OPTIMIZATION MODELS AND ALGORITHMS FOR SOLVING LARGE-SCALE NETWORK DESIGN, ROUTING AND SCHEDULING PROBLEMS

By

SUAT BOĞ

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2010

Dedicated to my parents (Bingül and Yakup),
my wife (Gözde),
and my brother (Murat)

ACKNOWLEDGMENTS

First of all, I would like to thank my advisor Dr. Ravindra K. Ahuja for his support and guidance. He was a great mentor. He gave me enough freedom to work on my research but he was also available to guide me when I needed help. I also want to thank him for opportunities he provided.

I would like to thank Dr. J. Cole Smith, Dr. Panos M. Pardalos and Dr. Sanjay Ranka for their willingness to serve on my committee, for their time and valuable feedback. I also want to thank Dr. Joseph Geunes for his support, smiling face and his endless humors in our office.

I want to thank my friend Yusuf with whom I started this journey with. He had been a great roommate. With him the first two years of my Ph.D. had been very enjoyable. I would like to acknowledge my friend and colleague Gonca for her support and for numerous delightful conversations we had on almost everything.

I am grateful to my parents and my brother who always supported me. Finally, I want to thank my beloved wife Gözde. She always supported and encouraged me. She was awake and always with me when I was working late. Her trust, confidence and support gave me great motivation and energy to complete this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

OPTIMIZATION MODELS AND ALGORITHMS FOR SOLVING LARGE-SCALE NETWORK DESIGN, ROUTING AND SCHEDULING PROBLEMS

By

Suat Boğ

August 2010

Chair: Ravindra K. Ahuja
Major: Industrial and Systems Engineering

Freight transportation is an important component of economy and constitutes largest portion of the logistics cost. Decisions regarding freight transportation have a high impact on customer service levels, economic efficiency and competitiveness of the firms. In this dissertation, we study network design, routing and scheduling problems arising in freight transportation industry. We focus on optimization problems faced mainly by consolidation-based freight carriers. The emphasis is given on developing computationally efficient optimization models and algorithms which capture all the real life complexities of studied problems and result into near-optimal solutions in reasonable time limits.

We first consider transportation network disruption problem encountered by carriers for maintenance of their tangible assets. In this context, we study the Curfew Planning Problem (CPP) encountered by rail carriers for the maintenance of railway tracks. Maintenance regions which are under absolute curfew cause a complete blockage of the rail traffic. While providing high customer service levels, railroads need to perform maintenance on their tracks with minimum possible disruptions in transportation network. We propose four iterative algorithms that decompose the problem into efficient smaller integer programming models working on shorter time horizons. These models are flexible to run with dynamically changing data sets throughout iterations. Hence, one can enter a partial maintenance schedule and the algorithm can complete the rest of the schedule. Proposed algorithms are applied on the whole rail network, for the complete

yearly planning horizon and can capture all the real life complexities affecting the implementability of the resulting solution. They provide a successful way for handling complicating decisions and constraints implicitly. We tested our algorithms on real life instances from a major North American Railroad company and obtained very good solutions in practical time limits.

Next, we consider integrated transportation planning and propose a novel model to solve combined network design and commodity routing problem. In this problem, our emphasis is on non-bifurcated flow of commodities where each commodity should flow on a single path from its origin to its destination. Non-bifurcated flow arises frequently in freight transportation for consolidation-based carriers such as less-than-truckload trucking, express package delivery and railway freight routing. Earlier methods mostly solved network design and commodity routing problems separately in a sequential manner for bifurcated network design problems. We propose a holistic approach to solve these highly interrelated problems in an integrated manner. The proposed model involves binary path based design and path based flow variables. Traditional network design models in the literature use arc based design variables, and hence they are not suitable for incorporating asset management related constraints into the model. We adapted our generic model on an instance obtained from a major railroad company to solve combined train routing and block-to-train assignment problem. The model could handle many asset management related constraints required by the railroad and it resulted in good quality solutions within reasonable time limits.

