum_{c \in C} \sum_{w \in W} \theta_w f_{i,c,w} q_{i,c,w} - \sum_{l \in L_k} \sum_{t \in T_k} c'_{lt}\ x_{lt}$$

subject to

$$\sum_{t \in T_k} x_{lt} = 1, \quad \forall\ l \in L_k \tag{6a}$$

$$\sum_{l \in L_k} bf_{ln}\ x_{lt} + \sum_{g \in G_t} bg_{gn}\ y_g = 0, \quad \forall n \in N_t,\ \forall t \in T_k \tag{6b}$$

$$\sum_{l \in CS_t} x_{lt} + \sum_{g \in CS_t} y_g \leq A_t, \quad \forall\ t \in T_k \tag{6c}$$

$$\sum_{i \in \Pi(l)} \sum_{c \in C} q_{i,c,w} \leq \sum_{t \in T_k} Cap_t x_{lt}, \quad \forall\ l \in\ L_k,\ \forall\ w \in W \tag{6d}$$

$$\sum_{i \in \Pi(l)} \sum_{c \in C} q_{i,c,w} \leq CapAss_l, \quad \forall\ l \in\ L - L_k,\ \forall\ w \in W \tag{6e}$$

$$q_{i,c,w} \leq \mu_{i,c,w}, \quad \forall\ i \in\ \Pi,\ \forall\ c \in C,\ \forall\ w \in W \tag{6f}$$

$$x_{lt} \in \{0, 1\}, \quad \forall\ l \in L_k,\ \forall t \in T_k \tag{6g}$$

$$y_g \geq 0, \quad \forall\ g \in G_t,\ \forall t \in T_k \tag{6h}$$

$$q_{i,c,w} \geq 0, \quad \forall\ i \in\ \Pi,\ \forall\ c \in C,\ \forall\ w \in W. \tag{6i}$$

Notice that the variant mean demands provide more accurate information for the assignment decision than the aggregated mean demands. However, the daily fleet assignment decisions are still left invariant until flight departures are very near, i.e., the DDD stage.

3.6.4 DDR with Recapture (DDR-R)

In the DDR models presented above (Models 3, 4 and 5), we assume that passengers either fly on the specific path that they have demanded to fly or they will be lost. In reality, however, a passenger, who cannot get on the path of her first choice, can be *recaptured* by the airline's other paths between her origin and destination. Thus, demands on paths with common origin-destination pairs will be highly dependent. This important effect needs to be considered in the DDR. Therefore, we now extend our basic DDR model to also consider this aspect of demand management. We will refer to the resulting model as the *DDR model with recapture* (in what follows, this is referred to as DDR-R).

We first review some notation defined in Section 3.3. Recall that $\Pi_{o,d}$ is the set of paths having origin o and destination d, $(o,d) \in OD$, and $\Pi(l)$ is the set of paths that cross through leg l, $l \in L$. Similarly, (o_i, d_i) refers to the origin and destination of path i, $i \in \Pi$. In addition, we define β_{ij} as the proportion of *excess* (i.e., spilled) demand on path i that *can be* routed to a different path j, $\forall i \in \Pi$, $j \in \Pi_{o_i,d_i}$, $j \neq i$. This is the proportion of passengers whose first choice is path i, but who can be routed to path j, in case their first choice is not available. We will refer to parameters β_{ij} as *substitution factors*; these are inputs to our model. In practice, passenger substitution will depend on several factors such as the airline's flight network and schedule, market share, and competition coming

from other airlines, and thus, these parameters are very difficult to determine. However, some researchers have recently started analyzing the use of *discrete choice logit models* for airline forecasting; see van Ryzin (2000), who uses discrete choice theory to predict an individual consumer's choice by considering such factors as price, flight characteristics, customer characteristics, schedules, etc. The literature on retail operations that considers customer substitution could also be useful in providing insights; see, for example, Mahajan and van Ryzin (1998).

In order to simplify notation, we consider below only path level demands; extensions to path-class and/or DOW or DOM levels are straightforward. We let q^r_{ij} denote the demand on path i that is flown on path j, $\forall i \in \Pi, j \in \Pi_{o_i,d_i}$, when *recapture* is considered in the analysis; these are our decision variables. (Note that the variable q_i of Model 2, for example, relates to variable q^r_{ij} in the present context.) Thus, our objective function and demand management constraints are revised as shown in Model 6 below.

Model 6 DDR Model Considering Recapture (DDR-R):

$$\textit{Maximize} \quad \sum_{i\in\Pi} f_i \sum_{j\in\Pi_{o_i,d_i}} q^r_{ji} - \sum_{l\in L_k}\sum_{t\in T_k} c'_{lt}x_{lt} - \sum_{i\in\Pi}\sum_{j\in\Pi_{o_i,d_i}:\ j\neq i} \delta_{ij}q^r_{ij}$$

subject to

$$\sum_{t\in T_k} x_{lt} = 1, \quad \forall\ l \in L_k \tag{7a}$$

$$\sum_{l\in L_k} bf_{ln}\ x_{lt} + \sum_{g\in G_t} bg_{gn}\ y_g = 0, \quad \forall n \in N_t,\ \forall t \in T_k \tag{7b}$$

$$\sum_{l\in CS_t} x_{lt} + \sum_{g\in CS_t} y_g \leq A_t, \quad \forall\ t \in T_k \tag{7c}$$

$$\sum_{i\in\Pi(l)}\sum_{j\in\Pi_{o_i,d_i}} q^r_{ji} \leq \sum_{t\in T_k} Cap_t x_{lt}, \quad \forall\ l \in\ L_k \tag{7d}$$

$$\sum_{i \in \Pi(l)} \sum_{j \in \Pi_{o_i,d_i}} q^r_{ji} \leq CapAss_l, \quad \forall\ l \in\ L - L_k \tag{7e}$$

$$\sum_{j \in \Pi_{o_i,d_i}} q^r_{ij} \leq \mu_i, \quad \forall\ i \in\ \Pi \tag{7f}$$

$$q^r_{ij} \leq \beta_{ij}(\mu_i - q^r_{ii}), \quad \forall\ i \in\ \Pi, \quad j \in \Pi_{o_i,d_i},\ j \neq i \tag{7g}$$

$$q^r_{ij} \geq 0, \quad \forall\ i \in\ \Pi, \quad j \in \Pi_{o_i,d_i} \tag{7h}$$

We add a small penalty term, δ_{ij}, for accepting a path i passenger on path j, when $i \neq j$, and include this in the objective function. This ensures that in an optimal solution each path demand is satisfied on that path itself, if capacity permits. Only when there is not enough capacity on that path will the *excess* path demand (spill) be routed to other paths having slack capacity. Furthermore, similar to this Model 1 variation, we can replace the demand management constraints in Models 4 and 5 with constraints (7d)–(7h) with obvious modifications in variable definition.

Observe that as noted above, variables q^r_{ii} refer to the demand on path i that is flown (accepted) on path i, for $i \in \Pi$. Thus, the special case of DDR-R with $\beta_{ij} = 0, \forall i \in \Pi, j \in \Pi(o_i, d_i), j \neq i$, reduces to the basic model DDR (DDR-B).

Chapter 4

Model Structure Analysis

In this chapter, we study the polyhedral structure of the DDR model to explore ways for tightening its representation and for deriving certain classes of valid inequalities. In Section 4.1, we present several enhanced representations of the DDR model, and generate valid inequalities for it. In Section 4.2, we derive a class of valid inequalities that yield facets of the convex hull of solutions feasible to a substructure of DDR-B. On the other hand, Section 4.3 directly constructs the entire convex hull description for this substructure.

4.1 Tightening Representation of and Valid Inequalities for Model DDR

Consider the following model for DDR-B for family k (in the established notation):

DDR-B Model for Family k:

$$\textit{Maximize} \quad \sum_{i \in \Pi} f_i \; q_i - \sum_{l \in L_k} \sum_{t \in T_k} c'_{lt} \; x_{lt} \tag{8a}$$

subject to

$$\sum_{t \in T_k} x_{lt} = 1, \quad \forall\, l \in L_k \tag{8b}$$

$$\sum_{l \in L_k} bf_{ln}\ x_{lt} + \sum_{g \in G_t} bg_{gn}\ y_g = 0, \quad \forall n \in N_t,\ \forall t \in T_k \tag{8c}$$

$$\sum_{l \in CS_t} x_{lt} + \sum_{g \in CS_t} y_g \leq A_t, \quad \forall\, t \in T_k \tag{8d}$$

$$\sum_{i \in \Pi(l)} q_i \leq \sum_{t \in T_k} Cap_t x_{lt}, \quad \forall\, l \in\ L_k \tag{8e}$$

$$\sum_{i \in \Pi(l)} q_i \leq CapAss_l, \quad \forall\, l \in\ L - L_k \tag{8f}$$

$$q_i \leq \mu_i, \quad \forall\, i \in\ \Pi \tag{8g}$$

$$(x, y, q) \geq 0, \tag{8h}$$

$$x\ binary. \tag{8i}$$

4.1.1 Tightening Representation of (8e).

Consider the following result that prescribes a tighter valid replacement for the set of constraints (8e).

Proposition 1 *The following set of constraints are a valid, tighter replacement for (8e):*

$$\sum_{i \in \Pi(l)} q_i \leq \sum_{t \in T_k} \tilde{c}_{lt} x_{lt}, \quad \forall\, l \in L_k, \tag{9a}$$

where

$$\tilde{c}_{lt} = \ min\ \{Cap_t, \sum_{i \in \Pi(l)} \mu_i\}, \quad \forall\, l \in L_k,\ t \in T_k. \tag{9b}$$

Proof: Clearly, (9a) is tighter than (i.e., at least as tight as) (8e). To establish validity, it is sufficient to show that for any feasible solution $(\bar{x}, \bar{y}, \bar{q})$ to (8), we have that (9a) is satisfied. Consider any $l \in L_k$, and noting (8b),(8i), suppose that $\bar{x}_{lt^*} = 1$, so that $\sum_{i \in \Pi(l)} \bar{q}_i \leq Cap_{t^*}$.

If $\tilde{c}_{lt^*} \equiv Cap_{t^*}$, then (9a) is satisfied. Otherwise, we have that the right-hand side of (9a) for $x = \bar{x}$ is given by (using (8g))

$$\tilde{c}_{lt^*} \equiv \sum_{i \in \Pi(l)} \mu_i \geq \sum_{i \in \Pi(l)} \bar{q}_i$$

or that again, (9a) is satisfied. This completes the proof. □

Example 1

Consider a DDR problem for family k in which there are 2 types to be assigned to 4 legs. The network structure is shown in Figure 5. Note that each type has a copy of this same network, so only one is shown in the figure. Some of the important parameters are:

Types in family k: $T_k = \{1,\ 2\}$;

Types in the entire fleet: $T = \{1,\ 2,\ 3\}$, where 3 is a type from some other family;

Legs assigned to family k: $L_k = \{l1,\ l2,\ l3,\ l4\}$, indexed by 1 through 4;

Legs assigned to the entire fleet: $L = L_k \cup \{l5\}$;

Capacities for the two types: $Cap_1 = 30,\ Cap_2 = 80$.

In addition, the paths passing through each leg are listed in the figure. Demands for the nine paths are: (12, 17, 8, 26, 11, 40, 19, 5, 34).

We examine $l = 4 \in L_k$. The constraint (8e) is given by

$$q_4 + q_7 \leq 30\, x_{41} + 80\, x_{42} \tag{10a}$$

From (9b), noting that

$$\sum_{i \in \Pi(4)} \mu_i = \mu_4 + \mu_7 = 26 + 19 = 45,$$

Path index	Legs	Path index	Legs
1	l1	6	l1-l2
2	l2	7	l3-l4
3	l3	8	l2-l5
4	l4	9	l1-l2-l5
5	l5		

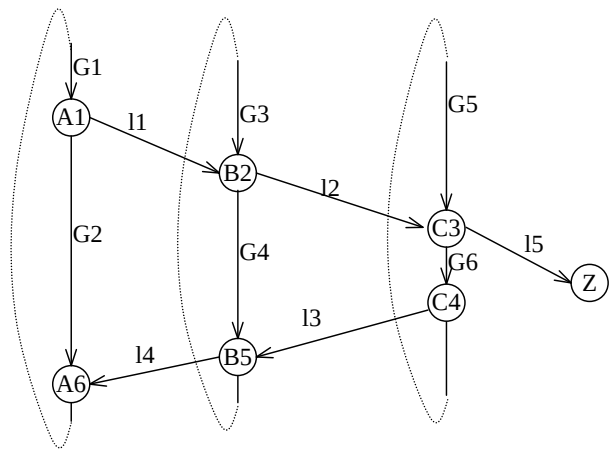


Figure 5: An Illustrative Example for the DDR Problem.

we get $\tilde{c}_{41} = 30$ and $\tilde{c}_{42} = 45$. Hence, (9a) yields a tighter representation than (10a) as given by

$$q_4 + q_7 \leq 30\, x_{41} + 45\, x_{42}. \tag{10b}$$

Note that in the continuous (fractional) relaxation solution to (8) for this test case, we have $q_4 = 26$, $q_7 = 19$, $x_{41} = 0.7$, and $x_{42} = 0.3$. Thus, (10b) deletes this solution (since the left-hand side in (10b) is 45, while the right-hand side is 34.5 at this solution).

