COLUMN GENERATION AND NETWORK MODELING IN LARGE-SCALE LOGISTICS NETWORKS

by

ZHE LIANG

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Doctor of Philosophy

Graduate Program in Industrial and Systems Engineering

Written under the direction of

Dr. Wanpracha Art Chaovalitwongse

And approved by

_____

_____

_____

_____

_____

New Brunswick, New Jersey

MAY, 2011

ABSTRACT OF THE DISSERTATION

COLUMN GENERATION AND NETWORK MODELING IN LARGE-SCALE LOGISTICS NETWORKS

By ZHE LIANG

Dissertation Director:
Dr. Wanpracha Art Chaovalitwongse

Decomposition has been used in solving numerous problems in mathematics, computer science, engineering, management, and operations research. In this dissertation, we use decomposition methods to solve three practical combinatorial optimization problems arising in telecommunication and airline planning.

In the first part of the dissertation, we study a redundant multicast routing problem with group diverse constraint (RMRGD) that arises in many network applications such as communication systems, power supply distribution systems, transportation networks, etc. We propose three mixed integer programming (MIP) models, an edge-based, a path-based, and a tree-based model, to solve RMRGD. We proposed two decomposition methods based on the column generation and branch-and-price to solve the path-based and tree-based models. Our empirical results suggest that the edge-based model is superior in solving small and mid-sized problems, whereas the tree-based model performs better for large problems.

In the second part of the dissertation, we study the flight conflict resolving problem (FCR). The purpose of flight conflict re-scheduling problem is to provide a flight schedule that minimizes the total penalty cost of schedule changes, while maintaining the FAA separation standard between aircrafts. We propose a set-partitioning-based flight sequence model (FSM) that selects an optimal set of flight sequences to minimize the total penalty cost. We also extend the FSM to consider equity among airlines because such corporate decision making (CDM)-feature is necessary and critical for the future aviation systems. The computation results show the proposed solution methods outperform other solution methods, and solve the real life test cases optimally within reasonable time

In the third part of the dissertation, the aircraft maintenance routing problem is studied. The aircraft maintenance routing problem is aimed at scheduling the aircraft rotations so that adequate maintenance opportunities are provided to every aircraft in the fleet. In this dissertation, we present two new compact rotation-tour network representations for the daily aircraft maintenance routing problem (AMR) and the weekly aircraft maintenance routing problem (WAMR), and propose new mixed-integer linear programming formulations to solve these two problems. The computational study suggests the proposed models are able to solve large real-life test instances optimally in reasonable time.

Acknowledgements

This dissertation represents the final chapter of my study at Rutgers. In my five years journey at Rutgers, I have had many people lead me, support me, and encourage me. I would like to express my sincerest gratitude to all my teachers, friends, and family.

First and foremost, I would like to thank my thesis advisor Dr. Art Chaovalitwongse for his intellectual support and patient guidance. I was very fortunate to work with Dr. Art who provided me with many opportunities in my research. He has spent countless hours advising me, teaching me how to formulate our ideas, guiding me to write scientific papers, encouraging me to participate in competitions, and pushing for excellence in my work. My gratitude toward him is beyond words.

I would like to thank Dr. Elsayed for his advice and guidance. I learned in many ways from Dr. Elsayed — from reliability to scheduling, from exploring new research projects to writing good scientific papers. I thank Dr. Albin for numerous occasions where she gave me thoughtful support and encouragement, and helped me to walk through the hardest time during my Phd study. I would like to thank the members of my thesis committee, Dr. Pham and Dr. Resende for their help and advice.

Special thanks also to my friends at IE department: Yaju Fan, Yada Zhu, Joe Chou, Shouyi Wang, and Yaping Wang, for the helps and discussions on the course work, qualify exams, and researches. I will cherish the wonderful moments we shared. Thanks to my friends outside the IE department: Fuguo Jiang, Kaiyuan He, Yu Han, Peng Yang, and Hui Li. They made my life in New Jersey full of fun.

My parents never had an opportunity to attend a college, but they provided me the best education in world: love. Without their unconditional love and support, this thesis would not have been possible. Words are never enough to express my love to them.

Finally, I would like to thank the most important person in my life, my wife, Tian Sun. She was the first girl I met in U.S. and became my soul mate soon after. She was always

the first audience of my presentations and the most loyal reader of my publications. She was always there to cheer me up and encourage me through all of the ups and downs. She made me a better man and a happy father. I could never thank her enough.

Dedication

To my parents, my wife, Tian, and my son, Daniel

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In operations research, when a problem is too large and/or complex to be solved at once, it might be decomposed into smaller and easier problems. Choosing a sound decomposition method is critical to resolving difficult problems successfully. Extensive studies have been made on solving numerous real-life problems using decomposition methods, which requires considerably related knowledge and analysis because decomposition techniques are highly problem-dependent. In this dissertation, we focus on solving three diverse problems that arise in telecommunication and airline logistics using decomposition methods.

In the first part of the dissertation, we study a redundant multicast routing problem with risk group diverse constraints (RMRGD). RMRGD arises in many network applications such as communication systems, power supply distribution systems, transportation networks, etc. In these applications, there are common needs to transmit or to deliver specific information or objects from a single source to a set of destinations [Paul and Raghavan, 2002, Garey and Johnson, 1979, Khoury and Pardalos, 1996]. The edges used in multicast transmission form a multicast tree. In order to provide reliable and resilient multicast services, a common practice is to find two redundant multicast trees from separated sources. The redundant multicast tree has to be disjoint from the original multicast tree so that a single edge failure does not disable the multicast service to any destination [Medard et al., 1999, Irava and Hauser, 2005]. In the real life, it is ordinary that some edges are subject to a common risk, and form a risk group [Yuan and Jue, 2005, Hu, 2003, Zang et al., 2003, Shen et al., 2005, Guo et al., 2005]. For example, in communication systems, all fibers in a conduit form a risk group because a conduit break may disable all the optical fibers contained. In the power distribution network, all power lines located in a geographic area form a risk group because these power lines are subject to common risks

like nature disaster (e.g., earthquakes, floods, etc.) or terrorism attack. It is important to note that an ordinal edge/link disjoint redundant routing is not necessarily reliable in these cases, because both paths to a destination could fail together in a risk group failure. Therefore, finding redundant multicast trees with group diverse constraints becomes very critical for reliable multicast services. In this dissertation, we propose three classes of models, an edge-based, a path-based, and a tree-based model, to solve RMRGD. The path-based and tree-based models can be viewed as different Dantzig-Wolfe decompositions of the edge-based model. Therefore, column generation and branch-and-price methods are used to solve the path-based and tree-based models. The computational results show that the edge-based model provides the best results for small and mid size problems, whereas the tree-based model outperforms the other two for large problems.

In the second part of the dissertation, we study the flight conflict re-scheduling problem, which is to provide a flight schedule that minimizes the total penalty cost of schedule changes while maintaining the FAA separation standard between aircrafts. We propose two optimization models for this problem. The first model is a basic absolute value model (BAVM) that explicitly presents the penalty cost as a nonlinear function. This model is extremely difficult to be solved because the linearized model is an integer programming problem with a large number of Big-Ms, and there is no obvious way, if possible, to decompose it. Then, we reformulate the problem as a set-partitioning-based flight sequence model (FSM) that selects an optimal set of flight sequences that minimizes the total penalty cost. Because there are an exponential number of flight sequences, we propose a column generation framework with a bilinear pricing subproblem to solve the linear relaxation of FSM, and use a branch-and-price method with a new branch-on flight-assignment rule to find the integer optimal solution. Both models are tested on ten simulated test instances randomly constructed based on a real dataset, and compared with two other heuristic methods currently employed at the air route traffic control centers (ARTCCs). The results show that the FSM outperforms all other methods in all test instances. We also extend the FSM to consider equity among airlines. Although the equity concept has not been incorporated in the current ARTCC operations, such corporate decision making (CDM)-feature is necessary and critical for the future aviation systems such as 4-D trajectory

system. Our study demonstrates that the proposed solution method can be extended to handle the equity constraints easily. The computation results show the proposed solution methods can solve the FSM with equity constraints within reasonable time.

In the third part of the dissertation, we study the aircraft maintenance routing problem. The aircraft maintenance problem is one of the important logistic problems in the airline industry. It is aimed at scheduling the aircrafts routing so that adequate maintenance opportunities are provided to every aircraft in the fleet. In this dissertation, we present two compact network representations for the daily aircraft maintenance routing problem (AMR) and the weekly aircraft maintenance routing problem (WAMR), and propose new mixed-integer linear programming formulations to solve these two problems. The quality of these models are assessed on real life test instances from major US carriers. The computational results show that the proposed models are able to obtain the optimal solutions to all test instances in reasonable time. This study suggests that these two models can be applied to integrated problems of the aircraft maintenance routing problem and other planning problems such as the fleet assignment problem and the crew pairing problem.

This dissertation is organized as follows:

In Chapter 2, a brief review of popular decomposition techniques for linear and integer programming problems is presented, and the relationships between different decomposition methods are revealed. In Chapter 3, we study a redundant multicast routing problem with risk group diverse constraints. In Chapter 4, we present the flight conflict re-scheduling problem and the solution methodology. In Chapter 5, we discuss the daily aircraft maintenance routing problem, and propose a new model to solve the problem. In Chapter 6, we present a compact model and the solution methodologies to solve the weekly aircraft maintenance routing problem.

Chapter 2

Preliminaries

In 1960, two American mathematicians, George Dantzig and Philip Wolfe, developed a decomposition algorithm for solving linear programming problems with special structures, and now this algorithm is known as Dantzig-Wolfe decomposition Dantzig and Philip [1960]. One year later, Gilmore and Gomory proposed the use of column generation to solve the cutting stock problem Gilmore and Gomory [1961]. From then on, a number of decomposition methods and the relevant theories have been developed to solve linear and integer programming problems Jr. [1961], Benders [1962], Walker [1969], Geoffrion [1974], Barnhart et al. [1998a]. In the last three decades, with the evolution of computational capability, these decomposition methods have been successfully applied to numerous large-scale practical problems. The aim of this chapter is to provide a brief overview on several well-known decomposition methods for linear and integer programming problems, which include column generation, Dantzig-Wolfe decomposition, Lagrangian relaxation, and branch-and-price.

2.1 Column Generation

The simplex method Bazaraa et al. [1990] is one of the most successful methods in solving linear programming problems. In every iteration of the simplex method, we look for a non-basic variable to price out and enter the basis. Consider the following linear problem.

$$\min \sum_{j \in J} c_j x_j \tag{2.1}$$

$$\text{s.t.} \sum_{j \in J} a_{ij} x_j \geq b \qquad \forall i \in I, \tag{2.2}$$

$$x_j \geq 0 \qquad \forall j \in \bar{J}. \tag{2.3}$$

Given non-negative dual variables $\phi_i, \forall i \in I$, we want to find a x_j with the minimum reduced cost $\bar{c}_j^*$, that is, $j = \arg\min_{j \in J} \bar{c}_j$, where $\bar{c}_j = c_j - \sum_{i \in I} \phi_i a_{ij}$. In the simplex method, this is accomplished by computing the reduced cost $\bar{c}_j$ for all $j \in J$ and then selecting the most negative one. When the number of variables is very large, computing the reduced cost for every j becomes time consuming, because it involves computing the inverse of the non-basic matrix. For some large problems, it is even impractical to enumerate all the variables explicitly.

The basic principle of column generation Wolsey [1998], Desaulniers et al. [2005] is similar with the simplex method. That is, in every iteration of column generation, we select profitable variables to enter the basis. However, there are two main differences between the simplex method and column generation. Firstly, instead of working with the complete set of variables, column generation only deals with a reasonably small subset of variables, and this partial problem is called the *restricted master problem*. Secondly, the pricing step of column generation is more efficient than the simplex method for large problems. Rather than computing the reduced cost $\bar{c}_j$ for all $j \in J$, the pricing operation of column generation is to find a column with the most negative reduced cost by solving an optimization problem, called the *pricing subproblem*, as follows:

$$\bar{c}_j^* = \min c_j - \sum_{i \in I} \phi_i a_{ij} \tag{2.4}$$

s.t.

$$\{a_{1j}, a_{2j}, \ldots, a_{|I|j}\} \text{ is a feasible variable index vector.} \qquad \forall j \in J. \tag{2.5}$$

If there exists some algorithm to solve the pricing subproblem so quickly that the solution time of the pricing subproblem is less than the time of computing the reduced costs for all non-basic variables, column generation becomes more computationally efficient than the simplex method. Furthermore, because column generation only deals with a subset of variables, it need much less computational space than the simplex method. The detailed column generation algorithm is shown as follows:

Initialization We first select a subset of $\bar{J} \subset J$, which provides a feasible restricted

master problem as shown in Eqs. (2.6)-(2.8).

$$\min \sum_{j \in \bar{J}} c_j x_j \tag{2.6}$$

$$\text{s.t.} \sum_{j \in \bar{J}} a_{ij} x_j \geq b \qquad \forall i \in I, \tag{2.7}$$

$$x_j \geq 0 \qquad \forall j \in \bar{J}. \tag{2.8}$$

Solving Restricted Master Problem After solving the restricted master problem, we get the dual variables $\phi_i,\ \forall i \in I$.

Optimality Check & Generation of New Variables In order to check optimality of the current solution, we solve the pricing subproblem in Eqs. (2.4)-(2.5). If the objective value of the pricing subproblem is less than 0, the corresponding variable is added into the restricted master problem, and we continue the next step; otherwise, the current LP solution is optimal, and the algorithm stops.

Computing Lower and Upper Bounds Note any solution to the restricted master problem is feasible to the original problem. Let $\bar{z}$ denote the optimal objective value of the restricted master problem in an iteration. It is obvious that $\bar{z}$ is an upper bound of the problem. If there exists an upper bound for the sum of all the variables, that is $\mu \geq \sum_{j \in J} x_j$, $\bar{z} - \mu \bar{c}^*$ is a lower bound to the original problem. In other words, the optimal value z^* of the original problem satisfies the following relation:

$$\bar{z} - \mu \bar{c}^* \leq z^* \leq \bar{z}. \tag{2.9}$$

This is because we cannot reduce $\bar{z}$ by more than μ times of the smallest reduced cost $\bar{c}^*$.

Stopping Criteria After computing and updating the lower bound of the problem, we stop the algorithm if the stopping criteria is met; otherwise, we go to step 2, and the algorithm continues.

The key step of column generation is how to model the pricing subproblem and solve

it efficiently. As we mentioned before, the modeling of the subproblem takes considerable experience and analysis. In Table 2.1, we list a set of well-known problems that have been solved successfully using column generation. We also list the corresponding subproblems.

Problems	**Column Generation Subproblems**
Bin Packing Problem Scholl et al. [1997], de Carvalho [1999, 2002]	Knapsack Problem
Crew Pairing Problem Barnhart et al. [1998a, 2003]	Constrained Shortest Path Problem
Cutting Stock Problem Gilmore and Gomory [1961], Marcotte [1985], Vance [1998] Vanderback [1999], Degraeve and Schrage [1999], Degraeve and Peeters [2003]	Knapsack Problem
Job Shop Problem Chen and Powell [1999a,b, 2003], Gelinas and Soumis [2005]	Single Machine Sequencing Problem with Time Window
Multicommodity Network Flow Problem Ahuja et al. [1993], Barnhart et al. [2000]	Shortest Path Problem
Multi-Item Lot-Sizing Problems Eppen and Martin [1987]	Single-Item Problem
Vehicle Routing Problem Desrochers et al. [1992], Kohl et al. [1999] Chabrier [2006], Irnich and Villeneuve [2006]	Traveling Sales Man Problem

Table 2.1: Selected well-known problems solved using column generation.

Some very important techniques may speed up the implementation of column generation greatly. For example, instead of searching for the variable with the most negative reduced cost, any variables with negative reduced cost can improve the current solution of the restricted master problem. Hence, many heuristics can be utilized to achieve this goal. Applying this method often speeds up the early stages of column generations. Another widely used approach is to return multiple columns to the restricted master problem in every iteration. This normally decreases the number of iterations needed for optimal solutions.

We have only briefly introduced the idea of column generation in this section. Much more about theories, implementation issues, and applications on column generation can be found in the literatures. For the theories of column generation, please see Chapter 11 of Wolsey [1998], Chapter 1 of Desaulniers et al. [2005], Chapter 7 of Bazaraa et al. [1990]; for the implementation issues of column generation, please see Chapter 12 of Desaulniers et al. [2005], Vanderback [1999]; and there are numerous applications of column generation in many fields of operations research, industrial engineering, and management science.

2.2 Dantzig Wolfe Decomposition

In this section, we present Dantzig Wolfe Decomposition for integer problems. Consider a mixed integer problem of the following form:

$$\min \mathbf{c}(x) \tag{2.10}$$

$$\text{s.t. } \mathbf{A}(x) \geq \mathbf{b}, \tag{2.11}$$

$$\mathbf{D}(x) \geq \mathbf{d}, \tag{2.12}$$

$$x \in Z^+, \tag{2.13}$$

where $\mathbf{A} \in \mathbf{Q}^{l\times(n)}$, $\mathbf{D} \in \mathbf{Q}^{m\times(n)}$, are rational matrices and $\mathbf{c} \in \mathbf{Q}^n$, $\mathbf{b} \in \mathbf{Q}^l$, and $\mathbf{d} \in \mathbf{Q}^m$ are rational vectors. Assume Eq. (2.11) represents difficult constraints, and Eq. (2.12) represents more tractable constraints that it alone can be solved efficiently. For example, $\mathbf{B}(x) \geq \mathbf{b}$ has a block diagonal structure or is totally unimodular Wolsey [1998], Schrijver [1998], Nemhauser and Wolsey [1999], which can be easily solved. Dantzig-Wolfe decomposition takes the advantage of such structure and solves the problem in Eqs. (2.10)-(2.13) efficiently.

Dantzig-Wolfe decomposition uses the concept of variable redefinition. If the convex set $conv(X^D) = \{(x) : \mathbf{D}(x) \geq \mathbf{d}, x \in Z^+\}$ can be represented by a combination of extreme points $\{x_p\}_{p\in P}$ and a non-negative combination of extreme rays $\{x_r\}_{r\in R}$, each x can be rewritten in the following form:

$$x = \sum_{p\in P} x_p\lambda_p + \sum_{r\in R} x_r\lambda_r, \tag{2.14}$$

$$\sum_{p\in P} \lambda_p = 1, \tag{2.15}$$

where the index set P and R are finite. Also, by applying linear transformations, we have $c_p = \mathbf{c}^T\mathbf{x}_p$, $\mathbf{a}_p = A\mathbf{x}_p$ and $c_r = \mathbf{c}^T\mathbf{x}_r$, $\mathbf{a}_r = A\mathbf{x}_r$. Substituting x in Eqs. (2.10)-(2.13), we

have the following form:

$$\min \sum_{p\in P} c_p\lambda_p + \sum_{r\in R} c_r\lambda_r \tag{2.16}$$

$$\text{s.t.} \sum_{p\in P} a_p\lambda_p + \sum_{r\in R} a_r\lambda_r \geq \mathbf{b}, \tag{2.17}$$

$$\sum_{p\in P} \lambda_p = 1, \tag{2.18}$$

$$\lambda_p \geq 0, \tag{2.19}$$

$$\sum_{p\in P} x_p\lambda_p + \sum_{r\in R} x_r\lambda_r = x, \tag{2.20}$$

$$x \in Z^+. \tag{2.21}$$

Eq. (2.18) is the convexity constraint. If the integrality constraints of x are relaxed, we can ignore the constraints in Eqs. (2.20)-(2.21). Since $|P|$, the number of extreme points of $conv(X^D)$, is usually very large, it is impractical to enumerate all the extreme points explicitly. A more efficient way is to work with a reasonably small subset of extreme points first, and then solve the entire problem using column generation. Define the dual variable vector of constraints in Eq. (2.17) as ϕ, and the dual variable of constraints in Eq. (2.18) as ϕ_0. The pricing subproblem of the problem becomes as follows:

$$\bar{c}^* = \min\{(\mathbf{c}^T - \phi^T A)\mathbf{x} - \phi_0 | x \in X^D\}. \tag{2.22}$$

When matrix $\mathbf{D}$ has a block diagonal structure as follows:

$$\mathbf{D} = \begin{pmatrix} D^1 & & & \\ & D^2 & & \\ & & \cdots & \\ & & & D^k \end{pmatrix}, \qquad \mathbf{d} = \begin{pmatrix} d^1 \\ d^2 \\ \cdots \\ d^k \end{pmatrix}, \tag{2.23}$$

Dantzig-Wolfe decomposition yields k subproblems, each with its own constraints and associated dual variables. The reduced cost of each subproblem is computed as follows:

$$\bar{c}^{k*} = \min\{(\mathbf{c}^{kT} - \phi^{kT} A^k)\mathbf{x}^k - \phi_0^k | x \in X^k\}, \qquad \forall k \in K. \tag{2.24}$$

From Eq. (3.44) together with the convexity constraint in Eq. (2.18), we have the upper and lower bounds in every iteration of column generation for Dantzig-Wolfe decomposition as follows:

$$\bar{z} - \bar{c}^* = \bar{z} - \sum_{k \in K} \bar{c}^{k*} \leq z^* \leq \bar{z}. \tag{2.25}$$

It is obvious that Dantzig-Wolfe decomposition is closely related with column generation. There are two main advantages of Dantzig-Wolfe decomposition over the original model: first, it often leads to a stronger LP bound than the original formulation Geoffrion [1974]; second, we may decompose the original complex problem into a set of well-studied easy problems and use existing algorithms and methods to solve the subproblem efficiently. Next, we illustrate the above points in the following example.

Example: Resource Constrained Shortest Path

Consider the network $G(N, A)$ shown in Figure 2.1. Each arc $(i, j) \in A$ has two

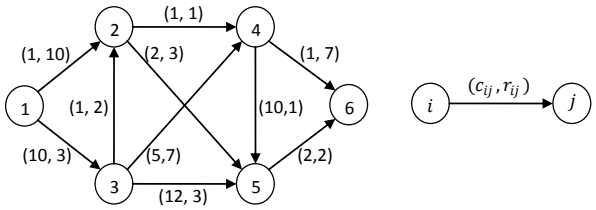


Figure 2.1: Resource constrained shortest path problem Ahuja et al. [1993], Desaulniers et al. [2005].

attributes: the first attribute c_{ij} is the travel cost, and the second attribute r_{ij} is the resource consumed. We want to find a path from node 1 to node 6 with minimum cost and the total resource consumed is less than or equal to 14.

One nature formulation is an edge-based model shown in Eqs. (2.26)-(2.31), in which we define a binary variable x_{ij} such that $x_{ij} = 1$ if edge (i, j) in the shortest path

and 0 otherwise.

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{2.26}$$

$$\text{s.t.} \sum_{j:(1,j)\in A} x_{1j} = 1, \tag{2.27}$$

$$\sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = 0 \qquad \forall i = 2,3,4,5, \tag{2.28}$$

$$\sum_{i:(i,6)\in A} x_{i6} = 1, \tag{2.29}$$

$$\sum_{(i,j)\in A} r_{ij}x_{ij} \leq 14, \tag{2.30}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A. \tag{2.31}$$

The constraints in Eqs. (2.27)-(2.29) are the flow balance constraints. The constraint in Eq. (2.30) is resource limitation constraint. The resource constrained shortest path problem has been proven to be NP-complete Garey and Johnson [1979]. However, the flow balance constraints in Eqs. (2.27)-(2.29) are totally unimodular, and it alone can be solved easily. In network theory, the extreme point of the polytope defined by Eqs. (2.27)-(2.29) corresponds to a path $p \in P$ in the network. Therefore, we can simply substitute x_{ij} by extreme point x_p as follows:

$$x_{ij} = \sum_{p|(i,j)\in p} \lambda_p \qquad \forall (i,j) \in A, \tag{2.32}$$

$$\sum_{p\in P} \lambda_p = 1, \tag{2.33}$$

$$\lambda_p \geq 0 \qquad \forall p \in P. \tag{2.34}$$

Dantzig-Wolfe decomposition of the model in Eqs. (2.26)-(2.31) is shown as follows:

$$\min \sum_{(p\in P)\in A} c_p\lambda_p \tag{2.35}$$

$$\text{s.t.} \sum_{p\in P} \lambda_p = 1, \tag{2.36}$$

$$\sum_{p\in P} r_p\lambda_p \leq 14, \tag{2.37}$$

$$x_{ij} = \sum_{p|(i,j)\in p} \lambda_p \qquad \forall (i,j) \in A, \tag{2.38}$$

$$\lambda_p \geq 0 \qquad \forall p \in P, \tag{2.39}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A. \tag{2.40}$$

Here, $c_p = \sum_{(i,j)\in p} c_{ij}$, and $r_p = \sum_{(i,j)\in p} r_{ij}$. We first solve the LP relaxation of the above problem using column generation. Define the dual variable of constraint in Eq. (2.36) as ϕ_0 and dual variable of constraint in Eq. (2.37) as ϕ_1. The reduced cost of a path is:

$$\bar{c} = \sum_{(i,j)\in p} c_{ij} - \phi_1 \sum_{(i,j)\in p} r_{i,j} - \phi_0 \qquad \forall p \in P. \tag{2.41}$$

The pricing subproblem is then formulated as:

$$\bar{c}^* = \min \sum_{(i,j)\in A} (c_{ij} - \phi_1 r_{ij})x_{ij} - \phi_0 \tag{2.42}$$

$$\text{s.t.} \sum_{j:(1,j)\in A} x_{1j} = 1, \tag{2.43}$$

$$\sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = 0 \qquad \forall i = 2,3,4,5, \tag{2.44}$$

$$\sum_{i:(i,6)\in A} x_{i6} = 1, \tag{2.45}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A. \tag{2.46}$$

This problem is simply the shortest path problem and can be solved in polynomial time Papadimitriou and Steiglitz [1998], Corman et al. [2001].

In Table 2.2, we show the solution procedure using column generation. Specifically,

Iteration	Columns	Objective	ϕ_0	ϕ_1	Reduced Cost	Solution
1	$p_1 = 1 \to 2 \to 4 \to 6$	11.4	40.8	-2.1	-2.8	$\lambda_{p_1} = 0.6, \lambda_{p_2} = 0.4$
	$p_2 = 1 \to 3 \to 5 \to 6$					
2	$p_3 = 1 \to 3 \to 2 \to 5 \to 6$	9.0	30.0	-1.5	-2.5	$\lambda_{p_1} = 0.5, \lambda_{p_3} = 0.5$
3	$p_4 = 1 \to 2 \to 5 \to 6$	7.0	35.0	-2.0	0	$\lambda_{p_3} = 0.2, \lambda_{p_4} = 0.8$

Table 2.2: Column generation solution precedure.

in iteration 1, we generate two initial paths p_1 and p_2 using shortest path algorithm, where $p_1 = 1 \to 2 \to 4 \to 6$ has the least cost, and $p_2 = 1 \to 3 \to 5 \to 6$ uses the least resource. Node that p_2 also guarantees the feasibility of the problem. After solving the restricted master problem, we have dual cost $\phi_0 = 40.8$ and dual cost $\phi_1 = -2.1$. By plugging in dual values into Eqs. (2.42)-(2.46), we generate the third path $1 \to 3 \to 2 \to 5 \to 6$ and column generation continues. As shown in Table 2.2, column generation terminates in three iterations with the objective value of 35. It is worth mentioning that the LP solution contains fractional values $\lambda_{p_3} = 0.2, \lambda_{p_4} = 0.8$, and we need to use branch-and-price described in Section 2.4 to obtain the integer solution.

2.3 Lagrangian Relaxation

When solving the integer problem in Eqs. (2.10)-(2.13), we can also relax the complex constraints in Eq. (2.11), and penalize the term $(\mathbf{A}(x) - \mathbf{b})$ with non-negative cost vector $\mathbf{u}$. Therefore, the original problem becomes:

$$\min \mathbf{c}(x) - \mathbf{u}(\mathbf{A}x - \mathbf{b}) \tag{2.47}$$

$$\text{s.t. } \mathbf{D}(x) \geq \mathbf{d}, \tag{2.48}$$

$$x \in Z^+. \tag{2.49}$$

This problem is called a *Lagrangian relaxation* of the original problem. Note the feasible region of the above formulation is at least as large as the original formulation. Also the objective value of the above formulation is a lower bound to the original problem, because for all $\mathbf{u} \geq 0$ and $\mathbf{A}(x) \geq \mathbf{b}$, $\mathbf{c}(x) - \mathbf{u}(\mathbf{A}(x) - \mathbf{b}) \leq \mathbf{c}(x)$ for all x satisfying constraints in Eqs. (2.11)-(2.12) Bazaraa et al. [1990], Wolsey [1998]. Here, we can view $\mathbf{u}$ as the dual

cost or *Lagrangian multiplier* of the complex constraint in Eq. (2.11). Note the values of $\min \mathbf{c}(x) - \mathbf{u}(\mathbf{A}x - \mathbf{b})$ for all $\mathbf{u} \geq 0$ provide a set of lower bounds. In order to find the largest lower bound for all $\mathbf{u}$, we solve the following *Lagrangian dual problem*.

$$\max_{u \geq 0} \{\min \mathbf{c}(x) - \mathbf{u}(\mathbf{A}x - \mathbf{b})\} \tag{2.50}$$

$$\text{s.t. } \mathbf{D}(x) \geq \mathbf{d}, \tag{2.51}$$

$$x \in Z^{+}. \tag{2.52}$$

It is known that this function of $\mathbf{u}$ is concave and piecewise linear Nemhauser and Wolsey [1999], Wolsey [1998], Bazaraa et al. [1990]. A subgradient optimization method can be used to obtain the optimal Lagrangian multiplier Wolsey [1998], Nemhauser and Wolsey [1999]. Specifically, the optimal value of u is computed using an iterative calculation as follows:

$$u_{k+1} = \max \{u_k + \theta_k(Ax - b), 0\}, \tag{2.53}$$

where u_k is the Lagrangian dual at iteration k, and θ_k is the step length of iteration k. In order to guarantee the convergence of the u_k, the step length of iteration k has to satisfy the following condition: $\lim_{k\to\infty} \sum_k \theta_k = \infty$ and $\lim_{k\to\infty} \theta_k = 0$. It is important to note that the step length θ_k is critical in implementing the subgradient algorithm, and different methodologies have been developed in determining effective step length θ_k for different applications.

Next we show the relationship between Dantzig-Wolfe decomposition and Lagrangian relaxation method. If we replace x in Eqs. (2.50)-(2.52) by extreme points x_p and extreme ray x_r of $conv(X^D)$, we have the following:

$$\max_{u \geq 0} \left\{ \min_{p \in P} \mathbf{c}^T(\mathbf{x}_p) - \mathbf{u}^T(\mathbf{A}\mathbf{x}_p - \mathbf{b}) \right\} \tag{2.54}$$

$$\text{s.t. } (\mathbf{c}^T - \mathbf{u}^T\mathbf{A}x_p)\mathbf{x}_r \geq 0 \qquad \forall r \in R. \tag{2.55}$$

Here, the constraints in Eq. (2.55) ensure that the problem is bound. We can rewritten

the Eqs. (2.54)-(2.55) in a linear problem form as shown below:

$$\max u_0 \tag{2.56}$$

$$\text{s.t. } \mathbf{u}^T(\mathbf{A}\mathbf{x}_p - \mathbf{b}) + u_0 \leq \mathbf{c}^T\mathbf{x}_p \qquad \forall p \in P, \tag{2.57}$$

$$(\mathbf{c}^T - \mathbf{u}^T\mathbf{A}x_p)\mathbf{x}_r \geq 0 \qquad \forall r \in R, \tag{2.58}$$

The dual of the above problem is shown as follows:

$$\min \sum_{p\in P} c_p x_p \lambda_p + \sum_{r\in R} c_r x_r \lambda_r \tag{2.59}$$

$$\text{s.t. } \sum_{p\in P} Ax_p\lambda_p + \sum_{r\in R} Ax_r\lambda_r \geq \mathbf{b}\sum_{p\in P}\lambda_p, \tag{2.60}$$

$$\sum_{p\in P}\lambda_p = 1, \tag{2.61}$$

$$\lambda_p \geq 0. \tag{2.62}$$

This is the same as Dantzig-Wolfe decomposition of the problem as shown in Eqs. (2.16)-(2.18). Also, for a given vector $\mathbf{u}$ of Lagrangian multiplier and a constant u_0, we have:

$$\bar{z} + \bar{c}^* = c^T x - (\phi^T(Ax - b)) = (\mathbf{u}^t b + u_0) + \min_{x\in conv(X^D)} (c^T - \mathbf{u}^T A)x - u_0 \tag{2.63}$$

This illustrates that the lower bound provided by Dantzig-Wolfe decomposition in Eq. (3.44) is the same as the bound provided by Lagrangian relaxation.

There are several advantages of Lagrangian relaxation. First, same as Dantzig-Wolfe decomposition, it allows us to decompose the complex problem into some well-studied problems, and solve them efficiently. Second, it is usually easier to solve the Lagrangian relaxation using subgradient optimization than Dantzig-Wolfe decomposition with linear or integer subproblems. In the following example, we show how to use the Lagrangian relaxation to solve the constrained shortest path problem mentioned in Section 2.2.

Example: Resource Constrained Shortest Path Problem (Continuation)

In the resource constrained shortest path, the constraint in Eq. (2.30) is hard constraint, and Lagrangian relaxation of the problem is as follows:

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} + u(\sum_{(i,j)\in A} r_{ij} - 14) \tag{2.64}$$

$$\text{s.t.} \sum_{j:(1,j)\in A} x_{1j} = 1, \tag{2.65}$$

$$\sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = 0 \qquad \forall i = 2,3,4,5, \tag{2.66}$$

$$\sum_{i:(i,6)\in A} x_{i6} = 1, \tag{2.67}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A. \tag{2.68}$$

Let $u_1 = 0$ and $\theta_k = \frac{1}{k}$. In iteration 1, we have the first path $1 \to 2 \to 4 \to 6$, and $u_1 = 4$. In iteration 2, we generate the second path $1 \to 3 \to 2 \to 5 \to 6$. In iteration 3, we generate the third path $1 \to 2 \to 5 \to 6$. As the number of iteration increases, the Lagrangian multiplier u converges to 2, and the generated paths change between $1 \to 3 \to 2 \to 5 \to 6$ and $1 \to 2 \to 5 \to 6$. This is also verified by the dual cost of constraints in Eq. (2.37) of Dantzig-Wolfe decomposition solution.

For detailed information about Lagrangian relaxation, please see Chapter 10 of Wolsey [1998], Chapter II.3 of Nemhauser and Wolsey [1999], Chapter 16 of Ahuja et al. [1993], and Chapter 24 of Schrijver [1998].

2.4 Branch-and-Price

Column generation is an efficient framework to solve the large-scale linear programs. However, when the master problem contains integer variables, column generation does not guarantee the integrality of the solution. In order to solve the large-scale integer problems, a method, namely branch-and-price, is designed to combine column generation and branch-and-bound Barnhart et al. [1998a], Desaulniers et al. [2005]. In this subsection, we discuss the branch-and-price algorithm for large-scale integer problems.

First, we review the branch-and-bound method for the integer problem briefly. The

basic principle of branch-and-bound is divide-and-conquer. The branch in the branch-and-bound is to divide the original problem into smaller problems; and the bound refers to lower and upper bounds that can be used to speed up the solution procedure and to prove the optimality of the solution.

When we solve the LP relaxation of an integer problem in Eqs. (2.69)-(2.71), we may obtain a solution $\mathbf{x}^0$.

$$\min \sum_{j \in J} c_j x_j \tag{2.69}$$

$$\text{s.t.} \sum_{j \in J} a_{ij} x_j \geq b \qquad \forall i \in I, \tag{2.70}$$

$$x_j \in Z^+ \qquad \forall j \in J. \tag{2.71}$$

If all the elements in $\mathbf{x}^0$ are integers, $\mathbf{x}^0$ is an optimal solution to the integer problem. Otherwise, we split the problem into two subproblems by adding two mutually exclusive constraints. For example, if $x^0_{j_1}$ of $\mathbf{x}^0$ is fractional, the two subproblems are

$$\min \sum_{j \in J} c_j x_j \tag{2.72}$$

$$\text{s.t.} \sum_{j \in J} a_{ij} x_j \geq b \qquad \forall i \in I, \tag{2.73}$$

$$x_j \in Z^+ \qquad \forall j \in J, \tag{2.74}$$

$$x_{j_1} \geq \lceil x^0_{j_1} \rceil. \tag{2.75}$$

and

$$\min \sum_{j \in J} c_j x_j \tag{2.76}$$

$$\text{s.t.} \sum_{j \in J} a_{ij} x_j \geq b \qquad \forall i \in I, \tag{2.77}$$

$$x_j \in Z^+ \qquad \forall j \in J, \tag{2.78}$$

$$x_{j_1} \leq \lfloor x^0_{j_1} \rfloor. \tag{2.79}$$

The solution to the original problem has to be in one of the above two subproblems. Therefore, we solve two subproblems and recheck the solutions. If the solution of a subproblem

is integral, it is an upper bound to the original problem, because the optimal integer solution has to be as good as the current integer solution; otherwise, we split the subproblem again into smaller subproblems. If the original problem is bounded, we are able to find optimal integer solutions by repeating this procedure in finite number of steps. We can view this process as a tree shown in Figure 2.2. The root node of the tree represents the LP

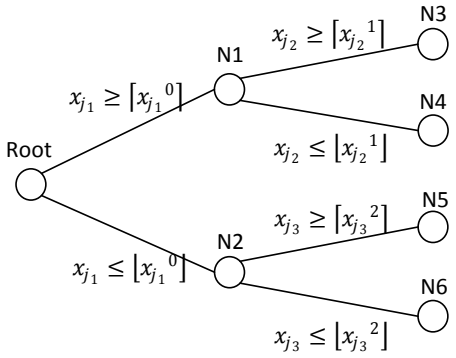


Figure 2.2: Branch-and-bound tree for integer problem.

relaxation of the original integer problem. Each node of the tree represents a subproblem.

When we search the branch-and-bound tree, we can always keep track of a global upper bound and a global lower bound. The global lower bound is the objective value of the root node, and the global upper bound is the best integer solution found so far. During the search of the branch-and-bound tree, there are four possible outcomes from a branch-and-bound node, which are listed below:

1. If the LP solution at the current node is greater or equal to the lower bound and smaller than the upper bound, and it is fractional, we need to split this node into subproblems and solve LP relaxation of these subproblems;

2. If the LP solution at the current node is greater than the lower bound and smaller than the upper bound, and it is integral, we need to update the current upper bound;

3. If the LP solution at the current node is equal to the lower bound, and it is integral, we find the optimal integer solution of the original problem;

4. If the LP solution at the current node is greater or equal to the upper bound, or it is infeasible, we stop search from this node because it is impossible for the remaining nodes to provide a better IP solution. This process is called pruning.

There are several implementation issues when implement branch-and-bound algorithm, which are listed below:

Branching Rules Selecting a right variable to branch may effect the efficiency of the branch-and-bound drastically. Computational experience shows that a balanced tree normally enables efficient pruning process. In order to obtain a balanced tree, one common choice is to start with the most fractional variable. For example, for a binary integer problem, we want to choose the variable x_j such that $j = \arg\max_{j \in J} x_j^0, 1 - x_j^0$.

Searching Procedure A depth-first search is usually used to obtain a feasible integer solution quickly. This integer solution provides an upper bound and makes the pruning possible. Also, the space requirement of depth-first search is not much comparing to other searching procedures such as breadth-first search. Other searching procedures may also be used, such as best-bound-first search.

In branch-and-price, we combine the well-known idea of column generation and branch-and-bound. When the number of variables is too large to be enumerated, we use column generation to obtain the LP relaxation at each node of the branch-and-bound tree. However, there are several difficulties when combining column generation and branch-and-bound. First, when we add constraints in Eqs. (2.74) and (2.78) into the subproblem, the structure of column generation pricing problem may be affected. This could bring difficulties when applying the existing algorithms to subproblems. Second, solving the LP to optimality at every node of the branch-and-bound tree may not be computationally efficient. Third, a traditional branching strategy for integer problems may not be efficient because the branch-and-bound tree may be highly unbalanced. In general, there is no universal branch-and-price procedure and strategy to avoid these difficulties. Instead, we need to study and analysis individual application and formulation to obtain an efficient and effect branch-and-price rule. However, some previous studies may provide us some insights and experience. For example, one of the most widely used branching strategies is branch-on-follow-on, which is developed for branch-and-price to solve crew scheduling problem Ryan and Foster [1981]. The branch-on-follow-on strategy normally provides a

more balanced branch-and-bound tree, and has been used in a large number of branch-and-price applications. In one of the pioneer papers on branch-and-price Barnhart et al. [1998a], two classes of models, a general model and a set partitioning model, are solved using branch-and-price. In Danna and Pape [2005], a local search heuristics is used together with branch-and-price to produce good solution efficiently.

Chapter 3

Redundant Multicast Routing Problem with Risk Group Diverse Constraints

This chapter presents a redundant multicast routing problem in multilayer networks that arises from large-scale distribution of realtime multicast data (e.g., Internet TV, video-casting, online games, stock quotes). Since these multicast services commonly operate in multilayer networks, the communications paths need to be robust against a single router or link failure as well as multiple such failures due to shared risk link groups (SRLGs). The objective of this problem is, therefore, to find two redundant multicast trees, each from one of the two redundant sources to every destination, at a minimum total communication cost whereas two paths from the two sources to every destination are guaranteed to be SRLG-diverse (i.e., links in the same risk group are disjoint). In this chapter, we present three mathematical programming models, a edge-based, a path-based, and a tree-based model, for the redundant multicast routing problem with risk group diverse constraints. Decomposition methods are used to solve the path-based and tree-based models. This study is motivated by emerging applications of internet-protocol TV service, and we evaluate the proposed approaches using real life network topologies. Our empirical results suggest that the edge-based model performs well on small and mid size test instances, and the tree-based model outperforms the other two models on large size test instances.

3.1 Introduction

Multicast networking has been widely used in many applications in communications systems, including multi-location video conferencing, multimedia broadcasting, wireless/mobile multicast in civil and military applications, web caching, software distribution, video-on-demand, and virtual reality simulation Oliveira and Pardalos [2005a], Oliveira et al. [2005].