Finally, we consider service network design problem on hub-and-spoke networks. We propose a novel network shrinking based decomposition which allows the generation of a smaller time space network of hub to hub connections. In this three phased decomposition, we apply all three types of consolidation. In the first phase of the decomposition, we perform facility based consolidation on a space network where we sort and consolidate shipments for their next location. In the second phase, we

perform temporal consolidation by holding shipments in hubs to be able to generate larger shipments and in the third phase, we perform multi-stop consolidation to improve direct non-hub to hub and hub to non-hub connections. This generic approach can be utilized by all consolidation-based carriers operating on hub-and-spoke networks, such as express package delivery, less-than-truckload service providers, freight rail carriers, etc. We could solve a fairly large scale practical problem using the proposed decomposition scheme. We applied our algorithm on a real life instance for a less-than-truckload motor carrier and obtained considerable improvements in transportation costs and load capacity utilizations in a reasonable time limit.

CHAPTER 1
INTRODUCTION

1.1 Background and Motivation

Freight transportation is one of the key activities of a logistics system and a major sector contributing to economy. Decisions regarding freight transportation affect all supply chain players, cost of final products, network design, location of facilities, resource allocations and utilizations, customer service levels, economic efficiency and competitiveness in the market. Freight transportation costs constitute the largest portion of logistics costs in US over the last two decades according to Annual State of Logistics Reports released by the Council of Supply Chain Management Professionals (CSCMP). 20th Annual Report released in 2009 indicates that total US logistics costs was $1.3 trillion in 2008 and freight transportation costs represent 64% of total logistics costs which is equivalent to 6.1% of Gross Domestic Product (GDP) (CSCMP, 2009). In 2008, transportation activities contributed to 9.5% of GDP (USDOT RITA BTS, 2010d) and total employment in transportation sector was 9.7% of total US labor force (USDOT RITA BTS, 2010c). Freight transportation is a highly dynamic field. Transportation activities are affected by various factors such as growth or decline in economic activity, globalization, energy prices, infrastructure capacity, environmental concerns, changes in regulations, advance of technology, internet shopping, and just-in-time inventory management (Crainic, 2003; USDOT RITA BTS, 2010a). Typically freight transportation industry requires a large investment in capital assets. Transportation companies need to make complex set of interrelated decisions which have trade-offs among each other. To manage their assets effectively, in a cost efficient way and to maintain high customer service levels, they need to plan and optimize their transportation activities. Otherwise, it might not be possible to stay competitive and make profit in this dynamic, complex and demanding industry.

There are five major transportation modes used to move goods between supply and demand points: trucking, water transportation, rail transportation, air transportation, and pipelines. In 2007, rail transportation and trucking accounted for 39.5% and 28.6% of total ton-miles of domestic freight respectively. They constitute the most commonly used transportation modes and are followed by pipelines, water transportation and air transportation. Compared to 1990, ton-miles of domestic freight for rail and trucking increased by 71% and 55.2% respectively (USDOT RITA BTS, 2010b).

1.1.1 Consolidation-based Freight Transportation

Consolidation is a common method used to obtain considerable cost savings in freight transportation. The process takes advantage of economies of scale principle by consolidating small shipments into larger ones. Consolidation-based freight transportation is applied by various service providers such as less-than-truckload (LTL) motor carriers, postal services, railways, shipping lines, etc. There are three major types of consolidation: Facility, temporal and multi-stop. In facility consolidation, inbound shipments are sorted and consolidated to form outbound shipments that are moved jointly to another hub. In temporal consolidation, shipments are hold and aggregated over time to be able to ship large shipments. Multi-stop consolidation is used for pick-up and delivery routes. Several customers are served together on a pick-up or delivery route instead of using one shipment for each customer.

1.1.2 Network Design Models

Planning problems for freight transportation are often expressed using network design models. These models are usually different extensions of generic multicommodity capacitated network design (MCND) formulation that has many applications in transportation, telecommunication, energy, computer, and production-distribution systems (Balakrishnan et al., 1997; Magnanti & Wong, 1984; Minoux, 1989). In freight transportation context, network design models are frequently used to construct and improve networks, build service routes and schedules, and allocate resources to jobs. Collection of these interrelated

problems is called as service network design in the literature. For all type of carriers, designing a reliable, time and cost efficient service network is crucial in order to operate profitably and maintain high customer service levels. Crainic (2000) makes a good review of network design models, relevant solution approaches and service network design formulations in freight transportation. In a recent survey, Wieberneit (2008) reviews different formulations and solution frameworks for service network design problems.