Henceforth, we assume that Model DDR-B in (8) has been preprocessed using the following two steps:

Step 1: For each $i \in \Pi$, first replace μ_i with

$$\min\{\mu_i,\ CapAss_l\ \forall\ l \in (L - L_k) : i \in \Pi(l),\ \max_{t \in T_k}(Cap_t) \text{ if } i \in \Pi(l) \text{ for some } l \in L_k\} \tag{11}$$

Note: This assumes that each type $t \in T_k$ is available for each leg $l \in L_k$.

Step 2: Then apply Proposition 1 to derive constraints (9a) via (9b).

Example 2

Suppose that for some $l \in L_k$, we have the constraint

$$q_1 + q_2 \leq 4x_{l1} + x_{l2}, \tag{12a}$$

where

$$0 \leq q_1 \leq 2,\ 0 \leq q_2 \leq 2. \tag{12b}$$

Then, Proposition 1 gives no further reduction of coefficients in (12a) based on (12b). However, suppose that we also have the restriction for some $l \in L - L_k$ that

$$q_1 + q_2 \leq 1. \tag{12c}$$

Then by (11), using (12c), we can reduce the upper bounds in (12b) to $\mu_1 = 1$, and $\mu_2 = 1$, and then Proposition 1 permits us to tighten (12a) to

$$q_1 + q_2 \leq 2x_{l1} + x_{l2}. \tag{12d}$$

4.1.2 Generalized Coefficient Reduction Using Generalized Upper Bound (GUB) Surrogate Constraints

Suppose that we solve the continuous relaxation of DDR-B (henceforth, we will assume that (8) has been preprocessed using the procedure described in Section 4.1.1), and we obtain a fractional solution. Consider a selection of constraints in (9a) and (8f), and let $L'_k \subseteq L_k$ be the set of legs corresponding to the selected constraints in (9a). Using constraint multipliers as given by the corresponding optimal dual multipliers, let us construct a surrogate constraint of the following type:

$$\sum_{l \in L'_k} [\sum_{t \in T_k} a_{lt}\, x_{lt}] \;\geq\; \sum_{i \in \Pi} b_i\, q_i - b_0 \tag{13}$$

where $a_{lt} \geq 0 \;\forall\; (l,t)$, $b_i \geq 0 \;\forall\; i$, and $b_0 \geq 0$ are the corresponding derived surrogate constraint coefficients. This surrogate constraint (13) might be constructed using all the constraints in (9a) and (8f), or it might be a surrogate of only the constraints (9a), or any subset thereof, where the latter includes the case when (13) is simply given by (9a) for some particular $l \in L_k$. Let

$$a_{lt_l} = minimum\ \{a_{lt} : t \in T_k\}, \quad \forall\ l \in L'_k \tag{14}$$

and let

$$b = maximum\ \{\sum_{i\in\Pi} b_i\, q_i - b_0 : \text{some continuous LP relaxation of DDR-B}\}. \tag{15}$$

Note from (8b) that if

$$\sum_{l\in L'_k} a_{lt_l} \geq b,$$

then (13) would be implied in the continuous sense by (8b) and (8h), and is not of interest (a coefficient reduction of each a_{lt} to a_{lt_l}, for each $l \in L'_k$, reduces to a redundant constraint in the continuous sense). Hence, suppose that

$$\sum_{l\in L'_k} a_{lt_l} < b. \tag{16}$$

Furthermore, for each $l \in L'_k$, compute

$$\alpha_l = b - \sum_{p\in L'_k, p\neq l} a_{pt_p} \quad \forall l \in L'_k, \tag{17}$$

and consider the following result.

Proposition 2 *Given (14) - (17), the following is a valid inequality for DDR-B:*

$$\sum_{l\in L'_k}\sum_{t\in T_k} [min\ \{a_{lt},\ \alpha_l\}]\, x_{lt} \ \geq\ \sum_{i\in\Pi} b_i\, q_i - b_0. \tag{18}$$

Proof: Consider any feasible solution $(\bar{x}, \bar{y}, \bar{q})$ to DDR. It is sufficient to show that $(\bar{x}, \bar{q})$ satisfies (18). Noting that $(\bar{x}, \bar{q})$ is feasible to the surrogate constraint (13), if $a_{lt} \leq \alpha_l\ \ \forall\,(l,t)$ such that $\bar{x}_{lt} = 1$, then (18) is also satisfied. Hence, suppose that for some $l^* \in L'_k$, we have that $\bar{x}_{l^*t^*} = 1$, where

$$a_{l^*t^*} > \alpha_{l^*}. \tag{19}$$

But note that for each $l \in L'_k$, we have from (16) and (17) that

$$a_{lt_l} < b - \sum_{p \in L'_k, p \neq l} a_{pt_p} = \alpha_l \quad \forall l \in L'_k \tag{20}$$

and so, from (14) and (20), we have,

$$min\ \{a_{lt_l},\ \alpha_l\} \equiv a_{lt_l} \leq min\ \{a_{lt},\ \alpha_l\} \quad \forall\ t \in T_k,\ \forall\ l \in L'_k. \tag{21}$$

Consequently, using (21), we have that the left-hand side of (18) at the solution $(\bar{x}, \bar{q})$ is, noting (15) and (17),

$$\geq a_{l^*t^*} + \sum_{l \in L'_k, l \neq l^*} a_{lt_l} > \alpha_{l^*} + \sum_{l \in L'_k, l \neq l^*} a_{lt_l} = b \geq \sum_{i \in \Pi} b_i\, \bar{q}_i - b_0$$

or that (18) is satisfied. This completes the proof. □

Example 3

Consider the following subset of DDR-B constraints of type (9a) and (8f).

$$(l = 1): \qquad q_1 + q_2 \leq x_{11} + \frac{1}{3} x_{12} \tag{22a}$$

$$(l = 2): \qquad q_2 + q_3 \leq x_{21} + \frac{3}{4} x_{22} \tag{22b}$$

$$(l = 3): \qquad q_1 + q_3 \leq x_{31} + \frac{3}{4} x_{32} \tag{22c}$$

$$(l = 4 \in L - L_k): q_1 + q_2 + q_3 \leq 1 \tag{22d}$$

$$0 \leq q_i \leq 1 \quad \forall i \tag{22e}$$

and where

$$x_{11} + x_{12} = 1, \quad x_{21} + x_{22} = 1, \quad x_{31} + x_{32} = 1, \tag{22f}$$

$$x \geq 0. \tag{22g}$$

Consider the surrogate formed by $\frac{(22a)+(22b)+(22c)}{2}$:

$$(\frac{1}{2}x_{11} + \frac{1}{6}x_{12}) + (\frac{1}{2}x_{21} + \frac{3}{8}x_{22}) + (\frac{1}{2}x_{31} + \frac{3}{8}x_{32}) \geq (q_1 + q_2 + q_3). \tag{23}$$

Noting (22d), we can use $b = 1$ in (23) (from (15)). Accordingly, noting that

$$\sum_{l=1}^{3} a_{lt_l} = \frac{1}{6} + \frac{3}{8} + \frac{3}{8} = \frac{11}{12} < b \equiv 1$$

as required by (16), we get from (17) that

$$\alpha_1 = 1 - \frac{3}{4} = \frac{1}{4}, \quad \alpha_2 = 1 - \frac{13}{24} = \frac{11}{24}, \quad \text{and } \alpha_3 = 1 - \frac{13}{24} = \frac{11}{24}. \tag{24}$$

Hence, from (23) and (24), we get that (18) of Proposition 2 gives

$$(\frac{1}{4}x_{11} + \frac{1}{6}x_{12}) + (\frac{11}{24}x_{21} + \frac{3}{8}x_{22}) + (\frac{11}{24}x_{31} + \frac{3}{8}x_{32}) \geq (q_1 + q_2 + q_3). \tag{25}$$

Observe that the solution

$$q_1 = q_2 = \frac{1}{4},\ q_3 = \frac{1}{2},\ (x_{11} = \frac{1}{4},\ x_{12} = \frac{3}{4}),\ (x_{21} = 0,\ x_{22} = 1),\ (x_{31} = 0,\ x_{32} = 1) \tag{26}$$

is a fractional vertex of (22) (at which the linearly independent constraints (22a)-(22d), (22f), and $x_{21} = x_{31} = 0$ from (22g) are binding). However, the left-hand side of (25) at this solution is $\frac{15}{16}$ which is less than the right-hand side value of 1. Hence (25) deletes this fractional vertex.

Example 4

A more straightforward application of Proposition 2 for the case of a single constraint itself of type (9a) is as follows. Consider the system of constraints:

$$(l \in L_k): \quad q_1 + q_2 \leq 2x_{l1} + \frac{3}{4}x_{l2} \tag{27a}$$

$$(l \in L - L_k) : q_1 + q_2 \leq 1 \tag{27b}$$

$$0 \leq q_i \leq 1 \quad \forall i \tag{27c}$$

$$\text{where} \qquad x_{l1} + x_{l2} = 1, \quad x \geq 0. \tag{27d}$$

Observe that treating (27a) itself as the "surrogate" (13), we have from (15) and (27b) that a permissible value of b equals 1. Accordingly, $\alpha_1 \equiv b = 1$, and using (18) of Proposition 2, we can tighten (27a) to

$$q_1 + q_2 \leq x_{l1} + \frac{3}{4} x_{l2} \tag{28}$$

Observe that the fractional solution

$$q_1 = 1, \; q_2 = 0, \; (x_{l1} = \frac{1}{5}, \; x_{l2} = \frac{4}{5}) \tag{29}$$

is a feasible vertex of (27), but is cut off by (28).

Remark 1

In general, it might be useful to apply Proposition 2 to each individual constraint (9a), rather than apply it to a single surrogate constraint of type (18). The tradeoff is that the number of LP problems of the type (15) increases, although the effort could be reduced by making this relaxation simpler. One alternative is to use the simple relaxation $0 \leq q_i \leq \mu_i \; \forall i$ in (15) and accordingly define $b = \sum_{i \in \Pi} b_i \, \mu_i - b_0$.

4.1.3 Replacing (8g) with a Tighter Set of Valid Inequalities

Consider the following result:

Proposition 3 *The following are valid inequalities for DDR-B*

$$q_i \leq \sum_{t \in T_k} min\{\mu_i,\ \tilde{c}_{lt}\} x_{lt} \quad \forall\ i \in \Pi(l),\ \forall\ l \in L_k \tag{30}$$

Moreover, using (30) within DDR-B causes (8g) to be implied in the continuous sense (hence, the latter can then be deleted) for all $i \in \bigcup_{l \in L_k} \Pi(l)$.

Proof: Consider any feasible solution $(\bar{x}, \bar{y}, \bar{z})$ to DDR-B, and let us show that (30) hold true. Consider any $l \in L_k$, and examine any $i \in \Pi(l)$. If $\bar{x}_{lt^*} = 1$ in (8b) where $t^* \in T_k$, then from (9a), (8g), and (8h) we have that $\bar{q}_i \leq \tilde{c}_{lt^*}$ and $\bar{q}_i \leq \mu_i$, or that

$$\bar{q}_i \leq min\{\mu_i,\ \tilde{c}_{lt^*}\} \equiv \sum_{t \in T_k} min\{\mu_i,\ \tilde{c}_{lt}\} \bar{x}_{lt}.$$

Hence (30) is satisfied, and therefore, it defines a valid inequality for each $i \in \Pi(l),\ l \in L_k$.

Moreover, since from (8b) and (8h) we have,

$$\sum_{t \in T_k} min\{\mu_i,\ \tilde{c}_{lt}\}\, x_{lt} \leq \sum_{t \in T_k} \mu_i\, x_{lt} = \mu_i$$

this asserts that (8b), (8h), and (30) imply (8g) in the continuous sense $\forall\ i \in \bigcup_{l \in L_k} \Pi(l)$. This completes the proof. □

Example 5

Consider the following system of constraints of type (8b), (9a), (8g), and (8h) for some $l \in L_k$:

$$x_{l1} + x_{l2} = 1 \tag{31a}$$

$$q_1 + q_2 \leq x_{l1} + 4x_{l2} \tag{31b}$$

$$q_1 \leq 2,\ q_2 \leq 2 \tag{31c}$$

$$(x,\ q) \geq 0. \tag{31d}$$

Note that the solution

$$(\hat{x}_{l1}, \hat{x}_{l2}, \hat{q}_1, \hat{q}_2) = (2/3, 1/3, 2, 0) \tag{32}$$

is a (fractional) extreme point of (31) as evidenced by the four linearly independent hyperplanes (31a), (31b), $q_1 = 2$ from (31c) and $q_2 = 0$ from (31d) being binding at this feasible solution. Note that the inequalities (30) for this example are given by

$$q_1 \leq x_{l1} + 2x_{l2} \ \text{ and } \ q_2 \leq x_{l1} + 2x_{l2}. \tag{33}$$

In particular, the first of these cuts off the fractional vertex (32).