Generally, a multicast network consists of a set of nodes, where some specific data of common interest is transmitted (or broadcasted) from a single source to a set of destinations, called *multicast group*. This feature provides the main difference between multicast networks and unicast networks, where data are sent between pairs of a source node and a destination node. Multicast transmission delivers the information to each node in a multicast group simultaneously over a network, which forms a *multicast tree*. The problem of finding a multicast tree is often called a multicast routing problem. Although there are several multicast routing protocols in the literature including open shortest path first (OSPF) Buriol et al. [2007] and intermediate system to intermediate system (IS-IS) routing Buriol et al. [2005], the multicast routing problem is still a hard combinatorial optimization problem Oliveira and Pardalos [2005b], Oliveira et al. [2006], Resende and Pardalos [2006].

Recently, a very practical concern about reliability and resiliency issues in provisioning a multicast transmission has been raised Oliveira et al. [2007]. As multicast networks should be protected from and be resilient to network outages or attacks, network operators normally utilize some communication links less than their limit capacity in order to increase the reliability and employ some fast restoration scheme against failures Levine and Garcia-Luna-Aceves [1998]. In an extreme case where the transmission needs to be always-on, network operators often employ a path protection scheme, where network routing finds a backup or redundant path that is disjoint from each working or primary path. Having redundant or multiple multicast trees that are disjoint will protect the multicast transmission from network failures or attacks. The problem of identifying disjoint redundant multicast trees from a single source to a set of destinations that can survive any single link failure was well studied in the literature Medard et al. [1999]. A protected multicast transmission can also be applied to an existing network infrastructure to maximize the coverage/yeild and minimize the cost and traffic load Chen et al. [2009], Kang et al. [2009].

Today's multicast applications are commonly operated on multilayer telecommunication networks but previous studies did not address the concept of *Shared Risk Resource Group* (SRRG) that arises due to a common structure of multilayer networks. SRRG is a

set of shared resources that always fail simultaneously. Such resources are often referred to a set of nodes or a set of links. For this reason, SRRG plays a very important role in diverse routing in multilayer networks because a single SRRG failure causes multiple link and node failure Datta and Somani [2004]. There are two types of SRRG: Shared Risk Link Group (SRLG) and Shared Risk Node Group (SRNG). Both SRNG and SRLG are special cases of SRRG, where the network contains only a group of nodes or links that are subject to a common risk. Multiple nodes in an SRNG share common resources like channel, power source, or connector (router), whose failure disrupts all nodes in the group. Multiple links in an SRLG share common resources, whose failure disrupts all links in the group, e.g., fiber cuts. In a mathematical modeling sense, the problems of finding redundant multicast trees with SRLG-diverse, SRRG-diverse, or SRNG-diverse constraints are equivalent.

Our study is motivated by emerging applications of Internet Protocol TV (IPTV), which deal with finding protected (two or more) redundant multicast trees in multilayer networks. Nowadays several major telecommunications providers around the world are exploring the employment of IPTV as a new revenue opportunity, considering the fact that broadband networks are becoming more and more popular all over the world, especially in Asia and Europe Wikipedia [2008]. The results of this study may be used to improve the efficiency and reduce the cost of IPTV employment and provisioning. Specifically, in this chapter we consider the problem of IPTV multicast routing from *two redundant multicast sources over a multilayer network*. Our objective is to *(a)* find two redundant multicast trees, each from one of the two sources to every destination, at a minimum total communication cost, and *(b)* guarantee that two paths from the two sources to every destination are SRLG-diverse (i.e., links in the same risk group are disjoint). We call this problem the redundant multicast routing problem with group diverse constraints (RMRGD), which is a generalization of the single-destination SRLG disjoint multicasing problem, which has been studied in Shen et al. [2005], Yuan and Jue [2005], Guo et al. [2005]. To the best of our knowledge, there are a few studies in the literature in this area, and most of them only propose simple heuristic approaches without giving rigorous mathematical modeling of protected multicast routing with the consideration of SRRG. In

addition, conventional path protection algorithms that are well studied in the literature (e.g., node/edge disjoint path Peleg and Upfal [1989], Robertson and Seymour [1990], Schrijver [1990]) cannot be directly applied to RMRGD.

In this chapter, we prove the NP-completeness of RMRGD by reducing the well-known NP-complete 3-SAT problem to it. Previously, NP-completeness has been shown for related problems based on a reduction of the maximum bipartite subgraph problem Hu [2003]. Our proof provides an alternative approach, in which a reduction of the 3-SAT problem is proposed. We present for the first time a new edge-based optimization model for RMRGD, and reformulate it as two different models, a path-based model with variables corresponding to paths from a source to a destination, and a tree-based model with variables corresponding to trees from a mutlicast source. These reformulations arise naturally from the Dantzig-Wolfe decomposition of the edge model. Since there are an exponential number of possible paths and trees for large real networks, we develop a heuristic approach to generate potentially high-quality paths and trees for the path-based model and the tree-based model respectively. We test and compare the performance of the edge-based, path-based, and tree-based model on real world network instances.

The rest of this chapter is organized as follows. Section 3.2 provides background on multicast routing and the concept of SRRG. Section 3.3 investigates the complexity of a special class of this multicast problem. Section 3.4 formally defines the problem, and provides the edge-based model. In Section 3.5, we present the path-based model, and propose three column generation methods to solve the LP relaxation of the problem. We also provide a branch-and-price method to obtain the integer solution of the problem. In Section 3.6, we present a column generation and branch-and-price solution methods for the tree-based model. Section 3.7 presents an comprehensive empirical study using real life test cases and the economic interpretation of the results we achieved by applying the described techniques. Section 3.8 presents a conclusion and discussion about future work.

3.2 Background

In contrast to unicast systems, multicast routing problems are hard combinatorial problems, which usually require the use of sophisticated solution methods such as approximation algorithms, distributed computing, multi-objective optimization, and mathematical programming. The most widely studied multicast routing problem is the minimum cost multicast routing problem Resende and Pardalos [2006], which is to find a multicast tree (a set of links) with a minimum cost from a given source to a set of destination nodes in a network. The problem can be easily seen as a generalization of the Steiner tree problem, which is a well-known NP-hard problem Garey and Johnson [1979], Pardalos and Khoury [1996]. For multicasting in communication networks, additional operational constraints such as capacity, delay, and reliability should be considered and added to the multicast routing research. For example, in most multicast routing problems, operational backbone network owners provision their networks such that the average load on each link and the average end-to-end propagation delay are below a certain threshold. The multicast system considered in this study is motivated by the fast growing Internet Protocol TV (IPTV) application. Since identical broadcast TV content is to be distributed across all video hub offices (VHOs) in IPTV networks, it is intuitive to consider the use of multicast to minimize the communication cost (e.g., bandwidth consumed) Oliveira et al. [2006], Resende and Pardalos [2006].

Operational networks of today's internet applications like IPTV involve physical/data-link layers for optical backbones and network layer for internet protocol (IP) backbones. Extensive research on multicasting and routing in IP networks can be found in Resende and Pardalos [2006], Buriol et al. [2007], Teixeira et al. [2007]. However, multicasting in multilayer networks (both optical and IP) is a relatively new research area and is in its infancy Ellinas et al. [2003], Xin and Rouskas [2004]. In order to make practical use of multilayer operations, research studies in multicasting need to invoke up to the network layer architecture of application-dependent network design. The multicasting routing problem in multilayer networks with delay or reliability constraints is in turn very practical, yet extremely challenging. The reliability constraints that are derived by the multilayer structure of operational networks are very intuitive and viable. The

concept of shared risk resource group (SRRG) has been introduced to represent a set of communication links and/or nodes that are simultaneously affected due to a single point of failure Li et al. [2001]. For instance, fiber links interconnecting network nodes are often routed over common sections of fiber or conduit Guo et al. [2005]. Specifically, in an optical network, a conduit carries a large number of fiber cables, each in turn carrying multiple channels (links). As a result, two diverse connections in the IP layer are not necessarily diverse in the physical/data-link layers (e.g., cable or conduit layer).

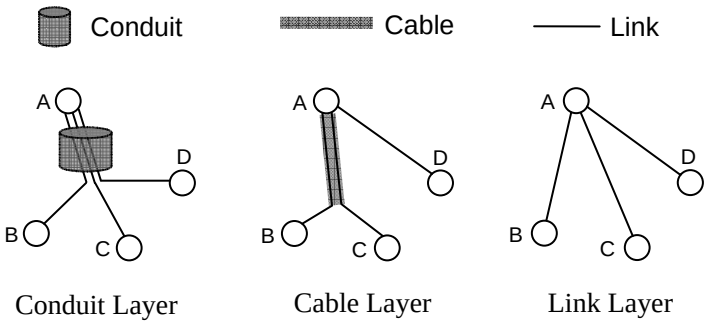


Figure 3.1: Multilayer network structure.

Figure 3.1 illustrates a simple example of the multilayer structure in today's communication networks. In the figure, the largest cylinder represents a conduit, typically carrying several fiber cables, each in turn carrying multiple channels (or wavelengths) to link between two nodes. Due to the multilayer structure, communication links connecting two distinct pairs of nodes may traverse the same conduit in physical networks. Thus, links that are diverse in the internet protocol or application layer are not necessarily diverse in the physical layer Li et al. [2001]. For example, in Figure 3.1, the links $A \leftrightarrow B$, $A \leftrightarrow C$ and $A \leftrightarrow D$ will fail if the conduit has a physical failure. We then group the links $A \leftrightarrow B$, $A \leftrightarrow C$ and $A \leftrightarrow D$ into an SRRG. Generally, the shared resources can often refer to a set of nodes (SRNG) and/or a set of links (SRLG). In the same example, the links $A \leftrightarrow B$, $A \leftrightarrow C$ and $A \leftrightarrow D$ specifically form an SRLG.

In order for network operators to provide reliable and protected communications, different protection schemes for multilayer networks have been studied in the literature. A common approach of protection schemes is to construct two simultaneous (or primary and backup) links that are SRRG-diverse between the source and destination Yuan and Jue

[2005]. However, most studies only address the issue of SRRG-diverse routing in unicast systems. It has been shown that, given an arbitrary set of links belonging to a common SRLG, the problem of finding SRLG-diverse paths between a given source and a destination (i.e., unicast traffic) is NP-complete Hu [2003]. In another unicast study, an approach for SRLG-diverse path protection in optical networks in WDM mesh networks was proposed in Zang et al. [2003].

3.3 Complexity of Redundant Multicast Routing Problem with Group Diverse Constraint (RMRGD)

In this section, we investigate the complexity issues of RMRGD. In Hu [2003], it is shown that a simpler version of RMRGD with a single destination can be reduced to the maximum bipartite subgraph problem Lewis [1978], Yannakakis [1978]. Here we will prove the NP-completeness of RMRGD by using the reduction of the 3-SAT problem, which is a restriction of the boolean satisfiability problemGarey and Johnson [1979].

Theorem 1. *RMRGD with 2 sources and 1 destination is an NP-complete problem.*

Proof. Here our proof follows the the NP-complete proof framework presented in Garey and Johnson [1979]. First we will show that RMRGD is NP by demonstrating that any two paths p_1 and p_2, each from one of the two sources to the destination, can be verified to be SRLG-diverse in polynomial time. One can verify this by comparing every edge in p_1 with every edge in p_2, which can be done in polynomial time. If any common edge pair between p_1 and p_2 belongs to the same SRLG, these two paths are not SRLG-diverse. Otherwise, these two paths are a solution to RMRGD.

Next, we will show that any instance of the well-known NP-complete 3-SAT problem reduces to an instance of RMRGD in polynomial time. We define an instance of the 3-SAT problem by a set of clauses $\{C_1 \wedge C_2 \wedge ... \wedge C_N\}$. Each clause consists of a disjunction of 3 literals, e.g., $C_1 = \{x_1 \vee x_2 \vee \bar{x}_3\}$, where each of the 3-SAT variables $x_1, x_2, ..., x_I$ yields a literal x_i or $\bar{x}_i$. Then we shall construct a 3-SAT graph $G = (V, E)$ such that this 3-SAT instance is satisfiable if and only if there exist SRLG-diverse paths in G. First we create a *special-purpose component* containing two nodes u_i and u_{i+1} for each literal (variable)

x_i (shown in Figure 3.2a). Nodes u_i and u_{i+1} are connected by two edges x_i and $\bar{x}_i$ (or $u_i u_{i+1}$ and $\overline{u_i u_{i+1}}$). We also create a *special-purpose component* containing two nodes v_n and v_{n+1} for each clause C_n (shown in Figure 3.2b). Nodes v_n and v_{n+1} are connected by three edges, each representing a literal (variable) in that clause. Specifically, if x_i is contained in clause C_n, then there is an edge c_{ni}; if $\bar{x}_i$ is contained in clause C_n, then the corresponding edge is $c_{n\bar{i}}$. An example in Figure 3.2b represents a special-purpose component for clause $C_n = \{x_1 \vee \bar{x}_2 \vee x_3\}$. Next we connect the two source nodes S_1

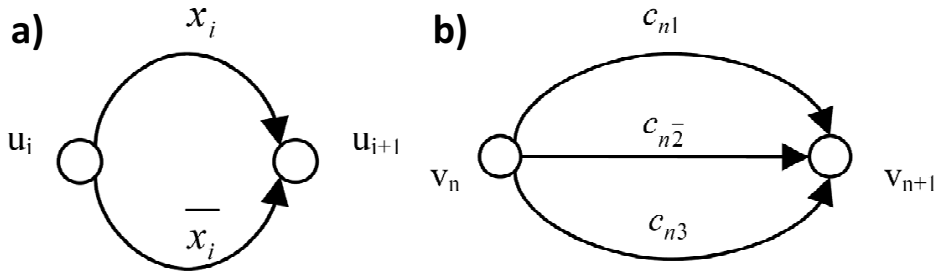


Figure 3.2: a) Special-purpose components for variable x_i; b) Special-purpose component for clause $C_n = \{x_{i1} \vee \bar{x}_{i2} \vee x_{i3}\}$

and S_2 with nodes u_1 and v_1, the destination node D with node u_{I+1} and v_{N+1}. It is easy to see that the constructed graph G network can represent any arbitrary instance of the 3-SAT problem, and obviously G can be constructed in polynomial time. Figure 3.3 illustrates an example of 3-SAT graph of an instance defined by $C_1 = \{x_1 \vee \bar{x}_2 \vee x_3\}$, $C_2 = \{x_1 \vee x_3 \vee \bar{x}_4\}$, $C_3 = \{\bar{x}_2 \vee \bar{x}_4 \vee x_5\}$.

After constructing graph G, we shall define four SRLGs as follows. The first SRLG contains edges $\{S_1u_1, S_2u_1\}$. The second SRLG contains edges $\{S_1v_1, S_2v_1\}$. The third SRLG contains c_{ni} and $\bar{x}_i$, and the last SRLG contains edges $c_{n\bar{i}}$ and x_i. The four SRLGs and their members are shown in Tab.3.1.

After constructing the graph and defining the SRLGs, we now show that this 3-SAT instance represented by the graph reduces to an instance of RMRGD. Suppose that this 3-SAT instance has a satisfying assignment, i.e., yes-instance or true-instance. Then in each clause C_n, one of its literal x_i or $\bar{x}_i$ must be true (yes). Note that each literal corresponds

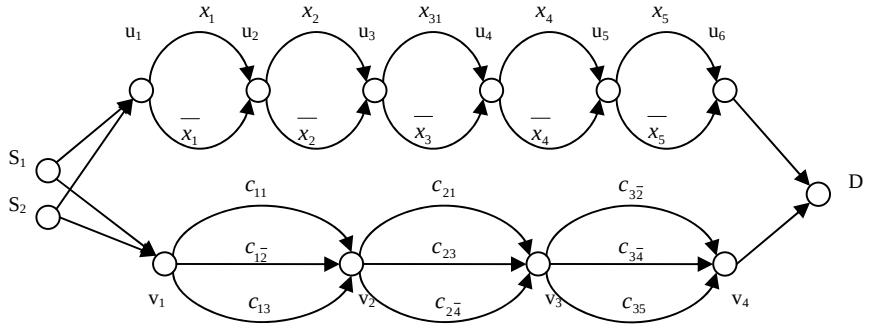


Figure 3.3: Constructed graph for a 3-SAT instance corresponding to $C_1 = \{x_1 \vee \bar{x}_2 \vee x_3\}$, $C_2 = \{x_1 \vee x_3 \vee \bar{x}_4\}$, $C_3 = \{\bar{x}_2 \vee \bar{x}_4 \vee x_5\}$

Group	Members		
SRLG1	S_1u_1,	S_2u_1	
SRLG2	S_1v_1,	S_2v_1	
SRLG3	C_{ni},	$\bar{x}_i$,	$\forall i \in I, \forall n \in N$
SRLG4	$C_{n\bar{i}}$,	x_i,	$\forall i \in I, \forall n \in N$

Table 3.1: SRLGs and their members in the 3-SAT graph.

to an edge, x_i for u_iu_{i+1} or $\bar{x}_i$ for $\overline{u_iu_{i+1}}$. By selecting one such "true" literal from each clause, an edge from the first special-purpose component and an edge from the second special-purpose component will be also selected. We claim that the two paths formed by connecting these selected edges are SRLG-diverse. Specifically, let the first path travel from S_1 to D through the the first special-purpose component (u nodes and their edges) and the second path travel from S_2 to D through the the second special-purpose component (v nodes and their edges). Clearly, for any literal $x_i = 1$, the first path transverses edge x_i and the second path transverses edge C_{ni}. Similarly, for any literal $\bar{x}_i = 1$, the first path transverses edge $\bar{x}_i$ and the second path transverses edge $C_{n\bar{i}}$. Based on the definition of the four SRLGs, edges x_i and C_{ni} are not in the same SRLG, and edges $\bar{x}_i$ and $C_{n\bar{i}}$ are not in the same SRLG. Thus we can now conclude that if the 3-SAT instance is a yes-instance, there must exist two SRLG-diverse paths in graph G.

We shall now prove the converse result. Suppose that G has two SRLG-diverse paths. It is obvious that one path $p1$ must travel all the u nodes (first path), and the other path $p2$ must travel all the v nodes (second path). We can assign "true" (yes) to each literal x_i in $p1$ and each clause $C_{n_{\bar{i}}}$ in $p2$ will not be true because of the SRLG constraint. Similarly, we can assign "true" (yes) to each $\bar{x}_i$ in $p1$ and each clause C_{n_i} in $p2$ will not be true.

Any variables that do not correspond to an edge in the first SRLG-diverse path may be set arbitrarily. For selected values of x_i and $\bar{x}_i$, a yes-instance can be obtained. Now, we can conclude any instance of the 3-SAT problem reduces to an instance of RMRGD. Thus RMRGD is NP-complete. □

It is easy to see that the above proof can be easily modified to prove the NP-Completeness of 2-source multiple-destination RMRGD. We can also show that the SRLG-diverse routing and SRNG-diverse routing problems are mathematically equivalent. We can do the simple transformations between SRLG and SRNG in both directed network and undirected network as shown in Figure 3.4. The detailed proof can be found in Liang et al. [2010]. Based on these observations, we claim that the SRRG-diverse routing and SRNG-diverse routing problems are NP-complete. In this study that focuses on telecommunication links, the concept of SRLG is more practical than that of SRNG because the link reliability (e.g., broken IP connection) is much more vulnerable than the node reliability (e.g., router failure). Therefore, in the remainder of this chapter, the mathematical formulation and solution approaches will focus on SRLG-diverse problem. However, these approaches can be generalized to SRRG-diverse and SRNG-diverse problems.

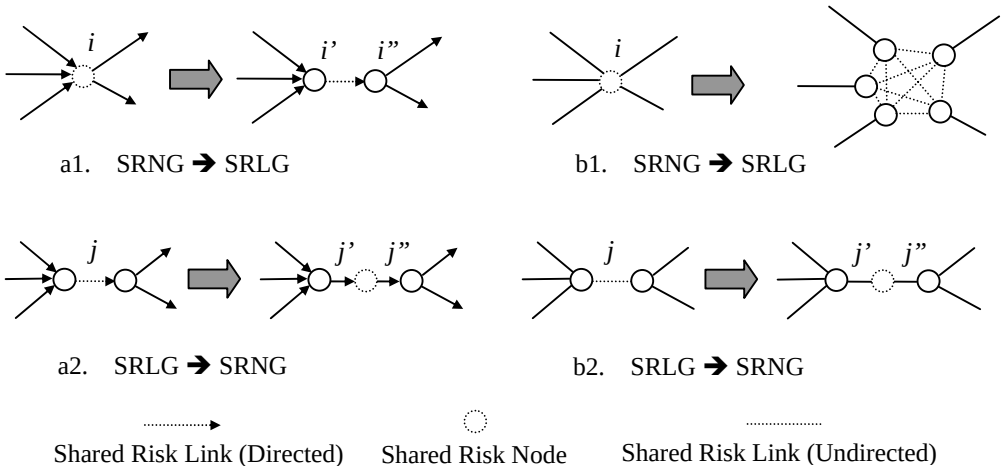


Figure 3.4: Transformation between SRLG and SRNG for: a) directed graph and b) undirected graph.

3.4 Edge-based Model

Given a directed network $G = (V, A)$, where V is a set of nodes and A is a set of arcs. There is a set of two sources denoted by $\{s_1, s_2\} = S \subset V$ and a set of destinations denoted by $D \subset V$. There is a set of *risk groups* denoted by B. Each risk group $b \in B$ contains a set of edges $A_b \subset A$, which are subject to a common risk. Each arc $(i, j) \in A$ has an associated cost c_{ij} and belongs to one or more risk groups. The objective of the problem is to find two multicast trees from each source $s \in S$ to all the destinations D with minimum total cost, and to ensures any pair of paths $p_d^{s_1}, p_d^{s_2}$ to a destination $d \in D$ do not use any common risk group $b \in B$. If an edge of A_b is used in a path, we say the risk group b is used in that path. We call this problem the redundant multicast routing problem with group diverse constraints (RMRGD).

The edge-based model ($RMRGD_E$) for RMRGD is formally defined as follows. Define binary decision variable y_{ij}^s such that $y_{ij}^s = 1$ if edge (i, j) is included in the multicast tree from source s, and $y_{ij}^s = 0$ otherwise. Define binary decision variable x_{ij}^{sd} such that edge $x_{ij}^{sd} = 1$ if edge (i, j) is used by a path from s to d in the solution, and 0 otherwise. Define decision binary variable $z_b^{sd} = 1$ if the path from source s to destination d uses risk group $b \in B$, and 0 otherwise. The $RMRGD_E$ is given by

$$\min \sum_{s \in S} c_{ij} y_{ij}^s \tag{3.1}$$

$$\text{s.t. } y_{ij}^s - x_{ij}^{sd} \geq 0 \qquad \forall\ (i,j) \in A,\ \forall s \in S,\ \forall d \in D, \tag{3.2}$$

$$\sum_{j|(i,j)\in A} x_{ij}^{sd} - \sum_{j|(j,i)\in A} x_{ij}^{sd} = \sigma_i^{sd} \quad \forall i \in V,\ \forall s \in S,\ \forall d \in D, \tag{3.3}$$

$$z_b^{sd} - x_{ij}^{sd} \geq 0 \qquad \forall (i,j) \in A_b,\ \forall b \in B,\ \forall s \in S,\ \forall d \in D, \tag{3.4}$$

$$\sum_{s \in S} z_b^{sd} \leq 1 \qquad \forall b \in B,\ \forall d \in D, \tag{3.5}$$

$$x_{ij}^{sd}, y_{ij}^s, z_b^{sd} \in \{0, 1\} \qquad \forall\ (i,j) \in A,\ \forall b \in B,\ \forall s \in S,\ \forall d \in D. \tag{3.6}$$

The objective function in Eq. (3.1) minimizes the total cost of two redundant multicast trees. The constraints in Eq. (3.2) are the logical constraints that ensure edge (i, j) must be selected in the multicast tree from source s, if it is used in the path from source s to destination d in the solution. The constraints in Eq. (3.3) are the flow balance constraints

for a path between source s and destination d. In particular, σ_i^{sd} is the net flow capacity at node i, which indicates a source, sink or transhipment node, in a path from source s to destination d. Thus $\sigma_i^{sd} = 1$ if $i = s$, $\sigma_i^{sd} = -1$ if $i = d$, and $\sigma_i^{sd} = 0$ otherwise. The constraints in Eq. (3.4) are the logical constraints ensuring that a risk group b is used in the path from source s to destination d, if any of its edge $(i, j) \in A_b$ is included in the path. The constraints in Eq. (3.5) ensure that the two paths to every destination are group diverse.

In particular, we notice the number of variables in $RMRGD_E$ is $|A| \times |S| \times (|D| + 1) + |B| \times |S| \times |D|$, and the number of constraints is $(|A| + |N| + \sum_{b \in B} |A_b|) \times |S| \times |D| + |B| \times |D|$. It is easy to see the size of the problem matrix increases quadratically with the number of destinations. Moreover, because there is no investigation on $RMRGD_E$ structure, limited practical and theoretical results are revealed for RMRGD.

3.5 Path-based Model

In this section, we propose two alternative path-based models: segregated and aggregated. We prove that the linear relaxation of the segregated model is equivalent to the linear relaxation of the edged model. We also prove the linear relaxation of the aggregated model is weaker than that of the edge-based model.

It notes the constraints in Eq. (3.3) determine the paths from every source to every destination. Therefore, it is intuitive to formulate the problem as a path-based model. In the path-based model, we correspond a decision variable to a path from source $s \in S$ to destination $d \in D$, which arises from the Dantzig-Wolfe decomposition of the edge model.

We shall define the following additional notations.

P is the set of all possible paths from any source $s \in S$ to any destination $d \in D$; we also denote P^{sd} as the set of all possible paths from s to d, where $s \in S$, $d \in D$. We have $P = \bigcup_{s \in S} \bigcup_{d \in D} P^{sd}$; similarly, we define P_{ij}^s as the set of paths from source s using edge (i, j).

α_{ij}^p is a binary parameter such that $\alpha_{ij}^p = 1$ if path p passes through edge (i, j), and $\alpha_{ij}^p = 0$ otherwise.

β_b^p is a binary parameter such that $\beta_b^p = 1$ if any edge of path p is a member of SRLG $b \in B$, and $\beta_b^p = 0$ otherwise.

u_p is a binary variable such that $u_p = 1$ if path p is selected to form the redundant multicast trees in the solution, and $u_p = 0$ otherwise.

3.5.1 Segregated Path-Based Model

Given the above additional notations, we can define the segregated path-based model ($RMRGD_{P1}$) as follows.

$$RMRGD_{P1} \quad \min \sum_{s \in S} \sum_{(i,j) \in A} c_{ij} y_{ij}^s \tag{3.7}$$

$$\text{s.t.} \quad \sum_{p \in P^{sd}} \alpha_{ij}^p u_p \leq y_{ij}^s \qquad \forall\ (i,j) \in A, \forall d \in D, \forall\ s \in S, \tag{3.8}$$

$$\sum_{p \in P^{sd}} u_p = 1 \qquad \forall\ s \in S,\ d \in D, \tag{3.9}$$

$$\sum_{s \in S} \sum_{p \in P^{sd}} \beta_b^p u_p \leq 1 \qquad \forall\ b \in B, \forall\ d \in D, \tag{3.10}$$

$$u_p, y_{ij}^s \in \{0,1\} \qquad \forall\ p \in P, \forall\ (i,j) \in A, \forall s \in S. \tag{3.11}$$

The objective function in Eq. (3.7) is the same as in Eq. (3.1). The constraints in Eq.(3.8) ensure that an edge must be selected if a path containing that edge is used in the multicast trees. The constraints in Eq. (3.9) ensure that each source-destination pair is connected by exactly one path. The constraints in Eq. (3.10) ensure that the two paths connecting the two sources to every destination are SRLG-diverse. Define the linear relaxations of $RMRGD_E$ and $RMRGD_{P1}$ as LM_E and LM_{P1}. Define the optimal solution values of LM_E and LM_{P1} as $\nu(LM_E)$ and $\nu(LM_{P1})$ respectively. We have the following lemma.

Lemma 2. *LM_{P1} is equivalent to LM_E. Every feasible solution for LM_{P1} corresponds to a feasible solution for LM_E.*

Proof. Since every path satisfies the flow balance constraints in Eq. (3.3), we know

$$\sum_{p \in P^{sd}} \alpha_{ij}^{p} u_p = x_{ij}^{sd} \qquad \forall\ (i,j) \in A,\ \forall\ s \in S,\ \forall d \in D, \tag{3.12}$$

$$\sum_{p \in P^{sd}} \beta_{b}^{p} u_p = z_{b}^{sd} \qquad \forall\ s \in S,\ \forall d \in D. \tag{3.13}$$

Therefore, we know constraints in Eq. (3.2) and constraints in Eq. (3.8) are equivalent because of the Eq. (3.34). Similarly, constraints in Eq. (3.4)-(3.5) are equivalent to constraints in Eq. (3.10) because of the Eq. (3.35). Hence, we conclude the segregated path-based model M_{P1} is the equivalent to the edge-based model M_E. □

3.5.2 Aggregated Path-Based Model

The aggregated path-based model ($RMRGD_{P2}$) can be formulated as follows.

$$RMRGD_{P2} \quad \min \sum_{s \in S} \sum_{(i,j) \in E} c_{ij} y_{ij}^{s} \tag{3.14}$$

$$\text{s.t.} \sum_{d \in D} \sum_{p \in P^{sd}} \alpha_{ij}^{p} u_p \leq |D| y_{ij}^{s} \qquad \forall\ (i,j) \in A,\ \forall\ s \in S \tag{3.15}$$

$$\sum_{p \in P^{sd}} u_p = 1 \qquad \forall\ s \in S,\ d \in D \tag{3.16}$$

$$\sum_{s \in S} \sum_{p \in P^{sd}} \beta_{b}^{p} u_p \leq 1 \qquad \forall\ b \in B,\ \forall\ d \in D \tag{3.17}$$

$$u_p, y_{ij}^{s} \in \{0,1\} \qquad \forall\ p \in P,\ \forall\ (i,j) \in A,\ \forall s \in S. \tag{3.18}$$

The formulation in Eqs. (3.14)-(3.18) is the same as formulation M_{P1} except constraints in Eq. (3.15) are in the aggregated form of constraints in Eq. (3.8).

Lemma 3. *LM_E is strictly stronger than LM_{P2}. The worst case ratio $\frac{\nu(LM_E)}{\nu(LM_{P2})}$ is $|D|$.*

Proof. Here, we need to prove $\nu(LM_E) \leq \nu(LM_{P2})$ for all instances, and there exists some instances such that $\nu(LM_E) < \nu(LM_{P2})$.

Assume the solution of LM_{P2} contains a set of paths P_s' from each source s, such that $u_p > 0, \forall p \in P_s'$. It is obvious the edges from each path satisfy the flow balance constraints in Eq. (3.3). Also, we can see that $z_b^{sd} = \sum_{p \in P^{sd}} \beta_b^p u_p$. Therefore, we know the constraints in Eq. (3.5) are satisfied. Hence, the edges from the optimal solution of the path-based

model LM_{P2} is a feasible solution for LP relaxation of the edge-based model. Furthermore, we know

$$\begin{aligned}
\nu(LM_{P2}) &= \sum_{s\in S}\sum_{(i,j)\in A} c_{ij}y_{ij}^{s} \\
&= \sum_{s\in S}\sum_{(i,j)\in A}\sum_{d\in D}\sum_{p\in P^{sd}} c_{ij}\frac{\alpha_{ij}^{p}u_p}{|D|} \\
&= \sum_{s\in S}\sum_{(i,j)\in A}\sum_{d\in D} c_{ij}\frac{\sum_{p\in P^{sd}}\alpha_{ij}^{p}u_p}{|D|} \\
&= \sum_{s\in S}\sum_{(i,j)\in A}\sum_{d\in D} c_{ij}\frac{x_{ij}^{sd}}{|D|} \qquad \text{due to Eq. (3.34)} \\
&= \sum_{s\in S}\sum_{(i,j)\in A} c_{ij}\frac{\sum_{d\in D}x_{ij}^{sd}}{|D|} \\
&\leq \sum_{s\in S}\sum_{(i,j)\in A} c_{ij}y_{ij}^{s} \\
&= \nu(LM_E).
\end{aligned}$$

Similarly, we have

$$\begin{aligned}
\nu(LM_{P2}) &= \sum_{s\in S}\sum_{(i,j)\in A} c_{ij}\frac{\sum_{d\in D}x_{ij}^{sd}}{|D|} \\
&\geq \sum_{s\in S}\sum_{(i,j)\in A} c_{ij}\frac{y_{ij}^{s}}{|D|} \\
&= \frac{1}{|D|}\nu(LM_E).
\end{aligned}$$

Now, consider an example in Figure 3.5. We have two source nodes s_1 and s_2 and a set of destination nodes from d_1 to d_n. We assume that $c_{s_1,a} = c_{s_2,a} = 0$ and $c_{a,d_1} = \cdots = c_{a,d_n} = 1$. We also assume each edge is a SRLG by itself. Then it is obvious $\nu(LM_E) = 2n$ and $\nu(LM_{P2}) = 2$. Hence, we know $\nu(LM_E) < \nu(LM_{P2})$ for some instance. Therefore, we know the edge-based model has stronger LP relaxation than the aggregated path-based model. □

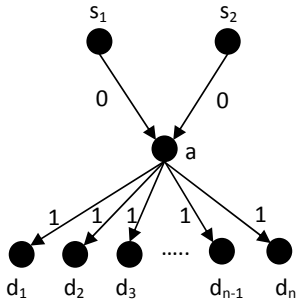


Figure 3.5: An example of RMR-SRLGD where $\nu(LM_E) \geq \nu(LM_{P2})$. Here, $\nu(LM_E) = 2n > \nu(LM_{P2}) = 2$.

Because $RMRGD_{P1}$ is stronger than $RMRGD_{P2}$, in the following sections, we only focus on how to solve $RMRGD_{P1}$, which will be referred as the path-based model in general.

3.5.3 Column Generation

In this section, we present three column generation pricing subproblem approaches for the path-based model.

3.5.3.1 Probability Approach

We first propose a probabilistic approach to generate a set of paths and select them probabilistically based on the depth of the paths. This approach starts with a breadth-first search to enumerate all possible paths between each $s - d$ pair. At every level of the breadth-first search tree, we select a set of paths in the tree to be included in $RMRGD_P$. Specifically, at level n of the search tree, we will randomly select $\frac{1}{n}$ of all the paths at the current level. By using this probability, shorter paths will have a higher probability to be included in $RMRGD_P$ than longer paths. This procedure terminates when the number of selected paths is greater than a pre-determined threshold. This probabilistic path generation approach is simple and easy to implement, and we employ it as a benchmark method.

3.5.3.2 Multi-Objective Greedy-Based Approach

Considering the constraints in Eqs. (6.3)-(6.4) of $RMRGD_P$, we observe that the two key factors playing an important role in improving solution quality are the cost and associated SRLGs of each path. It is intuitive that paths with lower costs and lesser number of associated SRLGs are desirable. This is a multi-objective problem. We herein propose a multi-objective greedy-based approach to generate only such desirable paths. We introduce a concept of path domination used to generate path candidates for $RMRGD_P$.

Let p_1 and p_2 be two distinct paths from s to d, where $s \in S$ and $d \in D$. p_1 *dominates* p_2 if and only if the following two conditions are satisfied, and either the strict inequality or strict subset holds.

- The cost of p_1 is less than or equal the cost of p_2 (i.e., $c_{p_1} \leq c_{p_2}$).
- The set of associated SRLGs of p_1 is a subset of or equal to the set of associated SRLGs of p_2 (i.e., $B_{p_1} \subseteq B_{p_2}$).

Path p is a *non-dominated path* if there is no paths dominating path p from s to d, $s \in S$ and $d \in D$.

It is not necessary to enumerate all possible paths in order to find all the non-dominated paths. We develop an intelligent way to search for all non-dominated paths. This problem can be in turn modeled as a multi-objective shortest path problem with one objective to minimize the cost and the other objective to minimize the set of associated SRLGs. There are several studies in the literature addressing this multi-objective shortest path problem Queiros and Martins [1984], Modesti and Sciomachen [1998], Perny and Spenjaard [2005]. Here, we modify the multiple labelling approach to find a set of non-dominated paths. The procedure of this approach, which is similar to the one in Queiros and Martins [1984], can be described as follows.

For each node, we keep a set of non-dominated labels (l), each being a combination of cost and associated SRLG set with two pointers i, l. Specifically, the l_ith label for node i is denoted by $[c_{p_i^s}, B_{p_i^s}, (j, l_j)]_{l_i}$, where (j, l_j) is the predecessor of the current label and $(i, j) \in A$. A permanent label of a node $i \in V$ is unchanged. Given a permanent label for node i, we can then assign a temporary label to node j if $(i, j) \in A$. Each label at every

Searching Procedure for Non-dominated Path

Input: The graph $G(V, A)$, a set of SRLGs B,
a source node s and a destination node d, where $s \in S$, $d \in D$
Output: A set of non-dominated paths from s to d

1 Assign the temporary label $[(0, \emptyset), (-, -)]$ to s;
2 **WHILE** the set of temporary label is not empty
3 Find the lth label of node i such that $c_{p_i^s}$ is minimum among all temporary labels;
4 **WHILE** some node $j \in V$ exists such that $(i, j) \in A$
5 Compute $c_{p_j^s} = c_{p_i^s} + c_{ij}$, and $B_{p_j^s} = B_{p_i^s} \cup B_{ij}$;
6 Set $[c_{p_j^s}, B_{p_j^s}, (i, l)]$ to be a new label of node j;
7 Delete all dominated temporary label from s to j;
8 **End**
9 **End**
10 Find all non-dominated label from s to d.

Figure 3.6: Pseudo-code of multi labeling algorithm for non-dominated path generation.

iteration of the algorithm corresponds to a unique path and a simple backtracking method can be used to construct these non-dominated paths. The detailed algorithm is shown in Figure 3.6.

It is important to note that the above multi-objective algorithm is a greedy approach. In most cases, greedy approaches tend to be trapped in a local optima, and cannot provide good solutions that are close to the global optimal. The concept of non-dominated paths also suffers from this downfall. It is very likely that we will not obtain the optimal solution by including only the non-dominated paths in our path formulation model. On the other hand, we do not want to include obviously bad paths as well. We herein enhance the multi-objective approach by introducing a concept of algorithm diversification, where we include not only non-dominated paths but also *nearly non-dominated* paths in $RMRGD_P$. The algorithm used to generate all the nearly non-dominated paths is a relaxation version of the algorithm in Figure 3.6. Specifically, we relax the multi-objective search algorithm so that a set of near non-dominated paths are included in the solution. We thus define nearly non-dominated paths as follows.

- A path p' is considered to be a nearly non-dominated path if $c_{p'} - c_p < \delta$ and $B_{p'} \subseteq B_p$ for any non-dominated path p, where δ is the relaxation (or diversification) factor.

It is obvious that the greater the δ value is, the larger number of nearly non-dominated paths are generated. Setting an appropriate value of δ is very critical to the performance of the proposed algorithm. If δ is too small, the paths included in $RMRGD_P$ maybe too

restricted so that the optimal or high-quality solutions may be excluded. If δ is too large, we may include too many bad paths so that the time used to solve $RMRGD_P$ drastically increases. In order to find a proper value of δ, we propose a simple line search scheme to balance the algorithm's greediness and diversification. In particular, we start with $\delta = 0$, which is equivalent to generating all the non-dominated paths, and then solve $RMRGD_P$. In the following iteration, we gradually increase the value of δ and resolve the problem. We stop the algorithm if the solution quality does not improve. It is obvious that the step size of the line search plays a critical role in this procedure. If the step size is too large, we have to generate too many unnecessary paths at once; if the step size is too small, we may not improve the solution quality after several iterations. In this study, we set the step size equal to an average edge cost divided by 2, which seems to be practical in terms of computational efficiency and solution quality.

3.5.3.3 Mathematical Programming Model for Pricing Subproblem

Let ω_{ij}^s be the non-positive dual variable associated with the constraints in Eq.(6.2), μ_d^s be the dual variable associated with the constraints in Eq.(6.3), and τ_{bd} be the non-positive dual variable associated with the constraints in Eq.(6.4). The reduced cost $\bar{c}_{p_d^s}$ of a path p_d^s can be computed using the following equation.

$$\bar{c}_{p_d^s} = -\sum_{(i,j)\in E} \alpha_{ij}^p \omega_{ij}^s - \mu_d^s - \sum_{b\in B} \beta_b^p \tau_{bd} \tag{3.19}$$

The pricing subproblem can be view as the problem of finding a sequence of edges from s to d and prices out to have negative reduced cost. Obviously, for any given $s \in S$ and $d \in D$, we can formulate the problem as an MIP problem as shown below.

$$\min \sum_{(i,j)\in A} c_{ij} y_{ij} + \sum_{b\in B_d} \tau_b z_b \tag{3.20}$$

$$\text{s.t.} \sum_{j:(i,j)\in A} y_{ij} - \sum_{j:(j,i)\in A} y_{ji} = \sigma_i^{sd} \qquad \forall\ i \in V \tag{3.21}$$

$$z_b - y_{ij} \geq 0 \qquad \forall\ (i,j) \in A_b,\ \forall\ b \in B_d \tag{3.22}$$

$$y_{ij}, z_b \in \{0,1\} \qquad \forall\ (i,j) \in A,\ \forall\ b \in B_d \tag{3.23}$$

By solving the above MIP problem, we are able to get good columns with negative reduced cost. However, only one good column can be generated by solving the MIP formulation each time. In fact, it is widely known that column generation approach will be much more efficient if we obtain multiple columns in each of the pricing iteration.

We also observe that if the pricing function in Eq.(4.10) only contains the first two parts, which is p_d^s is $-\sum_{(i,j)\in E} \alpha_{ij}^p \omega_{ij}^s - \mu_d^s$, then the problem can be simply formulated as a shortest path problem from s to d and solved by Dijkstra's algorithm as all the edge costs in our subproblem are non-negative and for a given s and d, μ_d^s is a constant. With extra bundle component in the pricing function, the Dijkstra's algorithm cannot be applied directly. For example, in Figure 3.7, we search for a path from A to F with the best reduced cost. The ω value of each arc is shown above the arc and the τ values

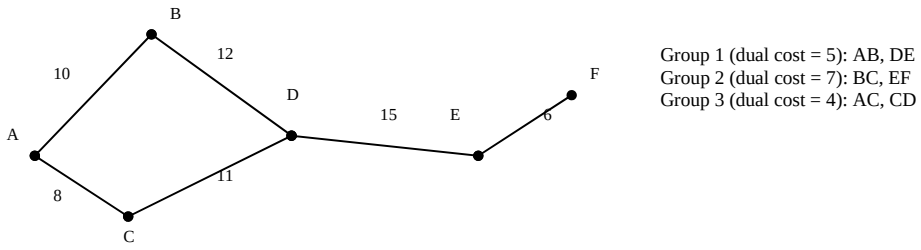


Figure 3.7: An example of searching for good path.

are shown on the right side of the figure. By using Dijkstra's algorithm, we know that the path $A \to C \to D$ (with cost 23) is superior to path $A \to B \to D$ (with cost 34). However, the path $A \to C \to D \to E \to F$ (with cost 56) is in fact worse than the path $A \to B \to D \to E \to F$ (total cost 55). This is because the necessary optimality condition of shortest path Ahuja et al. [1993] $c_{AF} \leq c_{AD} + c_{DF}$ is not satisfied. In particular, the cost of an arc is depending on the history of the path because of the group cost in the bundle constraints.