1.1.3 Transportation Planning Levels

Planning and optimization problems that need to be addressed by freight transportation managers can be classified according to three planning levels: Strategical, tactical and operational level planning (Crainic, 2000; Crainic & Laporte, 1997). Strategical planning includes long term decisions and requires large capital investments. Decisions related to ownership of resources and locations of the facilities are typical examples of strategical planning. Tactical planning decisions are related to optimal utilization of resources over a medium term horizon. Service network design is an example of tactical planning level and includes decisions regarding service selection, shipment routing, repositioning of empty vehicles and consolidation works at terminals. Operational planning decisions are given for a short term horizon in a dynamic setting. On this level, local managers and dispatchers perform adjustments on the tactical plans by controlling service and maintenance schedules, routing and dispatching of vehicles and crews.

1.2 Contributions and Overview

Transportation of resources is the driving component of the problems we study. These problems are mostly NP-hard problems, and for large-scale applications it is difficult to find optimal solutions or even feasible solutions. For these types of instances, we combine mixed integer programming with network optimization and heuristic techniques in novel ways. Such hybrid algorithms take advantage of both exact and heuristic methodologies. In this dissertation, we contribute to literature by designing computationally efficient algorithms that provide near-optimal solutions in reasonable time limits. Main application

area of our research has been freight transportation networks. We have conducted research in three important areas related to freight transportation: transportation network disruption, integrated transportation planning for non-bifurcated network design problems and service network design on hub-and-spoke networks. Below we highlight some of our main contributions in each of these fields of study.

1.2.1 Transportation Network Disruption

Freight transportation industry requires a large investment in capital assets. While operating in a reliable and cost efficient way, carriers have to perform regular maintenance activities for their expensive resources. They need to schedule required maintenance activities such that transportation network disruptions are at minimum possible levels. In this context, we study the Curfew Planning Problem (CPP) encountered by rail carriers for the maintenance of railway tracks. In railroad terminology, a region is under absolute curfew if a maintenance project causes a complete blockage of the rail traffic. While providing satisfactory service levels, railroads must perform maintenance on their tracks without causing disruptions in train schedules. The CPP is to design an annual timetable to complete a given set of repairs and replacement jobs (rail-work and tie-work) on the railway tracks for a set of teams specialized in rail-work (rail team) or tie-work (tie team). We develop the work schedule for each team such that the disruptions in train routes are minimized. Quality and implementability of a solution depends highly on the curfew related performance constraints. We published two papers on this problem. In our first paper Bog et al. (2010), we developed novel iterative decomposition algorithms to minimize the number of violations in performance constraints. Previous methods developed for this problem are applied on a single track of the whole rail network, for a short term horizon (a week) and are mostly useful to modify an existing timetable (see e.g. Budai et al., 2006; Higgins et al., 1999; Lake & Ferreira, 2002). In Bog et al. (2010), we proposed four iterative algorithms to minimize possible rail traffic disruptions. We tested our algorithms on real life instances from a major North American Railroad company

and obtained very good solutions in practical time limits. Our method was applied to the whole train network for a long term horizon (a year). We considered almost all of the real-life constraints affecting the implementability of the resulting solution. The proposed iterative algorithms are flexible in the sense that users can provide a partial schedule and algorithm can assign the rest of the schedule. In our second paper Nemani et al. (2010), we presented four different solution approaches for the CPP: (i) time-space network model, (ii) duty-generation model, (iii) column-generation model, and (iv) decomposition-based duty generation heuristics. This paper represents our thought process and all the developed methods in an ordered way during the solution of the CPP. The last solution approach (decomposition-based duty generation heuristics) takes insights from the iterative algorithms presented in the first paper. It is the best approach for generating solutions from scratch. Iterative approach is the best for incremental optimization. In this paper, we reduced performance constraint violations by 75% and increased crew work life quality by reducing travel distance along the year by 15%. With these two studies, we won a second place award in INFORMS student paper competition on Management Science in Railroad Applications in 2009.