The continuous relaxation solution to Example 1 after applying Proposition (1) gives

$$(\hat{x}_{11},\ \hat{x}_{12},\ \hat{x}_{21},\ \hat{x}_{22}) = (0.56,\ 0.44,\ 0.8,\ 0.2),$$

and $\hat{q}_6 = 40$. Constructing valid inequalities (30) for $l = 1$, $l = 2$ and $i = 6$ yields:

$$q_6 \leq min\{\mu_6,\ \tilde{c}_{11}\}x_{11} + min\{\mu_6,\ \tilde{c}_{12}\}x_{12} = 30x_{11} + 40x_{12}$$

and

$$q_6 \leq min\{\mu_6,\ \tilde{c}_{21}\}x_{21} + min\{\mu_6,\ \tilde{c}_{22}\}x_{22} = 30x_{21} + 40x_{22}.$$

The right-hand sides of these two inequalities are 34 and 32, respectively. Hence, since $\hat{q}_6 = 40$, these new constraints cut off the foregoing fractional solution.

Remark 2

In general one would only generate (30) for those cases that do not simply yield (8g) itself, and keep these along with (8g) for the remaining cases. (See Remark 3 below)

4.2 Generalized Class of Facetial Valid Inequalities

Motivated by Example 5 and Proposition 3, we now derive a class of valid inequalities that generalizes (30), and can in fact be demonstrated to yield facets of the convex hull of solutions feasible to a substructure of DDR-B.

Toward this end, for any $l \in L_k$, let

$$Z_l = \{(x_{lt} \text{ for } t \in T_k,\ q_i \text{ for } i \in \Pi(l)) :$$

$$\sum_{t \in T_k} x_{lt} = 1 \tag{34a}$$

$$\sum_{i \in \Pi(l)} q_i \leq \sum_{t \in T_k} \tilde{c}_{lt}\, x_{lt} \tag{34b}$$

$$q_i \leq \mu_i \quad \forall i \in \Pi(l) \tag{34c}$$

$$q \geq 0, \quad x \text{ binary}\}. \tag{34d}$$

Let $\bar{Z}_l$ denote the continuous relaxation of Z_l. Let us select any $I \subseteq \Pi(l)$ such that

$$\tau(I) = \{t \in T_k : \tilde{c}_{lt} < \sum_{i \in I} \mu_i\} \neq \emptyset. \tag{35a}$$

(Else, if $\tilde{c}_{lt} \geq \sum_{i \in \Pi(l)} \mu_i \ \forall\, t \in T_k$, then (34b) is implied by (34a), (34c) and the nonnegativity constraints in the continuous sense, and then, $\text{conv}\{Z_l\} \equiv \bar{Z}_l$ itself.) Also, let $\bar{\tau}(I) \equiv T_k - \tau(I)$, and assume that there exists a

$$\hat{t} \in \bar{\tau}(I) \text{ for which } \ \tilde{c}_{l\hat{t}} > \sum_{i \in I} \mu_i. \tag{35b}$$

Consider the following result.

Proposition 4 *For any* $l \in L_k$*, let* Z_l *be defined by (34), and consider any* $I \subseteq \Pi(l)$ *such that* $\tau(I)$ *defined by (35a) is nonempty, and* $\hat{t}$ *defined by (35b) exists. Then, the following inequality is valid for* Z_l *(and hence for DDR-B), and is moreover facet-defining for* $conv(Z_l)$*:*

$$\sum_{t \in \tau(I)} \tilde{c}_{lt} x_{lt} + [\sum_{i \in I} \mu_i] \sum_{t \in \bar{\tau}(I)} x_{lt} \geq \sum_{i \in I} q_i. \tag{36}$$

Proof: (**Validity:**) Consider any feasible solution $(\bar{x}, \bar{q}) \in Z_l$ and let us verify that this solution is feasible to (36). Noting (34a, 34d), suppose that $\bar{x}_{lt^*} = 1$ and $\bar{x}_{lt} \equiv 0 \; \forall \; t \neq t^*$. If $t^* \in \tau(I)$, then from (34b, 34d) we get that

$$\tilde{c}_{lt^*} \geq \sum_{i \in \Pi(l)} \bar{q}_i \geq \sum_{i \in I} \bar{q}_i$$

or that (36) is satisfied. If $t^* \in \bar{\tau}(I)$, then from (34c) we get

$$\sum_{i \in I} \mu_i \geq \sum_{i \in I} \bar{q}_i$$

or that again (36) is satisfied. Hence, (36) is valid.

(**Facet-defining:**) To show that (36) defines a facet of $\text{conv}(Z_l)$, denoting $|T_k| = n$ and $|\Pi(l)| = m$, let us exhibit the existence of some $(m + n - 1)$ linearly (and hence, affinely) independent points in Z_l at which (36) is binding. This would then indicate that the intersection of $\text{conv}(Z_l)$ and the hyperplane corresponding to (36) (as an equality) is $(m + n - 2)$, and since Z_l has a feasible solution at which (36) is inactive (for example, any feasible solution having $q \equiv 0$), this would assert that $\dim\{\text{conv}(Z_l)\} \geq (m + n - 1)$. However, due to the equality (34a), we know that $\dim\{\text{conv}(Z_l)\} \leq (m + n - 1)$. This in turn would establish that $\dim\{\text{conv}(Z_l)\} = (m+n-1)$ and that (36) defines a proper facet of $\text{conv}(Z_l)$.

Toward this end, pick any $t^* \in \tau(I) \neq \emptyset$, and consider the following collection of $(m+n-1)$ points, each defining a particular row of the matrix B given in (37) over the sets I, II, III, and IV (these are explained in the sequel), plus a dummy point defining the last row of B. Note that B is a square matrix of size $(m+n)$, and that $\bar{I} \equiv \Pi(l) - I$ is the complement of I.

Variables whose values are identified in each row of B

$B =$

	x_{lt^*}	x_{lt} for $t \in \tau(I)$ $t \neq t^*$	x_{lt} for $t \in \bar{\tau}(I)$ (including $x_{l\hat{t}}$)	q_i for $i \in I$	q_i for $i \in \bar{I}$
Set I ($\lvert I\rvert$ points)	1 ⋮ 1	0	0	H_1	0
Set II ($\lvert\tau(I)\rvert - 1$ points)	0	I	0	H_2	0
Set III ($\lvert\bar{\tau}(I)\rvert$ points)	0	0	I	H_3	0
Set IV ($\lvert\bar{I}\rvert$ points)	0	0	H_4	H_5	$\epsilon \cdot I$
Single dummy point	1	0 ... 0	0 ... 0	0 ... 0	0 ... 0

(37)

Set I: This is comprised of $|I|$ points feasible to Z_l at which (36) is binding as follows.

Consider the system

$$\sum_{i \in I} q_i = \tilde{c}_{lt^*} \tag{38a}$$

$$0 \leq q_i \leq \mu_i \quad \forall i \in I. \tag{38b}$$

Note that since $t^* \in \tau(I)$, we have from (35) that

$$0 < \tilde{c}_{lt^*} < \sum_{i \in I} \mu_i. \tag{39}$$

Hence, (38) is feasible, and moreover, the point $\hat{q} \equiv (\hat{q}_i,\ i \in I)$ where

$$\hat{q}_i = \left[\frac{\mu_i}{\sum_{r \in I} \mu_r}\right] \cdot \tilde{c}_{lt^*} \quad \forall\ i \in I, \tag{40}$$

satisfies (38a) with $0 < \hat{q}_i < \mu_i,\ \forall\ i \in I$ in (38b). Hence, since the dimension of the hypercube defined by (38b) is $|I|$, and since $\hat{q}$ is a feasible solution lying on the hyperplane (38a) that is interior to this hypercube, we have that the dimension of (38) is $|I|-1$. Hence, since the origin lies off the plane (38a), there exist a collection of $|I|$ linearly independent points $(q_i,\ i \in I)$ feasible to (38). Let these points define the rows of the $|I| \times |I|$ matrix H_1. This produces the points defining Set I in (37).

Set II: Each of the points in this set are defined by considering in turn each $t \in \tau(I),\ t \neq t^*$, and setting $x_{lt} = 1$ in (34) and finding a solution q such that $\sum_{i \in I} q_i = \tilde{c}_{lt}$, while $q_i = 0\ \forall i \in \bar{I}$. This is possible since from (35), we have $\tilde{c}_{lt} < \sum_{i \in I} \mu_i$. Defining the rows of H_2 accordingly, Set II contains $|\tau(I)| - 1$ points from Z_l at which (36) is binding.

Set III: These are a set of $|\bar{\tau}(I)|$ feasible points in Z_l at which (36) is binding, where each row of H_3 is identically equal to $(\mu_i,\ i \in I)$. (Note that feasibility to Z_l follows from (35) since $\tilde{c}_{lt} \geq \sum_{i \in I} \mu_i\ \forall\ t \in \bar{\tau}(I)$.)

Set IV: This collection of $|\bar{I}|$ points in Z_l at which (36) is binding are defined as follows. Each of these points has $x_{l\hat{t}} = 1$ and the remaining variables zeros, yielding the matrix H_4, and has $q_i = \mu_i \ \forall \ i \in I$, yielding the matrix H_5. Hence, (36) is binding at each of these points, in particular. Furthermore, noting from (35b) that $\tilde{c}_{l\hat{t}} > \sum_{i \in I} \mu_i$, there exists an $\epsilon > 0$ such that by taking $q_i = \epsilon$ for each $i \in \bar{I}$ in turn with the remaining $q_i = 0$ for $j \in I, \ j \neq i$, we would achieve feasibility of each of these points to Z_l.

To complete the proof, we exhibit linear independence of the set of $m + n - 1$ points defining the rows of B within the sets I, II, III, and IV by demonstrating that B is non-singular. Toward this end, consider the system $B\lambda = 0$ where $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5)$ is partitioned (into subvectors $\lambda_1, ...\lambda_5$) according to the respective five sets of columns of B identified in (37), and let us show that this system has a unique solution given by $\lambda \equiv 0$. From Set I, letting e denote a vector of $|I|$ ones, and noting that H_1^{-1} exists, we get

$$(e)\lambda_1 + H_1\lambda_4 = 0 \quad or \quad \lambda_4 = -H_1^{-1}(e)\lambda_1. \tag{41a}$$

Similarly, Sets II, III and IV respectively yield

$$\lambda_2 = -H_2\lambda_4 = [H_2 H_1^{-1} e]\lambda_1 \tag{41b}$$

$$\lambda_3 = -H_3\lambda_4 = [H_3 H_1^{-1} e]\lambda_1 \tag{41c}$$

$$\text{and } \epsilon\lambda_5 = -H_4\lambda_3 - H_5\lambda_4, \quad or \quad \lambda_5 = (1/\epsilon)[-H_4 H_3 H_1^{-1} e + H_5 H_1^{-1} e]\lambda_1. \tag{41d}$$

But the final equation yields $\lambda_1 \equiv 0$, and hence from (41), we have $\lambda \equiv 0$ as the unique solution. This completes the proof. □

Remark 3

Note the relevance of the existence of $\hat{t}$ in (35b) for (36) to be facetial. Otherwise, noting (35a), in case $\bar{\tau}(I) \neq \emptyset$, we would have $\tilde{c}_{lt} \equiv \sum_{i \in I} \mu_i$ for all $t \in \bar{\tau}(I)$, and then, the corresponding (valid) inequality (36) would be implied (in the continuous sense) by (34b) and $q \geq 0$.

Remark 4

Note that (30) is a special case of (36) for the case of $|I| = 1$. In this case, if there exists a $t^* \in T_k$ having $\tilde{c}_{lt^*} < \mu_i$ and a $\hat{t} \in T_k$ having $\tilde{c}_{l\hat{t}} > \mu_i$, then (35) holds true, and then by Proposition 4, (30) would define a facet of conv(Z_l). This was the case for Example 5.

Example 6

Consider a set Z_l of Equation (34) defined as follows:

$$
\begin{aligned}
Z_l = \{(x_{l1}, x_{l2}, q_1, q_2, q_3, q_4) :& \\
& x_{l1} + x_{l2} = 1 && \text{(42a)} \\
& q_1 + q_2 + q_3 + q_4 \leq x_{l1} + 4x_{l2} && \text{(42b)} \\
& q_i \leq 1 \quad \forall i = 1, ..., 4 && \text{(42c)} \\
& q \geq 0, \quad x \text{ binary}\}. && \text{(42d)}
\end{aligned}
$$

Note that the solution $(\hat{x}, \hat{q})$ given by

$$\hat{x}_{l1} = 2/3, \ \hat{x}_{l2} = 1/3, \ \hat{q}_1 = \hat{q}_2 = 1, \ \hat{q}_3 = \hat{q}_4 = 0 \tag{43}$$

defines a fractional vertex of the continuous relaxation $\bar{Z}_l$ of Z_l (as evidenced by the 6 linearly independent active constraints given by (42a), (42b), (42c) for $i = 1, 2$, and ($q \geq 0$ in (42d) for $i = 3, 4$). Observe that by Remark 4 we do not have any facetial constraints of type (30)(or (36) with $|I| = 1$). In fact, (30) reduces to (42c) $\forall\ i = 1, ..., 4$. However, using

$$I = \{1, 2\}, \quad \tau(I) = \{1\}, \quad \bar{\tau}(I) = \{2\}, \tag{44}$$

we see that (35a) and (35b) are satisfied and we derive the corresponding facetial valid inequality (36) as

$$x_{l1} + 2x_{l2} \geq q_1 + q_2. \tag{45}$$

Observe that (45) deletes the fractional solution (43).