In order to solve the price problem efficiently, we use a modified Dijkstra's algorithm to search good paths. We evaluate the reduced cost using Eq. (4.10). If the reduced costs of paths are less than 0, we add them into the restricted master problem, and the column generation continues. Otherwise, we use the mathematical model in Eqs. (4.15)-(4.21) to check the existence of the path with negative reduced cost.

3.5.4 IP Solution & Branch-and-Price

Given a fractional solution to the LP relaxation of $RMRGD_P$, we can identify a set of fractional y values. In the branch and price process, we branch on y value and use a depth first search algorithm. In particular, we always search the branch with maximum y value where $y \neq 1$, for the next branching decision. It is also noted that after a branch on y, it is possible that no feasible LP solution can be found based on existing columns in that branch, hence no subsequent pricing procedure can be continued. To overcome it, we create a set of dummy edges which connect each s and d directly. Also a set of dummy paths are created, each path contains a dummy edge. We set a very high penalty value for each dummy edge. In the branch-and-bound tree, branching on these dummy edges is restricted. Hence, in each of the branch in the tree, we always have the set of dummy paths connecting each s and d pair. Therefore, we can always find a feasible solution and continue the column generation.

3.6 Tree-based Model

In this section, we present the tree-based model for RMRGD problem. Define the set of additional notations as follows. Define T_s as a set of multicast trees connecting a source node $s \in S$ to all the destination nodes. T is the superset of T_s, and $T = \bigcup_{s \in S} T_s$. Define c_t as the cost of the multicast tree $t \in T$, and $c_t = \sum_{(i,j) \in T} c_{ij}$. Define binary parameter θ_{bd}^t such that $\theta_{bd}^t = 1$ if the path from s to d in a multicast tree t containing group d; $\theta_{bd}^t = 0$ otherwise. Since a path between any two nodes in a multicast tree is unique, we can explicitly identify the value of θ_{bd}^t for any destination-group pair (b, d) of tree t. Define the binary decision variable w_t such that $w_t = 1$ if multicast tree t is selected in the solution and $w_t = 0$ otherwise. Given the above new notations, we present the tree-based

model $RMRGD_T$ as following:

$$\min \sum_{s\in S}\sum_{t\in T_s} c_t w_t \tag{3.24}$$

$$\text{s.t.} \sum_{t\in T_s} w_t = 1 \qquad \forall\ s \in S, \tag{3.25}$$

$$\sum_{s\in S}\sum_{t\in T_s} \theta_{bd}^t w_t \le 1 \qquad \forall\ b \in B, \forall\ d \in D, \tag{3.26}$$

$$w_t \in \{0,1\} \qquad \forall\ t \in T, \forall\ (i,j) \in A. \tag{3.27}$$

Constraints in Eq. (3.24) minimize cost of the selected multicast trees. Constraints in Eq. (3.25) ensure a multicast tree is selected from each source. Constraints in Eq. (3.26) ensure the group diverse constraints for every destination d.

Theorem 4. *The LP relaxation of $RMRGD_T$ is strictly tighter than the LP relaxation of $RMRGD_E$.*

Proof. Denote the optimal LP solution of $RMRGD_T$ as LM_T, and its value as $\nu(LM_T)$; denote the optimal LP solution of $RMRGD_E$ as LM_E, and its value as $\nu(LM_E)$. We need to prove $\nu(LM_T) \ge \nu(LM_E)$ for all instances, and there exists some instances such that $\nu(LM_T) > \nu(LM_E)$.

Assume that LM_T contains a set of trees T'_s from each source s, such that $w_t > 0, \forall t \in T'_s$. We can construct a set of paths P_t from each tree $t \in T'_s$. Each path $p \in P_t$ is from source s to a destination $d \in D$. It is obvious that edges from each path satisfy the flow balance constraints in Eq. (3.3). Also, we can see that $z_b^{sd} = \sum_{t\in T_s} \theta_{bd}^t w_t$. Therefore, we know the constraints in Eq. (3.5) are satisfied. Hence, the edges from the optimal solution to the $RMRGD_T$ is a feasible solution to LP relaxation of $RMRGD_E$. Furthermore, we know

$$\nu(LM_T) = \sum_{s\in S}\sum_{t\in T'_s} c_t w_t = \sum_{s\in S}\sum_{t\in T'_s}\sum_{(i,j)\in t} c_{ij} w_t \ge \sum_{s\in S}\sum_{(i,j)\in \bigcup_{e\in T'_s}\{e\}} c_{ij} y_{ij}^s = \nu(LM_E).$$

Now, consider an example in Figure 3.8. We have two sources s_1 and s_2 and two destinations d_1 and d_2. We assume that each edge is a risk group by itself and the cost is shown in the figure. In LM_T, we have two trees from s_1: the first tree $t_{s_1}^1$ contains edges

$s_1 \rightarrow d_1$, $d_1 \rightarrow v_1$, $v_1 \rightarrow v_2$, and $v_2 \rightarrow d_2$; the second tree $t^2_{s_1}$ contains edges $s_1 \rightarrow d_2$, $d_2 \rightarrow v_1$, $v_1 \rightarrow v_2$, and $v_2 \rightarrow d_2$. There is a tree from s_2, which contains edges $s_2 \rightarrow d_1$ and $s_2 \rightarrow d_2$. We know $c_{t^1_{s_1}} = c_{t^2_{s_1}} = c_{t_{s_2}} = 6$, $w_{t^1_{s_1}} = w_{t^2_{s_1}} = 0.5$ and $w_{t_{s_2}} = 1$. Hence, it is obvious that $\nu(LM_T) = 12$.

However, in $RMRGD_E$, we have $y_{d_1,d_1} = y_{s_1,d_2} = y_{d_1,v_1} = y_{d_2,v_1} = y_{v_1,v_2}, = y_{v_2,d_1} = y_{v_2,d_2} = 0.5$, and $y_{s_2,d_1} = y_{s_2,d_2} = 1$. Therefore, $\nu(LM_E) = 11.5 < \nu(LM_T) = 12$. Hence, we know $\nu(LM_T) > \nu(LM_E)$ for this instance. Therefore, LM_T is strictly stronger than LM_E.

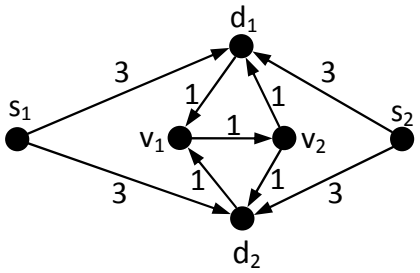


Figure 3.8: An example of RMRGD where $\nu(LM_T) > \nu(LM_E)$. Two multicast trees $t^1_{s_1}$ and $t^2_{s_1}$ from s_1 and one multicast tree t_{s_2} from s_2 are in the solution. $t^1_{s_1}$ contains edge $s_1 \rightarrow d_1$, $d_1 \rightarrow v_1$, $v_1 \rightarrow v_2$, and $v_2 \rightarrow d_2$. $t^2_{s_1}$ contains edges $s_1 \rightarrow d_2$, $d_2 \rightarrow v_1$, $v_1 \rightarrow v_2$, and $v_2 \rightarrow d_2$. In the LM_T, $w_{t^1_{s_1}} = w_{t^1_{s_2}} = 0.5$ and $w_{t_{s_2}} = 1$. $\nu(LM_T) = 12 > \nu(LM_E) = 11.5$.

□

We cannot enumerate all the multicast trees because the number of the trees increases exponentially with the size of the network. However, we would like to generate only necessary trees and ensure the optimality of the model. This can be achieved by using column generation method.

3.6.1 Column Generation Subproblem

Column generation has been widely and successfully applied to many large-scale, real-life optimization problems where the number of variables is too large to enumerate explicitly Desrocsiers et al. [1984], Lubbecke and Mesrosiers [2005], Wolsey [1998]. The column generation framework developed here for $RMRGD_T$ can be described as follows. First a subset of multicast trees are constructed by solving a set of Steiner tree problems from both sources without considering the risk groups. The restricted master problem of $RMRGD_T$

is then solved, and the dual cost of each constraint is calculated. To improve the restricted master (i.e., generate more columns), we solve the pricing subproblems based on the dual cost information, and generate new multicast trees with negative reduced costs. The lower and upper bounds of the problem are computed based on the dual information. These multicast trees are then added to the restricted master problem, and the updated restricted master problem is resolved. These iterative procedures repeat until no new multicast trees with negative reduced costs is found, which implies that the current LP solution is optimal.

3.6.1.1 Pricing Subproblem Model

Define α^s as the dual variable associated with the constraints in Eq. (3.25). Because the equal sign in Eq. (3.25) can be replaced by "$\geq$", we know $\alpha^s \geq 0$. Define β_b^d as the negative dual variable associated with the constraints in Eq. (3.26). That is, β_b^d evaluates the importance of group b to destination d. For a given a multicast tree t from source s, the reduced cost c_t' is defined as follows.

$$c_t' = c_t - \sum_{b \in B} \sum_{d \in D} \beta_b^d \theta_{bd}^t - \alpha^s = \sum_{(i,j) \in t} c_{ij} - \sum_{b \in B} \sum_{d \in D} \beta_b^d \theta_{bd}^t - \alpha^s \tag{3.28}$$

If we consider β_b^d as the risk cost of risk group b to destination d, the pricing subproblem can be viewed as the problem of finding a multicast tree with minimum edge cost and risk cost. It is obvious this problem is an extension of the Steiner tree problem Pardalos and Khoury [1996], Polzin and Daneshmand [2001a,b], in which only edge cost is considered. Therefore, we extend the well-known multicommodity flow model for Steiner tree problem to formulate the pricing subproblem, because it has been reported the multicommodity flow model provides very good LP relaxation of the Steiner tree problem. This minimum

cost tree problem can be formulated as following.

$$\min \sum_{(i,j)\in A} c_{ij} y_{ij} - \sum_{b\in B} \sum_{d\in D} \beta_b^d z_b^d \tag{3.29}$$

$$\text{s.t. } y_{ij} \geq x_{ij}^d \qquad \forall\ (i,j) \in A, \forall\ d \in D, \tag{3.30}$$

$$\sum_{(i,j)\in A} x_{ij}^d - \sum_{(j,i)\in A} x_{ji}^d = \sigma_i^d \quad \forall\ i \in V, \forall\ d \in D, \tag{3.31}$$

$$z_b^d \geq x_{ij}^d \qquad \forall\ (i,j) \in b, \forall\ b \in B, \forall d \in D, \tag{3.32}$$

$$x_{ij}^d, y_{ij}, z_b^d \in \{0,1\} \qquad \forall\ (i,j) \in A, \forall\ d \in D, \forall\ d \in D, \forall b \in B. \tag{3.33}$$

We call this model the basic pricing model (BPM). The objective function in Eq.(3.29) minimizes the total communication cost and the total risk cost of the multicast tree. We do not consider the dual cost α^s in the objective function because it is a constant for a given source $s \in S$. The constraints in Eq.(3.30) are the logical constraints to ensure that an arc must be selected if it is used in the path from the source to any destination. The constraints in Eq.(3.31) are the flow balance constraints for a path from the source to a destination. The constraints in Eq.(3.32) are the logical constraints to ensure that a risk group is selected if any of its arc is used in the path from the source to any destination. It is worth mentioning that we do not need to enumerate all the constraints for every combination of destination $d \in D$ and risk group $b \in B$ in Eq. (3.32), but only the constraints for destinations d and risk group b such that $\beta_b^d < 0$.

Although the structure of the multicommodity flow model should, in theory, provide tight LP relaxation bounds for the Steiner tree problem Polzin and Daneshmand [2001a,b], Goemans and Myung [1993], our computational experience shows that BPM does not provide a tight LP bound for this subproblem. In some instances, the IP and LP solution gap can be as large as 15%. It is also noted that the classical Steiner cut formulation Wong [1984], Aneja [1980] for the traditional Steiner tree problem cannot be extended to formulate the pricing subproblem, because the group variable z_b^d depends on the flow variable x_{ij}^d in Eq. (3.32), which is not explicitly shown in the cut formulation.

3.6.1.2 Valid Inequalities for the Pricing Subproblem

It is important to note that in practice most arcs in a risk group are geographically near to each other. It is very common that multiple arcs in a risk group share a common node. For example, in a telecommunication network, a group of arcs that share a common router could form a risk group with a common node; in the power distribution network, a set of power lines that share a common transmitter may form a risk group with a common node. If there exist multiple arcs starting/ending at a common node and included in the same risk group, we add a set of strong valid inequalities to tighten BPM, which is shown in the following theorem.

Theorem 5. *For any feasible integer solution to BPM in Eqs.(3.29)-(3.33), the following inequalities are valid.*

$$\sum_{(i,j)\in A_b} x_{ij}^d \leq z_b^d \qquad \forall i \in V,\ \forall b \in B,\ \forall d \in D, \tag{3.34}$$

$$\sum_{(j,i)\in A_b} x_{ji}^d \leq z_b^d \qquad \forall i \in V,\ \forall b \in B,\ \forall d \in D. \tag{3.35}$$

Proof. We only need to prove Eq.(3.34) and the result can be automatically applied to Eq.(3.35) because of symmetry. Based on the constraints in Eq.(3.31), we know that any feasible integer solution to BM will satisfy Eq.(3.36) below.

$$\sum_{(i,j)\in A_b} x_{ij}^d \leq \sum_{(i,j)\in A} x_{ij}^d \leq 1 \qquad \forall i \in V,\ \forall b \in B,\ \forall d \in D. \tag{3.36}$$

Intuitively, Eq.(3.36) simply states that the sum of all the inflow x_{ij}^d to a node i is less or equal to 1 for any path from s to d. Therefore, all the inflow x_{ij}^d from a risk group b to a node i is less or equal to 1 for any path from s to d. Because of Eqs. (3.32)-(3.33), if $x_{ij}^d > 0$, we have $z_b^d = 1$ where $(i,j) \in A_b$. If any $x_{ij}^d > 0$ in the left hand side of Eq.(3.36), we have

$$\sum_{(i,j)\in A_b} x_{ij}^d \leq 1 = z_b^d \qquad \forall i \in V,\ \forall b \in B,\ \forall d \in D.$$

If all $x_{ij}^d = 0$ in the left hand side of Eq.(3.36), we have

$$\sum_{(i,j)\in A_b} x_{ij}^d = 0 \leq z_b^d \qquad \forall i \in V,\, \forall b \in B,\, \forall d \in D.$$

Hence we know the inequalities in Eq. (3.34) are valid. □

In fact, it is easy to see Eqs.(3.34)-(3.35) are strictly stronger than constraints in Eq. (3.32). Specifically, if node i only contains one incoming or outgoing from group b, constraints in Eq. (3.32) is equivalent to constraints in Eqs. (3.34)-(3.35). Therefore, we replace Eq. (3.32) with the valid inequalities in theorem 5 in the basic pricing model, and the subsequent model is shown in Eqs.(3.37)-(3.42), denoted as the enhanced pricing model (EPM).

$$EPM: \min \sum_{(i,j)\in A} c_{ij} y_{ij} + \sum_{d\in D}\sum_{b\in B} \beta_b^d z_b^d \tag{3.37}$$

$$\text{s.t. } y_{ij} - x_{ij}^d \geq 0 \qquad \forall\ (i,j) \in A,\, \forall d \in D, \tag{3.38}$$

$$\sum_{j|(i,j)\in A} x_{ij}^d - \sum_{j|(j,i)\in A} x_{ji}^d = \sigma_i^d \quad \forall i \in V,\, \forall d \in D, \tag{3.39}$$

$$z_b^d \geq \sum_{(i,j)\in A_b} x_{ij}^d \qquad \forall i \in N,\, \forall b \in B,\, \forall d \in D, \tag{3.40}$$

$$z_b^d \geq \sum_{(j,i)\in A_b} x_{ji}^d \qquad \forall i \in N,\, \forall b \in B,\, \forall d \in D, \tag{3.41}$$

$$x_{ij}^d, y_{ij}, z_b^d \in \{0,1\} \qquad \forall (i,j) \in A,\, \forall b \in B,\, \forall d \in D. \tag{3.42}$$

Denote the optimal LP solution to BPM as LM_{BP}, and its value as $\nu(LM_{BP})$; denote the optimal LP solution to EPM as LM_{EP}, its value as $\nu(LM_{EP})$. Since LM_{EP} satisfies constraints in Eqs. (3.38)-(3.41), it must satisfy constraints in Eqs.(3.30)-(3.32). Therefore, the LM_{EP} is a feasible LP solution to BPM, and $\nu(LM_{EP}) \geq \nu(LM_{BP})$. In the following example, we show an instance with $\nu(LM_{EP}) > \nu(LM_{BP})$.

Example Consider the network in Figure 3.9. We have source node s and three destination nodes d_1, d_2 and d_3. All the arc costs are 1. There is an risk group containing $n_1 \to d_2$ and $n_2 \to d_2$ with the dual risk cost $\beta_b^{d_2} = 1$ for all destinations.

The optimal LP solution LM_{EP} is $x_{sn_1}^{d_1} = x_{n_1 d_1}^{d_1} = x_{sn_2}^{d_3} = x_{n_2 d}^{d_3} = 1$ and $x_{sn_1}^{d_2} = x_{sn_2}^{d_2} =$

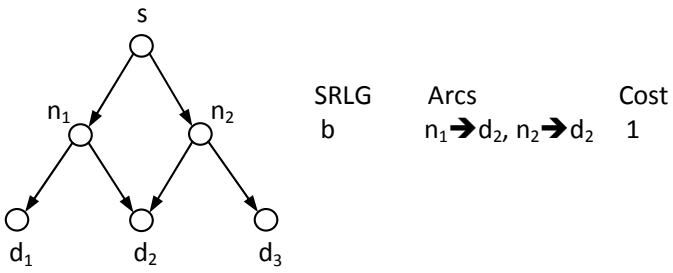


Figure 3.9: An example to show $\nu(LM_{EP}) > \nu(LM_{BM})$. We want to find a multicast tree from s to d_1, d_2 and d_3 with minimum arc cost and dual risk cost. All arc costs are 1. We have an risk group b containing two arcs $n_1 \to d_2$ and $n_2 \to d_2$, and the dual risk cost for $d2$ is $\beta_b^{d_2} = 1$. The optimal LP solution value of BPM is 5.5, and the optimal LP solution value of EPM is 6.

$x^{d_2}_{n_1d_2} = x^{d_2}_{n_2d_2} = 0.5$. We have $z_b^{d_2} \geq x^{d_2}_{n_1d_2} + x^{d_2}_{n_2d_2} = 1 \geq \max\{x^{d_2}_{n_1d_2}, x^{d_2}_{n_2d_2}\} = 0.5$. The total arc cost is 5 and total dual risk cost is 1.

The optimal LP solution LM_{BP} is $x^{d_1}_{sn_1} = x^{d_1}_{n_1d_1} = x^{d_3}_{sn_2} = x^{d_3}_{n_2d} = 1$, $x^{d_2}_{sn_1} = x^{d_2}_{sn_2} = x^{d_2}_{n_1d_2} = x^{d_2}_{n_2d_2} = 0.5$, and $z_b^{d_2} = x^{d_2}_{n_1d_2} = x^{d_2}_{n_2d_2} = 0.5$. The total arc cost is 5 and the total risk cost is 0.5. Hence, we have $\nu(LM_{EP}) = 6 > 5.5 = \nu(LM_{BM})$.

3.6.1.3 Lower Bound and Upper Bound of $RMRGD_T$

Let LM'_T be the optimal LP solution of the restricted master problem in an iteration of the column generation. It is obvious that $\nu(LM'_T)$ is an upper bound for the LP relaxation of $RMRGD_T$. We can also compute the lower bound of the $RMRGD_T$ after solving the pricing subproblems in every iteration. Specifically, define $t_{s_1} = \arg_{t \in T_{s_1}} \min c'_t$ and $t_{s_2} = \arg_{t \in T_{s_2}} \min c'_t$, where c'_t is computed using Eq. (3.28). In other words, t_{s_1} and t_{s_2} are the multicast trees with the most negative reduced costs from s_1 and s_2 respectively, and can be obtained by solving EPM optimally. The reduced cost of t_{s_1} is denoted as $c'_{t_{s1}}$, and the reduce cost of t_{s_2} is denoted as $c'_{t_{s2}}$. We have the following relations.

$$\nu(LM'_T) + c'_{t_{s1}} + c'_{t_{s2}} \leq \nu(LM_T) \leq \nu(LM'_T) \tag{3.43}$$

$\nu(LM'_T) + c'_{t_{s1}} + c'_{t_{s2}}$ is a valid lower bound because we cannot reduce $\nu(LM'_T)$ by more than $c'_{t_{s1}} + c'_{t_{s2}}$. To see this, consider the convexity constraints in Eq. (3.25). Because we can increase $w_{t_{s1}}$ and $w_{t_{s2}}$ by at most 1, the total decrease in the objective value cannot

exceed $c'_{t_{s1}} + c'_{t_{s2}}$. Furthermore, we know that $\nu(LM_{EP}) - \alpha_s \leq c_{t_s}$. We may have a weaker lower bound as follows:

$$\nu(LM'_T) + \sum_{s \in S} \{\nu(LP_{EP}) - \alpha_s\} \leq \nu(LM'_T) + c'_{t_{s1}} + c'_{t_{s2}} \leq \nu(LM_T) \leq \nu(LM'_T) \quad (3.44)$$

3.6.1.4 A Heuristics to Solve the Pricing Subproblem

Generally in column generation, one only need to find a set of good solution (multicast trees) to the pricing subproblem. Those solutions do not have to be optimal, especially in the early stage of the column generation. Any solution with negative reduced cost can be added to the restricted master problem and may improve the current LP solution. Also, it is beneficial to generate multiple columns in one iteration. This generally decreases the number of column generation iterations. It is noted that EPM is a mixed integer program, and to obtain the optimal integer solution might be time consuming. Furthermore, only one tree is obtained by solving a EPM problem. Thus, we develop a simple heuristics utilizing the LP solution provided by EPM. In the heuristics, we construct multiple multicast trees from the fractional LP EPM solutions. Note the edges used in fractional LP solution (identified by positive y variables) must be a superset of a multicast tree. Therefore, we randomly select necessary edges from the fractional solution edges to form proper multicast trees.

In this study, we propose a hybrid procedure that combines the heuristics and the MIP of EPM. First, we use heuristics to find good multicast trees with negative reduced costs. If no such trees are found, the MIP EPM is solved optimally to check the existence of the tree with negative reduced cost. If EPM provides a multicast tree with a negative reduced cost, the tree is then added to the restricted master $RMRGD_T$. Otherwise, the column generation stops and the optimal LP solution is obtained. We note that using the heuristics in the early stage drastically reduce the number of iteration of the column generation.

After iteratively generating columns based on the procedure described in the previous section, the optimal LP solution of the restricted master problem is eventually obtained. To find an integer optimal solution to $RMRGD_T$, we employ a branch-and-price algorithm,

in which a traditional branch-on-follow-on rule is used Ryan and Foster [1981], Barnhart et al. [1998a]. We use the depth-first-best-bound search algorithm to travel the branch-and-bound tree Barnhart et al. [1998b].

3.7 Computational Results

In this section, we discuss the computational study for the SRLG-diverse multicast problem. We first present the evaluation test instances used throughout the computational study. Then, we detail the computational study of the different models. In particular, we compare the LP and IP results of all three models.

3.7.1 Network Instances

Table 3.2 presents the characteristics (network topologies) of the six network instances used in this study. Each test instance is listed with the number of nodes, arcs, destinations, and the risk group information. NET1 and NET2 refer to the topologies of the Italian and US-NET networks published in Shen et al. [2005]. NET3 to NET6 are operational tier-1 backbones located across the US of a telecommunication company. Note that we preprocessed the backbone network topology to make it simple (i.e., a network without multiple arcs between the same pair of nodes) by adding new nodes and arcs. Based on the information about real fiber spans provided by the company, we identified a set of risk groups associated with interfaces, links, and fiber spans as follows. We first associate a unique risk group with each arc, interface, and fiber span that comprise the arc. A risk group may be used by multiple arcs, and likewise, multiple arcs may belong to a common risk group. Then, we remove those groups that are strict subset of other risk groups. We determined the locations of sources and destinations as follows. For example, consider test sets NET4. We mapped 40 largest cities in the US as potential destination locations and searched for the nodes that are located geographically closest to those potential endpoints. We identified 33 distinct backbone nodes as destinations for the first two test sets. Similarly, the same procedure was performed in test instance NET5 and NET6 with 60 and 100 destinations respectively. We also selected the location of two sources randomly, one from East Coast and the other one from West Coast. We set the arc cost c_{ij} as the leasing

cost for using the arc for service.

Test Set	# of Node	# of Edges	# of Destinations	# of Risk Group	Ave. Risk Group Size
NET1	21	72	10	22	2.9
NET2	24	86	10	32	2.8
NET3	38	136	10	41	8.4
NET4	178	886	33	212	7.5
NET5	178	886	60	212	7.5
NET6	178	886	100	212	7.5

Table 3.2: Test instances information.

3.7.2 Computational Settings and Implementation

In this subsection, we explain the computational experience in detail. We present the computational results of the LP and IP solution for $RMRGD_E$, $RMRGD_P$ and $RMRGD_T$. We first compare the different column generation algorithms proposed for $RMRGD_P$, and select the best algorithm. We then compare the computational results of $RMRGD_E$, $RMRGD_P$ and $RMRGD_T$.

All the experiments were implemented and performed on an Intel Dual Core 2.79GHz workstation with 1 gigabytes of memory running Windows XP. Computation times reported in the next section were obtained from the desktop's internal timing calculations, which include the time used for preprocessing, perturbation, and postprocessing. All the mathematical models and algorithms were implemented in C++. Each LP and MIP problem was solved through a callable CPLEX library version 10.0 with default settings. In order to reduce the heading-in and tailing-off effects of column generation Vanderbeck [2005], we use barrier LP solver in CPLEX to solve the LP relaxation of the restricted master of $RMRGD_P$ and $RMRGD_T$.

3.7.2.1 Computational Results for $RMRGD_P$

Table 3.3 presents the performance characteristics of the three path-based models tested on NET1 to NET4. The performance characteristics include computational time (seconds), objective function value (cost in thousand dollars) and the model size (numbers of rows and columns of the MIP models). From the table, the nearly non-dominated path generation algorithm and the mathematical programming path generation algorithm were able to find the best solutions for smaller test instances (NET1, NET2 and NET3). For the large

instance (NET4), the mathematical programming path generation algorithm obtained the best solution. It is worth mentioning that the nearly non-dominated path generation algorithm provided better solutions than the non-dominated path generation algorithm. This observation suggests that the diversification and relaxation concept plays an important role in improving the path generation process. It increased the solution quality by 2% on average for all four test instances from the non-dominated path approach.

	Column Generation	Problem Size		Solution	
Test Case	Algorithm	Cols	Rows	Cost	Time (in sec)
NET1	Probabilistic	3,611	1,680	1,950	2
	Non-Dominated	1,968	1,680	1,385	3
	Nearly Non-Dominated	2,669	1,680	1,385	2
	Math Programming	223	1,680	1,385	4
NET2	Probabilistic	3,766	2,060	2,640	17
	Non-Dominated	2,478	2,060	1,965	2
	Nearly Non-Dominated	3,375	2,060	1,955	5
	Math Programming	165	2,060	1,955	8
NET3	Probabilistic	88,728	3,150	1,328,541	1,284
	Non-Dominated	8,977	3,150	860,176	9
	Nearly Non-Dominated	75,974	3,150	841,326	135
	Math Programming	3,061	3,150	841,326	525
NET4	Probabilistic	102,117	65,538	7,227,782	4,037
	Non-Dominated	18,559	65,538	5,017,163	445
	Nearly Non-Dominated	100,242	65,538	4,888,562	4,810
	Math Programming	17,350	65,538	4,884,008	4,808

Table 3.3: Performance characteristics of different column generation algorithms for path-based model on four test instances.

Because the mathematical programming path generation algorithm provided the best solutions for NET1 to NET4, we used it in the remaining computational studies for $RMRGD_P$, and ignored other path generation algorithms.

3.7.2.2 Comparison Between Three Models

We solved LP relaxation of four models presented in the paper by enumerating all the possible variables for NET1 and NET2. When the number of the variables are too large to enumerate, we generate at least 500,000 variables. For NET3 and NET4, we use a column generation method to generate good variables (paths and trees). When solving the MIP

formulation of the problem, we stopped the CPLEX procedure if the computational time exceeded 4 hour.

Test Case	Solution Model	Problem Size Cols	Rows	LP Solution Time	Obj	LP Gap (%)	IP Solution Time	Obj	Optimality Gap (%)
NET1	$RMRGD_E$	2,024	3,356	1	1,375	0.72	2	1,385	0.00
	$RMRGD_P$	223	1,680	1	1,375	0.72	5	1,385	0.00
	$RMRGD_T$	44	222	13	1,385	0.00	14	1,385	0.00
NET2	$RMRGD_E$	2,532	4,312	1	1,955	0.00	2	1,955	0.00
	$RMRGD_P$	165	2,060	1	1,955	0.00	8	1,955	0.00
	$RMRGD_T$	30	322	10	1,955	0.00	11	1,955	0.00
NET3	$RMRGD_E$	3,812	10,778	4	823,995	2.06	9	841,326	0.00
	$RMRGD_P$	3,061	3,150	525	839,035	0.27	525	841,326	0.00
	$RMRGD_T$	56	412	29	841,326	0.12	35	841,326	0.00
NET4	$RMRGD_E$	74,240	182,162	21	4,421,522	3.54	1,941	4,582,897	0.00
	$RMRGD_P$	17,350	65,538	3,600	4,736,412	6.57	4,808	4,884,008	6.16
	$RMRGD_T$	203	6,998	1,320	4,425,280	3.44	1,652	4,583,367	0.00
NET5	$RMRGD_E$	133,532	331,440	141	6,660,709	2.07	6,945	6,801,491	0.00
	$RMRGD_P$	200,592	119,160	13,481	7,235,340	6.43	14,400	7,238,351	6.43
	$RMRGD_T$	432	12,722	4,946	6,697,647	1.56	6,573	6,803,761	0.03
NET6	$RMRGD_E$	221,372	552,012	14,400	-	-	-	-	-
	$RMRGD_P$	300,782	198,600	14,400	9,233,718	5.16	14,400	9,735,625	-
	$RMRGD_T$	675	21,202	6,946	8,573,590	2.71	9,378	8,812,521	-

Table 3.4: Comparison between different models. $RMRGD_E$ represents the edge-based model, $RMRGD_P$ represents the segregated path-based model, and $RMRGD_T$ represent the tree-based model.

In Table 3.4, we indicated, for four different models and both test cases, the problem sizes and the computational results. The problem size contains the number of columns and the number of rows in the model. The computational results contains the total computational times, objective values for LP and IP solutions, and the gaps between LP and optimal IP (computed as $\frac{OptimalIP-\nu(LM_1)}{OptimalIP}$), and IP solution optimality gap (computed as $\frac{OptimalIP-\nu(M_1)}{OptimalIP}$).

From Table 3.4, we obtained optimal solutions for NET1 and NET3 test cases in less than 5 minutes of computational time using all the methods. Within these models, $RMRGD_T$ provided the best LP-IP solution gap. For NET4, $RMRGD_E$ and $RMRGD_T$ obtained the optimal IP solution, and LP gap provided by $RMRGD_T$ is only 1/3 of the gap provided by $RMRGD_E$. However, the computational time of $RMRGD_T$ is longer than that of $RMRGD_E$, because the pricing subproblem of $RMRGD_T$ is time consuming. We did not get the optimal LP and IP solution using the $RMRGD_P$, because the column generate method of $RMRGD_P$ converges very slowly. For NET5, $RMRGD_T$ provided very tight LP bound and obtained optimal IP solution faster than $RMRGD_E$. Although $RMRGD_E$ generated the optimal IP solution within the time limit, CPLEX did not prove the optimality of the solution within the time limit. $RMRGD_P$ failed to obtain the

optimal IP solution within time limit because of the slow convergence and the poor LP bound. For NET6, $RMRGD_E$ failed to produce any solution within limited time. This could because the memory required for $RMRGD_E$ is too large for CPLEX to handle, e.g., the memory reqiured for $RMRGD_E$ is near 2G of memory, which is the limit for Windows 32-bit platform. However, $RMRGD_T$ provided the integer solution within 3% of optimality. From the computational results, we can see $RMRGD_E$ perform very good for small and mid-sized test cases (NET1-NET4), whereas $RMRGD_T$ perform better in very large test cases (NET5-NET6). This is because when the test problem is too large for $RMRGD_E$ to solve quickly, it is worth to decompose the problem into much smaller subproblems and then use $RMRGD_T$ to solve the problem.

For service providers, it is obviously very beneficial for then to increase the number of destinations due to the multicast nature. Although the increasing number of destinations will complicate RMRGD, it is worthwhile to consider such problems. For instance, we examine an average communication cost per destination as the number of destination increases based on the network topology used in NET4-NET6. The results are shown in Figure 3.10. From the figure, when the number of destination increases from 33 to 100,

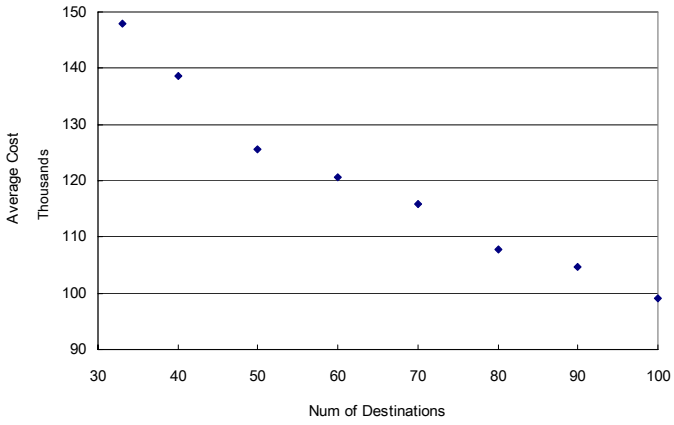


Figure 3.10: Average communication cost per destination of redundant multicast trees.

the total communication cost only doubles and the average cost per destination decrease by threefold.

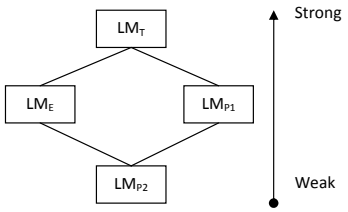


Figure 3.11: A hierarchy of linear relaxations of four formulations. M_E represents the edge-based model, M_{P1} represents the segregated path-based model, M_{P2} represents the aggregated path-based model, M_T represent the tree-based model. The models in the upper level provide better LP bounds.

3.8 Conclusion

Figure 3.11 summarizes the hierarchical relationships of all four mathematical formulations proposed here. The relaxations in the same level are equivalent. A line between two boxes means that the relaxation in the upper box are strictly stronger than the one in the lower box.

In a computational sense, a model with better LP bound may improve the computational time of an optimal integer solution dramatically. On the other hand, the difficulty of the LP relaxations of the different models could vary a lot. In $RNMRGD$, the number of possible paths increases exponentially with the number of edges in the problem, and the number of possible multicast trees increases exponentially with the number of possible paths. Therefore, it is almost impossible to enumerate all the paths and trees explicitly in the path-based and tree-based models. It is necessary and important to select the most suitable model to archive the best performance when dealing with the problems of various sizes.

Chapter 4

Flight Sequence Model for the Flight Conflict Resolving Problem

Everyday the Anchorage, Oakland and Tokyo air route traffic control centers (ARTCCs) receive a large number of requested flight plans, which detail the level, track and entry time for the flights entering the Pacific oceanic airspace. Because each airline independently optimizes its own flight plans, it is common that these requested flight plans request unbalanced usage of level and track capacities, and result in conflicting schedule that violates the Federal Aviation Administration (FAA) safety standards. The *flight conflict resolution problem* is to find a practical solution to such a common situation, and to provide a schedule that minimizes the total penalty cost of delay, level change, and track change while maintaining the FAA separation standards. In this chapter, we herein develop a computational framework to solve the flight conflict resolution problem effectively and efficiently. We propose two optimization models for this problem. The first model is a basic absolute value model (BAVM) that explicitly presents the penalty cost as a nonlinear function. The second model is a set-partitioning-based flight sequence model (FSM) that selects an optimal set of flight sequences that minimizes the total penalty cost. Because there are an exponential number of flight sequences, we propose a column generation framework with a bilinear pricing subproblem to solve the linear relaxation of the FSM and use a branch-and-price method with a new branch-on flight-assignment rule to find the integer optimal solution. Both models are tested on ten simulated instances randomly constructed based on a real dataset and compare with two other heuristics currently employed at the ARTCCs. The results show that the FSM outperforms all other methods in all test instances. We also extend the FSM to consider equity among airlines. Although the equity

concept has not been incorporated in the current ARTCC operations, such corporate decision making (CDM)-feature is necessary and critical for the future aviation systems such as 4-D trajectory system. Our study demonstrates that the proposed solution method can be extended to handle the equity constraints easily. The computation results show the proposed solution methods can solve the FSM with equity constraints within reasonable time.

4.1 Introduction

With the gradually recovering global economies, by December 2010 air travel volumes had reached a point that was 15% above the low of early 2009 and 4% above the pre-recession high of early 2008, and air freight was 1% above the pre-recession peak level of early 2008 IATA [2010]. The projected global economic growth has created an increasing demand in air transportation, which will require a significant expansion of the air traffic including the numbers of planes, passengers and cargos. Global air traffic growth between 2008 and 2027 is estimated to be at 4.2% annually for passenger traffic, and the passenger traffic within North America is predicted to increase 2.5% annually ICAO [2008]. The rapid growth of air traffic is likely to exceed the current airspace capacity under the current Federal Aviation Administration (FAA) separation standards and will require major changes in air traffic management (ATM) including operational changes and infrastructure improvement Zografos and Tsanos [2009]. There is an urgent need of new operations tools to effectively manage the increase in air traffic and accommodate the predicted traffic without compromising safety and separation standards.

This chapter addresses one of the most challenging ATM operations when the airspace is congested, and there are many conflicting flight schedules that violate the FAA safety standards (e.g., minimum vertical separation, minimum time separation). This study is motivated by the congestion of the Northern Pacific airspace. The air traffic control of the Pacific ocean falls under the jurisdiction of the Anchorage, Oakland, and Tokyo air route traffic control centers (ARTCCs). For flights that pass through the western direction of the Pacific ocean, the airline dispatchers transmit their flights' track, level and entry time requests to the Oakland ARTCC. Similarly, the Tokyo ARTCC will dispatch the flights'

track, level, and entry time for the flights that pass through the eastern direction of the Pacific ocean. Normally, airlines request the most fuel efficient *tracks* and *levels* for their flights based on the aircraft fleet type and the weather, and the *entry times* of the flights reflect the business needs of the airlines. Because the airspace capacity is limited and the safety standards must be met, it is very common to have flight conflicts and the requested plans need to be altered. If two flights request the same level on the same track and their requested entry times are within a predefined time interval set by FAA which is normally 20 minutes, the two flights are considered to be in *conflict*. The flight conflicts can be resolved by delaying one of the conflicting flights or changing one of the requested tracks or levels. Both options are undesirable. Delaying a flight will disrupt the airline's schedule (e.g., connecting flights), resource planning (e.g., aircraft utilization, crew pairing) and passengers' itineraries. This change will also impact passengers' satisfaction. Changing the track or level of a flight from what was requested might result in more fuel burn and/or arrival delays. This change is costly to the airlines as it may add up to hundreds of gallons per flight. When the conflicts occur, traffic controllers need to best resolve the conflicts by rescheduling the conflicting flights so that safety standards are met and the flight delay, track and level changes are minimized. This problem is called the *flight conflict re-scheduling problem (FCRP)*, which is one of the most common daily operational problems at the ARTCCs. The $FCRP$ can be formally defined as follows. Given a set of flights, each with requested track, level and entry time, the $FCRP$ reschedules the flight plans in order to minimize the extra fuel cost and the flight delay cost as well as satisfy the FAA airspace separation standards.

The $FCRP$ in the North Pacific airspace is especially challenging due to the airspace's fixed number of tracks and the limited entry points. This problem will be even more complicated when the traffic volume increases as the economy improves. In fact, it is expected (based on the Traffic Forecasting Group's report ICAO [2008]) that the volume will be increased by 20% within the next five years from 2008. This rapidly increased traffic volume will aggravate the $FCRP$ currently used by the ARTCCs and FAA. It has been pointed out by FAA [2010a] that although the traffic controllers in ARTCCs are able to optimize flight schedule for individual flights, there are no tools and control strategies

that can help them minimize the inefficiency of all the flights simultaneously without any conflicts. FAA has realized the urgent need of developing a trajectory control system that "utilizes integration of trajectory planning, management, and execution from strategic planning to tactical decision making" FAA [2010a].

In this chapter, we develop optimization models for the $FCRP$ to address the above mentioned concerns from the ARTCCs and FAA. In particular, we propose two mathematical models to reschedule the flight plans in order to meet the FAA safety standards and minimize the overall increased fuel cost (due to track and level changes) and the flight delay cost. In our model, we incorporate the corporate decision making (CDM) feature, particularly the equity consideration among airlines. This study is the first to develop optimization models of the $FCRP$ with CDM-feature and an efficient and practical solution approach. Specifically, we first present an explicit nonlinear formulation, called the *basic absolute value model (BAVM)*, that presents the cost objective function and the conflict between flights as a nonlinear programming problem. The $BAVM$ can be easily linearized as a mixed integer program (MIP) but solving the linearized $BAVM$ directly may not be computationally efficient. In order to efficiently solve the $FCRP$, we develop a new optimization model, called the *flight sequence model (FSM)*, which introduces a selection variable for each of the feasible flight sequences at every level or every track, and formulates the $FCRP$ as a set partitioning problem with assignment constraints. Because there are an exponential number of feasible flight sequences, we construct a column generation framework with bilinear pricing subproblems, and use a branch-and-price approach to search for the optimal integer solution. Both models are tested using ten flight traffic test instances generated randomly by a simulation module developed in Brewer [2005], and is developed based on the historical flight record of 238,778 flights in the Pacific airspace in 2002. To facilitate this model for the future aviation systems such as 4-D trajectory system, we extend the FSM to capture a CDM-feature that requires the equity consideration among airlines. It will be subsequently demonstrated that the proposed solution approach can be extended to handle the equity constraints easily.