In general, given a fractional solution to the continuous relaxation of DDR-B, we could develop a *separation routine* to (heuristically) generate a valid inequality of type (36) in order to delete the given fractional solution $(\hat{x}, \hat{q})$, say, if possible. We could examine each $l \in L_k$ in turn for which the corresponding variables $(x_{lt}, t \in T_k)$ are not binary valued. For any such l, having selected some $I \subseteq \Pi(l)$, observe that the valid inequality (36) is of the form

$$\sum_{t \in T_k} \min\{\tilde{c}_{lt}, \sum_{i \in I} \mu_i\} x_{lt} \geq \sum_{i \in I} q_i \tag{46}$$

We would like to select that $I \subseteq \Pi(l)$ for which (46) would be violated by the given fractional solution $(\hat{x}, \hat{q})$ to as great an extent as possible. Denoting

$$T_k^+ = \{t \in T_k : \hat{x}_{lt} > 0\}, \tag{47}$$

we wish to maximize (by determining $I \subseteq \Pi(l)$):

$$\sum_{i \in I} \hat{q}_i - \sum_{t \in T_k^+} \min\{\tilde{c}_{lt}, \sum_{i \in I} \mu_i\} \hat{x}_{lt}. \tag{48}$$

Defining binary variables

$$\psi_i = \begin{cases} 1, & \text{if } \ i \in I \\ 0, & \text{otherwise} \end{cases} \qquad \forall\ i \in \Pi(l), \tag{49}$$

we consider the following separation problem, based on (48).

$$\mathbf{SP(l)} : \underset{\psi\ binary}{Maximize} \{ \sum_{i \in \Pi(l)} \hat{q}_i\, \psi_i - \sum_{t \in T_k^+} \hat{x}_{lt}\ \min\{\tilde{c}_{lt}, \sum_{i \in \Pi(l)} \mu_i \psi_i\}\}. \tag{50}$$

Proposition 5

Below asserts that this separation problem can be equivalently solved as ***SP'(l)***, where $\xi_t\ \forall\ t \in T_k^+$ and $\beta_{it}\ \forall\ i \in \Pi(l)$ and $t \in T_k^+$ are auxiliary <u>continuous</u> variables.

$$\mathbf{SP'(l)} : Maximize \sum_{i \in \Pi(l)} \hat{q}_i\, \psi_i - \sum_{t \in T_k^+} \hat{x}_{lt} [\, \tilde{c}_{lt} \xi_t + \sum_{i \in \Pi(l)} \mu_i (\psi_i - \beta_{it})] \tag{51a}$$

subject to

$$0 \le \beta_{it} \le \xi_t \quad \forall\ i \in \Pi(l),\ t \in T_k^+ \tag{51b}$$

$$0 \le (\psi_i - \beta_{it}) \le (1 - \xi_t) \quad \forall\ i \in \Pi(l),\ t \in T_k^+ \tag{51c}$$

$$\psi\ binary. \tag{51d}$$

Consider again Example 6. Noting that $\hat{x}_{l1} = 2/3,\ \hat{x}_{l2} = 1/3,\ \hat{q}_1 = \hat{q}_2 = 1,\ \hat{q}_3 = \hat{q}_4 = 0,\ \mu_1 = \mu_2 = \mu_3 = \mu_4 = 1$, and $\tilde{c}_{l1} = 1,\ \tilde{c}_{l2} = 4$, we can formulate the corresponding $SP'(l)$ problem (51) as follows:

$$Maximize \quad \psi_1 + \psi_2 - \frac{2}{3}[\, \xi_1 + \sum_{i=1}^{4} (\psi_i - \beta_{i1})] - \frac{1}{3}[\, 4\, \xi_2 + \sum_{i=1}^{4} (\psi_i - \beta_{i2})] \tag{52a}$$

subject to :

$$0 \leq \beta_{it} \leq \xi_t \quad i = 1, ..., 4, \ t = 1, 2 \tag{52b}$$

$$0 \leq (\psi_i - \beta_{it}) \leq (1 - \xi_t) \quad i = 1, ..., 4, \ t = 1, 2 \tag{52c}$$

$$\psi \ \textit{binary}. \tag{52d}$$

The LP relaxation of $SP'(l)$ itself yields an optimal integer solution $\theta_1 = 1, \ \theta_2 = 2, \ \xi_1 = 1, \ \xi_2 = 0, \ \psi_1 = \psi_2 = 1$, and $\psi_3 = \psi_4 = 0$. The corresponding ψ vector yields $I = \{1, \ 2\}$, and generates the valid inequality (46): $x_{l1} + 2x_{l2} \geq q_1 + q_2$, which coincides with (45) as in Example 6.

Proposition 6 *Let (ψ^*, ξ^*, β^*) be an optimal solution for (51). Then ψ^* solves SP(l) with a common objective value ν^*.*

Proof: Consider any binary vector $\bar{\psi}$, and let $I = \{i \in \Pi(l) : \bar{\psi}_i = 1\}$ and $\bar{I} = \Pi(l) - I$. Observe that (51b) and (51c) assert that

$$\beta_{it} = \xi_t \ \forall \ i \in I \text{ and } \beta_{it} = 0 \ \forall \ i \in \bar{I}, \quad \text{for each } t \in T_k^+ \tag{53}$$

necessarily holds true. Hence, for $\psi = \bar{\psi}$ fixed, Problem $SP'(l)$ reduces to

$$\textit{Maximize} \sum_{i \in I} \hat{q}_i - \sum_{t \in T_k^+} \hat{x}_{lt} [\tilde{c}_{lt} \xi_t + (1 - \xi_t) \sum_{i \in I} \mu_i] \tag{54a}$$

$$\textit{subject to} \ 0 \leq \xi_t \leq 1 \quad \forall \ t \in T_k^+. \tag{54b}$$

Since $\hat{x}_{lt} > 0 \ \forall \ t \in T_k^+$, the optimal objective value for (54) is

$$\sum_{i \in I} \hat{q}_i - \sum_{t \in T_k^+} \hat{x}_{lt} \min\{\tilde{c}_{lt}, \ \sum_{i \in I} \mu_i\}. \tag{55}$$

Noting that the objective value (55) is the same as that obtained for $SP(l)$ in (50) by fixing $\psi = \bar{\psi}$, and since this is true for all such binary vectors $\bar{\psi}$, the proof is complete. □

Remark 5

Observe that the following are valid inequalities for $SP'(l)$:

$$\sum_{i\in\Pi(l)} \mu_i\,\beta_{it} \geq \tilde{c}_{lt}\,\xi_t \quad \forall t \in T_k^+ \tag{56a}$$

$$\sum_{i\in\Pi(l)} \mu_i(\psi_i - \beta_{it}) \leq \tilde{c}_{lt}(1-\xi_t) \quad \forall t \in T_k^+ \tag{56b}$$

because they respectively assert that the coefficient of $\hat{x}_{lt}$ in (51a) should be less than or equal to $\sum_{i\in\Pi(l)} \mu_i\,\psi_i$, and less than or equal to $\tilde{c}_{lt}$ as in (50). In fact, it is easy to verify that by including (56) into (51), for any binary vector $\bar{\psi}$, we would neccessarily have by $feasibility$ that $\xi_t \equiv 1$ if $\sum_{i\in I} \mu_i > \tilde{c}_{lt}$ and $\xi_t \equiv 0$ if $\sum_{i\in I} \mu_i < \tilde{c}_{lt}$ (where I is as defined in the proof of Proposition 6, and where any value of $\xi_t \in [0,1]$ is equivalent when $\sum_{i\in I} \mu_i = \tilde{c}_{lt}$). However, since the objective function of $SP'(l)$ performs this same function as in (54a), we have chosen to omit these constraints.

We now show that the LP relaxation of (51) automatically yields optimal integer solutions.

Proposition 7 *The LP relaxation of separation problem $SP'(l)$ yields optimal integer solutions.*

Proof: The problem can be transformed into:

$$\begin{aligned} &Maximize \sum_{i\in\Pi(l)} (\hat{q}_i - \mu_i)\,\psi_i - \sum_{t\in T_k^+} \hat{x}_{lt}\Big(\sum_{i\in\Pi(l)} \mu_i\,\beta_{it} - \tilde{c}_{lt}\xi_t\Big) \\ &subject\ to \end{aligned}$$

$$\beta_{it} \leq \xi_t \quad \forall\ i \in \Pi(l),\ t \in T_k^+ \tag{57}$$

$$\beta_{it} \leq \psi_i \quad \forall\ i \in \Pi(l),\ t \in T_k^+ \tag{58}$$

$$\psi_i - \beta_{it} \leq 1 - \xi_t \quad \forall\, i \in \Pi(l),\ t \in T_k^+ \tag{59}$$

$$0 \leq \psi_i \leq 1 \ \ \forall\, i, \quad \beta, \xi \geq 0. \tag{60}$$

First note that this problem is equivalent to (where $(f,\ g,\ h) \geq 0$):

$$\begin{aligned} &Maximize - \sum_{i \in \Pi(l)} f_i\, \psi_i - \sum_{t \in T_k^+} g_t\, \xi_t + \sum_{i \in \Pi(l)} \sum_{t \in T_k^+} h_{it} \beta_{it} \\ &subject\ to \end{aligned}$$

$$\beta_{it} \leq \xi_t \quad \forall\, i \in \Pi(l),\ t \in T_k^+ \tag{61}$$

$$\beta_{it} \leq \psi_i \quad \forall\, i \in \Pi(l),\ t \in T_k^+ \tag{62}$$

$$0 \leq \psi_i \leq 1 \ \ \forall\, i \in \Pi(l), \quad 0 \leq \xi_t \leq 1 \ \ \forall\, t \in T_k^+, \tag{63}$$

because at optimality, we would have $\beta_{it} = \min\{\xi_t,\ \psi_i\}\ \forall (i,t)$ cause of objective function and in either case, (59) would be satisfied because of (63). However, at extreme points of (61)–(63), any active constraints in (61) or (62) only determine β-variable values in terms of the ψ- and ξ-variable, establishing equalities among sets of the ψ- and ξ- variables, while these latter variable values are then necessarily determined by active constraints from (63). Hence, all variables are 0-1 at extreme points. This completes the proof. □

4.3 Direct Construction of conv$\{Z_l\}$

In this section, we demonstrate how to directly construct the entire convex hull of Z_l, which would then automatically imply the entire family of valid inequalities (36) (in addition to others). This representation is constructed in a higher dimension that involves a disaggregation of the variables q_i for $i \in \Pi(l)$ into components $Q_{ilt}\ \forall\ t \in T_k$, given any $l \in L_k$.

Consider the following result.

Proposition 8 *Let Z_l be given by (34) for any $l \in L_k$. Then conv$\{Z_l\} \equiv Z_l^c$, where*

$$Z_l^c = \{(x_{lt} \text{ for } t \in T_k,\ q_i \text{ for } i \in \Pi(l)) :$$

$$\sum_{t \in T_k} x_{lt} = 1 \tag{64a}$$

$$\sum_{i \in \Pi(l)} Q_{ilt} \leq \tilde{c}_{lt}\, x_{lt} \qquad \forall\ t \in T_k \tag{64b}$$

$$Q_{ilt} \leq \mu_i\, x_{lt} \qquad \forall\ i \in \Pi(l),\ t \in T_k \tag{64c}$$

$$q_i = \sum_{t \in T_k} Q_{ilt} \qquad \forall\ i \in \Pi(l) \tag{64d}$$

$$x_{lt} \geq 0 \qquad \forall\ t \in T_k$$

$$Q_{ilt} \geq 0 \qquad \forall\ i \in \Pi(l),\ t \in T_k\}. \tag{64e}$$

Proof: Let us first show that $Z_l \subseteq Z_l^c$. Given any $(\bar{x}, \bar{q}) \in Z_l$, define

$$\bar{Q}_{ilt} = \bar{q}_i\, \bar{x}_{lt} \quad i \in \Pi(l),\ t \in T_k. \tag{65}$$

Then by direct substitution, it is readily verified that $(\bar{x}, \bar{q}, \bar{Q})$ satisfies (64a-64e). Hence, $Z_l \subseteq Z_l^c$. To complete the proof, it is sufficient to show that at each extreme point $(\hat{x}, \hat{q}, \hat{Q})$ of Z_l^c, we necessarily have that $\hat{x}$ is binary with $(\hat{x}, \hat{q}) \in Z_l$.

Toward this end, consider the linear programming problem to

$$\text{maximize} \ \ \{c_1 \cdot x + c_2 \cdot q \ :\ (x, q) \in Z_l^c\} \tag{66}$$

for any objective gradient vector $(c_1,\ c_2)$ for which (66) has a unique optimum. Let us show that this optimum $(\hat{x}, \hat{q})$, say, has $\hat{x}$ necessarily binary valued with $(\hat{x}, \hat{q}) \in Z_l$. To solve (66), note that we could first substitute out for q in terms of Q using (64d). Next, observe

from (64b, 64c) that for each $t \in T_k$, these constraints are separable in the respective sets of variables $(Q_{ilt},\ i \in \Pi(l))$, and moreover, each of these separable constraints have their right-hand sides scaled by x_{lt}. Hence, by LP duality, the optimal values of Q_{ilt} can be obtained as linear functions of $x_{lt}\ \forall\ i \in \Pi(l)$, for each $t \in T_k$. Therefore, the linear program (66) effectively reduces to

$$\text{maximize}\ \ \{c_1' \cdot x\ :\ \sum_{t \in T_k} x_{lt} = 1,\ x \geq 0\} \tag{67}$$

for some objective vector c_1'. Since (66) has been assumed to have a unique optimum, then so does (67). Noting the structure of (67), this solution must be of the form

$$\hat{x}_{lt^*} = 1 \text{ and } \hat{x}_{lt} = 0 \quad \forall\ t \in T_k - \{t^*\},\ \text{for some } t^* \in T_k. \tag{68}$$

Hence, $\hat{x}$ is binary valued at each extreme point $(\hat{x}, \hat{q}, \hat{Q})$ of Z_l^c. Moreover, given (68), note from (64c)-(64e) that

$$\hat{Q}_{ilt} \equiv 0\ \forall\ i \in \Pi(l),\ t \in T_k - \{t^*\},\ \text{and so, } \hat{q}_i \equiv \hat{Q}_{ilt^*}\ \forall\ i \in \Pi(l). \tag{69}$$

Hence, from (64c), (64b) and (69), we have

$$0 \leq \hat{q}_i \leq \mu_i\ \forall\ i \in \Pi(l),\ \text{and}\ \sum_{i \in \Pi(l)} \hat{q}_i = \sum_{i \in \Pi(l)} \hat{Q}_{ilt^*} \leq \tilde{c}_{lt^*} = \sum_{t \in T_k} \tilde{c}_{lt} \hat{x}_{lt} \tag{70}$$

and so, from (68) and (70), we have that $(\hat{x}, \hat{q}) \in Z_l$. This completes the proof. □

Applying Proposition 8, the DDR-B Model can be tightened through the stated partial convex hull construction process as follows, which we refer to as Model DDR-B^c:

Model DDR-B^c for Family k:

$$\textit{Maximize} \quad \sum_{i \in \Pi} f_i\ q_i - \sum_{l \in L_k} \sum_{t \in T_k} c_{lt}'\ x_{lt} \tag{71a}$$

subject to

$$\sum_{t \in T_k} x_{lt} = 1, \quad \forall\, l \in L_k \tag{71b}$$

$$\sum_{l \in L_k} bf_{ln}\; x_{lt} + \sum_{g \in G_t} bg_{gn}\; y_g = 0, \quad \forall n \in N_t, \forall t \in T_k \tag{71c}$$

$$\sum_{l \in CS_t} x_{lt} + \sum_{g \in CS_t} y_g \leq A_t, \quad \forall\, t \in T_k \tag{71d}$$

$$\sum_{i \in \Pi(l)} Q_{ilt} \leq \tilde{c}_{lt}\, x_{lt}, \quad \forall\, t \in T_k, \quad \forall\, l \in\; L_k \tag{71e}$$

$$\sum_{i \in \Pi(l)} q_i \leq CapAss_l, \quad \forall\, l \in\; L - L_k \tag{71f}$$

$$Q_{ilt} \leq \mu_i\, x_{lt}, \quad \forall\, l \in L_k,\; i \in \Pi(l),\; t \in T_k \tag{71g}$$

$$q_i = \sum_{t \in T_k} Q_{ilt}, \quad \forall\, l \in L_k,\; i \in \Pi(l) \tag{71h}$$

$$(x, y, q, Q) \geq 0,\; x \textit{ binary}. \tag{71i}$$

Remark 6

While we have provided a direct proof of Proposition 8 above for the sake of completeness, note that the convex hull representation (64) can alternatively be derived by applying the special GUB-structured RLT approach described by Sherali *et al.* (1998). Furthermore, note that (36) can be derived by surrogating (64b) for $t \in \tau(I)$, given (35a), and (64c) for $i \in I$ and $t \in \bar{\tau}(I)$, and then using (64d) along with $Q \geq 0$. Moreover, with (35b) also holding true, one can verify that these surrogate multipliers correspond to an extreme direction of the associated projection cone, thereby providing an alternative proof for Proposition 4. However, we have preferred to adopt the more direct self-contained approach above for the sake of clarity.

Chapter 5

Numerical Experiments

Airline companies typically have large flight networks. The number of flights in each family and the total number of flights for any carrier can easily reach hundreds and even over thousands, let alone the number of paths formed by these flights, which can be tens of thousands. Considering the large sizes of real life networks, solving DDR-B directly as an MIP is usually implausible. In this chapter, we investigate how to efficiently apply the propositions stated in the previous chapter so that relatively larger problems can be solved at a realistic cost. In Section 5.1, we describe our numerical experimental design. In Section 5.2, we introduce the preliminary procedures designed to apply the various propositions, and explore different related approaches. In Section 5.3, we summarize the procedures in Section 5.2 by proposing a heuristic method and an optimality method to solve practical-sized DDR problems for deriving good quality solutions in an acceptable time frame.

5.1 Experimental Design

To test the efficiency of different approaches, we use realistic, hypothetical randomly generated flight networks. Each flight is generated by randomly selecting a departure station and an arrival station from a station pool. Departure times and flight durations are uniformly generated. Based on an observation of an airline's flight data, which indicate that all the flights are paired between stations, for each randomly generated flight, we create another flight departing from its arrival station and arriving at its departure station. The two flights have the same flight duration, but the departure times are independent variables uniformly distributed within 24 hours. Whether this pair of flights is in the designated family or not is arbitrarily determined.

Flights are generated until some pre-specified limit is reached. Paths consisting up to three legs are then formed based on this set, where each flight is taken as a single-leg path. By searching each flight's departure station, two-leg paths are formed if another flight arrives at this station at a time earlier than the departure time of the departing flight and the two flights do not form a cycle, i.e., the arriving flight's departure station is different from the departing flight's arrival station. A three-leg path is formed in a similar way by viewing a two-leg path as one aggregated "flight". In these generated test problems, unless specified otherwise, we assume four types of aircraft in the family, with larger capacities having higher operating costs.

Using this schema, we generated three sets of test networks. Those in the first set are relatively small in size. They have a maximum of 120 flights in total and no more than 50 flights in the family. The demands for these problems are uniformly generated. This

problem set is comprised of five groups, each having 30 networks. The groups are divided mainly according to the number of flights in total, and in the last two cases, also based on the distribution of the path demands. The problems in Groups 1, 2, and 3 have 60, 90, and 110 flights, respectively. In these problems, the smallest type has a capacity uniformly distributed between 40 and 75, and each type has 20 seats less than the immediately larger type. The problems in groups 4 and 5 problems have 100 and 120 flights respectively, and in addition, are assigned different path demands - for Group 4, the path demands are uniformly distributed between 0 and 6, and the capacities of the aircraft in the family range from 40 to 125, while for Group 5, the path demands are between 15 and 25, and the capacities range from 30 to 105. Costs of assigning different types of aircraft start from a mean of $6000 for the smallest type, and increases for each larger type by a mean of $1000. Standard deviations of these costs are taken to be $400. The flight fares are assumed to be uniformly distributed between $150 and $1500. Paths consisting of more legs tend to be more expensive. Each additional leg is assumed to yield a mean extra of $200 in the flight fare.

The second set consists of 51 larger problems. They are divided into three groups according to the number of flights in total: 150, 200, and 400. Half of the total flights are taken to belong to the designated family. The first two groups have four flight composition networks each, with each network being used to generate six different versions of the problem by accommodating different normally distributed demands. These six versions have mean demands of 5, 8, and 15, each being paired with a coefficient of variation of 0.15 and 0.45. Negative demands are truncated and reset to 0. The capacities for these two groups are

decided similar to the Set 1 problems with the exception that the numbers of seats differ by 15 between the adjacent types. The third group has only three networks, each generating one problem having a mean of 5 and a coefficient of variation of 0.15 for demands. The three problems have capacities {40, 55, 70, 85}, {60, 75, 90, 105}, and {75, 90, 105, 120}, respectively. Costs and fares are decided in the similar way as in the Set 1 problems.

Our experiments first consider the Set 1 problems. The different approaches are filtered out from further consideration if they yield poor results for these Set 1 test problems. The test cases of Set 2 are then used to further investigate the effectiveness of the more promising approaches, and for assessing their performance for different demand values.

The third set has two networks that are representative to the real sized networks in airlines. One network has 800 flights and 18196 paths in total and 300 flights in the designated family. The other network has 1060 flights and 33105 paths in total and 396 flights in the designated family. The problems in this set are used to test the final proposed approaches.

5.2 Preliminary Experiments

In Chapter 4, we presented several valid inequalities that tighten the LP relaxation for the DDR MIP formulation. However, adding all the valid inequalities simultaneously increases the number of constraints in the formulation, which may lead to an adverse overall effect. Consequently, we investigate in this section a variety of selection rules for the proposed valid inequalities. In particular, we attempt different ways to apply the individual propositions for the Set 1 problems, and select the best strategies to carry forward to further test using the larger problems in Set 2. The problems are solved by CPLEX 6.6.0 on a SUN Ultra 1

Workstation having 260 MB RAM and a clock-speed of 167 MHz. The results are presented in Tables 4 to 12. In these and the following tables, "time" refers to the CPU time used to solve the problem, excluding input and output processing times. All CPU times are shown in seconds, unless specified otherwise.

Our experiments follow the following sequence: (1) The a priori model improvements are attempted first, focusing on the CPLEX cuts and on Propositions 1 and 3. This is discussed in Section 5.2.1. (2) The convex hull of Z_l is tested next on the best model from (1). This is described in Section 5.2.2. (3) Valid inequalities are then generated based on Proposition 2 and the accompanying separation problem subroutine. These results are presented in Section 5.2.3. The foregoing tests are performed on the Set 1 problems. Only the models that yield good results are selected to test on the Set 2 problems, as detailed in Section 5.2.4. The most challenging problems from Set 2 are selected for further examination to investigate the application of CPLEX's options with respect to designing branching priorities and selecting various cut generation schemes. Section 5.2.5 discusses the related results obtained.

5.2.1 A Priori Model Improvement

Before applying the valid inequalities presented in the previous chapter, we tested the cuts provided by CPLEX itself. CPLEX applies some default preprocessing rules along with various branching, fathoming, and backtracking strategies in developing the branch-and-bound tree. These default options can be explored and changed according to the particular problem for improving the overall performance. We first examine CPLEX's cut generation options on the Set 1 problems. Later for the Set 2 problems, we will further explore various

branching strategies.

CPLEX provides six options of cuts: *cliquecuts* (clique inequalities), *covercuts* (cover inequalities), *flowcuts*, *fraccuts* (fractional inequalities), *gubcuts* (GUB inequalities), and *impliedcuts*. These cuts are valid inequalities that tighten the constraints for the LP relaxation while preserving the integer feasible solutions. By default, CPLEX performs internal tests on the problem, and applies cuts when the tests indicate that the cuts are likely to be beneficial. The generation of these cuts can also be specified by the user. We force these cuts to be generated to see whether any cuts are particularly effective for our problem. Table 4 shows the average computational time for the Set 1 problems under different cut options.

Table 4: Average CPU Time Under Different CPLEX Cutting Strategies

Group	CPLEX	cliquecuts		covercuts		flowcuts		fraccuts		gubcuts		impliedcuts	
	default	off	on	off	on	off	on	off	on	off	on	off	on
1	.90	.88	0.89	0.87	0.90	0.81	0.90	0.90	1.14	0.87	0.89	0.91	0.89
2	3.31	3.31	3.31	3.30	3.30	2.95	3.31	3.30	4.55	3.28	3.30	3.31	3.31
3	14.0	14.1	14.0	14.0	14.2	14.0	14.0	14.0	17.2	14.0	14.1	14.0	14.0
4	222	223	222	226	223	224	222	222	257	229	223	223	222
5	5.37	5.37	5.38	5.36	5.38	6.09	5.37	5.39	6.21	5.34	5.39	5.37	5.38

First, no restrictions on cuts are applied, i.e., by default, CPLEX determines whether to generate any types of cuts. To compare with this default setting, each cut is then turned off and on in turn while other cuts are set as default. Experiments indicate that the only cut that has a perceptible influence on the CPU time is fraccuts, which increases the CPU time when turned on. Since the default strategy with no cut generation restrictions is just as effective as the best of these options, we choose to apply the default settings in CPLEX

for the remaining experiments using Set 1 problems.

Propositions 1 and 3 are the most straightforward to be embedded into the original DDR model and are first examined. We tested (1) the original DDR-B model without any proposition used, (2) applying only Proposition 1 to the original model, i.e., using the valid replacement inequalities instead of the original constraints on total demand versus capacity, (3) the DDR model using the valid replacements from Proposition 1 and all the valid inequalities generated by Proposition 3, and, finally, (4) using Proposition 1 and controlled versions of Proposition 3, which are detailed in the following paragraph. Table 5 presents the numerical results for the CPU time and the number of branch-and-bound nodes enumerated for approaches (1) and (2). (Approach (3) was observed to be inferior in initial runs, and was therefore dropped from consideration.) Table 6 compares the results from several controlled versions of Proposition 3 on groups 1 to 5. Recall that if Proposition 3 is fully applied, the following valid inequalities are added to all the paths $i \in \Pi(l)$ for all the legs $l \in L_k$:

$$q_i \leq \sum_{t \in T_k} min\{\mu_i,\ \tilde{c}_{lt}\} x_{lt} \quad \forall\ i \in \Pi(l),\ \forall\ l \in L_k. \tag{72}$$

This increases the number of constraints markedly and is the major reason that approach (3) is slow. The purpose of approach (4) is to limit the number of constraints added by the above inequalities. We tested four criteria for selecting the valid inequalities, shown in Table 6 as C1, C2, C3, and C4. All of these options apply Proposition 1.