The rest of this chapter is organized as follows. In Section 5.2, we present background and rationale of the $FCRP$ as well as other related studies in the literature. In Section

4.3, we formally define the $FCRP$ and present the $BAVM$. In Section 4.4, we present the computational framework of the FSM and elucidate the details of our column generation approach and branch-and-price method for the $FCRP$. We provide the computational detail, experimental results, and the implications in finding efficient conflict resolving policy in Section 4.5. In Section 4.6, we extend the FSM with equity constraints, and modify the column generation for the equity model. Section 4.7 summarizes the chapter and provides some concluding remarks.

4.2 Background

4.2.1 Traffic Congestion and Track Advisory in the Pacific Airspace

The Pacific oceanic airspace includes flights between three continents: North America, Asia, Australia/New Zealand and islands like Hawaii and Guam. The Pacific oceanic airspace consists of five fixed track systems and one flexible track system, called the Pacific Organized Track System (PACOTS). The PACOTS comprises airways (tracks) used primarily for flights traveling between Japan and Southeast Asia and the mainland of the United States. On average, there are 700 flights every day on the PACOTS with a little variation throughout the year. The tracks in the PACOTS are generated twice daily based on the wind and temperature forecasts. In the PACOTS, each track contains several levels in the altitude range of 29,000 - 40,000 feet. The FAA safety standards require a 1,000 feet vertical separation being maintained between two adjacent levels of the same track. By convention, the flights traveling north and east occupy the odd-numbered flight levels, and the flights traveling south and west occupy the even-numbered flight levels.

Everyday the Oakland ARTCC publishes the west bound tracks through the PACOTS and receives the information (i.e., requested track, level and entry time) of all the flights through the Pacific ocean. It then compiles all the requests, sorts and modifies them to resolve any potential conflicts by delaying flights or requesting track and/or level changes. The *Track Advisory (TA)* system FAA [2010b] is a computer program used at the Oakland ARTCC that assigns the tracks, levels, and entry times for the flights utilizing the westbound PACOTS tracks in the high traffic hours between 1900 UTC and 0100 UTC for

the airlines that subscribe to this track advisory system. The current TA system resolves the conflicts in three sequential steps. In every step, the TA system tries to delay the conflicting flights or change flight levels or tracks. After the three-step procedure, if the conflicts still remain, the conflicting flights are delayed until all the conflicts are resolved. However, the current TA system does not consider the additional fuel cost when changing the level and track of the flight. Brewer [2005] improves the conflict resolving procedure of the TA system by allowing more level change and track change options through a sequential procedure. The improved approach is called the *sequential heuristic* (SH), which has been experimentally shown to provide better, yet practical, scheduling solutions than those generated by the TA system.

4.2.2 Airspace Flow Management

There are a number of studies in the literature on airspace flow management that deal with rescheduling flight plans due to adverse weather conditions and other congestion-causing circumstances (Terab and Odoni [1993], Bertsimas and Patterson [1998], Sherali et al. [2002, 2003, 2006b, 2009]). Terab and Odoni [1993] develop an approach for rescheduling flights such that the delay caused by airspace congestions and adverse weather conditions is minimized. Specifically, the approach provides an optimal assignment of take-off times for a set of flights departing from a single airport to reduce the extent and impact of airborne delays. Bertsimas and Patterson [1998] propose an approach for reducing the impact of congestion of the US national air traffic system by ground and airborne delay. The approach minimizes the total cost of delays incurred to satisfy the constraints at the origin and destination airports as well as the pre-specified sector capacity constraints. Bertsimas and Patterson [2000] solve the air traffic flow management problem with the consideration of dynamically rerouting aircrafts using a dynamic network flow approach. Sherali et al. [2002] propose a new comprehensive airspace planning model for selecting alternative flight plans when the flight delays and diversions are necessary because of special-use airspace restrictions for the adverse weather or space port launches. The model selects a set of flight plans from a given array of alternatives for each flight such that the ground holding delays and the fuel-cost-based objective function are minimized while

workload, safety, and equity restrictions are satisfied. In Dell'Olmo and Lulli [2003], the authors present a two-level hierarchical framework to solve a centralized air traffic flow management system. The first level optimizes the entire air route network, and its solution provides the air traffic flow on each arc of the network. The second level optimizes the traffic on individual airways. If the second level solutions are consistent with the first level solutions, the centralized air traffic flow problem is solved successfully; otherwise, the congested air route arcs are feedback to the first level for reoptimization, and the solution procedure continues iteratively. Vossen et al. [2003] define a general model for the single-resource air traffic management problem with equity consideration. Ball and Lulli [2004] solve a distance-based ground delay program. Different from the traditional ground delay program, the distance-based program only includes a set of flights whose origin airports are near to the destination airport. Sherali et al. [2003, 2006b] propose a new framework to enhance the management of air traffic at the national airspace system (NAS) through the development of the airspace planning and collaborative decision-making model (APCDM). The APCDM takes into account several FAA practices such as three-dimensional probabilistic conflict analysis, workload metrics based on peak load measures, and the equity among airlines in absorbing the costs of re-routing, delays, and cancellations. Specifically, the APCDM is an optimization model that selects an optimal set of flight plans subject to sector workload, collision safety, and airline equity considerations. The APCDM can also be used to generate alternative flight plans in response to a severe weather condition or spacecraft launches. Subsequently, to improve flight efficiencies, Sherali et al. [2009] extend the APCDM by integrating slot exchange mechanisms induced by multiple ground delay programs (GDPs), continuing flights in delineating surrogates for each flight, and alternative equity concepts. Bersimas et al. [2008] propose a model to the air traffic flow management problem with the complete representation of flight's taking-off, cruising, and landing. Short computational times are reported for instances with the size of US air traffic control system.

Although such previous studies deal with flight rescheduling problems to minimize the costs of delay and changes of flight plans, unlike this study the flight plans are not represented specifically by the levels, tracks and airspace entry times of the flights. Modeling

the flights' levels, tracks, and entry times explicitly increases the complexity of the flight rescheduling problems drastically. In addition, other studies are quite focused on a single origin or a specific origin-destination pair whereas we consider the use of the Pacific airspace as a "pipeline" for the flights through the Pacific ocean.

4.3 Problem Definition and Basic Formulation

Given a set of flights, each associated with a requested schedule (track, level and entry time), the $FCRP$ seeks a flight schedule that minimizes the cost of changing the flights' levels, tracks from what they have requested and the cost of flight delay, while satisfying the FAA separation standards. As mentioned in the previous section, these standards require a specific time interval (normally 20 minutes) between two adjacent flights on any track and level. The $FCRP$ can be mathematically defined as follows. Let F be a set of flights to be scheduled, L be a set of available flight levels and T be a set of available tracks. Define a requested entry level of a flight f as l_f, a requested entry track of a flight f as t_f, and a requested entry time of a flight f as r_f. If two flights are assigned to the same track and level, there must be a separation interval P between their entry times due to the FAA separation standards. There is a delay penalty A for every minute of flight delay from the requested time in the resulting schedule. It is worth mentioning that we assume that all the flights are homogenous and the delay cost is linear. For example, the cost of a ten-minute delayed flight is equal to the cost of any two five-minute delayed flights. There is a fuel penalty B_L for every flight level change in the resulting schedule, e.g., a flight changing from level 4 to 2 has a penalty of $2B_L$. Similarly, there is a fuel penalty B_T for every track change in the schedule. In practice the penalty of flight level change should be less than the penalty of flight track change, that is, $B_L < B_T$. The objective of $FCRP$ is to schedule the flights' entry times, tracks, and levels to meet the FAA separation standards such that the total penalty cost of level and track changes and flight delay is minimized.

We define the following set of variables. For a flight f, z_f is a decision variable of scheduled entry time, y_f^L is a scheduled flight level and y_f^T is a scheduled flight track. A flight f can only be assigned to a subset of tracks $T_f \subseteq T$ and a subset of levels $L_f \subseteq L$

because it may be impractical or infeasible to assign a flight to some tracks and/or levels due to aircraft's capability and takeoff weight. The explicit nonlinear $BAVM$ of the $FCRP$ is given by

$$(BAVM) \quad \min \sum_{f \in F} A(z_f - r_f) + B_L|y_f^L - l_f| + B_T|y_f^T - t_f| \tag{4.1}$$

$$\text{s.t. } P|y_{f_i}^L - y_{f_j}^L| + P|y_{f_i}^T - y_{f_j}^T| + |z_{f_j} - z_{f_i}| \geq P \quad \forall\ f_i,\ f_j \in F, \tag{4.2}$$

$$r_f \leq z_f \qquad \forall\ f \in F, \tag{4.3}$$

$$y_f^L \in L_f,\ y_f^T \in T_f, \qquad \forall\ f \in F. \tag{4.4}$$

The objective function in Eq. (4.1) minimizes the total penalty cost, which contains three elements: flight delay, level change and track change. The constraints in Eq. (4.2) are for the FAA separation standards, which ensure the minimum time P between the entry times of any two flights at the same level on the same track. The constraints in Eq. (4.3) ensure that a scheduled entry time z_f of a flight f is after its requested entry time (no early entry allowed). The constraints in Eq. (4.4) are the integral constraints for level and track variables. Note that the $BAVM$ is a nonlinear programming formulation that contains absolute value expressions in the objective function and constraints. It can be linearized to an MIP using a traditional absolute value linearization technique. The space complexity of linearized $BAVM$ is large, approximately $2|F|^2 + 5|F|$ variables and $4.5|F|^2 + 4|F|$ constraints (with $4|F|^2$ Big-M constraints). Solving the linearized BAVM directly may not be computationally efficient; therefore, we develop the FSM to heuristically solve the $FCRP$.

4.4 Flight Sequence Model and Column Generation Approach

In this section, we formulate the $FCRP$ as a set partitioning problem with decision variables of flight sequences and an objective function to find a subset of feasible flight sequences for every level and track at the minimum cost. The formulation is called the *flight sequence model (FSM)*, which is similar to the flight string model proposed in [Barnhart et al., 1998b]. A flight sequence is defined as a series of flights at any track and level that satisfies the FAA separation standards. Note that there is a cost of delay and/or level and

track changes associated with each sequence. The FSM can be formally defined as follows. Let S_{lt} be a complete set of possible flight sequences at a level $l \in L$ on a track $t \in T$. Define S as a set of all the possible flight sequences, and $S = \bigcup_{l \in L} \bigcup_{t \in t} S_{lt}$. Define the penalty cost of a flight sequence $s \in S_{lt}$ as $c_s = \sum_{f \in F_s} B_L|l - l_f| + B_T|t - t_f| + A(z_f - r_f)$. Define the binary parameter $\alpha_s^f = 1$ if flight f is included in sequence s, and 0 otherwise. For every flight sequence s, we introduce a binary decision variable x_s to be 1 if s is included in the solution schedule, and 0 otherwise. The mathematical formulation of the FSM is then given by

$$(FSM) \quad \min \sum_{s \in S} c_s x_s \tag{4.5}$$

$$\text{s.t.} \sum_{s \in S} \alpha_s^f x_s = 1 \qquad \forall\ f \in F, \tag{4.6}$$

$$\sum_{s \in S_{lt}} x_s \leq 1 \qquad \forall\ l \in L, \forall\ t \in T, \tag{4.7}$$

$$x_s \in \{0, 1\} \qquad \forall\ s \in S. \tag{4.8}$$

The objective function in Eq. (6.1) minimizes the total penalty cost of the selected flight sequences. The constraints in Eq. (6.2) are set partitioning constraints ensuring that each flight is included in only one of the selected sequences. The constraints in Eq. (6.3) ensure that at most one flight sequence is selected for every track and level. The constraints in Eq. (4.8) are the binary variable constraints. There are $|S|$ variables and $|F| + |L||T|$ constraints in this model. Note that enumerating the complete set of the flight sequences S is intractable and impractical because the number of possible flight sequences increases exponentially with the number of flights. We herein develop a column generation framework to generate good flight sequences, and use a branch-and-price method to obtain integer feasible solutions to the FSM.

Column generation has been widely and successfully applied to many large-scale, real-life optimization problems where the number of variables is too large to enumerate explicitly [Desrocsiers et al., 1984, Lubbecke and Mesrosiers, 2005, Wolsey, 1998]. The column generation framework developed here for the FSM can be described as follows. First a subset of feasible flight sequences is constructed by delaying all the conflicting flights. The

restricted master problem of the FSM is then solved, and the dual cost of each constraint is calculated. To improve the restricted master (i.e., generate more columns), we solve the pricing subproblems based on the dual cost information, and generate new flight sequences with negative reduced cost. These flight sequences are then added to the restricted master problem, and the updated restricted master problem is then resolved. These iterative procedures are repeated until no new flight sequences with negative reduced cost is found, which implies that the current LP solution is optimal. The flow chart of our computational framework for the FSM is shown in Figure 4.1.

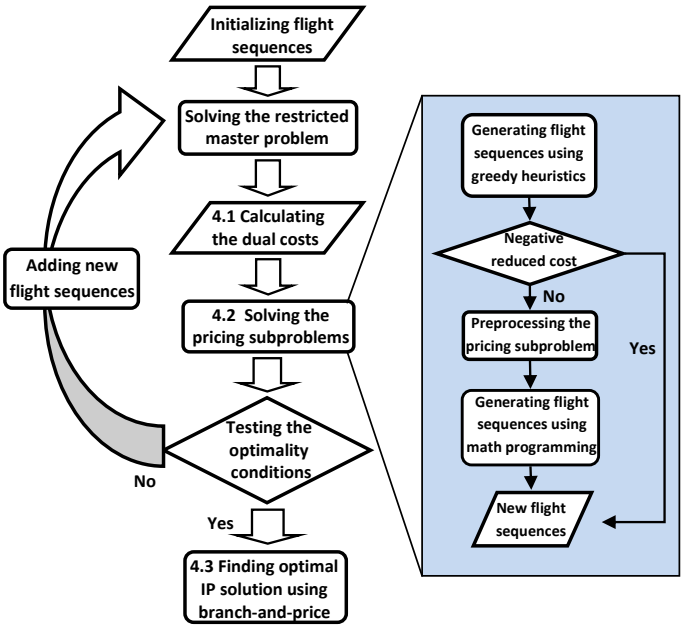


Figure 4.1: Flow Chart of the Computational Framework for the FSM.

4.4.1 Calculating the Dual Cost

After the restricted master problem of the FSM is solved, we obtain dual variable π_f associated with the partitioning constraint in Eq. (6.2) for flight $f \in F$. Because the equality sign (=) in Eq. (6.2) can be replaced by $\geq$ sign, we know π_f is non-negative. Similarly, we obtain non-positive dual variable λ_{lt} associated with the constraint in Eq. (6.3) for track $t \in T$ and level $l \in L$. The reduced cost $\bar{c}_s$ of flight sequence $s \in S_{lt}$ can be

calculated by

$$\overline{c}_s = c_s - \sum_{f \in F_s} \pi_f - \lambda_{lt}.$$

Here, F_s is the set of flights contained in the sequence s. As we discussed in Section 4.3, the cost c_s of flight sequence $s \in S_{lt}$ is the summation of the total penalty cost of all flights in s, and it is given by

$$c_s = \sum_{f \in F_s} \left(B_L|l - l_f| + B_T|t - t_f| + A(z_f - r_f) \right).$$

Denote $c_{l_f} = B_L|l - l_f|$ as the level change penalty and $c_{t_f} = B_T|t - t_f|$ as the track change penalty of flight f. Hence, the reduced cost of flight sequence $s \in S_{lt}$ can be rewritten as

$$\overline{c}_s = \sum_{f \in F_s} \left(c_{l_f} + c_{t_f} + A(z_f - r_f) \right) - \sum_{f \in F_s} \pi_f - \lambda_{lt} = \sum_{f \in F_s} \left(c_{l_f} + c_{t_f} - \pi_f + A(z_f - r_f) \right) - \lambda_{lt}. \tag{4.9}$$

Consequently, the pricing problem is to find a flight sequence with a negative reduced cost, i.e., $\overline{c}_s < 0$, which can be found by computing $\min_{s \in S_{lt}} \overline{c}_s$. Let us consider flight $f \in F$ at level l on track t. We first compute c_{l_f} and c_{t_f}. If $c_{l_f} + c_{t_f} - \pi_f \geq 0$, we know it is not beneficial to include flight f in sequence s because $c_{l_f} + c_{t_f} - \pi_f + A(z_f - r_f) \geq 0$. Otherwise, the flight f may be included in sequence $s \in S_{lt}$ with $z_f \in [r_f, r_f + \frac{\pi_f - c_{l_f} - c_{t_f}}{A})$. For simplicity, we denote $d_f = r_f + \frac{\pi_f - c_{l_f} - c_{t_f}}{A}$, and the reduced cost contributed by flight f is defined by

$$c_{l_f} + c_{t_f} - \pi_f + A(z_f - r_f) = A\left(\frac{c_{l_f} + c_{t_f} - \pi_f}{A} + z_f - r_f \right) = A(z_f - d_f).$$

Hence, d_f can be interpreted as the due time of flight f, and $(d_f - z_f)$ can be viewed as the earliness of flight f with respect to its entry time z_f of the airspace. Finally, the reduced cost in Eq. (4.9) can be rewritten as

$$\overline{c}_s = \sum_{f \in F_s} A(z_f - d_f) - \lambda_{lt}. \tag{4.10}$$

Note that λ_{lt} is a constant for every level $l \in L$ on every track $t \in T$, and d_f can be computed accordingly for every flight $f \in F$. Thus the pricing subproblem for level l and

track t becomes a problem of selecting a set of flights that maximizes the total earliness of flight sequence s.

4.4.2 Solving the Pricing Subproblems

In every iteration of the column generation framework, we need to solve $|L| \times |T|$ pricing subproblems, each to obtain an optimal flight sequence with the most negative reduced cost. The process is repeated until there are no flight sequences with negative reduced cost found at any level on any track, which implies that the current LP solution of the FSM is optimal. In practice, we need to determine the assignment of flights for every level l and track t (a selection problem) as well as the sequence of flights (a sequencing problem). In this study, we propose a necessary condition for optimal flight sequence so that we only need to solve the flight selection problem and significantly reduce the complexity of the pricing subproblem. The subsequent flight selection problem is formulated as a bilinear programming problem and can be linearized as an MIP problem. Then we propose a hybrid approach, which combines a greedy insertion heuristic and the linearized MIP subproblem, to solve the pricing subproblem efficiently. We also propose an efficient pre-processing method using clustering concept to reduce the size of the pricing subproblems.

4.4.2.1 Necessary Condition for Optimal Flight Sequences.

As noted earlier, in each iteration of the column generation framework, flights are selected and sequenced at each level on each track so that the reduced cost of the sequence is minimized. The problem of finding the best flight sequence at each level on each track can be viewed as a generalization of single-machine scheduling problem, in which a level on a track is viewed as a machine, and a flight is considered as a job. The release date of flight f is defined by r_f because no flight is allowed to enter the airspace before its requested time. The due date of each job is defined as d_f+P, where P is considered as a constant processing time for all jobs. The earliness of a flight f is defined as $d_f + P - z_f - P = d_f - z_f > 0$. Note that we do consider the level change and track change penalty costs here because those are included in the calculation of d_f. Here we are only interested in finding a flight sequence maximizing the total earliness. Apparently, our problem is more general than the

single-machine scheduling problem because not every job has to be selected and sequenced as in the single-machine scheduling problem. Instead, we only select the jobs with the largest cumulative earliness. In this work, we propose the following necessary condition for optimal flight sequences, called the *earliest flight request time rule*.

Theorem 6 (Earliest Flight Request Time Rule)**.** *A flight sequence at level $l \in L$ on track $t \in T$ is optimal only if it contains a series of flights arranged in an ascending order of request time r_f.*

Proof. By contradiction, we suppose that there exists an optimal schedule containing a track and level that do not satisfy Theorem 6. In this schedule there must be at least two adjacent flights, say f_i followed by f_j (denoted as $f_j \rightarrow f_i$), such that $r_{f_i} < r_{f_j}$. Define the earliest possible entry time for f_i and f_j as E, that is, all the previous flights have enter the track t and level l before time $E - P$. We know $z_{f_j} = \max\{E, r_{f_j}\}$ and $z_{f_i} = \max\{z_{f_j} + P, r_{f_i}\} = \max\{E + P, r_{f_j} + P, r_{f_i}\}$. Therefore, cost of these two flights $f_j \rightarrow f_i$ is

$$\begin{aligned}
C_{opt} &= B_L|y^L_{f_i} - l| + B_T|y^T_{f_i} - t| + A(z_{f_i} - r_{f_i}) + B_L|y^L_{f_j} - l| + B_T|y^T_{f_j} - t| + A(z_{f_j} - r_{f_j}) \\
&= B_L\left(|y^L_{f_i} - l| + |y^L_{f_j} - l|\right) + B_T\left(|y^T_{f_i} - t| + |y^T_{f_j} - t|\right) + \\
&\quad A\left(\max\{E, r_{f_j}\} - r_{f_j} + \max\{z_{f_j} + P, r_{f_i}\} - r_{f_i}\right) \\
&= C + A\left(\max\{E, r_{f_j}\} - r_{f_j} + \max\{E + P, r_{f_j} + P, r_{f_i}\} - r_{f_i}\right).
\end{aligned}$$

Here, $C = B_L\left(|y^L_{f_i} - l| + |y^L_{f_j} - l|\right) + B_T\left(|y^T_{f_i} - t| + |y^T_{f_j} - t|\right)$.

If we switch the order of f_i and f_j, we have the cost of new sequence $f_i \rightarrow f_j$ as follows.

$$C_{swt} = C + A\left(\max\{E, r_{f_i}\} - r_{f_i} + \max\{E + P, r_{f_i} + P, r_{f_j}\} - r_{f_j}\right).$$

By computing $(C_{opt} - C_{swt})$, we have

$$\begin{aligned}
C_{opt} - C_{swt} = A&\left(\max\{E, r_{f_j}\} - \max\{E, r_{f_i}\}\right) \\
&+A\left(\max\{E + P, r_{f_j} + P, r_{f_i}\} - \max\{E + P, r_{f_i} + P, r_{f_j}\}\right).
\end{aligned}$$

Since $r_{f_i} < r_{f_j}$, we know $\max\{E, r_{f_j}\} - \max\{E, r_{f_i}\} \geq 0$, and equality (=) only happens when $r_{f_i} < r_{f_j} \leq E$. Also, we have

$$\begin{aligned}&\max\{E+P, r_{f_j}+P, r_{f_i}\} - \max\{E+P, r_{f_i}+P, r_{f_j}\}\\ &\quad = \max\{E+P, r_{f_j}+P\} - \max\{E+P, r_{f_i}+P, r_{f_j}\} \geq 0,\end{aligned}$$

where and equality (=) only happens when $r_{f_i} < r_{f_j} \leq E$. Therefore, we know $C_{opt} < C_{swt}$ when $E < r_{f_j}$, which contradicts the assumption that C_{opt} is the optimal schedule. This completes the proof. □

Based on Theorem 6, we can eliminate flight sequences that do not satisfy the necessary condition of optimal flight sequences. Most importantly, it drastically simplifies the flight selection problem, which selects x flights to be scheduled at level l on track t yet there are $x!$ possible flight sequences. Theorem 6 provides a criterion of the best sequence out of $x!$ possible. In addition, one can use this result to estimate an upper bound of the objective of the FSM as $UpperBound = \sum_{t\in T}\sum_{l\in L} A \times P \times \frac{|F_{lt}|(|F_{lt}|-1)}{2} - \sum_{f\in F_{lt}} r_f$, where F_{lt} is the set of flights such that $l_f = l$ and $t_f = t$.

Finally, it is worth mentioning that Theorem 1 is only true with the assumptions that all the flights are homogenous and the delay cost is linear. Without these two assumptions, Theorem 1 may not be valid. For example, as we will present in Section 4.6, when the equity among airlines is considered in the model, the homogenous flight assumption is not valid. Therefore, Theorem 1 is no longer the necessary condition for the optimal solution with equity considerations.

4.4.2.2 Bilinear Pricing Problem.

Given a set of flights, an optimal flight sequence is uniquely defined as an ascending order of r_f (using the a result of Theorem 6). Hence, the objective of the pricing subproblem is to select a set of flights to be included in the optimal sequence such that the reduced cost is minimized. We can now formally define the pricing subproblem as follows. Define a set of flights F_{lt} at level $l \in L$ on track $t \in T$. Each flight is associated with request entry time r_f and due time $d_f = r_f + \frac{\pi_f - c_{l_f} - c_{t_f}}{A}$. We denote the flights as $f_1, f_2, \ldots, f_k$, where

$k = |F_{lt}|$, such that $r_{f_1} < r_{f_2} < \cdots < r_{f_k}$. Define binary variable u_f such that $u_f = 1$ if flight f is selected in the optimal sequence, and $u_f = 0$ otherwise. The pricing subproblem is to select a subset of F_{lt} and construct a flight sequence with the maximum earliness defined by Eq. (4.10). The bilinear programming of the pricing subproblem is given by

$$\max \quad \sum_{f \in F_{lt}} u_f(d_f - z_f) \tag{4.11}$$

$$\text{s.t.} \quad r_f \le z_f + (1 - u_f)M \qquad \forall f \in F_{lt}, \tag{4.12}$$

$$z_{f_i} + P - (1 - u_{f_i})M \le z_{f_j} + (1 - u_{f_j})M$$

$$\forall f_i, f_j \in F_{lt} \text{ and } r_{f_i} \le r_{f_j}, \tag{4.13}$$

$$u_f \in \{0, 1\},\ z_f \ge 0 \qquad \forall\ f \in F_{lt}. \tag{4.14}$$

The objective function in Eq. (4.11) maximizes the total earliness of the selected flights. The constraints in Eq. (4.12) ensure that the start time z_f of flight f is not earlier than the request entry time r_f when flight f is selected. The constraints in Eq. (4.13) ensure the separation time between two flights f_i and f_j to be at least P. The constraints in Eq. (4.14) are the binary and non-negative constraints of decision variables. Note that the objective function contains a bilinear term. We employed the linearization technique developed in Chaovalitwongse et al. [2004] to formulate this problem as an MIP problem, and define an additional continuous variable w_f as the earliness time of flight f. The linearized formulation of bilinear pricing subproblem is given by

$$\max \sum_{f \in F_{lt}} w_f \tag{4.15}$$

$$\text{s.t. } r_f \le z_f + (1 - u_f)M \qquad \forall f \in F_{lt}, \tag{4.16}$$

$$z_f \le M u_f \qquad \forall f \in F_{lt}, \tag{4.17}$$

$$z_{f_i} + P - (1 - u_{f_i})M \le z_{f_j} + (1 - u_{f_j})M \quad \forall f_i, f_j \in F_{lt} \text{ and } r_{f_i} \le r_{f_j}, \tag{4.18}$$

$$w_f \le M u_f \qquad \forall f \in F_{lt}, \tag{4.19}$$

$$w_f \le d_f - z_f \qquad \forall f \in F_{lt}, \tag{4.20}$$

$$u_f \in \{0, 1\},\ z_f, w_f \ge 0 \qquad \forall\ f \in F_{lt}. \tag{4.21}$$

The objective function in Eq. (4.15) maximizes the total earliness from the selected flights. The constraints in Eq. (4.16) is the same as the constraints in Eq. (4.12). The constraints in Eq. (4.17) ensure if flight f is not selected, $z_f = 0$. The constraints in Eq. (4.18) is the same as the constraints in Eq. (4.13). The constraints in Eqs. (4.19)-(4.20) ensure the earliness of flight f is equal to $d_f - z_f$ if flight f is included in the optimal sequence, and 0 otherwise. The constraints in Eq. (4.21) are the binary and non-negative constraints of decision variables.

We can tighten the linearized model in Eqs. (4.15)-(4.21) by replacing the Big-Ms with more realistic estimations. The result formulation is shown as follows:

$$\max \sum_{f \in F_{lt}} w_f \tag{4.22}$$

$$\text{s.t. } r_f \leq z_f + (1 - u_f) r_f \qquad \forall f \in F_{lt}, \tag{4.23}$$

$$z_f \leq d_f u_f \qquad \forall f \in F_{lt}, \tag{4.24}$$

$$z_{f_i} + P - (1 - u_{f_i})(P + d_{f_i}) \leq z_{f_j} + (1 - u_{f_j})(P + d_{f_i}) \qquad \forall f_i, f_j \in F_{lt} \text{ and } r_{f_i} \leq r_{f_j}, \tag{4.25}$$

$$w_f \leq (d_f - r_f) u_f \qquad \forall f \in F_{lt}, \tag{4.26}$$

$$w_f \leq d_f - z_f \qquad \forall f \in F_{lt}, \tag{4.27}$$

$$u_f \in \{0, 1\},\ z_f, w_f \geq 0 \qquad \forall\ f \in F_{lt}. \tag{4.28}$$

We can always select a set of flights that $c_{l_f} + c_{t_f} - \pi_f < 0$ for level l and track t as the input of the model in Eqs. (4.11)-(4.14). However, there are $|L| \times |T|$ pricing subproblems need to be solved in every iteration of the column generation, and each linearized MIP subproblem might not be solved to optimality easily and it can be time consuming. In the following subsections, we propose a clustering preprocessing method to reduce the size of pricing subproblem and a greedy heuristic to solve the pricing subproblems efficiently.

4.4.2.3 Flight Clustering

To speedup the computational performance of the pricing model in Eqs. (4.11)-(4.14), we develop a preprocessing method called flight clustering to reduce the size of pricing subproblems. Consider a set of flight candidates $\{f_1, f_2, ..., f_i, f_{i+1}, ..., f_k\}$ in a pricing subproblem such that r_f is sorted in an ascending order based on the result of Theorem 6. Define the latest start time of flight f_i as e_{f_i}. We have $e_{f_1} = r_{f_1}$ and $e_{f_{i+1}} = \min\{\max_{j\leq i}\{e_{f_j} + P, r_{f_{i+1}}\}, d_{f_{i+1}}\}$. This sequence of flights can be clustered into two subsequences $\{f_1, \ldots, f_i\}$ and $\{f_{i+1}, \ldots, f_k\}$ if there exists flight f_i and f_{i+1} such that $e_{f_j} + P \leq r_{f_{i+1}}, \forall j \leq i$. The optimal sequence obtained from flights in $f_1, \ldots, f_k$ is the optimal sequence from flights in $f_1, \ldots, f_i$ followed by optimal sequence from flights in $f_{i+1}, \ldots, f_k$. In Figure 4.2, we show an example of clustering flights in three subse-

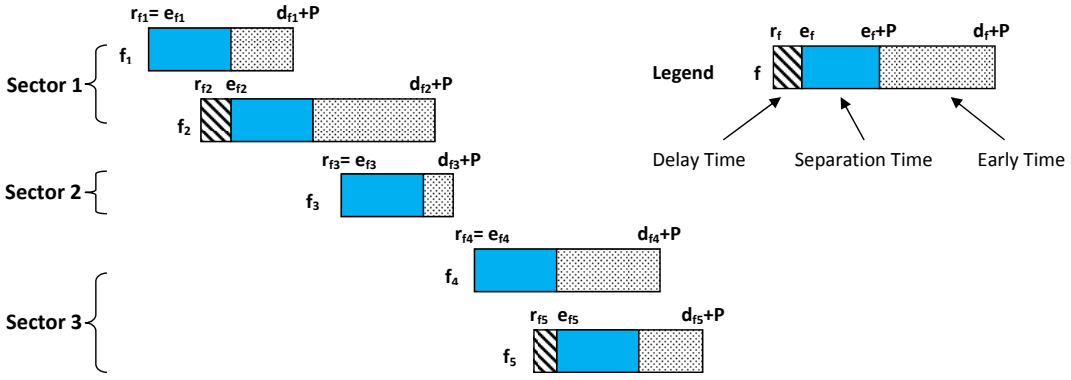


Figure 4.2: Grouping flights $\{f_1, ..., f_5\}$ into three clusters $\{f_1, f_2\}$, $\{f_3\}$ and $\{f_4, f_5\}$

quences. In particular, we have five flights $f_1, \ldots, f_5$ such that $r_{f_1} < r_{f_2} < r_{f_3} < r_{f_4} < r_{f_5}$ and $d_{f_1} < d_{f_2} < d_{f_3} < d_{f_4} < d_{f_5}$. We know $e_{f_1} = r_{f_1}$ and $e_{f_1} + P > r_{f_2}$. Therefore, flight f_1 and flight f_2 should be assigned the same cluster. However, $e_{f_2} + P = \min\{\max\{e_{f_1} + P, r_{f_2}\}, d_{f_2}\} + P = e_{f_1} + 2P < r_{f3}$. Therefore, we know flight f_2 and flight f_3 belong to the different clusters.

4.4.2.4 Hybrid Approach for Pricing Subproblem

Generally in column generation, one only needs to find a set of good solutions (flight sequences) to the pricing subproblems. Those solutions do not have to be optimal in the early stage of the column generation. Any solutions with negative reduced cost can be

added to the restricted master FSM and may improve the current LP solution. Thus we develop a simple greedy insertion algorithm to generate good flight sequences efficiently. In the greedy insertion algorithm, we insert a flight with the largest earliness $d_f - z_f$ into an initial sequence. We repeat this insertion procedure until there is no other flights with a positive earliness.

In this study, we propose a hybrid procedure that combines the greedy insertion heuristic and the linearized MIP subproblem model. First, the greedy heuristic is used to find flight sequences with negative reduced costs. If no such flight sequences are found, the MIP model in Eqs. (4.15)-(4.21) is solved optimally to check the existence of the flight sequence with a negative reduced cost. If the MIP subproblem model provides a flight sequence with a negative reduced cost, the sequence is then added to the restricted master FSM. The column generation framework iterates until the MIP subproblem model does not provide any flight sequence with a negative reduced cost. We note that using the greedy insertion heuristic in the early stage reduces the computational time drastically. Most importantly, the hybrid method guarantees the optimality of LP solution when the column generation is terminated.

4.4.3 Branch-and-Price Approach

After iteratively generating columns based on the procedure described in the previous section, the optimal LP solution of the restricted master problem is eventually obtained. To find an integer optimal solution to the FSM, we employ a branch-and-price algorithm, in which a good branching rule is crucial. We notice that the FSM is a set partitioning model with assignment constraints; therefore, it is intuitive to consider the branch-on-follow-ons rule for this problem [Ryan and Foster, 1981]. Barnhart et al. [1998a], Desrocsiers et al. [1984], Vance et al. [1997] have shown the effectiveness of this branching rule in solving large-scale set partitioning problems. The branch-on-follow-ons rule can be implemented here by constraining two flights, f_i and f_j, to be at the same level on the same track for one branch and constraining the complementary for the other branch. Nevertheless, it does not necessarily provide an integer solution in our case because the FSM involves multiple flight sequences that may contain both f_i and f_j and they can be in different tracks and/or

levels. For instance, a final solution found by the branch-on-follow-ons rule may contain a flight sequence with $f_i \rightarrow f_j$ at level 1 with a solution value of 0.5 and another sequence with $f_i \rightarrow f_j$ at level 2 with a solution value of 0.5. For this reason, we develop a new branching rule, called the *branch-on flight-assignment rule*, that exploits the property of flight assignment constraints. Given a fractional solution to the FSM, we identify a flight f being assigned to more than one level. This branch-on flight-assignment rule enforces flight f to be at level l on track t in one branch, and enforces the complimentary in other branch. In order to include flight f in the first branch, we can set a very high profit of including flight f in the subproblem. On the other hand, to exclude flight f in the other branch, we can set a penalty of including flight f in the subproblem. We then use a depth-first-best-bound-depth-first node choice rule proposed in Barnhart et al. [1998b]. Starting from the root node of the branch-and-bound tree, we use the depth first strategy to find a feasible integer solution. Then we choose the node with the best bound to start the depth first search until a feasible solution is found.

4.5 Computational Study

This section describes the characteristics of the realistic test instances used in this study. We then provide the detail of our computational settings and implementation. We study the computational experience of the FSM and compare the results of the $BAVM$ and FSM with those of the simulated TA system, currently used in the Oakland control center, as well as the SH approach proposed in Brewer [2005]. Finally, we investigate the effect of different flight planning policies through a sensitivity analysis of the penalty cost and alternative cost structure of flight schedules.

4.5.1 Test Instances

The realistic test instances used in this study are created by a simulation module developed in Brewer [2005]. This simulation module utilizes the historical flight record of 238,778 flights in the Pacific airspace in 2002 to randomly generate different scenarios of realistic flight traffic. The historical data is provided by the ARTCCs, referred to as ETMS data, and the Japan civil aviation bureau (JCAB). In this study, we generate ten test instances

of flight traffic with characteristics shown in Table 4.1. The characteristics include the number of flights, the number of tracks and levels, and the schedule time horizon. In particular, instances $s1$ to $s3$ are small test instances with 80 to 120 flights. Instances $m1$ to $m4$ are medium test instances containing more than 200 flights. Instances $d1$ to $d3$ are large instances of daily flight schedules.

Test Instance	Num of Flights	Num of Tracks	Start Time	End Time
$s1$	80	4	0:00	24:00
$s2$	86	8	10:00	12:00
$s3$	124	5	0:00	17:00
$m1$	200	4	0:00	24:00
$m2$	210	6	0:00	24:00
$m3$	212	6	0:00	24:00
$m4$	215	5	0:00	24:00
$d1$	415	12	0:00	24:00
$d2$	415	12	0:00	24:00
$d3$	422	12	0:00	24:00

Table 4.1: Characteristics of test instances

4.5.2 Computational Settings

In this subsection, we explain the computational experience in detail. We first present the computational results of the LP and IP solution for the FSM. After that, we compare the results of flight sequence model with other solution methods. Finally, we change the cost function of the airspace flight scheduling problem and show its effects on the final schedules.

All the experiments are implemented and performed on an Intel Dual Core 2.79 GHz workstation with 1 gigabytes of memory running Windows XP. Computational times reported in the next section are obtained from the desktop internal timing calculations, which include the time used for preprocessing, perturbation, and postprocessing. All the mathematical modeling and algorithms are implemented in C++. Each LP and MIP problem is solved through a callable CPLEX library version 10.0 with a default setting.

In this study, we set the delay penalty $A = 1$ for one minute of delay, level change penalty $B_L = 16$ and track change penalty $B_T = 31$ in order to make the optimization model allow at most 15/30-minute delay before changing the flight's level/track. The 15/30-minute delay is suggested by field operators at the Anchorage and Oakland ATCCs. In this study, we also explore different decision policies by changing the penalty settings

for A, B_L and B_T. We also make an assumption that any flight can be assigned to any track and at any level, i.e., $L_f = L, T_f = T$, $\forall f \in F$. As we mentioned previously, in real life T_f is a true subset of T for all flights and L_f may be a subset of L for some fleet types. We note that this assumption only makes the test instances more difficult because it allows more feasible flight sequences.

4.5.3 Column Generation and Branch-and-Price Implementation

To obtain an initial solution to the restricted master FSM in the column generation framework, we first create a set of feasible columns of flight sequences by delaying all conflicting flights for every level $l \in L$ and every track $t \in T$. The LP relaxation solution of the restricted master FSM in each iteration is obtained by using the barrier LP solver in CPLEX to reduce the heading-in and tailing-off effects [Vanderbeck, 2005]. Subsequently, the dual cost is calculated from the LP relaxation, and the pricing subproblem was preprocessed by clustering flights. The preprocessed pricing model presented in Eqs. (4.11)-(4.14) is then linearized and solved by the CPLEX MIP solver using a default setting.

4.5.4 Solution Characteristics of the FSM

4.5.4.1 LP Solution.

We first present a comparison of CPU time and computational effort needed to solve the LP relaxation of the FSM for ten test cases with different pricing subproblem methods. In particular, we evaluate the benefits of solving the LP pricing subproblem with hybrid method as opposed to only the pricing model in Eqs. (4.11)-(4.14) or the greedy heuristic. We first solve the pricing model presented in every iteration of the column generation. Then, we use the greedy heuristic to solve the pricing subproblem. Finally, we apply the hybrid subproblem method. We indicate, for three subproblem methods and all ten test cases, the total computational time, objective value as well as the number of column generation iterations (Num of Iterations) and the number of columns generated (Num of Cols). We can see from Table 4.2, for the small test cases, all three methods produced optimal solutions in a short time. However, by using only the pricing model, we cannot get good solutions in one hour time for the large test cases (Cases d1-d3). One can notice that

Test Case	Pricing Method	Obj Value	Comp Time	Num of Iters	Num of Cols
s1	Math	307.00	112	211	1,147
	Greedy	309.00	14	263	1,288
	Hybrid	307.00	32	413	1,645
s2	Math	726.00	548	232	1,861
	Greedy	726.00	29	295	3,192
	Hybrid	726.00	116	305	3,220
s3	Math	957.00	1,151	322	3,341
	Greedy	960.56	123	483	5,471
	Hybrid	956.00	217	596	5,547
m1	Math	725.00	2,396	588	5,716
	Greedy	725.00	455	950	9,556
	Hybrid	725.00	477	952	9,332
m2	Math	771.00	3,600	591	5,242
	Greedy	771.00	279	676	6,139
	Hybrid	771.00	971	1,159	7,955
m3	Math	951.00	3,600	562	4,721
	Greedy	951.00	282	678	6,541
	Hybrid	951.00	471	901	7,452
m4	Math	1,255.00	3,600	629	5,801
	Greedy	1,041.20	2,475	2,100	16,512
	Hybrid	1,041.00	2,735	2,112	16,313
d1	Math	2,071.35	3,600	787	7,032
	Greedy	1,926.00	667	1,113	14,415
	Hybrid	1,926.00	3,051	1,072	15,157
d2	Math	1,951.50	3,600	814	7,260
	Greedy	1,763.00	584	812	11,830
	Hybrid	1,760.00	3,600	1,065	13,769
d3	Math	1,778.00	3,600	648	6,686
	Greedy	1,686.50	377	772	12,854
	Hybrid	1,686.00	919	2,269	16,126

Table 4.2: Computational results for LP relaxation of the FSM using three different subproblem methods

the number of iterations and the number of columns generated using the pricing model are much less than the other two methods for the large problems (less than 50%). This is because the mathematical model of the subproblem is computationally expensive. On the other hand, by using only the greedy algorithm in the pricing problem, we obtain optimal or near optimal solutions in very short time for all test cases. Finally, the hybrid algorithm can provide the optimal solutions in an acceptable longer time than the greedy heuristic for all test cases. It is worth noting that the number of columns generated by hybrid method is only slightly larger than the number of columns by the greedy heuristic. This indicates the greedy heuristic submodule in the hybrid method serves more in improving the objective value, whereas the mathematical model submodule in the hybrid method is

more in proving the optimality of the solution.

4.5.4.2 Preprocessed Subproblems.

We investigate the benefit of the clustering method in reducing the sizes of pricing subproblems. We record the sizes of the pricing subproblems before and after the clustering methods (Table 4.3). Here the problem size is defined by the maximum and average number of flight sequences in Eqs. (4.11)-(4.14). Note that the statistics are collected after the LP relaxation of the FSM is solved optimally. From Table 4.3, the sizes of the pricing problems for mid-size and large instances decrease drastically after clustering method. In fact, in mid-size and large test cases, the maximum sizes of the subproblems are reduced by more than 30%. For instance, the largest subproblem size of all test cases is decreased from 90 to 61, and the average subproblem size of large-scale problems (instances $d1 - d3$) is decreased from 14 to less than 3.2. This reduction of the number of flights affects the performance of the pricing model greatly. On the other hand, the performance of the greedy heuristic is not significantly affected by the preprocessing methods because the computational complexity of the greedy heuristic only increases linearly with the number of flights considered. We also notice that the clustering method does not reduce the sub-

Test Instance	Before Preprocessing		Clustering Flights	
	Ave Sub Size	Max Sub Size	Ave Sub Size	Max Sub Size
$s1$	12.95	42	2.17	20
$s2$	5.02	36	4.76	36
$s3$	6.84	44	4.77	40
$m1$	11.93	42	2.05	22
$m2$	11.57	67	2.27	28
$m3$	14.24	64	2.82	29
$m4$	18.97	87	2.92	57
$d1$	14.21	90	3.15	61
$d2$	13.80	87	2.90	58
$d3$	11.48	71	2.52	46

Table 4.3: Sizes of the pricing subproblems before and after employing the clustering mathod.

problems of test case s2 and s3 significantly. This is because test cases s2 and s3 contain only the flights from a busy time window (as shown in Table 4.1). Therefore, it is hard to cluster flights into different groups. But this phenomenon usually does not occur in the large daily test cases.

4.5.4.3 IP Solution.

We investigate the characteristics of the best IP solutions found for all ten test instances. Those characteristics include the integrality gap, defined by ($\frac{IP-LP}{LP} \times 100\%$), the computational time, the number of branch-and-bound nodes, the number of columns generated, and the number of flights with fixed track and level at root node, which are presented in Table 4.4. We obtain nine optimal solutions out of ten test cases within the 90-minutes limit. It is also very important to note that, for all instances, the LP relaxation of the FSM provides good lower bounds. In most small and medium instances, the optimal solutions are found at the root node.

We also examine the suitability and efficiency of the proposed branch-on flight-assignment rule. At the root node of branch-and-bound tree, we count the number of flights with fixed track and level assignments. In other words, we count the number of flights that satisfy $\sum_{s \in S_{lt}} \alpha_s^f x_s = 1, \forall l \in L, \forall t \in T$ at the root node of branch-and-bound- tree. The procedure of our branch-on flight-assignment rule is very effective and efficient when there are a large number of flights with fixed track and level assignments, because it would be easier to obtain an integer feasible solution. From the last column of Table 4.4, we note that in every test instance the majority of flights had fixed track and level assignments at the root node. Therefore, by applying the branch-on flight-assignment rule, we can find the integer feasible solution efficiently for all the test cases.

Test Case	Best IP solution	Integrality gap (%)	Comp time	♯ of B&B nodes	♯ of col	% of fixed flights at root node
$s1$	307	0.00%	34	1	1,645	100.00%
$s2$	726	0.00%	118	1	3,220	100.00%
$s3$	956	0.00%	217	1	5,547	100.00%
$m1$	725	0.00%	475	1	9,332	100.00%
$m2$	771	0.00%	974	1	7,955	100.00%
$m3$	951	0.00%	475	1	7,452	100.00%
$m4$	1,062	2.02%	3,417	38	18,760	83.26%
$d1$	1,926	0.00%	3,059	1	15,157	100.00%
$d2$	1,760	0.00%	4,067	1	13,769	100.00%
$d3$	1,687	0.06%	1,092	14	21,100	95.97%

Table 4.4: Performance characteristics of the best LP and IP solutions for the FSM.

4.5.5 Performance Characteristics of Different Solution Methods.

Table 4.5 presents performance characteristics of the $BAVM$ and FSM in comparison with other solution methods for $FCRP$, the TA and SH approaches. TA denotes the track advisory system currently used at the Oakland ARTCC; SH denotes the sequential heuristic proposed in Brewer [2005]; $BAVM_N$ denotes the basic absolute value model without track changes; FSM_N denotes the flight sequence model without track changes. The performance characteristics are (1) the objective function value (the total penalty cost), (2) the computational time (in seconds), (3) the average and maximum delay time, (4) the percentage of flights being delayed, (5) the average and percentage of level and track changes, and (6) the percentage of flights with unchanged schedules. We will stop the computation when it reaches the predetermined time limit of one and half hour (5400 seconds). We note that the TA and SH approaches are programmed and implemented in Fortran but run on the same workstation as the $BAVM$ and FSM. In addition, because of the nature of their mechanisms, TA approach provides the solutions (conflict-resolved schedules) without any track changes. Thus, for a fair comparison, we modified both the FSM and $BAVM$ to disallow the track change option, and the modified models are denoted by $BAVM_N$ and FSM_N respectively.

As we can see from Table 4.5, in all cases the FSM provides the best solutions among all other methods tested. This is more evident in large test instances, where the FSM provides solutions that are much better (in terms of objective value) than those obtained by the TA and SH approaches whereas the $BAVM$ fails to obtain a feasible integer solution for the medium and large instances. We observe that a large number of Big-M type constraints in the $BAVM$ make the LP relaxation bound very poor and the CPLEX MIP solver has longer time searching the branch-and-bound tree. We also note that the FSM solutions are superior to the TA and $BAVM$ solutions in all performance characteristics in all test instances.

We also notice that the FSM_N needs shorter computational time than the FSM in all test cases. One possible reason is because the possible number of sequences in the FSM_N is much less than that of the FSM. Also, the objective values of the FSM_N are only marginally larger than that of the FSM except for test case d2, where a large number of

Test Instance	Solution Method	Obj Val	Comp Time	Ave Delay	Max Delay	% of Flight Delayed	Ave Level Change	% of Level Change	Ave Track Change	% of Track Change	% of Unchanged Flights
$s1$	TA	558	2	14.69	29	32.50	1.38	10.00	0.00	0.00	53.75
	SH	521	2	9.52	27	28.75	1.36	13.75	1.00	2.50	58.75
	$BAVM_N$	365	1,202	10.88	24	21.25	1.00	11.25	0.00	0.00	70.00
	$BAVM$	328	5,400	7.50	20	17.50	1.09	13.75	1.00	1.25	70.00
	FSM_N	365	8	10.88	24	21.25	1.00	11.25	0.00	0.00	70.00
	FSM	307	34	6.67	17	18.75	1.00	13.75	1.00	1.25	70.00
$s2$	TA	1,077	2	18.66	29	44.19	1.10	24.42	0.00	0.00	34.88
	SH	897	3	19.92	35	16.28	1.00	31.40	1.00	6.98	48.84
	$BAVM_N$	864	3,279	12.00	25	27.91	1.33	31.4	0.00	0.00	47.67
	$BAVM$	1,250	5,400	17.74	26	54.65	1.11	20.93	1.00	3.49	24.42
	FSM_N	863	93	12.00	25	27.91	1.33	31.4	0.00	0.00	47.67
	FSM	726	118	6.91	24	26.74	1.06	19.77	1.00	10.47	48.84
$s3$	TA	1,171	3	18.15	27	37.10	1.17	14.52	0.00	0.00	50.00
	SH	1,084	4	20.37	35	15.32	1.15	20.97	1.00	5.65	59.67
	$BAVM_N$	No Fea	-	-	-	-	-	-	-	-	-
	$BAVM$	No Fea	-	-	-	-	-	-	-	-	-
	FSM_N	1,113	171	8.02	34	31.45	1.52	21.77	0.00	0.00	46.77
	FSM	956	244	6.32	24	27.42	1.04	19.35	1.00	0.09	50.00
$m1$	TA	1,205	6	14.61	20	1.36	27.00	11.00	0.00	0.00	63.50
	SH	1,205	7	7.90	15	10.00	1.26	19.00	1.50	3.00	70.50
	$BAVM_N$	No Fea	-	-	-	-	-	-	-	-	-
	$BAVM$	No Fea	-	-	-	-	-	-	-	-	-
	FSM_N	729	320	10.45	30	16.50	1.00	12.00	0.00	0.00	73.50
	FSM	725	475	10.21	30	16.00	1.00	12.50	0.00	0.00	74.00
$m2$	TA	1,563	6	9.65	33	44.76	1.21	16.19	0.00	0.00	41.90
	SH	1,331	7	4.96	26	13.33	1.16	24.29	1.38	3.81	61.43
	$BAVM_N$	No Fea	-	-	-	-	-	-	-	-	-
	$BAVM$	No Fea	-	-	-	-	-	-	-	-	-
	FSM_N	780	131	6.74	15	16.67	1.06	15.24	0.00	0.00	70.95
	FSM	771	974	8.34	19	22.38	1.00	11.90	1.00	1.90	64.76
$m3$	TA	1,518	7	12.19	28	33.96	1.21	15.57	0.00	0.00	52.83
	SH	1,496	7	8.14	26	10.38	1.22	23.58	1.18	5.19	64.15
	$BAVM_N$	No Fea	-	-	-	-	-	-	-	-	-
	$BAVM$	No Fea	-	-	-	-	-	-	-	-	-
	FSM_N	971	167	8.52	25	24.53	1.00	15.09	0.00	0.00	64.62
	FSM	951	475	7.91	19	21.70	1.00	12.73	1.00	2.36	65.57
$m4$	TA	2,382	7	22.12	33	40.00	1.15	12.09	0.00	0.00	49.76
	SH	1,780	7	6.01	21	20.47	1.13	22.33	1.24	7.91	51.16
	$BAVM_N$	No Fea	-	-	-	-	-	-	-	-	-
	$BAVM$	No Fea	-	-	-	-	-	-	-	-	-
	FSM_N	1,537	1,118	14.49	81	29.30	1.18	15.35	0.00	0.00	60.47
	FSM	1,037	3,417	6.10	16	17.67	1.04	13.02	1.00	5.12	65.58
$d1$	TA	4,598	11	10.62	30	45.54	2.58	14.94	1.00	0.24	42.17
	SH	3,916	13	20.46	21	14.70	1.20	25.78	1.35	4.82	58.07
	$BAVM_N$	No Fea	-	-	-	-	-	-	-	-	-
	$BAVM$	No Fea	-	-	-	-	-	-	-	-	-
	FSM_N	1,954	941	7.94	24	25.78	1.08	15.18	0.00	0.00	63.37
	FSM	1,926	3,059	7.89	30	23.86	1.05	13.25	1.00	1.69	63.86
$d2$	TA	4,265	11	7.96	36	38.07	2.89	15.66	0.00	0.00	48.67
	SH	3,394	13	12.25	31	9.67	1.12	30.12	1.26	5.54	59.76
	$BAVM_N$	No Fea	-	-	-	-	-	-	-	-	-
	$BAVM$	No Fea	-	-	-	-	-	-	-	-	-
	FSM_N	2,281	3,977	13.46	66	21.93	1.10	14.46	0.00	0.00	68.67
	FSM	1,760	4,067	7.00	26	18.07	1.02	12.53	1.00	3.13	68.92
$d3$	TA	3,534	11	8.38	33	33.89	2.47	13.98	0.00	0.00	54.74
	SH	2,881	13	6.02	19	11.12	1.28	23.93	1.17	4.27	63.98
	$BAVM_N$	No Fea	-	-	-	-	-	-	-	-	-
	$BAVM$	No Fea	-	-	-	-	-	-	-	-	-
	FSM_N	1,732	410	8.42	29	20.38	1.05	14.22	0.00	0.00	67.54
	FSM	1,687	1,092	7.68	22	19.19	1.00	12.56	1.00	1.66	68.25

Table 4.5: Performance characteristics of the $BAVM$ and FSM in comparison with the TA and SH approaches.

flights request the same track during a short time window. Without changing the track, a portion of these flights have to be delayed.

4.5.6 Dynamic Cost Structures

In reality, the cost structure of $FCRP$ can vary from day to day. For example, the cost of changing track and level can vary widely depending on the different weather conditions; the cost of changing track and level might not increase linearly in some extreme conditions. Also, it is very important to provide the controllers of ARTCCs with the flexibility of cost to handle different real life scenarios. In our computational study, we change the penalty of flight track and level changes and delay with two alternatives.

In the first alternative, we assume the cost of level and track change is not proportional to the number of level or track changes. Specifically, we assume that a penalty cost is incurred when a flight track or level is changed, but the cost is the same whether it changes 1, 2, 3 or more levels or tracks. This scenario makes the calculation of the penalty cost more complicated because the cost function of flight sequence is nonlinear, which is given by

$$c_{s_{lt}} = \sum_{f \in F_{s_{lt}}} A(z_f - r_f) + B_L \min(1, |l - l_f|) + B_T \min(1, |t - t_f|). \tag{4.29}$$

However, the complicated cost structure does not affect the column generation because the subproblem does not change structurally. With the same computational settings, the performance characteristics of the FSM with a nonlinear penalty cost function on three large test cases is given in Table 4.6. We obtain the optimal IP solutions for all large test cases within the limited time. Also, it is interesting to note that the numbers of unchanged flights with non-linear cost do not vary significantly when compared with the results of original linear cost structure (less than 3%).

Test Case	Obj Value	Comp Time	Ave Delay	% Flights with Delay	% of Flights with Level Change	% of Flights with Track Change	% of Unchanged Flights
d1	1,684	1,479	6.99	18.55%	14.46%	1.45%	66.51%
d2	1,602	4,265	6.92	14.94%	12.53%	2.65%	70.36%
d3	1,553	1,543	6.59	19.67%	14.46%	0.24%	69.91%

Table 4.6: Performance characteristics of the FSM with nonlinear penalty cost function.

In the second alternative, we change the penalty value of B_L and B_T to investigate the sensitivity of the $FCRP$ and its effect on the solutions. We initially set the cost of changing the flight level $B_L = 16$ and the cost of changing the flight track $B_T = 31$ based on the policy of 15/30-minute delay window and the preference of changing the level over changing the track. In practice, it is always less desirable to change the flight track over the flight level. However, it is not always desirable to change the flight level over longer-than-15-minute delay. Here we vary the penalty cost into four settings: $(B_L = 6, B_T = 11)$, $(B_L = 11, B_T = 21)$, $(B_L = 16, B_T = 31)$, and $(B_L = 21, B_T = 41)$. We are particularly interested in the large test instances, which are daily schedules, because different policies should be applied to every flight in a particular day.

Table 5.4 presents the solution characteristics of different penalty cost settings. It is observed that the resulting schedules do not change drastically across the different settings. The total number of unchanged flights only varies within 3%. It is also expected that the higher the penalty cost, the more delay in the schedule, the less level and track changes. In addition, the level change and track change occur only between adjacent levels and tracks. This observation is also consistent with the one in Table 4.5. It may be used to improve the pricing subproblem method. Specifically, when solving the pricing subproblem for level l and track t, we need to consider only the flights whose requested levels and tracks are in $\{l-1, l, l+1\}$ and $\{t-1, t, t+1\}$ respectively.

Test Case	Penalty Setting	Ave Delay	Max Delay	% Flights with Delay	Ave Level Change	% of Flights with Level Change	Ave Track Change	% of Flights with Track Change	% of Unchanged Flights
d1	$B_L = 6, B_T = 11$	3.71	16	8.43	1.03	18.80	1.00	6.25	64.58
	$B_L = 11, B_T = 21$	6.82	23	17.83	1.04	18.55	1.00	2.65	63.31
	$B_L = 16, B_T = 31$	7.89	30	23.86	1.05	13.25	1.00	1.69	63.86
	$B_L = 21, B_T = 41$	8.84	33	26.51	1.10	13.25	1.00	0.72	63.61
d2	$B_L = 6, B_T = 11$	4.33	14	10.36	1.03	17.83	1.00	4.10	70.84
	$B_L = 11, B_T = 21$	5.67	23	13.01	1.00	18.07	1.00	3.37	71.08
	$B_L = 16, B_T = 31$	7.00	26	18.07	1.02	12.53	1.00	3.13	68.92
	$B_L = 21, B_T = 41$	8.88	32	22.17	1.02	10.84	1.00	1.93	67.23
d3	$B_L = 6, B_T = 11$	4.24	19	9.00	1.00	17.54	1.00	4.74	70.38
	$B_L = 11, B_T = 21$	7.03	21	11.37	1.00	17.30	1.00	3.08	69.43
	$B_L = 16, B_T = 31$	7.68	22	19.19	1.00	12.56	1.00	1.66	68.25
	$B_L = 21, B_T = 41$	9.28	25	22.27	1.00	10.66	1.00	0.96	68.01

Table 4.7: Comparison of solution characteristics with different penalty settings.

4.6 Equity Considerations Among Airlines

It has been recognized that the equity consideration is critical to the airline operation planning. Although the slot ownership concept and intra-airline slot-exchange procedures are not currently used for the Pacific oceanic airspace, in the foreseeable future the ARTCCs are very likely to consider this option since the collaborative and slot ownership concepts are likely beneficial to the airlines and the ARTCCs. The equity concept has been integrated in the APCDM for an airspace traffic flow program [Sherali et al., 2002, 2003, 2006b, 2009], ground delay problem [Vossen et al., 2003], and airline crew scheduling problem [Boubaker et al., 2010]. In Sherali et al. [2002], equity is modeled as a set of hard constraints, so that the equity values of different airlines are restricted within a predefined range. In Vossen et al. [2003], Boubaker et al. [2010], equity is captured in the objective function as a soft constraint. In other words, the undesired variance of equity is penalized in the objective function. In Sherali et al. [2003, 2006b, 2009], both methods are applied

simultaneously. A set of hard constraints is used to restrict the equity values within some predefined range. Then the variance of the restricted equity values is penalized in the objective function. In this section, we extend the FSM to consider the equity measurement.

4.6.1 Mathematical Formulation and Solution Methods

The equity measurement proposed here is similar to the ones developed in Sherali et al. [2002, 2003, 2006b]. However, in this study we only model the equity measurement as a set of hard constraints. Particularly, we require that the difference between the equity values of any airline to be less than a predefined threshold. For illustration purpose, we simplify the equity measurement by assuming that all the airlines and all flights of an airline have an equal weight. However, the weighted case can be handled in a similar manner.

Formally, we introduce the following notations. Define G as a set of airlines, and G_s as the airlines in the sequence s. Define F_g as the set of flights for airline $g \in G$, and F_{gs} as the flights for airline g in sequence s. Define g_f as the airline of flight f, therefore, $G_s = \cup_{f \in F_s} g_f$. Define c_{fs} as the penalty cost of flight f in sequence s. Therefore, we define the average penalty cost per flight for airline g as c_g, which can be computed as follows:

$$c_g = \frac{\sum_{f \in F_g} \sum_{s \in S} \alpha_s^f c_{fs} x_s}{|F_g|} \qquad \forall g \in G.$$

The average airlines delay c_G is then computed as $c_G = \frac{\sum_{g \in G} c_g}{|G|}$. We define the equity index E_g for airline g as $E_g = c_g - c_G$. In order to ensure the equity among airlines, we restrict the equity index $E_g \leq E$, where E is a predefined maximum limit of inequity. Therefore, the equity constraints can be written as follows.

$$c_g = \frac{\sum_{f \in F_g} \sum_{s \in S} \alpha_s^f c_{fs} x_s}{|F_g|} \qquad \forall g \in G, \tag{4.30}$$

$$E_g = c_g - \frac{\sum_{g \in G} c_g}{|G|} \qquad \forall g \in G, \tag{4.31}$$

$$E_g \leq E \qquad \forall g \in G. \tag{4.32}$$

Constraints in Eqs. (4.30)-(4.32) can be simplified in the following equation:

$$\sum_{s\in S} x_s \left(\frac{(|G|-1)\sum_{f\in F_{gs}} c_{fs}}{|G||F_g|} - \sum_{g'\neq g} \frac{\sum_{f\in F_{g's}} c_{fs}}{|G||F_{g'}|} \right) \leq E \qquad \forall g \in G. \quad (4.33)$$

Define dual cost for constraints in Eq. (4.33) as θ_g. The dual cost of a sequence can be rewritten in the following format.

$$\begin{aligned}
\overline{c}_s = & \sum_{f\in F_s} \big(c_{l_f} + c_{t_f} + A(z_f - r_f)\big) - \sum_{f\in F_s} \pi_f - \lambda_{lt} \\
& - \sum_{g\in G_s} \left(\frac{|G|-1}{|G||F_g|} \sum_{f\in F_{gs}} c_{fs} - \sum_{g'\in G_s\setminus g} \frac{\sum_{f\in F_{g's}} c_{fs}}{|G||F_{g'}|} \right) \theta_g \\
= & \sum_{f\in F_s} \big(c_{l_f} + c_{t_f} - \pi_f + A(z_f - r_f)\big) - \lambda_{lt} - \sum_{f\in F_s} c_{fs} \frac{(|G|-1)\theta_{g_f} - \sum_{g'\in G_s\setminus g_f} \theta_{g'}}{|G||F_{g_f}|}
\end{aligned}$$

As we can see from the above equation, the dual cost contributed by equity constraints contains two parts for flight f in sequence s. The first part is the positive contribution towards airline g_f when inserting flight f in the flight sequence s, which is equal to $\frac{(|G|-1)\theta_{g_f}}{|G||F_{g_f}|}$. The second part is the negative contribution to the rest of the airlines $G_s \setminus g_f$ when inserting flight f in the sequence s, which is equal to $\frac{\sum_{g'\in G_s\setminus g_f} \theta_{g'}}{|G||F_{g_f}|}$. For simplicity, we denote $m_{fs} = \frac{(|G|-1)\theta_{g_f} - \sum_{g'\in G_s\setminus g_f} \theta_{g'}}{|G||F_{g_f}|}$. Then we have

$$\begin{aligned}
\overline{c}_s &= \sum_{f\in F_s} \big(c_{l_f} + c_{t_f} - \pi_f + A(z_f - r_f)\big) - \lambda_{lt} - \sum_{f\in F_s} m_{fs} c_{fs} \\
&= \sum_{f\in F_s} \big(c_{l_f} + c_{t_f} - \pi_f + A(z_f - r_f) + (c_{l_f} + c_{t_f} + A(z_f - r_f)) m_{fs}\big) - \lambda_{lt} \\
&= \sum_{f\in F_s} \big((1 - m_{fs})(c_{l_f} + c_{t_f} + A(z_f - r_f)) - \pi_f\big) - \lambda_{lt}
\end{aligned}$$

Because we have decomposed the reduced cost of a flight sequence into the summation of the cost associated with each flight, the pricing problem can be conducted in a similar way as discussed in Section 4.4.2. In particular, we only need to consider flight f such that $\big((1 - m_{fs})(c_{l_f} + c_{t_f} + A(z_f - r_f)) - \pi_f\big) < 0$ in the pricing subproblem of the column generation. We can use the greedy heuristic to generate good flight sequences efficiently. However, it is important to note that Theory 6 is no longer valid when incorporating these

equity constraints. As a result, the preprocessing procedure and the bilinear pricing model in Eqs. (4.11)-(4.14) cannot be used in the pricing subproblem when considering equity constraints. Therefore, to solve the equity FSM ($EFSM$) efficiently, we first solve the original FSM to optimal, then we use the optimal LP solution of the original FSM as the initial solution for the $EFSM$.

4.6.2 Proof of Concept

We conduct the computational study on the $EFSM$ by creating the airline information based on the historical data. In particular, we first extract the airlines statistical information from the historical data mentioned in Section 4.5.1. We compute the airline frequency for every origin-destination pair in the historical data. Then based on the airline statistics, we randomly assign an airline to a flight based on the its origin and destination. Each test case contains 20 airlines.

When solving the $EFSM$, we first solve the LP relaxation of the origin FSM to optimal. Then we add the equity constraints to the model and start the column generation for the $EFSM$. We set the computational time for column generation of the $EFSM$ to thirty minutes. We only solve the LP relaxation of the $EFSM$ when either the optimal solution is obtained or the stopping criterion is met, then we obtain the IP solution using the existing sequences. No branch-and-price is performed for the $EFSM$. In order to investigate how equity constraints affect the solution quality, we increase the value of E in Eqs. (4.30)-(4.33) from 3 to 7 with step size 1. In Table 4.8, we record the computational results of the $EFSM$ for the three large test cases.

As we can see from Table 4.8, very good solutions are obtained in a reasonable time for all three test cases with equity consideration. The results of the $EFSM$ do not incur significantly incremental costs (less than 2%) in all test cases.

We also notice that when the value of E is too small, there is no feasible solution to the $EFSM$. This happens when multiple airlines bid for the same busy time slot of a track and level. For example, in test case d2, two small airlines bid for the same entry time of a track and level. Any delay or level and track change on these flights will increase the equity index of the corresponding airline drastically. Therefore, it is hard to find a

Test Case	E value	Obj value	Incremental cost	Comp time	Ave E_G	Std $E_g - E_G$	Max $E_g - E_G$
	∞	1,926	0.00%	3,050	4.67	3.23	6.33
	7	1,926	0.00%	3,271	4.67	3.23	6.33
d1	6	1,927	0.05%	3,378	4.71	3.14	5.29
	5	1,953	1.40%	3,359	4.48	2.77	4.77
	4	1,957	1.61%	3,216	4.50	2.69	3.42
	3	Infea	-		-	-	-
	∞	1,760	0.00%	4,067	4.10	3.98	17.2
	7	1,794	1.93%	4,978	3.27	6.59	6.19
d2	6	Infea	-	-	-	-	-
	5	Infea	-	-	-	-	-
	4	Infea	-	-	-	-	-
	3	Infea	-	-	-	-	-
	∞	1,687	0.00%	1,092	3.72	2.79	8.61
	7	1,687	0.00%	1,360	3.65	2.57	5.73
d3	6	1,687	0.00%	1,360	3.65	2.57	5.73
	5	1,689	0.12%	1,327	3.65	2.52	4.64
	4	1,691	0.24%	1,259	3.62	2.42	3.65
	3	Infea	-	-	-	-	-

Table 4.8: Comparison of solution characteristics with different equity values.

feasible solution in this situation. To resolve the infeasibility, we can either penalize the infeasibility in the objective function using a Lagrangian relaxation based method, or set different tolerance level E for the airlines of different size.

4.7 Concluding Remarks

Each airline has its own flight planning program that deals with the problem of determining the optimum flight plans (i.e., level, track, and entry time) to minimize the fuel usage for each flight based on the aircraft fleet type, takeoff weight and the weather data. Thus airlines always make the best flight plans independently and provide them to the ARTCCs. This practice has made the flight planning and scheduling very difficult as there is no collaborative decision making system among airlines. Often time the requested flight plans incur conflicts due to the FAA safety standards (i.e., 20-minute separation on the same track and same level), unbalanced level and track requests, and exhausted level and track capacity. A common approach to resolve the flight conflicts is to delay one of the conflicting flights or change one of their requested tracks or levels. This problem of resolving the flight conflicts is called the $FCRP$. In this study, we develop a new optimization framework for the $FCRP$. Specifically we develop two MIP models: $BAVM$ and FSM.

The $BAVM$ is a traditional model for minimizing the penalty cost of delay, level and track changes by linearizing the absolute term. However, it is shown here to be computationally inefficient. The FSM is a set-partitioning model, similar to the flight string model proposed in [Barnhart et al., 1998b]. We construct a column generation framework with a bilinear pricing subproblem. We develop a preprocessing method to reduce the size of the pricing subproblem based on a necessary condition of optimal flight sequences. To solve the pricing subproblem efficiently and effectively, we propose a hybrid approach that combines the linearized pricing subproblem with a greedy heuristic. A branch-and-price method with a new branch-on flight-assignment rule is employed to find the optimal integer solution to the FSM. Both $BAVM$ and FSM are tested on ten simulated instances, based on a real dataset. Their performance characteristics are compared with those obtained by two other heuristics: TA and SH. The FSM allows us to efficiently and effectively solve large instances, and its performance is superior to all other methods.

We also extend the FSM to consider equity among airlines. Although the equity concept has not been incorporated in the current ARTCC operations, such corporate decision making (CDM)-feature is necessary and critical for the future aviation systems such as 4-D trajectory system. Our study demonstrates that the proposed column generation and branch-and-price methods can be extended to handle the equity constraints easily. The computation results show the proposed solution methods can solve the FSM with equity constraints within reasonable time.

There are many other real-life applications that our mathematical model and solution method can be applied. For example, the problem of assigning workers with different skill levels can be modeled into our problem[Norman et al., 2002, Ni and Abeledo, 2007], where each worker can be viewed as a track and level and each task is a flight. Another example is the critical equipment scheduling in the healthcare centers [Rais and Viana, 2011, Houdenhoven et al., 2008, Erdogan and Denton, 2010]. Typically, in a hospital, there is a certain set of critical equipments, which patients need to use after a particular time. There is a penalty if a patient is waiting for the equipment. The objective function is to minimize the total waiting penalties for the patients. Vehicle routing/scheduling is another example that our approach can be naturally applied, where each vehicle can be

viewed as a track and level, each customer is a flight, and the customers' waiting time is the flight delay time [Bramel and Simchi-Levi, 1997]. Although the above mentioned problems may have different structures in the pricing subproblems, the flight sequence model and the proposed solution methods may serve as a prototype or example for these problems. The proposed equity extension and the column generation solution method can also be modified or extended to the needs of various applications as well.

Chapter 5

A New Rotation-Tour Network Model for Aircraft Maintenance Routing Problem

The airline industry currently has a $40-billion plus market and is expected to grow rapidly with the population growth and growth in the overall economy. Everyday, thousands of aircrafts undergo maintenance, repair and overhaul. The aircraft maintenance problem is one of the important logistic problems in the airline industry. It is aimed at scheduling the aircrafts routing so that enough maintenance opportunities are provided to every aircraft in the fleet. In this chapter, we present a new compact network representation of the aircraft maintenance routing problem (AMR), and propose a new mixed-integer linear programming formulation to solve the problem. The quality of this model was assessed on four real test instances from a major US carrier, and compared with the flight string model proposed in Barnhart et al. [1998b], Cohn and Barnhart [2003]. The computational results show that the proposed model is able to obtain the optimal solutions to all test instances in reasonable time. This study suggests that this model can be applied to integrated problems of the AMR and other planning problems such as the fleet assignment problem and the crew pairing problem.

5.1 Introduction

As the safest public transportation system, US domestic airlines operate 5,000 flights per day while offering over 4 million fares to serve over 10,000 markets. As the demand on air transportation fluctuates along with the population growth and growth in the overall economy, the airline aircrafts fly more intensively without compensating the safety considerations. However, there is a reasonable concern raising about the aircraft maintenance. In a recent article of USA Today Stoller [Feb. 3 2010], it is reported that “millions of

passengers have been on at least 65,000 U.S. airline flights that shouldn't have taken off because planes weren't properly maintained during the past six years." The investigation also shows that maintenance was 'a cause, factor or finding' in 18 accidents since January 1, 2000, and caused 43 people died and 60 people injured. It is suggested that the uncertificated mechanics and outsourced repairs might be the cause of the situation. In fact, how to schedule the aircrafts with enough maintenance opportunities using limited resource is always one of the major airline planning operations, known as the aircraft maintenance routing problem (AMR). In this chapter, we are investigating a new network-based model to solve the AMR, with the consideration of the resource limitations at maintenance stations. Generally, the airline planning consists of four major sequential operations: *flight schedule design*, *fleet assignment (FAM)*, *aircraft maintenance routing (AMR)*, and *crew scheduling*. The *flight schedule* is usually designed by the airline marketing department based on traffic forecasts, airline network analysis and profitability analysis over several months Soumis et al. [1980], Phillips et al. [1991], Lohatepanont and Barnhart [2004]. After a flight schedule is obtained, a variety of aircraft fleets are assigned to individual flights based on passenger demands (both point-to-point and continuing services), revenues, operating costs etc, so that the total profit is maximized Subramanian et al. [1994], Hane et al. [1995], Sherali et al. [2006a], Belanger et al. [2006]. Given an assigned aircraft fleet, the *aircraft maintenance routing* Clarke et al. [1997], Barnhart et al. [1998b], Gopalan and Talluri [1998], Talluri [1998], Boland et al. [2000], Mak and Boland [2000], Sriram and Haghani [2003], Elf and Kaibel [2003], Sarac et al. [2006] is to determine the rotation of individual aircraft, so that enough maintenance opportunities are provided to each of the aircrafts. Subsequently, the *crew scheduling* is to determine the best set of *crew pairings (CP)*, crew trips spanning one or more working days separated by a break period, to cover all the aircraft fleetsRyan and Foster [1981], Anbil et al. [1992], Desaulniers et al. [1997a], Vance et al. [1997], Barnhart and Shenoi [1998], Barnhart et al. [2003], AhmadBeygi et al. [2009] and to construct personalized monthly schedules (rosters) for crew members Gamache and Soumis [1998], Dawid et al. [2001].

In the last decade, there has been an increasing interest in solving the integrated problems of two or more planning operations, such as integrating FAM and AMR Barnhart

et al. [1998b], Haouari et al. [2009], CP and AMR Cordeau et al. [2001], Klabjan et al. [2002], Cohn and Barnhart [2003], Mercier et al. [2005], Mercier and Soumis [2007], Weide and D. Ryan [2010], FAM and CP Sandhu and Klabjan [2007], etc. As the evolution of the computational capability, it has become quite worthwhile to solve more than one planning operations simultaneously. For example, it has been shown in Cordeau et al. [2001], Klabjan et al. [2002], Cohn and Barnhart [2003], Mercier et al. [2005], Mercier and Soumis [2007], Weide and D. Ryan [2010] that a substantial saving on crew cost can be obtained by solving CP and AMR simultaneously. In order to solve an integrated problem with AMR, the integrated problem is usually decomposed into a multi-stage problem, and solved using iterative algorithms (e.g., Bender Decomposition algorithm, cutting planning algorithm, etc.). In these iterative algorithms, it is quite common that AMR needs to be solved for many times, and the solution speed of AMR effects the overall performance of the integrated problem greatly. In this chapter, we are not solving any new integrated problem or a competing solution method for any existing integrated problems, rather, we are proposing a new network-based model to solve AMR more efficiently. Currently, AMR is usually solved using a string-based model (FSM) Barnhart et al. [1998b], Clarke et al. [1997], Boland et al. [2000], Cordeau et al. [2001], Elf and Kaibel [2003], Mercier et al. [2005], Mercier and Soumis [2007], Weide and D. Ryan [2010], and it again needs to be solved iteratively using column-generation and branch-and-price, where the pricing problem was modeled as a constrained shortest path problem on a time-space network. This requires considerable experience to customize the algorithms and parameters to overcome the implementation difficulties such as heading-in and tailing-off effect Vanderbeck [2005]. Although FSM can be solved in reasonable time using column-generation and branch-and-price, to the best of our knowledge, there is no one-step solution methods for FSM because the number of strings grows exponentially with the number of flights in AMR, and it is impossible to enumerate all the strings in the model.

In this chapter, we propose a new rotation-tour network model (RTNM) built on a modified time-space network. RTNM introduces special time-reversible arcs to cater the maintenance operations. RTNM is very compact and scalable compared to the string model. The size of RTNM is relative to the maximum number of days allowed between

two consecutive maintenances while there are exponential number of feasible flight strings that can be included in the string model. Because the size of RTNM is polynomial in theory and relatively small even for the large real-life test cases, it can be solved by most commercial integer solvers in one step. The computational study shows that RTNM can get the optimal solutions for real network test problems in reasonable time. The results suggest that the proposed RTNM may improve the solution time of integrated problems of AMR and other planning problems such as the fleet assignment and crew pairing problem. We also compare the performance characteristics of RTNM with those of the flight string model (FSM) proposed in Barnhart et al. [1998b], Cohn and Barnhart [2003] on four real network test problems.

The remainder of the chapter is organized as follows. In Section 5.2, we give a background on AMR and present the widely used time-space network structure. In Section 5.3, we present the new rotation-tour network representation of AMR mathematically. Section 5.4 presents the rotation-tour network optimization model (RTNOM) of AMR, and compares the well-known FSM with RTNOM. Section 5.5 presents the computational results of the proposed network model and compares its performance characteristics with those of the flight string model. Finally, we conclude our work and discuss some future studies in Section 5.6.

5.2 Background of Aircraft Maintenance Routing Problem (AMR)

In this section, we first introduce the background information of AMR. We then present the traditional time-space network representation of the airline network.

5.2.1 Problem Definition

AMR is to determine how to assign flights of a fleet to every individual aircraft, while ensures each aircraft has enough maintenance opportunities to meet the regulations set by an airline or country. Generally, different countries and airlines have slightly different requirements for their aircraft maintenances. There are several types of common maintenances for US domestic airlines Hessburg [Apr. 2000], FAA [2002]. The most frequent maintenance is daily check (also called night-stop check, or service check), which is needed

for every 24 to 60 hours of accumulated flight time. A daily check includes a walk around inspection, check on lights, emergency equipment, servicing engine oil. It takes a mechanic about one to three hours to finish and several hours to fix any items (e.g., changing tire/brake). Normally there is not enough time during the day to fix many items, therefore a daily check is usually done at night Talluri [1998], Gopalan and Talluri [1998], Sriram and Haghani [2003]. The next higher level of maintenance check is called Type A check, and it is performed approximately every month or longer. Then two heavy checks called Type C check and Type D check are performed every 18 months and 5 years approximately. Note a maintenance check can only be performed at certain airports called maintenance stations. Each maintenance station has limited resource and is only capable to perform certain number of maintenance checks every night. Usually, only daily check is considered in AMR, whereas the less frequent maintenances can be incorporated into FAM.

In order to ensure the equal utilization of every aircraft in a fleet, some airlines required aircrafts fly the same sequence of flights, also called *big cycle constraints* Klabjan [2005]. For example, considering fours daily flights $LGA(10:10) \rightarrow DFW(13:15)$, $DFW(15:15) \rightarrow LGA(19:55)$, $LAX(8:10) \rightarrow DFW(13:10)$, and $DFW(15:20) \rightarrow LAX(16:30)$. Assume we have two aircrafts to cover these four flights. There are two possible scenarios: the first scenario is aircraft A flies $LGA \rightarrow DFW \rightarrow LGA$ and aircraft B flies $LAX \rightarrow DFW \rightarrow LAX$ everyday; an alternative scenario is to let every aircraft flies all four flights $LGA \rightarrow DFW \rightarrow LAX \rightarrow DFW \rightarrow LGA$ (and forms a big cycle) in every two days. That is, on day 1, aircraft A takes flights $LGA \rightarrow DFW \rightarrow LAX$ and spends the night at LAX, and aircraft B flies flights $LAX \rightarrow DFW \rightarrow LGA$, and spends the night at LGA. On day 2, aircraft A flies $LAX \rightarrow DFW \rightarrow LGA$ and aircraft B flies $LGA \rightarrow DFW \rightarrow LAX$. Notice both aircrafts have the same flight sequences in the rotations. By taking the second scenario, the equal utilization rate of aircrafts is guaranteed. However, it is worthy mentioning that a big cycle of many flights can hardly be operated without perturbation in real implementations. Therefore, this constraint is relaxed in a number of studies Cohn and Barnhart [2003], Mercier et al. [2005], Mercier and Soumis [2007], Haouari et al. [2009], Weide and D. Ryan [2010].

A feasible solution to AMR in the planning stage normally contains a generic aircraft

route during a rolling time horizon. Consider the example mentioned before, a feasible AMR solution could be a rotation of $LGA \rightarrow DFW \rightarrow LAX \rightarrow DFW \rightarrow LGA$ followed by a maintenance at LGA, assuming the maximum duration between two maintenances is two days. This generic solution does not assign individual aircraft to flights explicitly. However, in the operational stage, this solution can be served as a reference for assigning individual aircrafts.

There are several possible costs associated with AMR, which include the (negative) through revenue cost, penalty cost for undesired connections, the maintenance costs, etc. The through revenue between two connected flights is measured by the number of passengers who stay on the same aircraft between two connected flights in the rotationClarke et al. [1997]. The penalty cost between two connected flights in a rotation incurs when the connection/ground time between two flights is close to *the minimum ground time*. The minimum ground time is the minimum time for offloading and reloading of passengers, bags, cargo, fuel, cleaning of the cabin, etc.. A rigid ground time may introduce serious disruptions for crews, passengers and the operations of the following schedule. For example, in Klabjan et al. [2002], Cohn and Barnhart [2003], Mercier et al. [2005], it is pointed out that if the connection time between two flights is less than the minimum time for crews to change aircrafts (normally longer than the minimum ground time), called a *short connection*, it may lead the infeasibility to the crew pairing problem. Finally, the maintenance cost is associated with the manpower and facility cost at the maintenance stations Sriram and Haghani [2003]. The maintenance cost can also be associated with the aircraft flying time since the last maintenance. In particular, if the flying time since the last maintenance is too short, a new maintenance could lead to potential waste of maintenance opportunities. On the other hand, if the flying time since the last maintenance is too long (near the maximum allowed time), a new maintenance could restrict the flexibility of the schedule and bring difficulty for necessary reschedule of the rotation in future. Therefore, we can penalize the maintenance cost if the aircraft flying time is too short or too long. It is also worthy mentioning some airlines consider AMR as a pure feasibility problem Klabjan [2005] because the cost of AMR is relatively small comparing with other planning operations such as FAM and CP, and it is hard to identify an accurate

penalty cost for AMR.