Firstly, in C1, Proposition 3 is applied to legs l only if $\tilde{\mu}_i < \tilde{c}_{lt},\ \forall\ i \in \Pi(l),\ t \in T$. If some $\tilde{\mu}_i > \tilde{c}_{lt}$, Proposition 1 alone is sufficient as it provides tighter constraints. However, this criterion is not too restrictive and still increases the number of constraints significantly.

Table 5: Average CPU Time and Number of Branch-and-Bound Nodes with Propositions 1 and 3.

group	Original DDR		With Prop. 1	
	time	node	time	node
1	0.8	6	0.79	7
2	3.09	10	3.03	10
3	13.12	225	12.83	225
4	219.35	2470	220.00	2468
5	5.43	11	5.36	11

Table 6: Numerical Results for Partial Applications of Proposition 3.

Group	Only Prop. 1		C1		C2		C3		C4	
	time	node	time	node	time	node	time	node	time	node
1	0.79	7	1.659	11	0.993	11	1.588	11	0.994	11
2	3.03	10	14.35	7	3.53	9	15.16	7	3.50	9
3	12.83	225	99.72	208	17.88	245	104.0	208	17.56	245
4	220.00	2468	1048.9	1849	202.2	1930	867.8	1517	206.6	1930
5	5.36	11	16.55	12	5.873	12	17.38	12	5.878	12

Three other alternatives are further tested. In C2, the valid inequality (72) is added for only one leg and one path, where the selected leg has the maximum total demand on all the paths crossing it, and the selected path has the minimum demand among these paths. In C3, the criterion is to select the paths that have the minimum demand among all the paths, while applying no restrictions to the legs. In C4, one leg is selected as in C2, while all paths crossing this leg are selected. The results show that C1 and C3 give almost identical results while C2 and C4 are similar and superior to C1 and C3. This may indicate that the decision on which leg to choose is critical.

The results from using approach (2) appear to be the best, with approach (1) being slightly inferior. Hereafter, we use the original DDR with only Proposition 1 applied as a

baseline to measure the efficiency of other approaches. Although when adding the controlled versions of Proposition 3 is not as good, C2 and C4 show some promise when the problems become more challenging. Hence, approach (2) and approach (4) under C2 are carried forward for later tests using the Set 2 problems.

5.2.2 Convex Hull Construction

Recall that in the previous chapter, we developed a direct construction of the entire convex hull of Z_l. This involved a disaggregation of the variables q_i for $i \in \Pi(l)$ into components Q_{ilt} $\forall\ t \in T_k$, given any $l \in L_k$. However, adding all the Q-variables into the model greatly increases the problem size. This prompts us to study only partial constructions of this convex hull to incorporate into the formulation. (We refer to this partial process in terms of adding the corresponding Q-variables into the model.)

Firstly, Q-variables are added a priori for some critical legs in the original DDR-B model before it is solved as an MIP. Five rules are applied in selecting the critical legs, as shown in columns 2 to 6 of Table 7. Rule 1: Q-variables are added when $\sum_{i\in\Pi(l)} \mu_i > \tilde{c}_{lt}$; Rule 2: Q-variables are added when $\sum_{i\in\Pi(l)} \mu_i < \max_{t\in T} Cap_{lt}$; Rule 3: Q-variables are added when $\tilde{c}_{lt} < \sum_{i\in\Pi(l)} \mu_i < \max_{t\in T} Cap_{lt}$; Rule 4: Q-variables are added for the flights that have the maximum total fare ($\max_{i\in\Pi(l)} f_i\,\mu_i$); and Rule 5: Q-variables are added for the flights that have the maximum average fare ($\max_{i\in\Pi(l)}\{\frac{f_i\,\mu_i}{|\Pi(l)|}\}$). Results indicate that the second and third rules are both very good. However, later experiments indicate that the fourth and fifth rules perform much better for larger problems.

Table 7 also shows results from a posteriori application of this idea. The model is solved in two stages: at Stage 1, the LP-relaxation is solved. At Stage 2, Q-variables are

incorporated for the fractional legs and the model is solved as an MIP. During Stage 2, we adopt two versions: the integral legs obtained from Stage 1 are not fixed in the first version, while they are fixed in the second. While the second version does not guarantee optimality, its objective value is very near to the optimum value in our experimental runs (see the "obj%" column which shows the average percent of the objective value obtained to the global optimum value), and the CPU times are smaller than for version 1. Still both versions are rather slow.

Table 7: Average CPU Time for the Q-variable Application for Critical Flights

group	a priori					a posteriori		
	rule 1	rule 2	rule 3	rule 4	rule 5	non fixing	fixing	
	time	time	time	time	time	time	obj%	time
1	8.28	0.92	0.92	1.71	1.02	4.65	99.98	3.29
2	54.24	3.33	3.33	8.22	4.28	35.49	99.86	25.44
3	247.35	16.04	16.03	49.60	18.9	175.77	99.96	155.19
4	362.42	142.77	143.03	234.70	94.8	246.41	100	210.01
5	232.41	7.72	7.72	23.80	11.1	173.88	99.91	153.14

We also attempted to incorporate Q-variables both a priori and a posteriori for certain critical flights based on Rule 2 as described above. Here, in the second stage, the integral solutions from Stage 1 were fixed and the MIP was solved with another group of Q-variables incorporated via Rule 2. (Different ways to select Q-variables were also tested at Stage 2.) However, none of them outperformed the pure a priori version. Furthermore, although in the a priori version, Rules 2 and 3 performed well on the Set 1 problems, this performance deteriorated when the network grew in size. In fact, for the Set 2 larger problems, Rules 4 and 5 perform very well, yielding the best results in certain combinations of strategies. Hence, we recommend the foregoing options only for relatively small-sized problems of the

type contained in Set 1.

5.2.3 Generation of Valid Inequalities

Results from implementing the separation problems based on Proposition 3 are shown in Table 8, and are compared with the DDR-B model. Two versions of this approach were

Table 8: Average CPU Time for the Separation Problem Application

Group	SP	SP2	DDR
1	1.51	1.70	0.79
2	5.59	5.26	2.83
3	20.47	19.13	13.55
4	97.57	96.35	218.31
5	15.02	14.41	5.68

tested. In the version SP, we simply solved the separation problem, and generated the associated cut if one was obtained (i.e., the objective value of the separation problem was positive). The revised LP solution was then fed back into the separation problem for generating another cut. This loop was repeated until no cut was generated from the separation problem subroutine. In the second version called SP2, a GUB cut generated from Proposition 2 was also embedded into the foregoing SP approach. First, the separation routine was used to generate a cut and to update the corresponding DDR model LP solution as above. If no new cut was generated by this process, we exited this routine. Otherwise, we examined next the leg that yielded the maximum difference between $\max_{t \in T} \tilde{c}_{lt}$ and a_{lt_l} (where $a_{lt_l} = \min_{t \in T} a_{lt}$) to see if a GUB cut of Proposition 2 could be generated. If so, the DDR model's LP solution was updated after incorporating this new GUB cut. The procedure then reverted to solving the separation routine. This loop was repeated until either

no cut was generated from the separation problem, or a pre-specified number of loops was reached. This version SP2 yielded very encouraging results, especially when the problems became larger in size. As we shall see later, SP2 performed extremely well on the Set 2 test problems.

5.2.4 Further Experiments on Set 2 Problems

Based on the numerical results from Set 1, we experimented with the most promising methods using the Set 2 problems. Recall that we have three groups of networks in the Set 2 problems, each having a total number of flights of 150, 200, and 400, respectively. We first solved the 150- and 200- leg group problems using the selected promising approaches as described below. The average CPU times and the average percentage of optimality achieved by the heuristically run versions are summarized in Table 9.

The following algorithmic compositions were tested (see the columns of Table 9). The original DDR model was first solved, followed by the application of Proposition 1 (Prop1). It is noticed that when the problems become relatively more challenging, especially when the original DDR model requires more than one hour (3600 seconds) to solve the problem, applying Proposition 1 usually performs better. Thus in the following experiments, Proposition 1 is always applied.

Next, we attempted to generate valid inequalities using the separation problem routine. Two different approaches were adopted after adding all the constraints generated from the separation problem. In SPF, we fix the variables that are already binary valued after adding the generated cuts before solving the resulting DDR problem as an MIP. In the second version SPU, the cut-augmented DDR model is solved exactly as an MIP without

Table 9: Numerical Results from Different Approaches when Applied to the Set 2 Problems.

$\|L\|$	μ	cv	DDR	Prop1	SPF		SPU	Q (1)	Q (2)	Q-sp-prop2		Q-sp-prop2,3	
			time	time	obj%	time	time	time	time	obj%	time	obj%	time
150	5	.15	583	581	100	278	701	1298	1211	100	358	100	387
150	5	.45	2161	2155	100	520	3460	967	703	99.98	546	99.98	538
150	8	.15	85	84	100	103	154	273	208	100	103	100	106
150	8	.45	154	156	100	113	202	891	199	100	108	100	107
150	15	.15	59	57	99.99	69	95	179	96	100	72	100	73
150	15	.45	54	55	99.99	62	74	188	76	99.99	63	99.99	63
200	5	.15	3657	2492	100	1173	3338	3410	3523	100	1043	100	1040
200	5	.45	5432	5408	100	1655	5211	7971	3953	100	1351	100	1602
200	8	.15	963*	1156*	100	517	863*	6451	5673	100	552	100	541
200	8	.45	1934*	1931*	100	682	2840*	6688	6049	100	713	100	686
200	15	.15	158	159	100	87	136	267	220	100	83	99.99	94
200	15	.45	152	159	99.98	121	180	246	181	99.98	107	99.98	114

*: One network could not be solved within the six hour limit, and was therefore not included in the average calculations.

any such fixings. While the version SPF may not yield an optimal solution, it is shown to produce solutions very close to optimality (at an average, more than 99.99% of optimality) and the computational time is typically half and sometimes one third of that for SPU (on average, the CPU times for SPF are 54.3% of that for SPU). Hence, in the following tests, we apply the fixed version strategy SPF.

As mentioned in Section 5.2.2, the first three rules stated therein when applied a priori for the Q-variables reformulation did not appear to be encouraging for solving larger sized problems. However, Rules 4 and 5 performed very well on the Set 2 problems. Their results are given in columns Q(1) and Q(2), respectively. Recall that Rule 4, or Q(1), adds the Q-variables for the flights that have the maximum total fare, and Rule 5, or Q(2), adds the Q-variables for the flights that have the maximum average fare. Q(2) performs better

and is selected to be experimented with further. A combination of Rules 4 and 5 was also tested, i.e., Q-variables are incorporated for flights that have either the maximum total fare or the maximum average fare. The results were not as good as for applying Q(2) alone.

The average CPU times for an integrated approach using the separation routine with the Q-variable formulation, as well as using Propositions 2 and 3, are given in the last two columns of Table 9. In this integrated approach, the Q-variables are added a priori for those flight legs having the maximum average fare, as in the version Q(2) described above. Proposition 3, when used, is applied partially as for the strategy C2 described in Section 5.2, i.e., the valid inequality is added for only one leg and one path, where the leg has the maximum total demand on all the paths crossing it, and the path has the minimum demand among these paths. The SP2 version was shown to be very promising for the Set 1 problems. In the present integrated approach, it is applied to the model having Q-variables in a likewise fashion. These two integrated approaches, called *Q-sp-prop2* and *Q-sp-prop2,3*, respectively, improve CPU times remarkably. In several cases, they reduce the computation time by a factor of 17 in comparison with the original DDR-B formulation (at an average, the reduction in effort is by a factor of 2.3). In the 200-leg group, when the problems become very hard to solve, the *Q-sp-prop2* approach outperforms SPF, the best approach so far.

The results also indicate that the means and variances of demands influence the computational effort. The problems become more challenging when the demands become smaller. When the mean demand is small, an increase in the variance increases the difficulty of solving the problem. It is worth noticing that the SPF strategy outperforms the integrated

approaches when the demands are larger, while otherwise, when the demands are smaller and the problems take hours to solve, the integrated approaches perform better. In our test problems, in particular, when the demand distribution has a mean of 5, or a mean of 8 and a coefficient of variation of 0.45, the problems take very long to solve. It is these problems that are in most need of good algorithms. In the following, we focus our tests on such problems only.

5.2.5 Node Selection and Branching Priorities and CPLEX Cuts

At each node of the branch-and-bound tree, CPLEX associates a value that is used to select nodes in the enumeration process. By default, the "best bound" rule is applied, i.e., the lower bound on the integer optimum as obtained from the LP subproblem solved at the node is used as the designated value. An alternative is to use the "best estimate" rule, which computes an estimate of the best objective value that can be achieved from the node subproblem. This alternative significantly reduced the computational time in our test problems. In several cases, it solved the problem in only half of the CPU time as compared with the default "best bound" option. In Table 10, we compare the computational times on the 200-leg group test problems for these two node selection alternatives when using *Q-sp-prop2*.

Table 10: Average CPU Times for Two Node Selection Strategies.

$\|L\|$	μ	c.v.	best bound	best estimate
200	5	0.15	1043	582.1
200	5	0.45	1350.8	1100.9
200	8	0.45	713.0	630.0

In addition, CPLEX users can also prioritize the integer variables, so that the variables having higher priorities are branched earlier at any selected node. In our experiment, we tested three strategies for prioritizing the flight variables. First, we give higher priorities to the flights having larger total demands ($\{\sum_{i\in\Pi(l)} \mu_i\}$ for $l \in L_k$). Second, we give higher priorities to the flights having larger average demands ($\{\sum_{i\in\Pi(l)} \frac{\mu_i}{|\Pi(l)|}\}$ for $l \in L_k$). Both of these strategies (as well as their reverse versions) degraded CPLEX's performance. Hence, we used CPLEX's default strategy.

Recall that we tested on the Set 1 problems the cut generation options within CPLEX. We next re-tested these options on the Set 2 problems using the integrated approach *Q-sp-prop2* with the best estimate node selection strategy. Table 11 compares the average CPU times for the different cut options applied to this combination strategy.

Table 11: Average CPU Times for Different Cuts Options.

$\|L\|$	μ	c.v.	default	cliques	covers	flows	fracs	gubs	implieds
200	5	0.15	582	726	721	570	526	719	735
200	5	0.45	1101	1154	1118	1097	1009	1131	1126
200	8	0.45	630	704	697	617	651	700	703

The flowcuts is the only option that improves the performance. The fraccuts option also enhances the performance sometimes, but degrades it in some cases, and is not stable. We further tested a strong application of the flowcuts option, but the results were not as good as the default application.

Based on the results from the 150- and 200-leg group problems, we selected SPF and the *Q-sp-prop2,3* approaches with the "best estimate" node selection option and the flowcuts option to test on the 400-leg group problems. Table 12 presents the results from the original

DDR model, the SPF approach, and the *Q-sp-prop2,3* approach. We first tried to solve the problem using the original DDR model. However, among the three networks, only the first network could be solved within the pre-specified 10-hour time limit. The second and the third networks only returned the best integral solutions that were detected within 10 hours, and therefore we do not have a verified optimal objective value for these two instances. When using the (heuristic) approaches that fix the binary-valued variables after the final node-zero LP relaxation, the second network could be solved by SPF and *Q-sp-prop2,3* much faster and achieved the same objective value. The third network could also be solved within the time limit, and in this case, formed slightly improved objective values. The *Q-sp-prop2,3* approach improved the best objective value found for the original DDR model run by 0.3%, and the SPF approach improved this DDR value by 0.2%.

Table 12: CPU Times for Three 400-leg Problems.

Problem	DDR	SPF		Q-sp-prop2,3	
	time	obj%	time	obj%	time
1	20400	99.99	4869	99.99	3041
2	36000	100	5452	100	4299
3	36000	100.20	9634	100.30	11339

5.3 Solution Approaches

In this section, we summarize the integrated approach by proposing a heuristic approach in Section 5.3.1, and an optimal algorithm in Section 5.3.2. Both approaches were tested on real sized networks and the results are presented in Section 5.3.3.

5.3.1 Summary of the Proposed Heuristic Approach

Based on the integrated approaches, we propose the following heuristic to solve the DDR-B model:

Initialization:

Preprocess the DDR-B model established in Section 3.6.1 by the two steps proposed in Section 4.1.1 so that Proposition 1 is applied to the set of constraints (4d).

Step 1: Replace (4f) with the valid inequality $q_i \leq \sum_{t \in T_k} min\{\mu_i,\ \tilde{c}_{lt}\} x_{lt}$ from Proposition 3 for the leg that has the maximum total demand on all the paths crossing it, and the path that has the minimum demand among these paths. Add the Q-variables for the flights that have the maximum average fare. Solve the resulting LP relaxation by relaxing the binary constraints.

Step 2: If the solution to the LP relaxation satisfies the binary restrictions, stop; otherwise, use the current solution to solve the separation problem. If the objective value of the separation problem is zero (no cut is generated), go to Step 4; otherwise, generate a cut based on the solution and re-solve the DDR-B model's LP relaxation with this new cut. Proceed to Step 3.

Step 3: If a binary valued solution is obtained, stop; otherwise, use the leg that yields the maximum difference between $\max_{t \in T} \tilde{c}_{lt}$ and a_{lt_l} among all the fractional variables (i.e., $\max_{l \in L_k : x_{lt}\ fractional} (\max_{t \in T} \tilde{c}_{lt} - a_{lt_l})$) to generate a GUB inequality of Proposition 2. If this inequality cuts off the current solution, add it to the model and resolve. Return to Step 2.

Step 4: If the solution is binary, stop; otherwise, fix the variables that are binary

valued in the LP relaxation solution at their current values, and fix x_{lt} at 1 if it is greater than 0.95. Solve the resulting MIP problem by setting the node selection option to "best estimate", and switching on the "flowcuts" in CPLEX.

5.3.2 The Proposed Optimal Approach

Prompted by the heuristic approach, and noticing that the integrated procedure achieves near-optimal solutions, we modify the final step of the heuristic method and propose the following exact algorithm. Notice that this approach retains the LP processing steps from the heuristic procedure from Section 5.3.1.

Initialization:

Preprocess the DDR-B model described in Section 3.6.1 by the two steps proposed in Section 4.1.1 so that Proposition 1 is applied to the set of constraints (4d).

Step 1: Replace (4f) with the valid inequality $q_i \leq \sum_{t \in T_k} min\{\mu_i,\ \tilde{c}_{lt}\} x_{lt}$ from Proposition 3 for the leg that has the maximum total demand on all the paths crossing it, and the path that has the minimum demand among these paths. Add the Q-variables for the flights that have the maximum average fare. Solve the resulting LP relaxation by relaxing the binary constraints.

Step 2: If the solution to the LP relaxation satisfies the binary restrictions, stop; otherwise, use the current solution to solve the separation problem. If the objective value of the separation problem is zero (no cut is generated), go to Step 4; otherwise, generate a cut based on the solution and re-solve the DDR-B model's LP relaxation with this new cut. Proceed to Step 3.

Step 3: If a binary valued solution is obtained, stop; otherwise, use the leg that yields

the maximum difference between $\max_{t \in T} \tilde{c}_{lt}$ and a_{lt_l} among all the fractional variables (i.e., $\max_{l \in L_k : x_{lt} \ fractional} (\max_{t \in T} \tilde{c}_{lt} - a_{lt_l})$) to generate a GUB inequality of Proposition 2. If this inequality cuts off the current solution, add it to the model and resolve. Return to Step 2.

Step 4: If the solution is binary, stop; otherwise, prioritize the solution variables by imposing a higher priority to the set of the variables that are still fractional valued in the LP relaxation solution and a lower priority to the set of the binary valued variables. Allow an ϵ- optimality tolerance with the MIP solver (CPLEX 6.6.0) and solve the resulting mixed-integer problem by setting the node selection option to "best estimate", and switching on the "flowcuts" in CPLEX. (In our computations, we used $\epsilon = 0.05$.)

We remark here that in practice, airlines would have the original fleet assignment at hand, which is a feasible solution to our model. This original fleet assignment value can be set as a lower bound to speed up the fathoming process in both of the above approaches.

5.3.3 Experimental Results Obtained from Real-sized Problems

We tested the proposed approaches on the Set 3 problems. This set has two networks. The first one has 800 legs and 18196 paths in total, with 300 legs in the family, and the second one has 1060 legs and 33105 paths in total, with 300 legs in the family. Both networks are tested on two-type, three-type, and four-type problems with different means and coefficients of variation on demands. Table 13 compares the average CPU time for solving these real sized problems with the original DDR model and the proposed ϵ-optimal and heuristic methods.

In Table 13, ORG, OPT, and HEU columns show the computing times by using the

Table 13: CPU Times for Real Sized Problems.

Capacities					95,112,(126)			65,75,(85)			65,90,(115)		
$\|L\|$	$\|L_k\|$	$\|T\|$	μ	c.v.	ORG	OPT	HEU	ORG	OPT	HEU	ORG	OPT	HEU
800	300	2	5	0.15	26	51	45	52	144	107	**	4051	4123
			8	0.45	38	86	111	36	110	96	3660	1491	490
			15	0.45	24	74	70	32	207	94	76	224	115
		3	5	0.15	1183	535	446	**	14673	1464	**	**	**
			8	0.45	13816	13397	1757	**	8239	1391	**	**	**
			15	0.45	85	279	243	235	378	261	1843	631	462
1060	396	2	5	0.15	25	148	93	60	173	150	**	**	**
			8	0.45	64	242	160	91	198	152	**	3294	7269
			15	0.45	76	178	167	75	180	142	**	666	619
		3	5	0.15	1415	1972	673	**	5634	871	**	**	**
			8	0.45	**	12132	2302	2145	3164	712	**	**	**
			15	0.45	179	440	308	240	413	269	**	1325	569

Note: ** denotes the problem is not solved because of "out of memory" error.

original DDR model, the proposed ϵ-optimal method, and the proposed heuristic method, respectively. We summarize the following findings from the results:

1) If a problem was solved fast by the original model, it could also be solved relatively fast using the proposed methods.

2) The proposed methods are more reliable than the original model. Eight problems were not solvable by the original model, but were solved by the proposed methods with the prescribed limits. (Within 5 hours by CPLEX 6.6.0 on a SUN Ultra 1 Workstation having 260 MB RAM and a clock-speed of 167 MHz.)

3) Comparing the optimal method with the heuristic method, even though the former consumes greater CPU time by an average factor of 2.7, the number of problems solvable by this method does not decrease.

4) The proposed heuristic provides very good quality solutions. For the 31 problems

that are solvable by OPT, the heuristic yielded 100% optimality for 29 of them, and 99.99% for the other two, reaching an average of 99.999% of optimality of those solved by the (ϵ)-optimal algorithm.

Problems having 4 types of aircraft in the family were also tested with the proposed heuristic. The network having 800 flights was solved in 4, 6, and 14 hours, and the network having 1060 flights was solved in 6 hours. These CPU times are considered acceptable as this model is solved in the planning stage.

Chapter 6

Summary, Conclusions and Future Research Directions

This chapter provides a summary and conclusions for our work in Section 6.1, and propose some recommendations for future research in Section 6.2.

6.1 Summary and Conclusions

This thesis presents models and solution approaches for the DDR (Demand Driven Refleeting) problem. In the airline scheduling process, the initial fleet assignments are completed with only preliminary demand information. The objective of the DDR problem is to revise the initial fleeting assignments as and when better demand forecasts become available, in order to provide a better match between demands and aircraft capacities, so as to maximize the total revenue.

We first proposed an overall three-stage framework for supply fleet management in airlines taking advantage of inherent flexibilities in system capacity. In this framework,

continuously updated demand forecast information is utilized so that the capacities assigned to flights are better matched with the demand, and the spill and spoil costs are consequently reduced.

In Chapter 3, we presented four DDR models: the basic DDR model, the DDR model considering path-class levels, the DDR model with DOW/DOM (Day of Week and Day of Month) variations, and the DDR model with re-capture (i.e., a proportion of lost path level demands can be satisfied on other paths, if needed). Prior to our work, fleet assignment and revenue management models have traditionally been solved in a sequential process. The fleet assignment is decided at an earlier stage when only preliminary demand information is available, and a revenue management study is used to maximize the total revenue, given the prescribed fleet assignment. One of the contributions of the DDR models is that by integrating the fleeting and revenue management decision processes, the fleeting decisions are adjusted based on the feedback from the revenue management for the potential of a better profit.

A third contribution of these models is the inclusion of the path level demand information. Prior to these models, demands on each leg were considered to be independent. However, after the introduction of hub-and-spoke systems in airline networks, about 40% of the passenger trips in the US consist of more than one leg. Therefore, demands are highly dependent between flight legs. Path level demands introduce more accurate information to the fleeting and revenue management decisions, but on the other hand, lead to a significant increase in the size of the problem to be solved. In our test problems, we constructed paths having up to 3 legs. On average, the total number of the paths for the networks having 200,

400, 800, and 1060 flights in our tests are 4424, 8347, 18196, and 33105, respectively.

Even though we apply the model to a single family, the number of paths still make it implausible to solve the model directly as a traditionally formulated MIP. We proposed several reformulation, partial convex hull constructions, and various classes of valid inequalities to tighten the formulation, and used randomly generated realistic networks to test for the performance of different algorithmic compositions of these derived strategies. We discovered that incorporating entire sets of the proposed valid inequalities can increase the size of the model significantly and make it intractable, while versions having carefully selected sets of such valid inequalities can generally enhance the overall performance. Moreover, when the LP relaxation is solved for these tightened versions, a relatively greater number of variables turn out to be binary valued. When these binary variables are (heuristically) fixed in value to reduce the size of the MIP that is fed into CPLEX, the CPU times to solve the problem are greatly reduced, while the quality of the solution is only mildly degraded (the average objective value attained 99.99% of optimality).

The best of these methods tested turned out to be an *integrated approach* that employ selected valid inequalities based on Propositions 1, 2, and 3, generating cuts from a separation routine, and constructing a particular partial convex hull representation. Using this integrated approach along with properly selected CPLEX strategies cuts the CPU time by an average factor of 7.48 for the 200-leg test problem group. Based on the integrated approach, we proposed a heuristic approach and an optimality approach. These two approaches were tested on some practical size problems and yielded very encouraging results. For example, a typical large-scale practical sized problem involving 1,060 flights and 33,150

paths, with 396 flights and 4 types belonging to the designated family, was solved within 6 hours, on a SUN Ultra 1 Workstation having 260 MB RAM and a clock-speed of 167 MHz. This is acceptable considering that such models are solved at a planning stage in the decision process.