In this chapter, we only consider the daily check in AMR, and we assume the maintenance is performed during the night. Also, we are solving the planning stage of the aircraft maintenance routing problem. That is, we will not consider assigning individual aircraft to flights. Instead, we try to find a generic route satisfying maintenance requirements. We assume the cost of AMR contains two parts. The first part is the connection cost, which is negative for through connections and positive for short connections. The second part is the maintenance cost.

5.2.2 Time-Space Network Representation

The time-space network is widely used in airline planning operations, i.e., FAM, AMR, CP, etc. The time-space network first appears in Hane et al. [1995] to solve FAM. In a daily time-space network (shown in Figure 6.1), a time line represents a station, which consists a series of event nodes occurring sequentially with respect to a specific time at the station. Each event node represents an event of flight departure or arrival at the station. In order to allow connection between flights, a minimum ground time is added on the actual flight arrival time for computing the time of arrival node. In the time-space network, there are three types of arcs: ground arc, flight arc and overnight (also called wrap-around) arcs. Ground arcs represent one or more aircrafts staying at the same station for a period of time. Flight arcs represent flights between airports. Overnight arcs ensure the continuity of the aircraft routing from the current planning period to the next one. With ground arcs and overnight arcs, it is possible to preserve the aircraft balance and allow all possible connections between the arrival flights and the following departure flights at a station. In Figure 6.1, we show a daily time-space network containing eight flights between two stations, and the time progresses horizontally from left to right.

Two preprocessing methods, namely *node aggregation* and *island isolation*, are presented in Hane et al. [1995] to reduce the size of time-space network. Node aggregation allows to combine consecutive arrival nodes and subsequent consecutive departure nodes and eliminate unnecessary ground arcs. Island isolation can eliminate a ground arc if it is not necessary to have aircrafts on the ground arc during the specific period. Considering

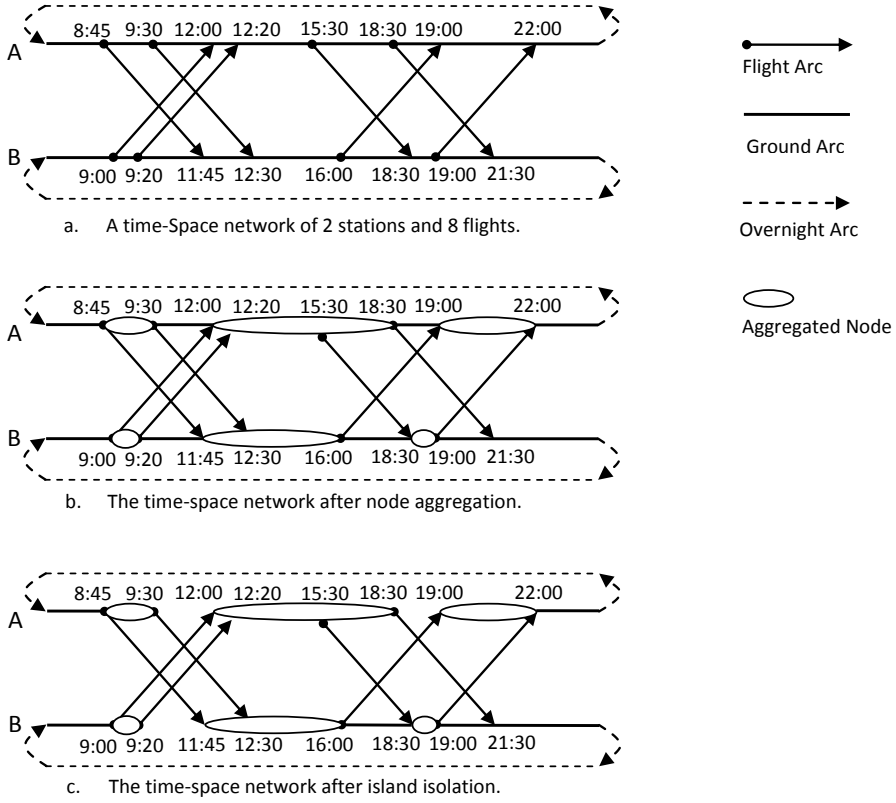


Figure 5.1: Time-space network with three stations and eight flights.

the example shown in Figure 6.1. There are 8 flights and hence 16 nodes in the time-space network. In Figure 6.1b, we show the time-space network after node aggregation, where only 7 nodes (6 aggregated nodes and one original node) and 9 ground arcs are necessary. In Figure 6.1c, we show the network after island isolation, where 6 ground arcs are necessary. It has been reported that these preprocessing methods reduce the size of the time-space network greatly [Hane et al., 1995, Sherali et al., 2006a].

5.3 Rotation-Tour Time-Space Network

In this section, we first present the new rotation-tour network to represent a daily AMR. We then introduce the new rotation-tour network model (RTNM) to formulate the daily AMR as a feasibility problem for the sake of exposition. In section 5.4, we will present the complete optimization model formally.

5.3.1 Network Modeling of AMR

We propose a novel rotation-tour network model modified on a the traditional time-space network to represent AMR, which can be constructed as follows. We construct a $|D|-$day time-space network $G(N, E)$, where $D = \{0, 1, \ldots, |D|\}$ is a set of possible days between two consecutive maintenances for an aircraft and $|D|$ is the maximum number of days allowed between the two consecutive maintenances. We define N as a set of nodes (events) at every station, where each node on a time line represents an event of flight arrival or departure at a station (both maintenance and non-maintenance). There are three types of arcs in the rotation-tour time-space network: ground, flight and maintenance arcs. Ground arcs and flight arcs are constructed in the same way as in the traditional time-space network. The maintenance arcs, which are capacitated, start at the end of each day, at maintenance stations and end at the beginning of the same time lines (*time reversible*). Here the end of a day at a station for AMR is defined as the last scheduled arrival time at the station, and beginning of a day is defined as the first scheduled departure time at the station. Notice that the beginning/end of a day could be different at different stations. The duration between the end of a day and the beginning of the next day is normally longer than the maintenance duration. The capacity of maintenance arc is the maximum number of aircrafts allowed to receive the service at a maintenance station per day.

There are two main differences between the rotation-tour network and the traditional time-space network presented in Section 5.2.2. First, there is only a one-day period in the traditional time-space network for a daily schedule, but a $|D|$-day duration in the rotation-tour network for a daily AMR. Second, instead of having overnight arcs at every station in the traditional network as time-reversible arcs, the maintenance arcs at the maintenance stations are time-reversible, and there is no time-reversible arcs at non-maintenance stations in the rotation-tour network. Figure 5.2a illustrates an example of daily flight schedules that contains 6 flights among three stations, JFK, CVG, and MEM. The two maintenance stations are JFK and MEM. Figure 5.2b depicts the corresponding rotation-tour network representation such that the maximum time between maintenances is two days.

It is obvious that in the rotation-tour network, there exists no sequence of connected

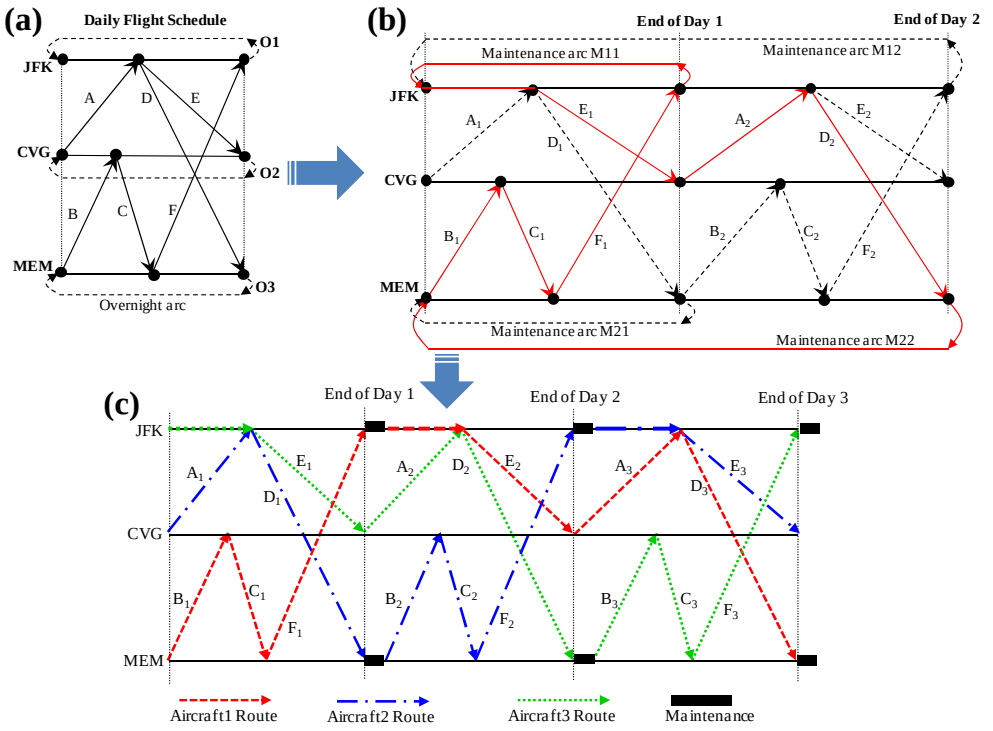


Figure 5.2: (a) An example of daily flight schedule feasible to 2-day maintenance constraints; (b) Rotation-tour time-space network constructed from the flight schedules in (a) and a feasible rotation tour solution to RTNM shown in red arcs; (c) Flight sequences of 3 aircrafts produced from the rotation tour in (b).

flights that violates the maintenance constraints. Because duration of the rotation-tour network is $|D|$, it is impossible to have a flight sequence (a path in rotation-tour network) last more than $|D|$ days without a maintenance arc. That is, the $|D|$-day maximum maintenance constraints are guaranteed in the rotation-tour network. Also, by constructing $|D|$ maintenance arcs (each arc starts at the end of $d \in D$ day) at a maintenance station, an aircraft can perform maintenance after any feasible flying days. Finally, all the maintenance arcs end at the beginning of the maintenance station time lines, so it is possible to have $|D|$-day long flight sequences after a maintenance.

Note red eye flights can also be handled using the rotation-tour network. Unlike the daytime flights, we duplicate a red eye flight $|D| - 1$ times, each starts on day d and ends on day $d + 1$, where $d \in \{1, \ldots, |D| - 1\}$. Because we assume the maintenances are only performed at night, it is obvious that the maximum red eye flights an aircraft can fly without a maintenance is $|D| - 1$. Hence, the maintenance feasibility of the schedule with red eye flights is guaranteed using the rotation-tour network.

5.3.2 Formulating Daily AMR as a Feasibility Problem

In this subsection, we illustrate in detail how to formulate AMR using rotation-tour network. In particular, we only consider AMR as a feasibility problem (with no objective costs) for explanatory purpose.

Denote a set of nodes (events) in the rotation-tour network by N, a set of daily flights by F. Denote $D = \{0, 1, \ldots, |D|\}$ as a set of possible days between two consecutive maintenances for an aircraft, and the day index $d \in D$ represents the day in a rotation-tour network. Denote K as the size of the scheduled fleet. Define a set of maintenance stations as M. The maximum number of maintenances allowed per day at maintenance station m is denoted by Q_m, where $m \in M$. Define the binary indicator α^+_{fdn} such that $\alpha^+_{fdn} = 1$ if flight f on day d starts at node n and 0 otherwise. Define the binary indicator α^-_{fdn} such that $\alpha^-_{fdn} = 1$ if flight f on day d ends at node n and 0 otherwise. Similarly, define the binary indicator β^+_{mdn} such that $\beta^+_{mdn} = 1$ if maintenance arc at station m on day d starts at node n and 0 otherwise; define the binary indicator β^-_{mdn} such that $\beta^-_{mdn} = 1$ if maintenance arc at station m on day d ends at node n and 0 otherwise. Define the binary decision variable x_{fd} such that $x_{fd} = 1$ if flight f is flown on day d in the rotation-tour network, and 0 otherwise. Define the integer decision variable z_{md} as the number of aircrafts in maintenance at station m at the end of day d in the rotation-tour network. Define the ground arc before node n as l^+_n, and the ground arc after node n as l^-_n. Define the ground arc variable w_l as the number of the aircrafts on the ground arc l.

The proposed RTNM for AMR is given by

$$(RTNM)\ \min \quad 0 \tag{5.1}$$

$$\text{s.t.} \sum_{d\in D} x_{fd} = 1 \qquad \forall\ f \in F, \tag{5.2}$$

$$\sum_{f\in F}\sum_{d\in D} \alpha^{+}_{fdn} x_{fd} + \sum_{m\in M}\sum_{d\in D} \beta^{+}_{mdn} z_{md} + w_{l^{+}_{n}} = \sum_{f\in F}\sum_{d\in D} \alpha^{-}_{fdn} x_{fd} + \sum_{m\in M}\sum_{d\in D} \beta^{-}_{mdn} z_{md} + w_{l^{-}_{n}} \quad \forall\ n \in N, \tag{5.3}$$

$$\sum_{m\in M}\sum_{d\in D} d \cdot z_{md} \le K \tag{5.4}$$

$$\sum_{d\in D} z_{md} \le Q_m \qquad \forall\ m \in M, \tag{5.5}$$

$$x_{fd} \in \{0,1\} \qquad \forall\ f \in F, \forall d \in D, \tag{5.6}$$

$$z_{md} \in \{0,1,...,Q_m\} \qquad \forall\ m \in M,\ d \in D, \tag{5.7}$$

$$w^{-}_{n}, w^{+}_{n} \ge 0 \qquad \forall\ n \in N. \tag{5.8}$$

The objective function is 0 because we consider a feasibility problem. The assignment constraints in Eq.(5.2) ensure that each flight is covered once in the solution rotation. The flow balance constraints in Eq.(5.3) ensure that the number of inbound aircrafts is equal to the number of outbound aircrafts at each node. The fleet size constraint in Eq.(5.4) ensures that the total number of aircrafts used is not greater than the size of the fleet. In particular, we use the day index d of maintenance arcs to calculate the total number of aircrafts needed. Specifically, if a maintenance arc at the end of day d is used in the solution, it implies that there exists a sequence of flights (or string) lasting for d days. As a result, d aircrafts are needed to cover this sequence of flights. Consider the example in Figure 5.2. The rotation tour $B_1 \to C_1 \to F_1 \to M_{JFK1} \to E_1 \to A_2 \to D_2 \to M_{MEM2}$ shows that maintenance arc M_{JFK} on day 1 is used, which implies that an aircraft is needed to service the flights B_1, C_1, F_1 on that day and be maintained at the end of the day. Also maintenance arc M_{MEM} on day 2 is used, which implies that two aircrafts are needed to service the flights in last two days (one for the flight E_1 on day 1, and the other for the flights A_2, D_2 on day 2). From this observation, to serve all daily flight schedules, the fleet size of 3 is needed and can be verified easily from Figure 5.2c. The capacity constraints

in Eq.(5.5) ensure that the number of aircrafts being maintained at each station is not greater than the station's capacity. It is worthy mentioning that day index d does not effect the usage rate of maintenance stations. For example, consider the rotation tour shown in Figure 5.2 b and c. M_{MEM2} implies an aircraft need a maintenance at MEM after two days. However because two aircrafts are needed for the flight sequences before the maintenance at MEM, when the first aircraft perform the maintenance at MEM after two days, the second aircraft has complete the flights of the first day and will go maintenance after the next day's flights. Hence a maintenance is needed at MEM everyday without effected by day index d.

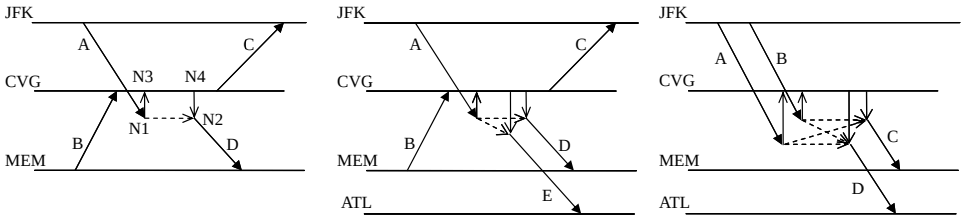
Note that the solution to RTNM does not provide details of routing sequences. Instead, the solution provides a set of flights, ground arcs and maintenance arcs. We can use the Eulerian tour algorithm Chartrand and Oellermann [1993] to construct a generic routing sequence in polynomial time. It is worthy mentioning although RTNM does not guarantee the big cycle constraints mathematically, our computational results never show a violation on the big cycle constraints for RTNM solutions. This might be because of the hub-and-spoke structure of the airline networks.

5.4 Rotation-Tour Network Optimization Model (RTNOM)

The proposed RTNM described in the previous section is to find a feasible rotation tour for AMR. However, in the rotation-tour network, the connections between flights might be implicity, which bring the difficulty to handle the profit/cost of flight connections. Nevertheless, the main objective of the AMR is to maximize the profit and minimize the cost. We herein extend the rotation-tour network to represent the profit connections and short connections. We then present the optimization version of RTNM, namely RTNOM for daily AMR. At the end of the section, we compare RTNOM with the well-known flight string model in terms of flexibility, space complexity, and solution quality. Because we are solving daily AMR, we assume the cost and profit of flight connections do not vary from day to day .

5.4.1 Modeling Through Value Connections

In order to capture the profit for every flight connection, we introduce a type of connection arcs called *through value arcs*. Through value arcs represent profitable flight connections. In the optimization version of RTNM, we can maximize the total profit value (i.e., number of through value arcs used in the solution) over the network in the objective function. If a through value arc is contained in the maintenance solution, the two connected flights must be flown by the same aircraft. We incorporate this profit model in RTNM by creating negative-cost through value arcs for every profitable connection. To allow other non-profitable connections that might be in the solution, we introduce a set of 0-cost arcs connecting flights and station time lines, called touching arcs. Formally, given I profitable connections and A arrival flights and B departure flights involved in these connections, we construct $|I|$ additional through value arcs, $|A|$ additional touching arcs into the station timeline, and $|B|$ additional touching arcs from the station timeline. An example



a. Single through value arc b. Two through value arcs from a single flight c. Multiple through value arcs between flights

Figure 5.3: Construction of through value arcs. (a) A single profitable connection $A \rightarrow D$ between arrival flight A and departure flight D; (b) Two profitable connections $A \rightarrow D$ and $A \rightarrow E$ from arrival flight A; (c) Multiple profitable connections $A \rightarrow C$, $A \rightarrow D$, $B \rightarrow C$, and $B \rightarrow D$ between arriving flights A, B and departure flights C, D.

of through value arc construction is shown in Figure 5.3a. Assume that the connection between flights A and D is a profitable connection, we created a through value arc between A and D directly (arc $N1 \rightarrow N2$). We subsequently created two touching arcs, $N1 \rightarrow N3$ and $N4 \rightarrow N2$, to allow non-profitable connections (e.g., $A \rightarrow C$ or $B \rightarrow D$). In a maximization problem, the through value arc $N1 \rightarrow N2$ will be selected in the solution. In Figure 5.3 b and c, we show that multiple through value arcs are constructed for profitable connections. Specifically, in Figure 5.3b, flight A has multiple profitable connections with flight D and E. Therefore, we construct two through value arcs between A and D, and

A and E. In Figure 5.3c, flights A and B have profitable connections with flights C and D, and four through value arcs are constructed for all profitable connections. As we can see, the method for constructing through value arcs can be extended to handle multiple profitable connections easily.

5.4.2 Modeling Short Connections

We create a set of connection arcs, called *penalty arcs*, for all short connections at every station. We try to avoid short connections in the rotation tour by minimizing the total cost of the penalty connections in the objective function. In particular, for every arrival flight at a station, we find a set of departure flights that form short connections and create the corresponding penalty arcs. It is important to note that the cost of penalty arcs should be proportional to the short time of the connections. Then we find a non-short-connected departure flight with least turn time, and create a 0-cost connection arc. Two examples of penalty arc construction are shown in Figure 5.4. In Figure 5.4a, there is only short connection $A \rightarrow F$. By adding 0-cost arc $A \rightarrow E1$, we ensure the cost-free connection between A and other departure flights at E_1 or later, because other subsequent connections can go through this 0-cost arc. In Figure 5.4b, there are four short connections between arrival flights A and C_1 and departure flights F and E_1. We create four penalty arcs for each short connection and two 0-cost arcs $A \rightarrow E_2$ and $C_3 \rightarrow E_2$ to ensure other cost free connections from flights A and C_3. Formally, given a set of flight schedules with I short connections and J arrival flights involved in these short connections, we construct $|I|$ additional arcs for all penalty connections and $|J|$ additional 0-cost arcs for non-penalty connections.

5.4.3 Formulation of RTNOM

After we extend the rotation-tour network to represent the through value connections and short connections, the objective of RTNOM is to minimize the total cost of the undesirable connections. The mathematical formulation of RTNOM is an extension of RTNM's mixed integer program in Eqs. (5.1)-(5.8). We first need to introduce the following additional parameters and variables. We define the set of additional arcs within a day as H (including

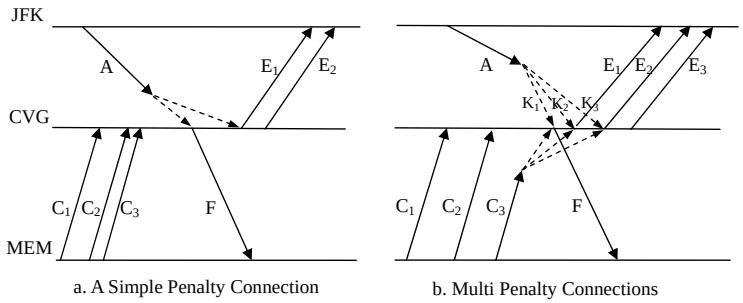


Figure 5.4: Construction of penalty arcs. (a) A penalty arc and a 0-cost connection arc are constructed for short connection $A \rightarrow F$; (b) Four penalty arcs and two 0-cost connection arcs are constructed for four short connections $A \rightarrow F$, $A \rightarrow E_1$, $C_3 \rightarrow F$, and $C_3 \rightarrow E1$.

penalty arcs, touching arcs, and connection arcs). The cost for arc $h \in H$ is defined as c_h. Note that here we consider only penalty arcs, we shall assume that $c_h \geq 0$ for penalty arc, $c_h = 0$ for through value and touching arcs. In future, if the information about the flight profit margin is given, we can incorporate it into this modeling by setting up a cost (profit) for every arc c_h. The maintenance cost at maintenance station $m \in M$ at day $d \in D$ is denoted as c_{md}. Note the maintenance cost is associated with day index d as we discussed in Section 5.2.1. The indicator parameter $\gamma^+_{hdn}(\gamma^-_{hdn})$ has value 1 if arc h at day d starts (ends) at node n; and 0 otherwise. The decision variable y_{hd} has value 1 if arc h at day d is included in the maintenance solution and 0 otherwise. Therefore, we formulate

RTNOM as follows:

$$(RTNOM) \quad \min \sum_{h\in H}\sum_{d\in D} c_h y_{hd} + \sum_{m\in M}\sum_{d\in D} c_{md} z_{md} \tag{5.9}$$

$$\text{s.t.} \sum_{d\in D} x_{fd} = 1 \qquad \forall\ f \in F, \tag{5.10}$$

$$\sum_{f\in F}\sum_{d\in D} \alpha^{+}_{fdn} x_{fd} + \sum_{m\in M}\sum_{d\in D} \beta^{+}_{mdn} z_{md} + \sum_{h\in H}\sum_{d\in D} \gamma^{+}_{hdn} y_{hd} + w_{l_n^+} =$$
$$\sum_{f\in F}\sum_{d\in D} \alpha^{-}_{fdn} x_{fd} + \sum_{m\in M}\sum_{d\in D} \beta^{-}_{mdn} z_{md} + \sum_{h\in H}\sum_{d\in D} \gamma^{-}_{hdn} y_{hd} + w_{l_n^-}$$
$$\forall\ n \in N, \tag{5.11}$$

$$\sum_{d\in D} z_{md} \le Q_m \qquad \forall\ m \in M, \tag{5.12}$$

$$\sum_{m\in M}\sum_{d\in D} d \cdot z_{md} \le K \tag{5.13}$$

$$x_{fd}, y_{hd} \in \{0,1\} \qquad \forall\ f \in F, \forall\ d \in D, \tag{5.14}$$

$$z_{md} \in \{0,1,...,Q_m\} \qquad \forall\ m \in M, \forall\ d \in D, \tag{5.15}$$

$$w_n^-, w_n^+ \ge 0 \qquad \forall\ n \in N. \tag{5.16}$$

The objective function in Eq.(6.1) minimizes the total penalty cost. The constraints in Eq.(6.3) are the new balance constraints which include additional arcs. The rest of constraints are the same with the constraints in RTMN presented in Eqs.(5.1)-(5.8).

5.4.4 Comparison Between FSM and RTNOM

The first flight string model (FSM) for AMR was proposed in Barnhart et al. [1998b]. A *string* was defined as a sequence of connected flights that begin and end at maintenance stations followed by a maintenance at the end. It satisfies the flow balance and is maintenance feasible by following all FAA and carrier-specified maintenance requirements (e.g., the maximum number of days allowed between the two consecutive maintenances). The departure time of a string is the departure time of the first flight in the sequence. The arrival time of a string is the arrival time of the last flight in the sequence plus the required maintenance time. The string model formulates AMR as a set partitioning problem with side constraints as follows.

Define S as a set of feasible strings, where $s \in S$ represents a feasible string for an

aircraft with a cost c_s. Define a binary parameter γ_{fs} such that $\gamma_{fs} = 1$ if flight f is included in string s, and $\gamma_{fs} = 0$ otherwise. Define a time-space network $G(N_M, E)$, where N_M is a set of nodes at maintenance stations representing the start and the end time of strings, and E is a set of arcs consisting two parts, one part representing all the string arcs S and the other part representing all the ground arcs L. Define the set of incoming and outgoing strings at node n as S_n^+ and S_n^-. Define the incoming and outgoing ground arcs at node n as l_n^+ and l_n^-. Define the integer parameter δ_s as the number of times flight sequence s crosses an arbitrary time O known as count time, and define L_O as a set of ground arcs spanning the count time O. Then we define a binary decision variable u_s such that $u_s = 1$ if string s is selected in the solution, and $u_s = 0$ otherwise. Define an integer variable w_l as the number of aircrafts on ground arc l. The MIP formulation of string model is given by

$$(FSM) \min \sum_{s \in S} c_s u_s \tag{5.17}$$

$$\text{s.t.} \sum_{s \in S} \gamma_{fs} u_s = 1 \qquad \forall\ f \in F, \tag{5.18}$$

$$\sum_{s \in S_n^+} u_s + w_{l_n^+} = \sum_{s \in S_n^-} u_s + w_{l_n^-} \qquad \forall\ n \in N_M, \tag{5.19}$$

$$\sum_{s \in S} \delta_s u_s + \sum_{g \in L_O} w_l \leq K, \tag{5.20}$$

$$u_s \in \{0, 1\} \qquad \forall\ s \in S, \tag{5.21}$$

$$w_{l_n^+}, w_{l_n^-} \geq 0 \qquad \forall\ n \in N. \tag{5.22}$$

The objective function in Eq. (5.17) minimizes the cost of the chosen route strings. The constraints in Eq. (5.18) are the partitioning constraints ensuring that each flight must be included in exactly one string. The constraints in Eq. (5.19) are flow balance constraints, which ensure that the flow of strings form a cycle. The constraint in Eq. (5.20) ensures that the number of strings and ground arcs crossing the count time O does not exceed the number of aircrafts in the fleet. Because of the flow balance constraints in Eq. (5.19) and binary constraints in Eq. (5.21), the integrality of the ground arc variables can be relaxed as denoted in Eq. (5.22).

As we stated before, RTNOM solves daily AMR problem under the assumption that

the maintenance is only performed during the night at maintenance stations. Also, RTNOM only considers the flight connection cost and the maintenance cost. Although the assumptions of the RTNOM are realistic, FSM is able to handle a broad range of AMR problems without these assumptions. For example, FSM is able to model the weekly AMR problem, capture more complex cost structure, and assume that maintenance can be performed at anytime of the day. Nevertheless, in this chapter we assume these assumptions are true for FSM in order to compare RTNOM with FSM. Specifically, we assume the string cost c_s can be decomposed into the connection cost c_h and maintenance cost c_{m_s}, that is, $c_s = \sum_{h \in H_s} c_h + c_{m_s}$, where H_s is the set of connections contained in string s, and c_{m_s} is cost of the maintenance at the end of the string s. The connection cost can be either negative for profit (as shown in Figure 5.3) or positive for penalty connections (as shown in Figure 5.4).

The total number of variables in RTNOM is $|D| \times (|F| + |M| + |H|) + |N|$, the number of constraints is $|F| + |N| + |M| + 1$, and the number of non-zero entries in the problem matrix is $|D| \times (3|F| + 2|H| + 4|M|) + 2|N|$. Because $\mathbf{O}(|N|) = |F|$, and $\mathbf{O}(|H|) = |F|^2$, we know the space complexity of RTNOM is $\mathbf{O}(|D||F|^2)$. By comparing the space complexity of RTNOM with that of FSM $\mathbf{O}(2^{|F|})$, the proposed RTNOM is more scalable than FSM. The number of variables in RTNOM is much less than the possible number of strings in FSM. Although the number of constraints in RTNOM is larger than that in FSM, preprocessing steps such as such as node aggregation and construction of islands, which are discussed in Section 5.2, can drastically reduce the number of nodes in RTNM network.

RTNOM not only has less space complexity than FSM, but has a provably equal linear relaxation as well. To see this, we first note that any solution to the LP relaxation of FSM has a corresponding solution to the LP relaxation of RTNOM with the same cost. To construct the corresponding solution of RTNOM from a FSM solution, we need to define the following notations for string s. Define the binary parameter $\theta^s_{fd} = 1$ if flight f is in dth day of string s, and 0 otherwise. Define $\rho^s_{hd} = 1$ if connection arc h is in contained in the dth day of string s, and 0 otherwise. Define $\phi^s_{md} = 1$ if maintenance at station m is performed after d days flights in string s, and 0 otherwise. We can construct a feasible RTNOM solution from a FSM solution using the following relations, i.e., let

$x_{fd} = \sum_{s \in S} \theta^s_{fd} u_s$, $y_{hd} = \sum_{s \in S} \rho^s_{hd} u_s$, and $z_{md} = \sum_{s \in S} \phi^s_{md} u_s$. It is easy to see that the constructed LP solution of RTNOM is at least as good as the LP solution of FSM. In fact, because the cost of string is decomposable, FSM can be view as a Danzig-Wolfe decomposition of RTNOM.

5.5 Computational Experience

In this section, we report empirical results of the proposed model on four real flight schedule datasets from a major US carrier. All test problems were solved using an Intel Dual Core 2.79GHz workstation with 1 GB of RAM memory running on Windows XP platform. Computational times reported in this section were obtained from the desktop's internal timing calculations, which include time used for preprocessing, perturbation, and postprocessing. All the mathematical modeling and algorithms were implemented in C++ language. All LP and MIP problems were solved using a CPLEX callable library version 10.0. We also compare the performance characteristics of RTNOM with those of FSM, which was implemented and solved using a column generation approach described in Barnhart et al. [1998b] as follows. First we find a feasible initial solution to the master problem by solving a constrained shortest path problem to generate a list of strings. Then we solve the LP relaxation of the master problem optimally, and subsequently solve the original IP master problem using a set of generated strings. Note that in order to get the optimal IP solution, we should implement a branch-and-price algorithm. However, this procedure is very time consuming. For the scope of this chapter in which we use FSM to compare the network representation and mathematical formulation, we only solve the root node of the branch-and-price tree and report the IP solution at the root node.

5.5.1 Test Instances

The test instances used as benchmark problems in this study are acquired from real flight schedules of a major US carrier. Here we consider a 4-day maintenance routing problem, where the maximum number of days between two consecutive maintenances is 4. Note that the 4-day horizon comes from a realistic estimate of FAA regulations. We shall call our four test instances as: *P1*, *P2*, *P3* and *P4*. The problem size and network characteristics

are shown in Table 6.1.

Test Instances	*P1*	*P2*	*P3*	*P4*
Size of the Fleet	14	20	39	70
Number of Flight Legs	46	92	172	352
Total Number of Stations	9	17	29	46
Number of Hub Stations	1	1	1	2
Number of Maintenance Stations	4	6	9	18
Number of Non-Maintenance Stations	5	11	20	28

Table 5.1: Characteristics of all four test problems.

5.5.2 Performance Characteristics

Table 5.2 presents the space complexities of RTNOM and FSM models in all four test instances. In FSM case, *All Columns* represents the numbers of possible flight sequences (strings) in the test instances while *G-Columns* represents the numbers of columns generated by the column generation approach that we implement. If we want to guarantee that FSM will eventually solve the test problems to optimality, we need to generate a large number of strings even with the column generation approach. The *Non-0* column represents the total number of non-zero entries in the mathematical formulations of RTNOM and FSM. As we can see from the table, the proposed RTNOM is much more compact and scalable than FSM.

Test	RTNOM			FSM			
Instances	Columns	Rows	Non-0	All Columns	G-Columns	Rows	Non-0
P1	376	168	1,104	109,220	6,103	82	55,590
P2	819	415	2,310	$> 1M$	73,434	162	939,520
P3	1,679	816	4,798	$> 1M$	581,536	289	4,994,162
P4	3,205	1,550	9,146	$> 1M$	1,000,121	590	12,184,268

Table 5.2: Model space complexity.

Table 5.3 presents the performance characteristics of RTNOM & FSM on all four test instances. The Solution Gap column represents the gaps between the values of the solutions, provided by RTNOM and FSM, and those of the optimal solutions. *Opt* means that the obtained solution is the optimal solution to the test instance, i.e., solution gap = 0. The $CPU\ Time$ column records the solution computational time in seconds. The $LP - IP\ Gap$ column records the gaps between the optimal LP solution and the optimal

IP solution, computed as $LP - IP\ Gap = (Opt\ IP - Opt\ LP)/Opt\ IP$. In our study, we were able to optimally solve all four test instances using RTNOM while only two test instances (*P1* and *P2*) were optimally solved using FSM. We also observed that RTNOM approach was able to obtain optimal (or integer) solutions to test instances in much less time comparing with FSM, because the size of RTNOM is much smaller than that of FSM, and RTNOM can be solved by CPLEX solver in one-step. On the other hand, we need to solve FSM iteratively using column-generation, and the initial solution to the string model may not be of high-quality (i.e., close to the true optimal solution). For small test problems (*P1* & *P2*), both RTNOM and FSM were able to obtain very good LP relaxation solutions with no gap to integer solutions. For large test problems, RTNOM yielded slightly better LP solutions than FSM, because of the tailing-off effect of column generation when solving FSM.

	RTNOM			FSM		
Test Instance	Solution Gap	CPU Time (sec)	LP-IP Gap	Solution Gap	CPU Time (sec)	LP-IP Gap
P1	*Opt*	2	0.000%	*Opt*	41	0.000%
P2	*Opt*	4	0.000%	*Opt*	65	0.000%
P3	*Opt*	6	0.901%	4.12%	169	0.978%
P4	*Opt*	15	0.372%	3.79%	418	0.393%

Table 5.3: Performance characteristics of RTNOM & FSM on all four test instances.

5.6 Conclusion and Future Works

In this chapter, we presented a new rotation-tour network representation of AMR. Based on this representation, we proposed a new mixed-integer linear programming formulation for the aircraft maintenance routing problem, namely *Rotation-Tour Network Optimization Model (RTNOM)*. To assess the performance of RTNOM, we compared its performance with the flight string model proposed in Barnhart et al. [1998b], Cohn and Barnhart [2003], which was considered to be the most well-known and state-of-the-art mathematical programming model for AMR. We tested both models on four real test instances (i.e., fleet assignments and flight schedules) from a major US carrier. The computational results showed that the proposed model was very compact and scalable, and was able to find the optimal solutions in much less time than the flight string model in all four test instances.

This research provides a compact formulation for AMR, which is beneficial to the researches on integrated planning problems of AMR and other planning problems such as FAM and CP. Evaluating the proposed model with the integrated planning problem is another interesting future research direction. For example, we note that the proposed RTNOM model can be extended to the integrated fleet assignment and routing problem (similar to the implementation of flight string model in Barnhart et al. [1998b]). This can be done by introducing a set of fleets and adding one more dimension (fleet selection) to the model. The performance of the extended model, compared to the flight string model, remains to be tested in our future study.

Chapter 6

A Network Model for Weekly Aircraft Maintenance Routing Problem and Integration with the Fleet Assignment Problem

In airline operation planning research, most studies focus on planning a daily schedule, and only a very few consider a weekly schedule. The weekly problems are normally much harder than the daily problems because the complexity increases drastically from daily problems to weekly problems. In this chapter, we present a novel weekly rotation-tour network representation for the *weekly aircraft maintenance routing problem* (WAMRP). Based on this representation, we propose a new network-based mixed-integer linear programming formulation for the WAMRP, namely *weekly rotation-tour network model* (WRTNM). The size of WRTNM only increases linearly with the size of the weekly schedule. We propose a simple variable fixing heuristic to solve WRTNM efficiently and effectively. To assess the performance of WRTNM, we test the WRTNM using eight real life test cases. The computational results show that the proposed model is very compact and scalable, and is able to find the optimal solutions to the schedule with 5700 flights and 330 aircrafts, approximately the size of world's largest airlines fleet, within five minutes. We also propose an integrated model to solve the weekly fleet assignment problem (WFAP) and the WAMRP simultaneously. We test the integrated model on nine self-constructed test cases. The computational results show that the integrated model generates near optimal solutions to the schedules with 1700 flights, 8 fleets with 120 aircrafts, approximately a medium-sized airline, in reasonable time. The computational results show that WRTNM and the integrated model provide very good LP relaxation bounds for all test cases.

6.1 Introduction

In the last decade, airlines' profit margin has been continuously pressured by their growing exposure to a high-cost low-fare environment. The increasing cost in capital, labor, and fuel, and raising competition from budget carriers has tied the airlines' profitability to the current economic downturn. In 2009, five out of nine major passenger airlines in the U.S. (i.e., US Airways, Continental, United, Delta and American) suffered net losses, where American airlines alone lost about $1.5 billion. As a whole, these nine airlines collectively lost $3.4 billion [McCartney, Jan. 28 2010]. How to efficiently utilize their expensive resources and generate the maximum profit is always the main challenge in the airline planning operations. This challenge has attracted numerous researchers from industry and academia, and generated far more than a thousand papers. However, a great majority of these papers consider a "daily schedule", where the schedule and the profitability of the flights are considered to be the same everyday in the week. For the U.S. domestic airlines, the schedules are normally the same during the weekdays and slightly different during the weekends because of the demand variation. For international airlines, on the other hand, the "daily schedule" assumption is no longer valid. For example, Lufthansa airlines only offer three flights per week from Frankfurt to Anchorage. Apparently, planning based on "weekly schedule" is very necessary for international airlines. Even for US domestic airlines, considering "weekly schedule" in the planning stage may increase their revenues. However, a very few papers in the airline schedule planning literature deal with the "weekly schedule". One of the possible reason is that the weekly problems are normally much harder than the daily problems because the complexity increases drastically from daily problems to weekly problems. Although the daily models and solution methodologies can be adopted to the weekly problems by extending the planning time from a single day to an entire week, the size of the models and the computational efforts usually increase exponentially when changing from "daily problems" to "weekly problems". To overcome this computational obstacles, various mathematical and heuristic decomposition methods have to be developed to solve the "weekly problems" [Desaulniers et al., 1997a, Barnhart et al., 1998a, Barnhart and Shenoi, 1998, Ioachim et al., 1999, Sriram and Haghani, 2003, Belanger et al., 2006, Gronkvist, 2006, Haouari et al., 2011, Weide and D. Ryan, 2010].

We focus our research on the weekly aircraft maintenance routing problem (WAMRP). The aircraft maintenance routing problem (AMRP) is to determine the flight routes for every aircraft such that the maintenance requirements are satisfied. The sizes of the traditional models for the AMRP usually increase exponentially with the number of flights in the schedule, and these models needs to be solved iteratively using column generation [Desaulniers et al., 1997b, Barnhart et al., 1998a, Cordeau et al., 2001, Elf and Kaibel, 2003, Mercier et al., 2005, Mercier and Soumis, 2007] or row generation [Clarke et al., 1997, Boland et al., 2000] approaches. Recently, Liang et al. [2011] proposed a compact rotation-tour network model for the daily AMRP, where the size of the model only increases linearly with the number of flights in the schedule. However, direct extension of this daily AMRP model to the WAMRP is not obvious, because simply extending the planning time-space network used in the model from a single day to an entire week does not preserve the critical properties to model maintenance requirement.

In this chapter, we propose a novel network-flow based model for the WAMRP. This model is inspired by the network model proposed in Liang et al. [2011] for the daily AMRP. The advantage of the proposed model is two-fold. First, the size of the proposed mathematical model only increases linearly with the number of flights to be scheduled. Therefore, this compact and scalable model can be solved directly by most commercial mathematical programming softwares even for very large real life schedules. Second, we notice from our computational experience that the proposed time-space network-flow based model provides very tight linear programming (LP) relaxation bounds, which can help to find good integer solutions efficiently. In addition to that, we further extend the proposed model to solve the integration of the WAMRP with the weekly fleet assignment problem (WFAP). As the LP relaxation bounds of the integrated model are very tight, we also propose a simple variable fixing heuristic to efficiently solve the integrated model.

The remainder of the chapter is organized as follows. In Section 2, we give a background on airline schedule planning, especially on the fleet assignment problem (FAP) and the AMRP, and present the widely used time-space network structure. In Section 3, we mathematically present the new rotation-tour network model of WAMRP. Section 4 presents the mathematical model to integrate the WFAP and the WAMRP, and the

solution methodology. Section 5 presents the computational results of WAMRP and integrated problem. Finally, we conclude our work and discuss some future studies in Section 6.

6.2 Background

In this section, we first introduce several major optimization problems in airline schedule planning. Then we discuss in detail about the AMRP and its integration with AMRP. At the end of the section, we present the traditional time-space network representation of the airline network, and its extension to the AMRP.

6.2.1 Airline Planning Operations

Generally, airline planning is consisted of four major optimization problems: flight schedule design, fleet assignment, aircraft maintenance routing, and crew scheduling. The *flight schedule design problem* is to decide which flights should be offered based on traffic forecasts, airline network analysis and profitability analysis [Soumis et al., 1980, Lohatepanont and Barnhart, 2004, Phillips et al., 1991]. After a flight schedule is obtained, a variety of aircraft fleets are assigned to individual flights based on passenger demands, revenues, and operating cost, so that the total profit is maximized [Subramanian et al., 1994, Hane et al., 1995, Sherali et al., 2006b, Belanger et al., 2006]. This operation is referred as the *fleet assignment problem* (FAP). Given an assigned aircraft fleet, the *aircraft maintenance routing problem* (AMRP) is to determine the rotation of individual aircrafts, so that adequate maintenance opportunities are provided to each and every aircraft [Clarke et al., 1997, Barnhart et al., 1998a, Barnhart and Shenoi, 1998, Gopalan and Talluri, 1998, Talluri, 1998, Boland et al., 2000, Mak and Boland, 2000, Sriram and Haghani, 2003, Elf and Kaibel, 2003, Sarac et al., 2006, Liang et al., 2011]. The *crew scheduling problem* is to determine the best set of crew pairings (referred as the crew pairing problem (CPP)) to cover all the aircraft fleets [Ryan and Foster, 1981, Anbil et al., 1992, AhmadBeygi et al., 2009] and to construct personalized monthly schedules (referred as the rostering problem) for crew members [Gamache and Soumis, 1998, Dawid et al., 2001].

Traditionally, airlines solve these planning operations separately and sequentially because of their enormous problem sizes and intractable complexity. Obviously, the sequential approach usually leads to suboptimal solutions. In the last decade, with the evolution of the computational capability, researchers have been able to develop better solution approaches by integrating multiple planning operations and solving them simultaneously. For example, it has been shown that a substantial saving on the crew cost can be obtained by solving the crew pairing problem (CPP) with the FAP and/or the AMRP together [Cordeau et al., 2001, Klabjan et al., 2002, Cohn and Barnhart, 2003, Mercier et al., 2005, Mercier and Soumis, 2007, Sandhu and Klabjan, 2007, Weide and D. Ryan, 2010]. In Yan and Tseng [2002], Lohatepanont and Barnhart [2004], Sherali et al. [2010], it has been shown that significant benefits can be achieved by solving the integration of the schedule design problem with the FAP simultaneously.

In this chapter, we are especially interested in the AMRP and its integration with the FAP. In the following subsections, we provide a detailed literature review on the AMRP and the recent advances of its integration with the FAP.

6.2.2 AMRP

Given a set of flights and a fleet of aircrafts, the AMRP is to determine the flight routes for every aircraft such that the maintenance requirements, which are set by Federal Aviation Administration (FAA) and individual airline companies, are satisfied. There are several types of common aircraft maintenances for US domestic airlines. The most frequent maintenance is called daily check, which is needed every two to four days. It includes a walk around inspection, a checkup on lights, emergency equipment, servicing engine oil, etc., and repairs if needed. It normally lasts from one to three hours for a checkup, and several hours for a repair. In order not to affect the aircraft utilization during the day, a daily check is usually performed at night [Talluri, 1998, Gopalan and Talluri, 1998, Sriram and Haghani, 2003]. Only certain airports, called maintenance stations, are capable of performing the maintenance operation. Usually, only daily check is considered in the AMRP, whereas other types of maintenances, which are much less frequent, are typically considered in other planning operations. A feasible solution to the AMRP normally contains a

generic aircraft route during a rolling time horizon. This generic solution does not assign individual aircraft to flights explicitly. However, in the operational stage, this solution can serve as a reference for assigning individual aircrafts.

Two common objectives considered in the AMRP include a *short connect* penalty cost and a *through revenue* between connecting flights. A short connect happens when the *turn time* between two connecting flights is less than the *minimum sit time*. The turn time is the time required to unload an aircraft after its arrival at the gate and to prepare it for the next departure, and the sit time is the time for a crew to change aircrafts between two connecting flights. A short connect is undesirable because it may lead to an infeasible schedule or a high cost for the CPP. A through revenue between two connecting flights is measured by the number of passengers who stay on the same aircraft between two connecting flights. It is worth mentioning that some airlines consider the AMRP as a pure feasibility problem because the cost of AMRP is relatively small comparing with other planning operations such as FAP and CPP.

The solution methodology for the AMRP can be categorized into three approaches. The most commonly used approach in the literature is to model the AMRP as a set partitioning problem with side constraints, which was first proposed in Desaulniers et al. [1997b], Barnhart et al. [1998a]. This set-partitioning based model has been further modified and extended in Cordeau et al. [2001], Elf and Kaibel [2003], Mercier et al. [2005], Mercier and Soumis [2007]. In this approach, the decision variables represent the maintenance feasible flight sequences between two maintenance stations with a maintenance at the end. Various column generation and branch-and-price solution approaches were developed to solve this type of models. The second approach models the AMRP as an Euler tour problem or asymmetric traveling salesman problem with side constraints [Clarke et al., 1997, Gopalan and Talluri, 1998, Talluri, 1998, Boland et al., 2000]. The last and most recent approach models the AMRP as a network flow problem [Liang et al., 2011]. The network model has been shown to be very compact and scalable as it is able to solve the real life daily AMRP in a reasonable time.

6.2.3 Integration of the FAP and the AMRP

The integrated the FAP and the AMRP has continuously attracted many researchers to develop effective models and efficient solution approaches. The traditional FAP usually does not consider the feasibility of aircraft maintenance schedule. Therefore, when solving the FAP followed by the AMRP sequentially, the solution to the FAP might not be maintenance feasible for the following AMRP. Barnhart et al. [1998a] was the first to solve the integrated FAP and AMRP using a set-partitioning based model as mentioned previously. Ioachim et al. [1999] solve the weekly integrated problem with schedule synchronization constraints, which require the departure times for flights with same identifier (flight number) to be the same. However, no maintenances are considered in the model. A column generation method is developed to solve the model. A set of long-haul schedules with about 100 flights are solved within reasonable time. Recently, Haouari et al. [2009] propose a multi-commodity network flow model for a simplified version of the integrated problem, in which aircraft maintenances are not considered because of the special practice of the airline company. The computational results show that this method is able to generate near-optimal solutions within three minutes for all the schedules from TunisAir. Note this proposed model cannot be extended to the standard AMRP easily without introducing new variables and constraints. Haouari et al. [2011] propose two models, an assignment based and a set-partitioning based, to solve the same simplified version of the integrated problem with no maintenance considerations. A Bender's decomposition method is proposed to solve the assignment-based model, and a column generation method to solve the set-partitioning model. Computational results show that the branch-and-price approach performs better than Benders decomposition approach and delivers good solutions to real life test cases. In Mansour et al. [2011], the same group solves the integrated FAP and AMRP with the options of chartered aircrafts and flight retiming, and no maintenance are considered in the model. Two sequential heuristics are designed for the problem. The computational results demonstrate that the solution methods improve the current solutions used in TunisAir. Papadakos [2009] proposes an integrated model to solve the FAP, the AMRP, and the CPP simultaneously. An enhanced Benders decomposition method is combined with a column generation approach to solve the integrated model. Solutions to

realistic data sets are obtained and compared with the solutions using other methods from the literature. It is shown that the integrated approach reduces the cost significantly.

Note that most of the above papers for the integrated problem do not consider maintenance requirements explicitly in their model except Barnhart et al. [1998a], Papadakos [2009]. Without considering maintenance requirement, a network flow based model can be used to obtain a set of connecting flights for aircrafts, and no knowledge of the flight routes are necessary. On the other hand, considering maintenance requirement explicitly poses significant difficulties in both modeling and solution approaches because the flight routes information has to be built into the model.

6.2.4 Time-Space Network

The time-space network is widely used to model the airline planning operations including the FAP, the AMRP, the CPP, etc. The time-space network first appears in Hane et al. [1995] to solve FAP. In a daily time-space network (shown in Figure 6.1a), a time line represents a station, which consists a series of event nodes representing flight departure and/or arrival at the station. In order to allow connection between flights, a minimum

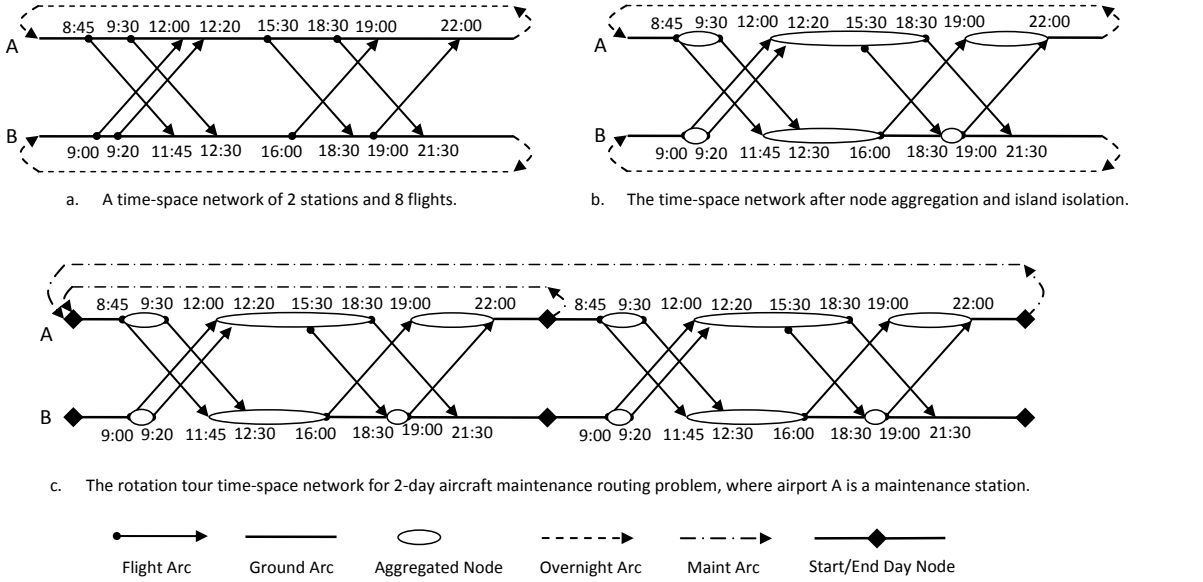


Figure 6.1: Time-space network with three stations and eight flights for FAP and Rotation-tour network for 2-day AMRP.

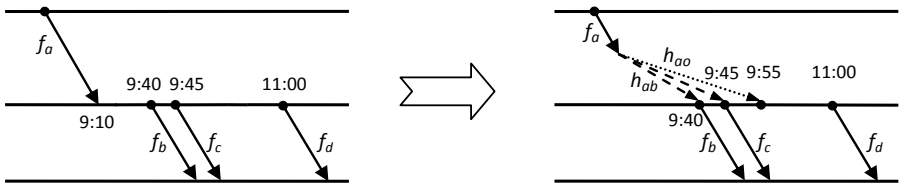
ground time is added on the actual flight arrival time for computing the time of arrival

node. In the time-space network, there are three types of arcs: ground arc, flight arc and overnight arcs. Ground arcs represent one or more aircrafts staying at the same station for a period of time. Flight arcs represent flights between airports. Overnight arcs ensure the continuity of the aircraft routing from the current planning period to the next one. With ground arcs and overnight arcs, it is possible to preserve the aircraft balance and allow all possible connections between the arrival flights and the following departure flights at a station. In Figure 6.1a, we show a daily time-space network containing eight flights between two stations, and the time progresses horizontally from left to right.

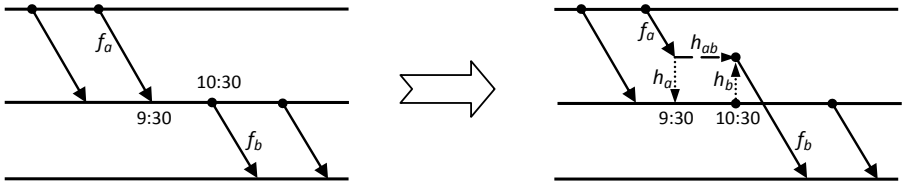
Two preprocessing methods, namely *node aggregation* and *island isolation*, proposed in Hane et al. [1995] can be used to reduce the size of time-space network. Node aggregation allows the combination of consecutive arrival nodes and subsequent consecutive departure nodes and the elimination of the unnecessary ground arcs. Island isolation can eliminate a ground arc if it is not necessary to have aircrafts on the ground arc during the specific period. Considering the example shown in Figure 6.1. There are 8 flights and hence 16 nodes in the time-space network. In Figure 6.1b, we show the network after node aggregation and island isolation, where only 6 ground arcs and 6 nodes are necessary.

Liang et al. [2011] propose a modified time-space network, namely rotation-tour network, to model the daily AMRP. They duplicate the daily time-space network for D times, where D is the maximal days allowed between two consecutive maintenances. They also remove all the overnight arcs in the traditional time-space, and create a set of time-reversible maintenance arcs to represent maintenance opportunities. The new maintenance arcs start at the end of every day in a maintenance station and end at the beginning of the same maintenance station time line. It is obvious that any flight sequences in the rotation-tour network cannot violate the D-day maintenance constraints. In Figure 6.1c, we show a 2-day rotation-tour network for the daily AMRP.

Liang et al. [2011] also propose two set of additional arcs to represent the undesired short connects and profitable connects with through revenue in the time-space network. In Figure 6.2a, we show how to model the short connects in the time-space network. For any short connect between an arrival flight and a departure flight, we construct a penalty arc to represent the short connect. Consider the example shown in Figure 6.2a, we have



a. To model two short connects between flights f_a, f_b, and flights f_a, f_c, we construct two penalty arcs h_{ab}, h_{ac}, and a zero cost arc h_{ao}. We assume the minimum turn time is 30 minutes and minimum sit time is 45 minutes.



b. To model the through connect between flights f_a and f_b, we construct a through arc h_{ab} and two zero cost arcs h_a and h_b.

Figure 6.2: Construction of penalty arcs for short connects and through arcs for through revenue connects. For illustration purpose, we assume that the end time of a flight arc is equal to the arrival time of the flight in this figure to avoid confusion. However in a general time-space network, the end time of a flight arc is normally set to the arrival time of the flight plus the minimum turn time.

two short connects between flights f_a, f_b, and flights f_a, f_c. Instead of connecting the arrival flight f_a with the ground arc directly, we let the penalty arcs h_{ab} and h_{ac} connect the arrival flight with the ground arc. The end time of a penalty arc h_{ab} (or h_{ac}) is the departure time of the short connect flight f_b (or f_c). In order to allow non-penalty connections between the arrival flight f_a and the later departure flights, we also create a zero cost arc h_{ao} connecting the arrival flight f_a with the ground arc. The end time of the zero cost arc h_{ao} equals to the arrival time of the flight f_a plus the minimum sit time.

In Figure 6.2b, we show how to model the through revenue connects. For any through connect, we build a through arc connecting the arrival flight with the departure flight. Consider the example shown in Figure 6.2b, we have a through connect between flights f_a and f_b. Instead of connecting f_a and f_b with the ground arc, we create a through arc h_{ab} connecting flights f_a and f_b. The profit of through arc h_{ab} is the number of passengers with the itinerary f_a followed by f_b. In order to allow non-through connects between f_a/f_b and other flights, we create a zero cost arc h_a/h_b to connect the f_a/f_b and the ground arc.

6.3 Weekly Rotation-Tour Time-Space Network Model

Given a weekly schedule containing F flights, the WAMRP is to find a generic cyclic aircraft route so that the overall penalty cost or (negative) through revenue is minimized. The number of aircrafts required to implement the solution route has to be less than or equal to the fleet size K. Additionally, an aircraft needs to undergo a maintenance every D day or less, and maintenances can only be performed at the set of maintenance stations denoted by M. There is a maintenance capacity Q_{mp} on p day of the week at maintenance station m, where $p \in P = \{1, 2, \ldots, 7\}$ and $m \in M$. Note the maintenance capacity of a station may vary on different day of the week (e.g., weekdays versus weekends).

In this section, we first present the weekly rotation-tour network to represent the WAMRP. We then introduce the new weekly rotation-tour network model (WRTNM) to formulate the WAMRP.

6.3.1 Construction of Weekly Rotation-Tour Network

The weekly rotation-tour network (WRTN) for the WAMRP can be constructed as follows (as shown in Figure 6.3). We first construct seven D-day time-space networks. Each D-day time-space network starts at day p and ends at day $p + D - 1$, where $p \in \{1, 2, \ldots, 7\}$. If $p + D - 1$ is greater than 7, we divide it by 7 and take the remainder to ensure the day index is smaller than or equal to 7. Without loss of generality, we assume all the day indexes calculated in the remaining of the chapter are smaller or equal to 7 by using the modular operation. We denote the D-day time-space network starting at day p, where $p \in \{1, 2, \ldots, 7\}$, as S_p network. For example, S_6 network, where $D = 3$, is the time-space network containing Saturday, Sunday, and Monday.

We define N as a set of nodes in the seven D-day time-space networks, where node $n \in N$ represents a flight departure or arrival at a station. There are three types of arcs in any S_p network: flight, ground, and connection arcs. The flight arcs and ground arcs are constructed in the same way as in the traditional time-space network. The connection arcs are constructed in the same way as in Liang et al. [2011] as discussed previously in Section 6.2.4 and Figure 6.2. It is noted that we need to construct D flight arcs, each in one of S_p

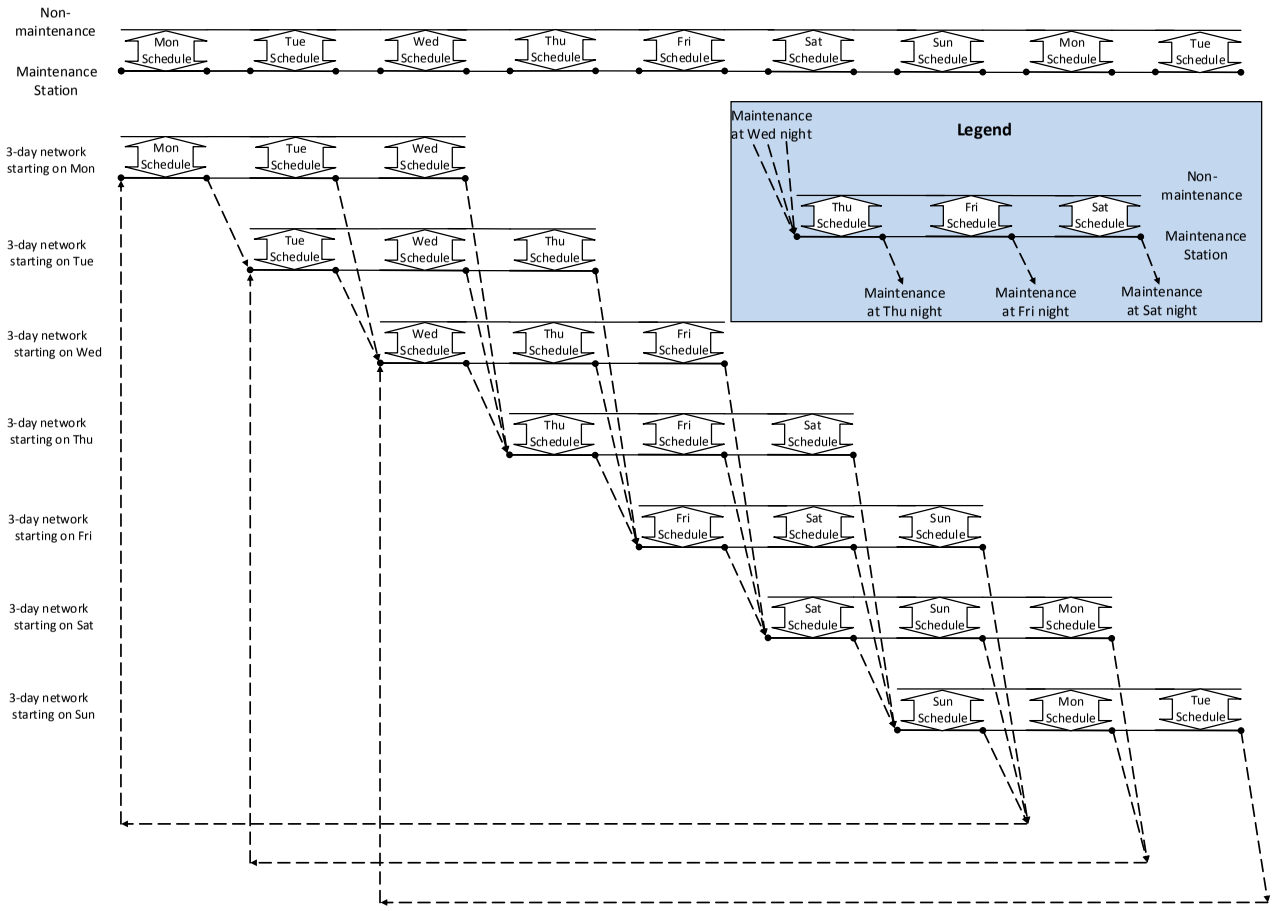


Figure 6.3: Weekly rotation-tour network for the WAMRP. The maximum days allowed between two consecutive maintenances is 3. Only the maintenance arcs, which connect seven D-day time-space networks, are shown in the figure explicitly; whereas the arcs within every D-day time-space are represented by two-sided bold arrows implicitly.

network where $p \in \{i, \ldots, i + D - 1\}$, for every flight f departing on day $i + D - 1$. For example, we need to construct 3 flight arcs for a flight departing on Saturday in S_4, S_5, and S_6 network respectively.

We also create a set of capacitated maintenance arcs connecting seven S_p networks. Specifically, a maintenance arc starts at the end of day i at maintenance station m in S_p network, where $i \in \{p, \ldots, p + D - 1\}$, and ends at the beginning time line of maintenance station m in S_{i+1} network. For example, as shown in Figure 6.3, for S_2 network (here $D = 3$) spanning on Tuesday, Wednesday, and Thursday, we create three maintenance arcs leaving this time-space network at the end of Tuesday, Wednesday, and Thursday. The maintenance arc that starts at the end of Tuesday ends at the beginning of S_3, and the maintenance arc that starts at the end of Wednesday ends at the beginning of S_4. It is important to note that "the end of day" at a station is defined as the last departure/arrival event at the station, and "the beginning of a day" is defined as the first scheduled arrival/departure time at the station. The duration between the end of a day and the beginning of the next day is normally longer than the maintenance duration. Other operational concerns such as red eye flights can be handled in a similar way as in Liang et al. [2011].

By constructing the WRTN, it is guaranteed that no aircraft rotations violate the maximum D-day maintenance constraints. This is because without going through any maintenance arcs, an aircraft only stays in a single S_p network in WRTN, which lasts for D days. In order to build a flight route lasting more than D days, a maintenance arc has to be inserted in the route. Also, it is noticed that an aircraft can fly at most D days after a maintenance, because in the WRTN an aircraft will enter a new S_p network after going through any maintenance arcs. For any maintenance station of a S_p network, there are D outgoing maintenance arcs, each at the end of day $i \in \{p, \ldots, p + D - 1\}$. Therefore, an aircraft can perform a maintenance after $d \in \{1, ..., D\}$ days flying, and no maintenance opportunities are ignored in the network.

6.3.2 Mathematical Modeling for WAMRP

To facilitate the discussion of our model, we first list all the notations as follows.

Sets, Elements, and Constants

D: the maximum days between two consecutive maintenances.

p: $\in \{1, 2, \ldots, 7\}$ represent the day of week.

S_p: the D-day time-space network starts on day p.

N: the set of nodes (events) in the WRTN, indexed by n.

N_p: the set of node in S_p network, where $p \in \{1, 2, \ldots, 7\}$.

F: the set of flights in the weekly schedule, indexed by f.

F_p: the set of flight arcs in S_p network. It is worth mentioning that F_p are the arcs in network, whereas F are the flights in reality. Therefore, $\sum_{p\in\{1,\ldots,7\}} |F_p| = D \times |F|$, because each flight appears D times in the weekly network.

H: the set of connect arcs in WRTN (either for penalty connects or for through revenue connects), indexed by h.

H_p: the set of connect arcs in S_p network.

c_h: the cost of connect arc h, where $h \in H$.

M: the set of maintenance stations, indexed by m.

G: the set of maintenance arcs in WRTN, indexed by g.

g_{mpd}: the maintenance arc at station m at the end of day p after d days flying, where $m \in M$, $p \in \{1, \ldots, 7\}$, and $d \in D$. Notice that index (mpd) uniquely define a maintenance arc in the WRTN.

Q_{mp}: the maintenance capacity at maintenance station m on day p.

L: the set of ground arcs in the WRTN, indexed by l.

l_n^+: the ground arc before node n.

l_n^-: the ground arc after node n.

K: the size of the aircraft fleet.

O: the arbitrary count time for counting the number of aircrafts.

F_O: the set of flight arcs passing the count time O in WRTN.

H_O: the set of connect arcs passing the count time O in WRTN.

L_O: the set of ground arcs passing the count time O in WRTN.

G_O: the set of maintenance arcs passing the count time O in WRTN.

Indication Parameters

α^+_{fpn}: the binary indicator such that $\alpha^+_{fpn} = 1$ if flight f in S_p network starts at node n, and 0 otherwise.

α^-_{fpn}: the binary indicator such that $\alpha^-_{fpn} = 1$ if flight f in S_p network ends at node n, and 0 otherwise.

γ^+_{hpn}: the binary indicator such that $\gamma^+_{hpn} = 1$ if arc h in S_p network starts at node n, and 0 otherwise.

γ^-_{hpn}: the binary indicator such that $\gamma^-_{hpn} = 1$ if arc h in S_p network ends at node n, and 0 otherwise.

β^+_{mpdn}: the binary indicator such that $\beta^+_{mpdn} = 1$ if maintenance arc g_{mpd} starts at node n, and 0 otherwise.

β^-_{mpdn}: the binary indicator such that $\beta^-_{mpdn} = 1$ if maintenance arc g_{mpd} ends at node n; and 0 otherwise.

Variables

x_{fp}: the binary variable such that $x_{fp} = 1$ if flight f is flown in S_p network, and 0 otherwise.

y_{hp}: the binary variable such that $y_{hp} = 1$ if connect arc h is in S_p network, and 0 otherwise.

z_{mpd}: the integer variable representing the number of aircrafts on maintenance arc g_{mpd} in the weekly network.

w_l: the integer variable representing the number of aircrafts on ground arc l.

Given the above notations, a WRTNM for the WAMRP is presented as follows.

$$(WRTNM) \quad \min \sum_{h\in H_p} \sum_{p\in\{1,\ldots,7\}} c_h y_{hp} \tag{6.1}$$

$$\text{s.t.} \sum_{p:f\in F_p} x_{fp} = 1 \qquad \forall\ f \in F, \tag{6.2}$$

$$\sum_{p:f\in F_p} \sum_{f\in F} \alpha^{+}_{fpn} x_{fp} + \sum_{m\in M} \sum_{d\in D} \sum_{p\in\{1,\ldots,7\}} \beta^{+}_{mdpn} z_{mpd} + \sum_{p:h\in H_p} \sum_{h\in H} \gamma^{+}_{hpn} y_{hp} + w_{l^{+}_n} = \sum_{p:f\in F_p} \sum_{f\in F} \alpha^{-}_{fpn} x_{fp} + \sum_{m\in M} \sum_{d\in D} \sum_{p\in\{1,\ldots,7\}} \beta^{-}_{mdpn} z_{mpd} + \sum_{p:h\in H_p} \sum_{h\in H} \gamma^{-}_{hpn} y_{hp} + w_{l^{-}_n} \qquad \forall\ n \in N, \tag{6.3}$$

$$\sum_{d\in D} z_{mpd} \le Q_{mp} \qquad \forall\ m \in M,\ p \in \{1,\ldots,7\}, \tag{6.4}$$

$$\sum_{f_p\in F_O} x_{fp} + \sum_{g_{mpd}\in G_O} z_{mpd} + \sum_{l\in L_O} w_l \le K \tag{6.5}$$

$$x_{fp}, y_{hp} \in \{0,1\} \qquad \forall\ p \in \{1,\ldots,7\},\ \forall\ f \in F_p,\ \forall h \in H_p, \tag{6.6}$$

$$z_{mpd} \in \{0,1,...,Q_{mp}\} \qquad \forall\ m \in M,\ \forall\ p \in \{1,\ldots,7\},\ \forall d \in D, \tag{6.7}$$

$$w_l \in Z^{+} \qquad \forall\ l \in L. \tag{6.8}$$

The objective function in Eq.(6.1) minimizes the total penalty cost. The assignment constraints in Eq.(6.2) ensure that each flight is covered once in the rotation solution. The flow balance constraints in Eq.(6.3) ensure that the number of inbound aircrafts is equal to the number of outbound aircrafts at each node. The capacity constraints in Eq.(6.4) ensure that the number of aircrafts being maintained at each station is not greater than the station's capacity. The fleet size constraint in Eq.(6.5) ensures that the total number of aircrafts used is not greater than the size of the fleet. The constraints in Eqs.(6.6)-(6.8) are the binary, integrality constraints for variables. It is interesting to note that the integrality constraints in Eqs.(6.7)-(6.8) can be relaxed because of flow balance constraints in Eq.(6.3) and binary constraints in Eq.(6.6). The total number of variables and constraints in WRTNM is $\mathbf{O}(D|F|)$, and the number of non-zero entries in the problem matrix is $\mathbf{O}(D|F|^2)$.

Note that the solution to WRTNM does not provide details of routing sequences. Instead, the solution provides a set of connecting flight arcs, ground arcs, connection arcs, and maintenance arcs. We can use a general Eulerian tour algorithm [Chartrand and Oellermann, 1993] to construct generic routing sequences in polynomial time. To construct an Eulerian tour, we first convert the set of connecting arcs in the solution into a simple digraph. In particular, we replace a solution arc with value π by π parallel arcs, each with capacity one. Then we use the Euler tour algorithm to find a rotation, which covers all the arcs in the graph. We can also extract the individual flight strings from each S_p network by traveling the flight arcs, ground arcs, and connection arcs in that S_p network.

6.3.3 Variable Fixing Heuristic

Based on our preliminary computational experiment, the integrality gaps between optimal LP relaxations and integer solutions of the WRTNM are less than 0.1%. This results matches the observations from most time-space network based model from airline planning problems, e.g., [Hane et al., 1995, Barnhart et al., 1998a, Sherali et al., 2006b]. Also, the number of cuts added by CPLEX at the root node of the branch-and-bound tree [ILOG, 2002] is quite small comparing the problem matrix size, and these cuts improve the objective value insignificantly. The strong LP relaxation of WRTNM encourages us to apply a simple variable fixing heuristic to simplify the problem and obtain good solutions in a timely fashion.

Here, we propose an iterative heuristic to fix the variables based on their values in the LP solution. Given a fractional LP solution, we first fix all the variables whose values are equal to 1. Because of the constraints in Eq.(6.2), we fix the corresponding non-selected variables to 0. Then we resolve the simplified MIP using CPLEX solver. If no integer solution is obtained within a limited time, we then solve the LP relaxation of the model and fix the variables whose value are greater than $1 - \sigma$, where σ is the step size of the variable fixing heuristic. It is important to note that by rounding up multiple fractional variables, the problem might become infeasible. When an infeasibility is detected, we reduce the σ by half and the algorithm continues. The detailed algorithm is shown in Figure 6.4.

Variable Fixing Heuristic

Input: WRTNM problem matrix;
Heuristic step length σ;
Predefined IP solution time limit τ;
$K = 0$;
Solve the LP relaxation of WRTNM;
WHILE Current LP solution is fractional
  Select a set of variables $\hat{X}$ such that $x_{fp} \geq 1 - K \times \sigma$ for all $x_{fp} \in \hat{X}$;
  $K = K + 1$;
  Set lower bound of x_{fp} to 1, $\forall x_{fp} \in \hat{X}$;
  Set upper bound of $x_{f\bar{p}}$ to 0, $\forall x_{f\bar{p}}$ such that $x_{fp} \in \hat{X}, \bar{p} \neq p$;
  Solve the IP of restricted WRTNM;
  If feasible solution of restricted WRTNM is obtained within time τ;
    If optimal solution of restricted WRTNM is obtained;
      Select the best integer solutions available, algorithm ends;
    Else If feasible but not optimal solution is obtained;
      Record the solution if it is better than the current best solution;
    End If;
  End If;
  Solve the LP relaxation of the restricted WRTNM;
  If the restricted WRTNM is infeasible;
    Restore lower bounds for $x_{fp} \in \hat{X}$ and upper bounds for $x_{f\bar{p}}$ such that $x_{fp} \in \hat{X}, \bar{p} \neq p$;
    $\sigma = \sigma/2$;
    $K = K - 1$;
  End If
End While
Select the best integer solutions available, algorithm ends;

Figure 6.4: Pseudo-code of variable fixing heuristic

6.4 Integrated WFAP with WAMRP

It has been pointed out in Section 6.2.2 that the results of FAP do not guarantee the feasibility of the AMRP, because the maintenance requirements are not considered in FAP. Therefore, it is necessary to study the integrated WFAP with WAMRP. The network representation of WRTNM can be extended to the integrated WFAP and WAMRP. The objective of WFAP is to assign the flights to aircrafts of several fleets so that the total profit is maximized. To model the integrated WFAP with WAMRP, we introduce a new set of available fleets I, which increases the dimensionality of variables with fleet index. In other words, we create a WRTN for each of the fleet $i \in I$. To facilitate the discussion, we define the following additional notations.

Sets, Elements, and Constants

I**:** the set of fleets, indexed by i.

D^i**:** the maximum days between maintenances for fleet i.

S_p^i**:** the D-day time-space network starts on day p for fleet i.

N^i**:** the set of nodes (events) in the WRTN for fleet i, indexed by n.

N_p^i**:** the set of node in S_p^i network, where $p \in \{1, 2, \ldots, 7\}$.

F_p^i**:** the set of flight arcs in S_p^i network.

r_f^i**:** the revenue of assigning flight f to fleet i.

M^i**:** the set of maintenance stations for fleet i, indexed by m.

G^i**:** the set of maintenance arcs in WRTN for fleet i, indexed by g.

g_{mpd}^i**:** as the maintenance arc at station m at the end of day p after d days flying for fleet i.

Q_{mp}^i**:** the maintenance capacity at station m on day p for fleet i.

L^i**:** the set of ground arcs in the WRTN for fleet i, indexed by l.

K^i**:** the size of the aircraft fleet i.

F_O^i**:** the set of flight arcs passing the count time O in WRTN for fleet i.

L_O^i**:** the set of ground arcs passing the count time O in WRTN for fleet i.

G_O^i**:** the set of maintenance arcs passing the count time O in WRTN for fleet i.

Indication Parameters

α_{fpn}^{i+}**:** is 1 if flight f in S_p^i network starts at node n for fleet i; and 0 otherwise.

α^{i-}_{fpn}: is 1 if flight f in S^i_p network ends at node n for fleet i; and 0 otherwise.

β^{i+}_{mpdn}: is 1 if maintenance arc g^i_{mpd} starts at node n for fleet i; and 0 otherwise.

β^{i-}_{mpdn}: is 1 if maintenance arc g^i_{mpd} ends at node n for fleet i; and 0 otherwise.

Variables

x^i_{fp}: is a binary variable such that $x^i_{fp} = 1$ if flight f is flown in S^i_p network, and 0 otherwise.

z^i_{mpd}: is an integer variable representing the number of aircrafts on maintenance arc g^i_{mpd} in the WRTN for fleet i.

w^i_l is an integer variable representing the number of aircrafts on ground arc l for fleet i.

The MIP formulation of WFAP and WAMRP using WRTNM is given by

$$\max \sum_{i\in I}\sum_{p:f\in F^i_p}\sum_{f\in F} r^i_f x^i_{fp} \tag{6.9}$$

$$\text{s.t.} \sum_{i\in I}\sum_{p:f\in F^i_p} x^i_{fp} = 1 \qquad \forall\ f \in F, \tag{6.10}$$

$$\sum_{p:f\in F^i_p}\sum_{f\in F} \alpha^{i+}_{fpn} x^i_{fp} + \sum_{m\in M^i}\sum_{d\in D^i}\sum_{p\in\{1,\ldots,7\}} \beta^{i+}_{mdpn} z^i_{mpd} + w^i_{l^+_n} = \sum_{p:f\in F^i_p}\sum_{f\in F} \alpha^{i-}_{fpn} x^i_{fp} + \sum_{m\in M^i}\sum_{d\in D^i}\sum_{p\in\{1,\ldots,7\}} \beta^{i-}_{mdpn} z^i_{mpd} + w^i_{l^-_n} \qquad \forall\ n \in N^i, \forall i \in I, \tag{6.11}$$

$$\sum_{d\in D^i} z^i_{mpd} \le Q^i_{mp} \qquad \forall\ m \in M^i,\ p \in \{1,\ldots,7\},\ \forall i \in I, \tag{6.12}$$

$$\sum_{f_p\in F^i_O} x^i_{fp} + \sum_{g^i_{mpd}\in G^i_O} z^i_{mpd} + \sum_{l\in L^i_O} w^i_l \le K^i \qquad \forall i \in I, \tag{6.13}$$

$$x^i_{fp} \in \{0,1\} \qquad \forall\ f \in F^i_p,\ \forall\ p \in \{1,\ldots,7\},\ \forall i \in I, \tag{6.14}$$

$$z^i_{mpd} \in \{0,1,...,Q^i_{mp}\} \qquad \forall\ m \in M^i,\ \forall\ p \in \{1,\ldots,7\}, \forall d \in D^i, \forall i \in I, \tag{6.15}$$

$$w^i_l \in Z^+ \qquad \forall\ l \in L,\ \forall i \in I. \tag{6.16}$$

The objective function in Eq. (6.9) maximize the total profit of all the flights. Here, we do not consider the cost of WAMRP but only the profit r^i_f of the flights, because the cost of WAMRP is negligible comparing to the cost of WFAP. However, the connection costs and maintenance costs can be included in the integrated model easily as in WRTNM. The

constraints in Eq. (6.10) ensure each flight is assigned to a single fleet. The constraints in Eq. (6.11) are the flow balance constraints, which ensure the flow balance within the WRTN of any fleet. The constraints in Eq. (6.12) ensure the number of maintenance performed on day p, at maintenance station m, for fleet i does not exceed the maintenance capacity. The constraints in Eq. (6.13) are the plan count constraints for any fleet i. Because the size of the integrated model is compact, we use commercial solvers to solve the integrated model directly. If no satisfied solution is obtained in reasonable time, we use variable fixing heuristic presented in Section 6.3.3 to obtain the solution in a timely fashion.

6.5 Computational Results

In this section, we report empirical results of the proposed models. All test problems were solved using an Intel Dual Core 2.79GHz workstation with 3 GB of memory running on Windows XP platform. No parallel processes are implemented. Computational times reported in this section were obtained from the desktop's internal timing calculations, which include time used for preprocessing, perturbation, and postprocessing. All the mathematical modeling and algorithms were implemented in C++ language. All LP and MIP problems were solved using a CPLEX callable library version 10.0. We stop the solution algorithm if the optimality gap of a solution is less than 0.05%.

6.5.1 Computational Experience for WAMRP

The test instances used to benchmark WRTNM in this study are constructed based on the real life operational aircraft schedule from a major US airline, which are publicly accessible on the airline websites. In particular, we construct total eight test cases, in which the first six test cases are constructed from six different fleets respectively, and the last two are constructed by combining flights from multiple fleets. To construct the first six test cases, we first select six representable fleets, and extract the corresponding airline flights within a particular week of the published schedule. Because what we obtained is an operational schedule, it does not guarantee the cyclic constraints geographically, i.e., the number of incoming flights to an airport is not equal to the number of outgoing flights in

a particular week. In order to ensure the feasibility of WAMRP, we develop an Eulerian tour algorithm [Chartrand and Oellermann, 1993] to obtain the maximal set of flights, which are maintenance feasible. We obtain the fleet size information from the airline website. Since we have no information on maintenance stations, we assume an airport is a maintenance station if the number of flights departure/arrival on that airport is greater than a threshold. We minimize the total number of maintenance stations by increasing this threshold and maintaining the feasibility of the schedule at the same time. For example, for Boeing 757-200 flights, we assume an airport is a maintenance station if the pairs of in/out flights at the airport are greater than 30.

We combine the schedules of two fleets AIR-320 and 737-800 to create a larger test case SIM-001, and combine the schedules of three fleets 757-200, AIR-320, and 737-800 to create the largest test case SIM-002 just for testing purpose. It is interesting to note that the last test case SIM-002 is about the size of the world's largest fleet, Southwest Airline Boeing 737-700 (350 aircrafts). For all the test cases, we assume the maximum days between maintenance are 4, which is a realistic estimation of the airline regulation. The detailed information of eight test cases are shown in Table 6.1.

Test Cases	Flights	Fleet Size	Redeyes	Airports	Maint Station
757-300	352	22	14	19	15
737-500	680	40	0	38	19
CRJ-700	1,000	56	0	47	22
757-200	1,428	88	54	31	6
AIR-320	2,110	123	39	65	34
737-800	2,240	122	54	84	12
SIM-001	4,372	245	93	109	41
SIM-002	5,778	333	157	111	44

Table 6.1: Characteristics of eight test cases.

Table 6.2 presents the solution information of WRTNM with objective of minimizing the total penalty cost of short connects. *S-Connect* records the number of all possible short connects in the schedule. In our computation, we assume the minimum turn time for aircraft is 30 minutes and minimum sit time for crew is 45 minutes. We set the short

connect cost to be the short time (*short time = min(minimum sit time - real connection time, 0)*) of the connect. *Cols* and *Cols-P* represent the number of variables before and after preprocessing. Similarly, *Rows* with *Row-P*, and *NonO* with *NonO-P* represent the numbers of constraints, and non-zero entities in the problem matrix before and after preprocessing respectively. *LP* represents the solution values of LP relaxation of WRTNM. *IP* represents the integer solution to WRTNM. *Gap* represents the LP-IP gap of the model. *Time* records the total solution time of the problem. *Cut* records the number of cuts added by CPLEX solver when solving the model. *Nodes* records the number of branch-and-bound nodes CPLEX solver explored to obtain the solution.