Another re-fleeting model proposed by Jarrah *et al.* (2000) tests problems having 241 to 818 flights. Their model was solved on an HP C160 workstation with 256 MB RAM and a clock-speed of 160 MHz. Their solutions were obtained within one hour. The difference between our model and theirs lies in the more accurate and realistic path level demand information that we consider. As afore-mentioned, even the 200-flight problems have an average of 4424 paths, which increases the number of constraints drastically. Hence, our class of problems are significantly more challenging.

6.2 Future Research Directions

The largest test network considered in our research, having 1,060 flights in total, with 396 flights in the designated family, is comparable to real-sized networks in the airline industry. To further validate the proposed approach, we recommend that it be tested on more networks of this type, having a variety of demand distributions.

Hane *et al.* (1995) proposed node aggregation and island construction approaches for solving large-scale fleet assignment problems. These approaches were implemented by many researchers (see, for example, Jarrah *et al.*, 2000), and shown to be very effective. Hane *et al.*'s results show that node aggregation alone can lead to a reduction of the CPU times for the fleet assignment problem by a factor of 2 to 7. As a topic for future research, these

strategies could be applied to our model for enhancing the problem handling capability.

The model we presented solves the re-fleeting problem for a single family. In this case, the sequence in which these models for the families are solved will have an impact on the overall decision. Analyzing this issue is another research direction. Furthermore, DDR model can be extended such that re-fleeting is done for all families simultaneously (as long as each leg is re-assigned within its original family).

We have presented four graded models in this thesis, and have proposed a specified solution approach for the basic structured model in this set. The more advanced DDR models, as well as the model with maintenance constraints which is included in the appendix of this thesis, have certain additional features and constraints whose structure needs to be further explored in order to extend our solution approach to solving these problems.

In this thesis, we have focused on the model and solution approaches for the re-fleeting problem, without addressing any feedback process to the initial fleeting stage. An important future research direction can be a study on how the initial fleet assignment, the DDR problem, and the subsequent swapping process interact, and on the impact of the initial fleet assignment on the DDR phase and the swapping mechanisms.

Finally, our model integrates the fleet assignment model and the revenue management decision strategy with a focus on path level demand information. Further research can include other scheduling considerations such as aircraft routing and crew scheduling, as well as other yield management decisions, such as pricing and protection level decisions.

Appendix

Appendix 1 Extending FAM with Maintenance Constraints

As mentioned before, aircraft maintenance requirements are crucial constraints that need to be satisfied by the aircraft schedules. These maintenance requirements include major and minor part inspections and tool replacements that need to be performed regularly. The Federal Aviation Administration (FAA) requires four types of aircraft maintenance, known as the **A, B, C** and **D** checks (Gopalan and Talluri, 1998). Among them, **A** and **B** checks are shorter than 24 hours (Clarke, 1996). **A** checks are the most frequent (every 65 flight hours according to FAA's requirement), and are the shortest (about 4 hours), whereas **B** checks require 10 to 15 hours. However, airlines generally impose stricter maintenance constraints than FAA requirements to ensure safety. In addition, the type of maintenance, frequencies, durations, and the required equipment and maintenance stations vary depending on the configuration of the aircraft, and therefore, are usually different for different aircraft types. Thus, an input to the airline scheduling process includes a planned maintenance schedule, aggregated for each aircraft type, which designates a list of maintenance requirements during a specified time-window and at a particular maintenance station. Typically, a particular maintenance station has the facility to accommodate only a limited number of aircraft types.

An aircraft, then, has to be assigned to flights so that it spends enough time at a qualified station during its maintenance time.

It is possible to extend FAM such that it also includes *aggregate maintenance constraints* at the type level. In the following, we first introduce the additional notation that will be needed in this section, and then revise the formulation given in Model 1 to include these aggregate maintenance constraints.

<u>Additional Notation:</u>

PL	:	set of A and B maintenance requirements, indexed by p
$MC(p)$	:	set of maintenance arcs that are candidates for satisfying maintenance requirement $p,\ p \in PL$
M_t	:	set of maintenance arcs in type t's network, $t \in T$; indexed by m
S_p	:	number of aircraft needed to satisfy maintenance requirement $p,\ p \in PL$

$$bc_{mn} = \begin{cases} 1, & \text{if maintenance arc } m \text{ begins at node } n \text{ (in type } t\text{'s network)}, \\ & m \in M_t,\ n \in N_t,\ t \in T \\ -1, & \text{if maintenance arc } m \text{ ends at node } n \text{ (in type } t\text{'s network)}, \\ & m \in M_t,\ n \in N_t,\ t \in T \\ 0, & \text{otherwise.} \end{cases}$$

In addition, we let decision variables z_m denote the number of aircraft (of type t) on maintenance arc m in type t's network, $m \in M_t,\ t \in T$. We also update sets CS_t, $t \in T$, such that now they also include those maintenance arcs passing through the counting time line.

We then replace constraints (1b) and (1c) with constraints (73a) and (73b), and add constraints (73c) and (73d), as given below:

$$\sum_{l \in L} bf_{ln}\ x_{lt} + \sum_{g \in G_t} bg_{gn}\ y_g + \sum_{m \in M_t} bc_{mn}\ z_m = 0, \quad \forall n \in N_t,\ \forall t \in T \tag{73a}$$

$$\sum_{l \in CS_t} x_{lt} + \sum_{g \in CS_t} y_g + \sum_{m \in CS_t} z_m \leq A_t, \quad \forall\, t \in T \tag{73b}$$

$$\sum_{m \in MC(p)} z_m + zs_p = S_p, \quad \forall\, p \in PL \tag{73c}$$

$$z_m : \ integer, \quad \forall m \in M_t, \ \forall\, t \in T \tag{73d}$$

Observe that constraints (73a) and (73b) now include the aircraft under maintenance. Also, recall that the parameter S_p specifies the number of aircraft needed to satisfy maintenance requirement p, $p \in PL$. This is considered in constraints (73c). We allow this constraint to be violated at a penalty: this is done by adding slack variables zs_p, which correspond to the shortage in the required number, and by adding $ViolCost \ \sum_{p \in PL} zs_p$ to the objective function, where $ViolCost$ is the penalty per unit shortage in maintenance requirements. Note that because of the critical nature of the maintenance requirements, in lieu of a linear penalty term for the slack variables zs_p, we could alternatively have imposed some increasing rate penalty function, and/or bounded each zs_p from above by a maximum permissible violation in this maintenance requirement. Finally, constraints (73d) require each maintenance variable to be integral.

As mentioned previously, these aggregate maintenance constraints will only assure that the maintenance activities selected satisfy the aggregate requirements, but they cannot guarantee satisfaction of the maintenance requirements by each individual aircraft (tail). Thus, it is possible that a feasible maintenance schedule in the type level solution turns out to be infeasible in the individual aircraft tail level problem. Such conflicts need to be resolved in the routing model.

References

Abara, J. (1989), Applying Integer Linear Programming to the Fleet Assignment Problem. *Interfaces*, Vol. 19, 20–28.

Barnhart, C., Boland, N. L., Clarke, L. W., Johnson, E. L., Nemhauser, G. L., and Shenoi, R. G. (1998), Flight String Models for Aircraft Fleeting and Routing. *Transportation Science*, Vol. 32, No. 3, 208–220.

Belobaba, P. and Lee, S. (2000), PODS Update Large Network O-D Control Results. *AGIFORS RM Study Group Meeting*, New York City.

Berge, M. E. and Hopperstad, C. A. (1993), Demand Driven Dispatch: A Method for Dynamic Aircraft Capacity Assignment, Models and Algorithms. *Operations Research*, Vol. 41, No. 1, 153–168.

Cao, J. and Fan, H. (2000), Integration of Optimizing Flight Network, Fleet Assignments and Aircraft Routes. *Working paper*, Nanyang Technological University, Singapore.

Clarke, L. W., Hane, C. A., Johnson, E. L. and Nemhauser, G. L. (1996), Maintenance and Crew Considerations in Fleet Assignment. *Transportation Science*, Vol. 30, No. 3, 249–260.

Eglese, R. W. (1990), Simulated Annealing: A Tool for Operational Research. *European Journal of Operational Research*, Vol. 46, 271–281.

Feo, T. A. and Bard, J. F. (1989), Flight Scheduling and Maintenance Base Planning. *Management Science*, Vol. 35, 1415–1423.

Fourer, Robert, Gay, D. M. and Kernighan, B. W. (1993), AMPL: A Modeling Language for Mathematical Program. The Scientific Press.

Fourer, Robert, Gay, D. M. and Kernighan, B. W. (1997), AMPL: A Modeling Language for Mathematical Program with AMPL Plus 1.6 Student Edition for Microsoft Windows. Duxbury Press.

Gopalan, R. and Talluri, K. T. (1998), Mathematical Models in Airline Schedule Planning: A Survey. *Annals of Operations Research*, Vol. 76, 155–185.

Hane, C. A., Barnhart, C., Johnson, E. L., Marsten, R. E., Nemhauser, G .L., and Sigismondi, G. (1995), The Fleet Assignment Problem - Solving a Large-Scale Integer-Program. *Mathematical Programming*, Vol. 70, No. 2, 211–232.

Jacobs, T. L., Ratliff, R. M. and Smith, B. C. (2000), Soaring with Synchronized Systems. *OR/MS Today*, August 2000, 36–44.

Jarrah, A. I., Goodstein, J., Narasimhan, R. (2000), An efficient airline re-fleeting model for the incremental modification of planned fleet assignments. *Transportation Science*, Vol. 34, No. 4, 349–363.

Kirkpatrick, S., Gelatt, C. D. Jr. and Vecchi, M. P. (1983), Optimization by Simulated Annealing, *Science*, Vol. 220: 671–680.

Mahajan, S. and van Ryzin, G. (1998), Stocking Retail Assortments under Dynamic Consumer Substitution. *Working Paper*, Columbia University, NY.

McGill, J. and van Ryzin, G. (1999), Revenue Management: Research Overview and Prospects. *Transportation Science*, Vol. 33, No. 2, 233-256.

Mulvey, J. M., Vanderbei, R. J. and Zenios, S. A. (1995), Robust Optimization of Large-scale Systems. *Operations Research*, Vol. 43, No. 2, 264–279.

Pastor, J. (1999), Demand Driven Dispatch: A Stochastic Approach. *AGIFORS Symposium*, New Orleans, Louisiana.

Rushmeier, R. A. and Kontogiorgis, S. A. (1997), Advances in the Optimization of Airline Fleet Assignment. *Transportation Science*, Vol. 31, No. 2, 159–169.

Rexing, B., Barnhart, C., Kniker, T., Jarrah, A. and Krishnamurthy, N. (2000), Airline Fleet Assignment with Time Windows. *Transportation Science*, Vol. 34, No. 1, 1–20.

Sherali, H. D., Adams, W. P. and Driscoll, P. J. (1998), Exploiting Special Structures in Constructing a Hierarchy of Relaxations for 0-1 Mixed Integer Problems. *Operations Research*, Vol. 46, No. 3, 396–405.

Sherali, H. D. and Driscoll, P. J. (2000) Evolution and State-of-the-Art in Integer Programming. *Journal of Computational and Applied Mathematics*, Vol. 124, No. 1–2, 319–340.

Sherali, H. D. and Lee, Y. G. (1995), Sequential and Simultaneous Liftings of Minimal Cover Inequalities for Generalized Upper Bound Constrained Knapsack Polytopes. *SIAM Journal On Discrete Mathematics*, Vol. 8, No. 1, 133–153.

Smith, D. H., Hurley, S. and Thiel, S. U. (1995), Frequency Assignment Algorithms Radiocommunications Agency Agreement. Interim Report, Ref. RCCM 070.

Sosnowska, D. (2000), Optimization of a Simplified Fleet Assignment Problem with Metaheuristics: Simulated Annealing and GRASP. *Approximation and Complexity in Numerical Optimization: Continuous and Discrete Problems*, Kluwer Academic Publishers, 477–488.

Talluri, K. T. (1996), Swapping Applications in a Daily Airline Fleet Assignment. *Transportation Science*, Vol. 30, No. 3, 237–248.

Teodorovic, D. (1988), *Airline Operations Research.* Gordon and Breach Science Publishers, New York.

van Ryzin, G. (2000), Why We Need Discrete Choice Theory in Revenue Management? *AGIFORS Symposium*, New Orleans.

Yu, G. and Yang, J. (1998), Optimization Applications in the Airline Industry, in *Handbook of Combinatorial Optimization*, Vol. 2, Kluwer Academic Publishers.

Vita

Xiaomei Zhu was born on March 7, 1976 in Xi'an, China. She did her BS in Industrial International Trade from the Xi'an Jiaotong University. After completion of her BS, she continued graduate study in the Institute of Strategy and Decision-Making in the Xi'an Jiaotong University for two years. In August, 1999, she started to pursue her Master's degree in Industrial Engineering at Virginia Tech. During her MS, she maintained strong academic achievements. She worked for three semesters as a Teaching Assistant in the Grado Department of Industrial and Systems Engineering, and one semester as a Research Assistant with Dr. Bish on the Demand Driven Re-fleeting project.During her second year as a Master's student, she held the post of "Vice President of INFORMS", Virginia Tech Chapter.

After graduation, she joined Exeter Group and will be working as an IT consultant. Her research interests are in Airline Operations Research and Optimization.