Test Cases	S-Connect	Cols	Cols-P	Rows	Rows-P	Non0	Non0-P	LP	IP	Gap(%)	Time (sec)	Cuts	Nodes
757-300	21	5,613	3,737	4,243	2,367	13,343	9,591	0	0*	0	3	4	1
737-500	148	10,810	7,786	7,815	4,791	26,006	19,958	102	102*	0	7	1	1
CRJ-700	472	15,003	11,471	9,983	6,451	36,739	29,675	1,789	1,789*	0	12	17	1
757-200	412	18,777	12,509	12,995	6,727	45,183	32,647	25	25*	0	15	20	1
AIR-320	789	30,123	21,811	20,215	11,903	73,411	56,787	216	216*	0	40	7	1
737-800	656	31,011	20,971	22,061	12,021	74,221	54,141	82	82*	0	45	18	1
SIM-001	1,667	59,150	41,186	39,372	21,408	144,040	108,112	105	112*	6.250	309	39	1
SIM-002	3,521	82,666	59,746	50,331	27,411	205,475	159,635	64	64*	0	422	7	1

Table 6.2: Performance characteristics of WRTNM when minimizing the total penalty cost from short connects. (*) denotes the optimal solutions.

We can see from Table 6.2 that WRTNM produces optimal solutions to all test cases within about seven minutes. Also, it is important to note that the LP bounds provided by WRTNM are very tight. Zero LP-IP gaps are obtained for seven out of eight test cases. The tight LP bounds again help to find optimal solutions quickly. The number of cuts generated by CPLEX to WRTNM is very minimum comparing to the problem sizes. Also, the optimal integer solutions are found at the root node of the CPLEX branch-and-bound trees for all test cases. Finally, the preprocessing procedure reduces the problem sizes by about 1/3 for all the test cases.

Table 6.3 presents the solution information of WRTNM with objective of maximizing the total profit from through revenue. Similar information is recorded in Table 6.3 as in Table 6.2. Particularly, we record the number of possible through connects in the second column of Table 6.3. Since we do not have real life through connects information, we

assume a through connect only occurs on the airline hubs for one stop service with spoke-hub-spoke structure. In particular, we build a through connect between any arrival flight and departure flight if the connection time between them is between 45 minutes to 3 hours. We randomly generate a through value for any through connect such that the accumulative through value from/to any flight does not exceed the aircraft's capacity.

Test Cases	T-Connect	Cols	Cols-P	Rows	Rows-P	Non0	Non0-P	LP	IP	Gap(%)	Time (sec)	Cuts	Nodes
757-300	0	5,609	3,605	4,323	2,319	13,251	9,243	0	0*	0	2	11	1
737-500	1,222	15,554	11,946	8,263	4,655	44,678	37,462	1,755	1,755*	0	25	10	1
CRJ-700	1,496	20,583	15,315	11,467	6,199	57,979	47,443	2,087	2,087*	0	14	13	1
757-200	4,099	34,314	26,379	13,995	6,059	107,134	91,262	4,724	4,724*	0	40	11	1
AIR-320	5,260	49,856	38,572	22,279	10,995	151,436	128,868	5,921	5,921*	0	189	8	1
737-800	10,167	72,570	60,478	23,573	11,481	240,126	215,942	6,814	6,813*	0.015	89	58	1
SIM-001	10,647	98,979	74,984	43,528	19,536	301,795	253,811	9,615	9,608	0.050	508	40	1
SIM-002	8,243	108,030	74,922	56,855	23,747	307,956	241,740	8,153	8,153*	0	1,465	10	1

Table 6.3: Performance characteristics of WRTNM when maximizing the total revenue from through connects. (*) denotes the optimal solutions.

As we can see from Table 6.3, we obtain optimal solutions to 6 out of 8 test cases, except SIM-001 within 0.05% optimal. For the first 6 test cases, we are able to obtain optimal solutions in less than 3 minutes computational time. Similar with the results shown in Table 6.2, the LP bounds provided by WRTNM are very tight (within 0.05% optimal). The cuts added by CPLEX at the root node of the branch-and-bound tree is minimal and do not improve the LP objective value significantly. Optimal and near optimal solutions are found at the root node of the branch-and-bound tree.

For test cases SIM-001 and SIM-002, the computational times are longer than the other test cases significantly. Therefore, we test the variable fixing heuristic. Here, we set the step size $\sigma = 0.1$ and time limit for restrict IP $\tau = 300$ seconds. The computational results is recorded in Table 6.4.

Test Cases	Iteration	Threshold	Variables Up	Variables Down	Remaining Binaries	LP	IP	Time (sec)
SIM-001	1	1.0	1,932	25,125	32,669	9,615	9,608	270
SIM-002	1	1.0	3,485	24,606	29,040	8,153	8,153*	222

Table 6.4: Performance characteristics of variable fixing heuristic when maximizing the total revenue from through connects. (*) denotes the proven optimal solutions to the current restricted WRTNM.

As we can see from Table 6.4, the variable fixing heuristic is very efficient in solving large WRTNM problems. We fix more than 45% binary variables in the first iteration of the variable fixing heuristic, and obtain the solution to SIM-001 within 0.05% optimal and optimal solution to SIM-002 in less than 5 minutes.

6.5.2 Computational Experience for Integrated WFAP and WAMRP

The test cases used for the integrated model are constructed from multiple fleet schedules presented in Table 6.1. As we can see from the integrated model in Eqs. (6.9)-(6.16), the size of the integrated model primarily depends on both the number of flights in the schedule, and the number of fleets in WFAP. Therefore, we vary these characters when generating the integrated test cases. Specifically, we construct 2 sets of test cases, one set with 4 fleets and the other set with 8 fleets. There are five instances in the 4-fleet test set and four instances in the 8-fleet test set. We assume the demand for every flight is uniformly distributed between the 80 to 320. We also assume the capacities of the fleets in 4-fleet test set are 110, 170, 230, and 290 with 60 seats difference between two adjacent fleets. Similarly, we assume the fleet capacities for 8-fleet test set are from 95 to 305 with 30 seats difference between two adjacent fleets. The number of aircrafts in each fleet is same for all test cases. The profit obtained by assigning flight f to fleet i is computed by the following equation.

$$r_f^i = \min\{dmd_f, cap_i\} \times dur_f - 0.1 \max\{0, dmd_f - cap_i\} \times dur_f \tag{6.17}$$

Here, dmd_f is the randomly generated demand for flight f, cap_i is the capacity of aircraft fleet i, and dur_f is the flying time of flight i. The profit r_f^i in Eq. (6.17) contains two part: the first part computes the revenue, and the second part computes the loss sale penalty. For simplicity, we only consider flight-based profit, but not itinerary-based profit.

In Table 6.5, we present the nine test cases used to test the proposed integrated model. As we can see from Table 6.5, the numbers of flights vary approximately from 1000 to 2000, which are about the sizes of small to medium airlines. We also list the schedules used to construct the test cases in the last column of the Table 6.5.

Test Cases	Flights	Aircrafts	Fleets	Origin Flights
INT1-4	1,032	62	4	757-300, 737-500
INT2-4	1,352	78	4	757-300, CRJ-700
INT3-4	1,686	96	4	737-500, CRJ-700
INT4-4	1,780	110	4	757-200, 757-300
INT5-4	2,108	128	4	737-500, 757-200
INT1-8	1,032	62	8	757-300, 737-500
INT2-8	1,352	78	8	757-300, CRJ-700
INT3-8	1,686	96	8	737-500, CRJ-700
INT4-8	1,780	110	8	757-200, 757-300

Table 6.5: Characteristics of ten integrated WFAP and WAMRP test cases.

In our computational study, we assume every flight is eligible to be assigned to any fleet. In other words, we construct $|I|$ rotation-tour networks, and each network contains all flights F. In reality, if it is infeasible or unattractive to assign flight f to fleet i (e.g., the flight distance of f might be too long for some small fleet; the demand is too small for some fleet with large capacity), we can exclude flight f in rotation network for fleet i to reduce the size of the integrated model. We set the computational time to be three hours (10,800 seconds). We stop the CPLEX MIP solver if the optimality gap of CPLEX is less than 0.2%.

Test Cases	Cols	Rows	Non0	LP	LP Time (sec)	IP	IP Time (sec)	Gap(%)	Cuts	Nodes
INT1-4	40,692	22,752	104,492	138,085	77	138,085*	219	0.00	3	1
INT2-4	51,008	28,212	130,712	152,586	110	152,593*	309	0.00	5	1
INT3-4	62,252	34,138	159,452	175,975	156	175,991	8,543	0.01	13	41
INT4-4	54,708	26,240	143,136	194,392	245	194,392*	812	0.00	2	1
INT5-4	67,620	33,560	175,944	250,814	231	250,863	10,800	0.02	3	11
INT1-8	81,384	44,472	208,984	137,483	101	137,533	10,800	0.04	2	27
INT2-8	102,016	55,072	261,424	158,472	180	-	10,800	-	-	-
INT3-8	124,504	66,590	318,904	178,018	515	-	10,800	-	-	-
INT4-8	109,416	50,700	286,272	210,064	617	-	10,800	-	-	-

Table 6.6: Performance characteristics of integrated WFAP with WRTNM using CPLEX directly. (*) denotes the optimal solutions.

In Table 6.6, we present the solutions of integrated model using CPLEX directly. As we can see from Table 6.6, the weekly integrated model can solve all the 4-fleet test cases within 0.02% optimal. However, the computational time increase drastically from less than 4 minutes to 3 hours when the sizes of the 4-fleet test cases increase. Also, the number of

fleets effects the computational time greatly. The integrated model can only obtain one feasible solution for four 8-fleet test cases, and the computational time reaches the time limit of 3 hours. On the other hand, we notice the LP gaps provided by the integrated model are very tight, which matches our observation for WRTNM.

To obtain good solutions in reasonable time, we apply the variable fixing heuristic to the integrated weekly model. In Table 6.7, we present the solutions of integrated model using variable fixing heuristic. As we can see, we obtain the near optimal solutions to all nine

Test Cases	Iteration	Threshold	Var Up	Var Down	IP	IP Time (sec)	Gap(%)	Cuts	Nodes
INT1-4	1	1.0	339	5,061	138,085*	93	0.00	3	1
INT2-4	1	1.0	292	4,380	152,593*	95	0.00	7	1
INT3-4	1	1.0	340	5,100	175,996	167	0.01	15	1
INT4-4	1	1.0	482	7,170	194,392*	778	0.00	3	1
INT5-4	1	1.0	407	6,073	250,835	1,191	0.01	3	7
INT1-8	3	0.8	231	7,153	137,526	3,588	0.03	7	11
INT2-8	4	0.7	368	11392	158,727	2,014	0.16	1	2
INT3-8	3	0.8	401	12,431	178,311	1,909	0.16	2	1
INT4-8	3	0.8	456	14,072	213,003	10,800	1.38	4	1

Table 6.7: Performance characteristics of integrated WFAP with WRTNM using CPLEX directly. (*) denotes the optimal solutions.

test cases. Particularly, for 8 out of 9 test cases, the optimality gaps are within 0.2%. For test case INT4-8, the optimality gap is 1.38%. Also, by applying variable fixing heuristic, the computational times are reduced drastically. Specifically, for 4-fleet test cases, we only fix those flight variables with $x^i_{fp} = 1$ in the IP. We can obtain optimal and near optimal solutions within 20 minutes. We also notice that for 8-fleet test cases, the variable fixing thresholds are 0.7 to 0.8. That means we round up x^i_{fp} to 1 if $x^i_{fp} \geq 0.8$ or 0.7. Because of the variable rounding up, the LP solutions of the integrated model only increase marginally. We are able to obtain very good IP solutions with small optimality gaps for these test cases. Overall, by fixing 20% to 28% of the flights, we can quickly obtain very good IP solutions for the integrated model.

6.6 Conclusion

In this chapter, we present a new weekly rotation-tour network representation for the WAMRP. Based on this representation, we propose a new mixed-integer linear programming formulation for the WAMRP, namely *Weekly Rotation-Tour Network Model (WRTNM)*. To assess the performance of WRTNM, we test the model using eight test cases. The computational results show that the proposed model is very compact and scalable, and is able to find the optimal solutions to schedule with 5,700 flights and 330 aircrafts in minutes. We also propose an integrated model to solve the WFAP and the WAMRP simultaneously. A simple variable fixing heuristic is used to solve the integrated model efficiently. We test the integrated model on nine self-constructed cases. The computational results show that the integrated model generates near optimal solutions to the schedules less 2,000 flights and 120 aircrafts, approximately a medium-sized airline, in reasonable time. The computational results show that WRTNM and the integrated model provide good LP relaxation bounds for all test cases.

The current WRTNM can only handle simple cost structures. Extending the proposed network representation for more complex requirement and/or cost will be an interesting future research direction. The compact formulation of WRTNM might also be beneficial to the integration of WAMRP with other planning operations such as schedule design problem and CPP. Evaluating the new integrated problems with WRTNM might be anther interesting future research direction. Finally, the proposed WRTN representation might facilitate researchers for various weekly planning problems in other areas of transportation, scheduling, and networking.

Bibliography

S. AhmadBeygi, A. Cohn, and M. Weir. An integer programming approach to generating airline crew pairings. *Computers & Operations Research*, 36(4):1284–1298, 2009.

R. K. Ahuja, T. L. Magnati, and J. B. Orlin. *Network Flows: Theory, Algorithms and Appliations*. Prentice Hall, 1993.

R. Anbil, R. Tanga, and E. L. Johnson. A global approach to crew-pairing optimization. *IBM Systems Journal*, 31(1):71–78, 1992.

Y. P. Aneja. An integer linear programming approach to the Steiner problem in graphs. *Networks*, 10(2):167–178, 1980.

M. Ball and G. Lulli. Ground delay programs: Optimizing over included flight set based on distance. *Air Traffic Control Quarterly*, 12:1–25, 2004.

C. Barnhart and R. Shenoi. An approximate model and solution approach for the long-haul crew pairing problem. *Transportation Science*, 32(3):221–231, 1998.

C. Barnhart, E. L. Johnson, G. L. Nemhauser, , M. W. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316–329, 1998a.

C. Barnhart, E. L. Johnson N. L. Boland, L. W. Clark, G. L. Nemhauser, and R. G. Shenoi. Flight string model for aircraft fleeting and routing. *Transportation Science*, 32 (3):208–220, 1998b.

C. Barnhart, C. A. Hane, , and P. H. Vance. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research*, 48(2): 318–326, 2000.

C. Barnhart, A. Cohn, E. L. Johnson, D. Klabjan, G. L. Nemhauser, and P. H. Vance. Airline crew scheduling. In R. W. Hall, editor, *Handbook of Transportation Science*, pages 57–93. Kluwer Scientific Publishers, Boston, 2003.

M. S. Bazaraa, J. J. Jarvis, and H. D. Sherali. *Linear Programming and Network Flows*. John Wiley & Sones, 1990.

N. Belanger, G. Desaulniers, F. Soumis, J. Desrosiers, and J. Lavigne. Weekly airline fleet assignment with homogeneity. *Transportation Research B*, 40(4):306–318, 2006.

J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Methamatik*, 4:238–252, 1962.

D. Bersimas, G. Lulli, and A. Odoni. The air traffic flow management problem: an integer optimization approach. In *Proceedings of 13th International Conference, IPCO*, pages 36–46, 2008.

D. Bertsimas and S. Patterson. The air traffic flow management problem with enroute capacities. *Operations Research*, 46(3):406–422, 1998.

D. Bertsimas and S. Patterson. The traffic flow management rerouting problem in air traffic control: A dynamic network flow approach. *Transportation Science*, 34:239–255, 2000.

N. L. Boland, L. W. Clarke, and G. L. Nemhauser. The asymmetric traveling salesman problem with replenishment arcs. *European Journal of Operational Research*, 123(2): 408–427, 2000.

K. Boubaker, G. Desaulniers, and I. Elhallaoui. Bidline scheduling with equity by heuristic dynamic constraint aggregation. *Transportation Research B*, 44(1):50–61, 2010.

J. Bramel and D. Simchi-Levi. On the effectiveness of set covering formulations for the vehicle routing problem with time windows. *Operations Research*, 45(2):295–301, 1997.

T. A. Brewer. Algorithms for scheduling flights in the northern pacific airspace. Master's thesis, Industrial and Systems Engineering Department, Rutgers, 2005.

L. S. Buriol, M. G. C. Resende, C. C. Ribeiro, and M. Thorup. A hybrid genetic algorithm for the weight setting problem in OSPF/IS-IS routing. *Networks*, 46(1):36–56, 2005.

L. S. Buriol, M. G. C. Resende, C. C. Ribeiro, and M. Thorup. Survivable IP network design with OSPF routing. *Networks*, 49(1):51–64, 2007.

A. Chabrier. Vehicle routing problem with elementary shortest path based column generation. *Computers & Operations Research*, 33(10):2972–2990, 2006.

W. Chaovalitwongse, P. M. Pardalos, and O. A. Prokopyev. A new linearization technique for multi-quadratic 0-1 programming problems. *Operations Research Letters*, 32(6): 517–522, 2004.

G. Chartrand and O. Oellermann. *Applied and Algorithmic Graph Theory*. McGraw-Hill, 1993.

A. Chen, D. Lee, and P. Sinha. Efficient multicasting over large-scale wlans through controlled association. *Computer Networks*, 53(1):45–59, 2009.

Z.-L. Chen and W. B. Powell. A column generation based decomposition algorithm for parallel machine just-in-time scheduling problem. *European Journal of Operational Research*, 116(2):220–232, 1999a.

Z.-L. Chen and W. B. Powell. Solving parallel machine scheduling problems by column generation. *INFORMS Journal on Computing*, 11(1):79–94, 1999b.

Z.-L. Chen and W. B. Powell. Exact algorithms for scheduling multiple families of jobs on parallel machines. *Naval Research Logistics*, 50(7):823–840, 2003.

L. W. Clarke, E. L. Johnson, G. L. Nemhauser, and Z. Zhu. The aircraft rotation problem. *Annals of Operations Research*, 69(0):33–46, 1997.

A. Cohn and C. Barnhart. Improving crew scheduling by incorporating key maintenance routing decisions. *Operations Research*, 51(3):387–396, 2003.

J. Cordeau, G. Stojkoviac, F. Soumis, and J. Desrosiers. Benders decomposition for simultaneous aircraft routing and crew scheduling. *Transportation Science*, 35(4):375–388, 2001.

T. H. Corman, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithm.* McGraw-Hill Science/Engineering/Math, 2nd edition, 2001.

E. Danna and C. Le Pape. Branch-and-price heuristics: a case study on the vehicle routing problem with time windows. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column Generation*, pages 331–358. Springer, 2005.

G. B. Dantzig and Philip. Decomposition principle for linear programs. *Operations Research*, 8(1):101–111, 1960.

P. Datta and A. K. Somani. Diverse routing for shared risk resource groups (SRRG) failures in WDM optical networks. In *Proc. of the First International Conference on Broadband Networks (BROADNETS)*, pages 120–129, October 2004.

H. Dawid, J. Konig, and C. Strauss. An enhanced rostering model for airline crews. *Computers & Operations Research*, 28(7):671–688, 2001.

J. M. Valerio de Carvalho. Exact solution of bin-packing problems using column gneration and branch-and-bound. *Annals of Operations Research*, 86(3):629–659, 1999.

J. M. Valerio de Carvalho. Lp models for bin-packing and cutting stock problems. *European Journal of Operational Research*, 141(2):253–273, 2002.

Z. Degraeve and M. Peeters. Optimal integer solutions to industrial cutting-stock problems: part 2, benchmark results. *INFORMS Journal on Computing*, 15(1):58–81, 2003.

Z. Degraeve and L. Schrage. Optimal integer solutions to industrial cutting-stock problems. *INFORMS Journal on Computing*, 11(4):406–419, 1999.

P. Dell'Olmo and G. Lulli. A new hierachical architecture for air traffic management optimisation of airway capacity in a free flight scenario. *European Journal of Operational Research*, 144:179–193, 2003.

G. Desaulniers, J. Desrosiers, Y. Dumas, S. Marc, B. Rioux, M. Solomon, and F. Soumis. Crew pairing at Air France. *European Journal of Operational Research*, 97(1):245–259, 1997a.

G. Desaulniers, J. Desrosiers, Y. Dumas, M. Solomon, and F. Soumis. Daily aircraft routing and scheduling. *Management Science*, 43(6):841–855, 1997b.

G. Desaulniers, J. Desrosiers, and M. M. Solomon. *Column Generation.* Springer, 2005.

M. Desrochers, J. Desrocsiers, and M. M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40(3):342–354, 1992.

J. Desrocsiers, F. Soumis, and M. Desrochers. Routing with time windows by column generation. *Networks*, 14(4):545–565, 1984.

M. Elf and V. Kaibel. Rotation planning for the continental service of a European airline. In W. Jager and H. Krebs, editors, *Mathematics - Key Technologies for the Future: Joint Projects between Universities and Industry*, pages 675–689. Springer, 2003.

G. Ellinas, E. Bouillet, R. Ramamurthy, J. Labourdette, S. Chaudhuri, and K. Bala. Routing and restoration architectures in mesh optical networks. *Optical Networks Magazine*, pages 91–105, 2003.

G. D. Eppen and R. K. Martin. Solving multi-item capacitated lot-sizing problems using variable redefinition. *Operations Research*, 35(6):832–848, 1987.

A. Erdogan and B. Denton. Surgery planning and scheduling. *Wiley Encyclopedia of Operations Research and Management Science*, to appear, 2010.

FAA. Federal Aviation Regulations, 2002. URL `http://www.faa.gov/avr/afs`.

FAA. Oceanic work group meeting, 2010a. URL `http://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/enroute/oceanic/documents/owg/OWG_Jan132010.pdf`.

FAA. Track advisory, 2010b. URL `http://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/enroute/oceanic/documents/Pacific_Track_Advisory/TA_User_Guide_101206.pdf`.

M. Gamache and F. Soumis. A method for optimally solving the rostering problem. In G. Yu, editor, *Operations Research in the Airline Industry*, pages 124–157. Kluwer Academic Publishers, 1998.

M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman and Company, 1979.

S. Gelinas and F. Soumis. Dantzig-Wolfe decomposition for job shop scheduling. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column Generation*, pages 271–302. Springer, 2005.

A. M. Geoffrion. Lagrangian relaxation for integer programming. *Mathematical Programming Studies*, 2:82–114, 1974.

P. C. Gilmore and R. E. Gomory. A linear programming approach to the cutting-stock problem. *Operations Research*, 9(6):849–859, 1961.

M. X. Goemans and Y. Myung. A catalog of Steiner tree formulations. *Networks*, 23(1): 19–28, 1993.

R. Gopalan and K. Talluri. The aircraft maintenance routing problem. *Operations Research*, 46(2):260–271, 1998.

M. Gronkvist. Accelerating column generation for aircraft scheduling using constraint propagation. *Computers & Operations Research*, 33(10):2918–2934, 2006.

L. Guo, H. Yu, and L. Li. A new shared-path protection algorithm under shared risk link group constraints for survivable WDM mesh networks. *Optical Communication*, 246 (4-6):285–295, 2005.

C. A. Hane, C. Barnhart, E. L. Johnson, R. E. Marsten, G. L. Nemhauser, and G. Sigismondi. The fleet assignment problem: Solving a large-scale integer program. *Mathematical Programming*, 70(1-3):211–232, 1995.

M. Haouari, N. Aissaoui, and F. Z. Mansour. Network flow-based approaches for integrated aircraft fleeting and routing. *European Journal of Operational Research*, 193(2):591–599, 2009.

M. Haouari, H. D. Sherali, F. Z. Mansour, and N. Aissaoui. Exact approaches for integrated aircraft fleeting and routing at TunisAir. *Computional Optimization and Applications*, page In press, 2011.

J. Hessburg. What's this 'A' Check, 'C' Check stuff? Aircraft inspections defined, Apr. 2000. URL `http://www.amtonline.com/publication/article.jsp?pubId=1&id=964`.

M. Van Houdenhoven, J. M. van Oostrum, G. Wullink, E. Hans, J. L. Hurink, J. Bakker, and G. Kazemier. Fewer intensive care unit refusals and a higher capacity utilization by using a cyclic surgical case schedule. *Journal of Critical Care*, 23:222–226, 2008.

J. Q. Hu. Diverse routing in mesh optical networks. *IEEE Transactions on Communications*, 51(3):489–494, 2003.

IATA. Air transport market analysis December, 2010. URL `http://www.iata.org/whatwedo/Documents/economics/MIS_Note_Dec10.pdf`.

ICAO. Conference on the Economics of Airports and Air Navigation Services, 2008. URL `http://www.icao.int/ceans/Docs/Ceans_Wp_066_en.pdf`.

ILOG. *ILOG CPLEX 8.1 User's Manual*. ILOG, 2002.

I. Ioachim, J. Desrosiers, F. Soumis, and N. Belanger. Fleet assignment and routing with schedule synchronization constraints. *European Journal of Operational Research*, 119 (1):75–90, 1999.

V. S. Irava and C. Hauser. Survivable low-cost low-delay multicast trees. In *Proceeding of IEEE GLOBECOM 2005*, pages 110–115, 2005.

S. Irnich and D. Villeneuve. The shortest-path problem with resource constraints and k-cycle elimination for $k \geq 3$. *INFORMS Journal on Computing*, 18(3):391–406, 2006.

J. E. Kelley Jr. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1961.

J. Kang, K. Park, and S. Park. Optimal multicast route packing. *European Journal of Operational Research*, 196(1):351–359, 2009.

B. Khoury and P. Pardalos. A heuristic for the Steiner problem in graphs. *Computational Optimization and Applications*, 6(1):5–14, 1996.

D. Klabjan. Large-scale models in the airline industry. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column Generation*, pages 163–195. Springer, 2005.

D. Klabjan, E. L. Johnson, G. L. Nemhauser, E. Gelman, and S. Ramaswamy. Airline crew scheduling with time windows and plane count constraints. *Transportation Science*, 36 (3):337–348, 2002.

N. Kohl, J. Desrosiers, O. B. G. Madson, M. M. Solomon, and F. Soumis. 2-path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33(1):101–116, 1999.

B. N. Levine and J. J. Garcia-Luna-Aceves. A comparison of reliable multicast protocols. *Multimedia Systems*, 6(5):334–348, 1998.

J. M. Lewis. On the complexity of the maximum subgraph problem. In *Proc. 10th Annu. ACM Symp. Theory of Computing*, pages 265–274, 1978.

G. Li, R. Doverspike, and C. Kalmanek. Fiber span failure protection in mesh optical networks. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 4599, pages 130–141, August 2001.

Z. Liang, W. A. Chaovalitwongse, M. Cha, and S. Moon. Redundant multicast routing in multilayer networks with shared risk resource groups: Complexity, models and algorithms. *Computers & Operations Research*, 37(10):1731–1739, 2010.

Z. Liang, W. A. Chaovalitwongse, H.-C. Huang, and E. L. Johnson. On a new rotation-tour network model for aircraft maintenance routing problem. *Transportation Science*, 45(1):109–120, 2011.

M. Lohatepanont and C. Barnhart. Airline schedule planning: integrated models and algorithms for schedule design and fleet assignment. *Transportation Science*, 38(1): 19–32, 2004.

M. E. Lubbecke and J. Mesrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

V. Mak and N. Boland. Heuristic approaches to the asymmetric travelling salesman problem with replenishment arcs. *International Transactions in Operational Research*, 7 (4-5):431–447, 2000.

F. Z. Mansour, M. Haouari, H. D. Sherali, and N. Aissaoui. Flexible aircraft fleeting and routing at TunisAir. *Journal of Operational Research Society*, 62(2):368–380, 2011.

O. Marcotte. The cutting stock problem and integer rounding. *Mathematical Programming*, 33(1):82–92, 1985.

S. McCartney. The money ledger: Tallying 2009 airline profits and losses. The Wall Street Journal, Jan. 28 2010. URL `http://blogs.wsj.com/middleseat/2010/01/28/which-airlines-made-lost-money-last-year/`.

M. Medard, S. G. Finn, and R. A. Barry. Redundant trees for preplanned recovery in arbitrary vertex-redundant or edge-redundant graphs. *IEEE/ACM Transactions on Networking*, 7(5):641–652, 1999.

A. Mercier and F. Soumis. An integrated aircraft routing, crew scheduling and flight retiming model. *Computers & Operations Research*, 34(8):2251–2265, 2007.

A. Mercier, J. Cordeau, and F. Soumis. A computational study of Benders decomposition for the integrated aircraft routing and crew scheduling problem. *Computers & Operations Research*, 32(6):1451–1476, 2005.

P. Modesti and A. Sciomachen. A utility measure for finding mutiobjective shortest paths in urban multimodal transportation networks. *European Journal of Operational Research*, 111:495–508, 1998.

G. L. Nemhauser and L. Wolsey. *Integer and Combinatorial Optimization*. Wiley, 1999.

H. Ni and H. Abeledo. A branch-and-price approach for large-scale employee tour scheduling problems. *Annals of Operations Research*, 155:167–176, 2007.

B. A. Norman, W. Tharmmaphornphilas, K. L. Needy, B. Bidanda, and R. C. Warner. Worker assignment in cellular manufacturing considering technical and human skills. *International Journal of Production Research*, 40(6):1479–1492, 2002.

C. A. S. Oliveira and P. M. Pardalos. Construction algorithms and approximation bounds for the streaming cache placement problems in multicast networks. *Cybernetics and Systems Analysis*, 41(6):898–908, 2005a.

C. A. S. Oliveira and P. M. Pardalos. A survey of combinatorial optimization problems in multicast routing. *Computers & Operations Research*, 32(8):1953–1981, 2005b.

C. A. S. Oliveira, P. M. Pardalos, and T. M. Querido. A combinatorial algorithm for message scheduling on controller area networks. *International Journal of Operations Research*, 1(2):160–171, 2005.

C. A. S. Oliveira, P. M. Pardalos, and M. G. C. Resende. Optimization problems in multicast tree construction. In M. G. C. Resende and P. M. Pardalos, editors, *Handbook of Optimization in Telecommunications*, pages 701–723. Springer, Berlin, 2006.

C. A. S. Oliveira, O. A. Prokopyev, P. M. Pardalos, and M. G. C. Resende. Streaming cache placement problems: complexity and algorithms. *International Journal of Computational Science and Engineering*, 3(3):173–183, 2007.

N. Papadakos. Integrated airline scheduling. *Computer & Operations Research*, 36(1): 176–195, 2009.

C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, 1998.

P. M. Pardalos and B. Khoury. A heuristic for the Steiner problem in graphs. *Computational Optimization and Applications*, 6(1):5–14, 1996.

P. Paul and S. V. Raghavan. Survey of multicast routing algorithms and protocols. In *Proceedings of the 15th international conference on computer communication*, pages 902–926, 2002.

D. Peleg and E. Upfal. Constructing disjoint paths on expander graphs. *Combinatorica*, 9(3):289–313, 1989.

P. Perny and O. Spenjaard. A preference-based approach to spanning trees and shortest paths problem. *European Journal of Operational Research*, 162:584–601, 2005.

R. L. Phillips, D. W. Boyd, and T. A. Grossman. An algorithm for calculating consistent itinerary flows. *Transportation Science*, 25(3):225–239, 1991.

T. Polzin and S. V. Daneshmand. A comparison of Steiner tree relaxations. *Discrete Applied Mathematics*, 112(1-3):241–261, 2001a.

T. Polzin and S. V. Daneshmand. Improved algorithms for the Steiner problem in networks. *Discrete Applied Mathematics*, 112(1-3):263–300, 2001b.

E. Queiros and V. Martins. On a multicriteria shortest path problem. *European Journal of Operational Research*, 16:236–245, 1984.

A. Rais and A. Viana. Operations research in healthcare: a survey. *International Transactions in Operational Research*, 18(1):1–31, 2011.

M. G. C. Resende and P. M. Pardalos. *Handbook of optimization in telecommunications*. Springer, Berlin, 2006.

N. Robertson and P. D. Seymour. Outline of a disjoint paths algorithm. In Jirte B, L. Lovasz, H. J. Promel, and A. Schrijver, editors, *Paths, Flows and VLSI-Layout*, pages 267–292. Springer, Berlin, 1990.

D. Ryan and B. Foster. An integer programming approach to scheduling. In A. Wren, editor, *Computer Scheduling of Public Transport: Urban Passenger Vehicle and Crew Scheduling*, pages 269–280. Elsevier Science, Amsterdam, 1981.

R. Sandhu and D. Klabjan. Integrated airline fleeting and crew pairing decisions. *Operations Research*, 55(3):439–456, 2007.

A. Sarac, R. Batta, and C. Rump. A branch-and-price approach for operational aircraft maintenance routing. *European Journal of Operational Research*, 175(3):1850–1869, 2006.

P. Scholl, R. Klein, and C. Juergens. Bison: a fast hybrid procedure for exactly solving the one-dimensional bin-packing problem. *Computers & Operations Research*, 24(3): 627–645, 1997.

A. Schrijver. Homotopic routing methods. In Jirte B, L. Lovasz, H. J. Promel, and A. Schrijver, editors, *Paths, Flows and VLSI-Layout*, pages 329–371. Springer, Berlin, 1990.

A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, 1998.

L. Shen, X. Yang, and B. Ramanurthy. Shared Risk Link Group (SRLG)-diverse path provisioning under hybrid service level agreements in wavelength-routed optical mesh networks. *IEEE/ACM Transactions on Networking*, 13(4):918–931, 2005.

H. D. Sherali, J. C. Smith, and A. A. Trani. An airspace planning model for selecting flight-plans under workload, safety, and equity consideration. *Transportation Science*, 36(4):378–397, 2002.

H. D. Sherali, R. W. Staats, and A. A. Trani. An airspace planning and collaborative decision-making model: Part I - probabilistic conflicts, workload, and equity considerations. *Transportation Science*, 37(4):434–456, 2003.

H. D. Sherali, E. Bish, and X. Zhu. Airline fleet assignment concepts, models and algorithms. *European Journal of Operational Research*, 172(1):1–30, 2006a.

H. D. Sherali, R. W. Staats, and A. A. Trani. An airspace planning and collaborative decision-making model: Part II - cost model, data considerations, and computations. *Transportation Science*, 40(2):147–164, 2006b.

H. D. Sherali, J. M. Hill, and M. V. McCreaand A. A. Trani. Integrating slot-exchange, safety, capacity, and equity mechanisms within an airspace flow program, 2009. Manuscript, Grado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, VA.

H. D. Sherali, K.-H. Bae, and M. Haouari. Integrated airline schedule design and fleet assignment: Polyhedral analysis and Benders' decomposition approach. *INFORMS Journal on Computing*, 22(4):500–513, 2010.

F. Soumis, J. A. Ferland, and J.-M. Rousseau. A model for large scale aircraft routing and scheduling problems. *Transportation Research, Part B*, 14(1-2):191–201, 1980.

C. Sriram and A. Haghani. An optimization model for aircraft maintenance scheduling and re-assignment. *Transportation Research Part A*, 37(1):29–48, 2003.

G. Stoller. Planes with maintenance problems have flown anyway. USA Today, Feb. 3 2010. URL `http://www.usatoday.com/travel/flights/2010-02-02-1Aairmaintenance02_CV_N.htm`.

R. Subramanian, Jr. R. P. Scheff, J. D. Quillinan, D. S. Wiper, and R. E. Marsten. Coldstart: Fleet assignment at delta air lines. *Interfaces*, 24(1):104–120, 1994.

K. Talluri. The four-day aircraft maintenance routing problem. *Transportation Science*, 32(1):43–53, 1998.

R. Teixeira, T. G. Griffin, M. G. C. Resende, and J. Rexford. TIE breaking: tunable interdomain egress selection. *IEEE/ACM Transactions on Networking*, 15(4):761–774, 2007.

M. Terab and A. R. Odoni. Strategic flow management for air traffic control. *Operations Research*, 41(1):138–152, 1993.

P. Vance. Branch-and-price algorithms for the one-dimentional cutting stock problem. *Computational Optimization and Applications*, 9(2):211–228, 1998.

P. Vance, A. Atamturk, C. Barnhart, E. Gelman, E. L. Johnson, A. Krishna, D. Mahidhara, and G. L. Nemhauser. A heuristic branch-and-price approach for the airline crew pairing problem. Technical Report LEC-97-06, Georgia Institute of Technology, 1997.

F. Vanderback. Computational study of a column generation algorithm for binpacking and cutting stock problems. *Mathematical Programming*, 86(4):565–594, 1999.

F. Vanderbeck. Implementing mixed integer column generation. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column Generation*, pages 331–358. Springer, 2005.

T. M. Vossen, M. O. Ball, R. L. Hoffman, and M. C. Wambsganss. A general approach to equity in traffic flow management and its application to mitigating exemption bias in ground delay programs. *Air Traffic Control Quarterly*, 11:277–292, 2003.

W. E. Walker. A method for obtaining the optimal dual solution to a linear program using the Dantzig-Wolfe decompoisition. *Operations Research*, 17(10):368–370, 1969.

O. Weide and M. Ehrgott D. Ryan. An iterative approach to robust and integrated aircraft routing and crew scheduling. *Computers & Operations Research*, 37(5):833–844, 2010.

Wikipedia. IPTV, 2008. URL `http://en.wikipedia.org/wiki/IPTV`.

L. Wolsey. *Integer Programming*. Wiley, 1998.

R. T. Wong. A dual ascent approach for Steiner tree problems on a directed graph. *Mathematical Programming*, 28(3):271–287, 1984.

Y. Xin and G. N. Rouskas. Multicast routing under optical layer constraints. *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 4:2731–2742, 2004.

S. Yan and C.-H. Tseng. A passenger demand model for airline flight scheduling and fleet routing. *Computers & Operations Research*, 39:1559–1581, 2002.

M. Yannakakis. Node- and edge-deletion np-complete problems. In *Proc. 10th Annu. ACM Symp. Theory of Computing*, pages 253–264, 1978.

S. Yuan and J. P. Jue. Dynamic lightpath protection in WDM mesh networks under wavelength-continuity and risk-disjoint constraints. *Computer Networks*, 48(2):91–112, 2005.

H. Zang, C. Ou, and B. Mukherjee. Path-protection Routing and Wavelength Assignment (RWA) in WDM Mesh Networks under Duct-layer Constraints. *IEEE/ACM Transactions on Networking*, 11(2):248–258, 2003.

K. G. Zografos and C. S. Tsanos. Developing and applying a methodology for assessing the implementation potential of reduced separation minima. In *Proceedings of Transportation Research Board 88th Annual Meeting*, pages 09–0775, 2009.

Vita

Zhe Liang

EDUCATION

09/2006 - 04/2011 **Ph.D. Student** in Industrial and Systems Engineering
Rutgers University, Piscataway, NJ

01/2002 - 12/2003 **M. Eng.** in Industrial and Systems Engineering
National University of Singapore, Singapore

09/1997 - 11/2001 **B. Eng.** in Computer Engineering
National University of Singapore, Singapore

OCCUPATION

01/2008 - 06/2009 **Research Assistant**
Department of Industrial & Systems Engineering
Rutgers University, Piscataway, NJ

09/2007 - 12/2007 **Teaching Assistant**
Department of Industrial & Systems Engineering
Rutgers University, Piscataway, NJ

09/2006 - 06/2007 **Research Fellow**
Department of Industrial and Systems Engineering
Rutgers University, Piscataway, NJ

01/2004 - 12/2005 **Research Engineer**
Department of Industrial and Systems Engineering
National University of Singapore, Singapore

PUBLICATIONS

Journal Papers

Z. Liang, and W. A. Chaovalitwongse, "A Multicast Problem with Shared Risk Cost", *Optimization Letters*, online first, pages 1-14, 2011.

Z. Liang, W. A. Chaovalitwongse, H. C. Huang and E. L. Johnson, "A New Rotation-Tour Network Model for Aircraft Maintenance Routing Problem", *Transportation Science*, vol 45(1), pages 109-120, 2011.

A. D. Rodriguez , W. A. Chaovalitwongse, **Z. Liang**, H. Singhal, and H. Pham, "Master Defect Record Retrieval Using Network-Based Feature Association", *IEEE Transactions*

on Systems, Man, and Cybernetics Part C: Applications and Reviews, vol 40(3), pages 319-329, 2010.

Z. Liang, and W. A. Chaovalitwongse, "Bounds of Redundant Multicast Routing Problem with SRLG-diverse Constraints: Edge, Path and Tree Models", *Journal of Global Optimization*, vol 48(2), pages 335-345, 2010.

Z. Liang, W. A. Chaovalitwongse, A. D. Rodriguez, D. E. Jeffcoat, D. A. Grundel, and J. K. O'Neal, "Optimization of Spatiotemporal Clustering for Target Tracking From Multisensor Data", *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews*, vol 40(2), pages 176-188, 2010.

Z. Liang, W. A. Chaovalitwongse, M. Cha, and S. B. Moon. "Redundant Multicast Routing in Multilayer Networks with Shared Risk Resource Groups: Complexity, Models and Algorithms", *Computers & Operations Research*, vol 37(10), pages 1731-1739, 2010.

Working Papers

Z. Liang, and W. A. Chaovalitwongse, A Compact Rotation Tour Network Model for Weekly Aircraft Maintenance Routing Problem, submitted to *Transportation Research Part B: Methodological* , 2011.

Z. Liang, W. A. Chaovalitwongse, and E. A. Elsayed, "Flight Sequence Model for the Flight Conflict Resolving Problem", submitted to *Operations Research* (under 2nd review), 2011.

W. A. Chaovalitwongse, **Z. Liang**, M. Cha, S. B. Moon, A. Shaikh and J. Yates. "Mathematical Programming Approaches for Dual Multicast Routing Problem with Multilayer Risk Constraints", submitted to *Annals of Operations Research*, 2011.

C. A. Chou, **Z. Liang**, W. A. Chaovalitwongse, T. Berger-Wolf, B. Dasgupta S. Sheikh, S. L. Putrevu, M. V. Ashley, and I. C. Caballero, Column Generation Framework of Nonlinear Similarity Model for Reconstructing Sibling Groups, submitted to *INFORMS Journal of Computing* (under 2nd review), 2011.

Book Chapters

Z. Liang, and W. A. Chaovalitwongse, "The Aircraft Maintenance Routing Problem", In W. A. Chaovalitwongse, K. C. Furman and P. M. Pardalos, editors, *Optimization and Logistics Challenges in the Enterprise*, pages 327-348, Springer, New York, 2009.

M. Cha, W. A. Chaovalitwongse, **Z. Liang**, J. Yates, A. Shaikh, and S. B. Moon, "Integer Linear Programs for Routing and Protection Problems in Optical Networks", In C. A. Floudas, P. M. Pardalos, editors, *Encyclopedia of Optimization*, Second Edition, pages 1610-1617, Springer, New York, 2009.

W. A. Chaovalitwongse, H. Pham, S. Hwang, **Z. Liang**, and C. H. Pham. "Recent Advances in Data Mining for Categorizing Text Records". In H. Pham, editors, *Recent Advances in Reliability*, pages 223-240, Springer, New York, 2008.