1.2.2 Integrated Transportation Planning

Literature on Multicommodity Capacitated Network Design (MCND) problems mostly focus on bifurcated flow of commodities (see e.g. Crainic et al., 2000; Ghamlouche et al., 2003; Holmberg & Yuan, 2000; Magnanti et al., 1993). In bifurcated case, a commodity can flow on several paths from its origin to its destination. In this study, we focus on non-bifurcated flow of commodities where a commodity can flow on a single path from its origin to its destination. We can categorize our problem as Non-bifurcated Multicommodity Capacitated Network Design (NMCND) problem which belongs to the general class of MCND problems. NMCND problem arises in many real life systems such as computer networks, telecommunication networks, freight transportation in consolidation-based carriers (less-than-truckload trucking, express package delivery and

railway freight routing). The problem involves two closely interrelated decisions: network design and commodity routing. Many studies in the literature divide this problem into two separate stages which are solved separately in a sequential manner. In this study, we propose a novel mathematical model which efficiently solves these two problems in an integrated manner. The proposed model involves binary path based design and path based flow variables. Traditional network design models in the literature use arc based design variables, and hence they are not suitable for incorporating asset management related constraints into the model. We tested our model on an instance obtained from a major railroad company to solve combined train routing and block-to-train assignment problem. The model could handle many asset management related constraints required by the railroad and it resulted in good quality solutions within reasonable time limits.

1.2.3 Service Network Design

Bringing the freight at the right time, to the right place is very critical for carriers. Most carriers announce their service commitments and provide strict delivery times for their customers. In this study, we focus on service network design problem of consolidation-based freight carriers which operate on hub-and-spoke networks. Inputs of the problem are service commitments and network of terminal locations which include hub locations and end-of-line terminals. Hub-and-spoke networks are frequently utilized to solve consolidation problems where, instead of sending each shipment directly to its destination, shipments are combined into loads and routed through hubs. The goal is to install loads on the links of the given the network and decide on routes for each shipment such that total transportation costs (total mileage costs) are minimized. Each route is a sequence of loads a shipment should take to travel from its origin to destination. We assume that shipments' demand quantities are not large enough to fill load capacities. Once a shipment is loaded from its origin, it travels hub locations and finally reaches its destination terminal. For this problem, we propose a decomposition approach based on an innovative network shrinking idea. Proposed approach is needed by all consolidation based

carriers, such as express package delivery, less-than-truckload (LTL) service providers, freight rail carriers, etc. Using the proposed decomposition scheme, we solved fairly large scale practical problem of an LTL carrier and were able to decrease weekly costs of the carrier by 15% and increase their load capacity utilization by 5%.

The outline of this dissertation is as follows. In Chapter 2, we develop four iterative algorithms to solve the curfew planning problem (CPP) encountered by railroads. The goal of the problem is to schedule maintenance of railway tracks such that possible transportation network disruptions are minimized. We simplify the original problem by first solving 1-weekly models iteratively. We then extend this approach by solving k-weekly models and using backtracking idea. Backtracking helps to have more far-sighted approach by solving for multiple k-weekly or 1-weekly periods and modifying the current period's solution if a possible future violation is foreseen.

In Chapter 3, we describe the non-bifurcated network design model we have developed for solving network design and commodity routing problems simultaneously. Non-bifurcated flow of commodities where each commodity flows on a single path is frequently observed in freight transportation networks. We first give an overview of bifurcated network design models in the literature, then we explain details of our proposed non-bifurcated network design model.

In Chapter 4, we study combined train routing and block-to-train assignment problems which constitute an important part in developing a railroad's operating plan. We illustrate how we adapt our generic non-bifurcated network design model for these two highly interrelated problems. We also extend the basic model by incorporating many real life constraints to generate an implementable solution. Computational tests on a real life instance show the effectiveness of our approach over an iterative method and railroad's solution.

In Chapter 5, we focus on service network design problem on hub-and-spoke networks. We propose a novel decomposition approach based on a network shrinking idea. We

provide details of our three phased decomposition scheme which is composed of facility consolidation, temporal consolidation and multi-stop consolidation phases. We illustrate our computational experience on a real life instance for a less-than-truckload motor carrier.

CHAPTER 2
ITERATIVE ALGORITHMS FOR THE CURFEW PLANNING PROBLEM

2.1 Introduction

Railroads are vital to America's economical power and competitiveness, moving 40 percent of the nation's freight (in ton-miles). Freight railroads spend more than 20 billion dollars each year for track and equipment maintenance, renewal and expansion. In 2006, freight railroads in the United States had 54 billion dollars of total revenue (Association of American Railroads, 2008a). The investment and revenues are expected to grow since it is projected that demand for freight railroad will rise 87.6 percent by 2035 compared to 2002 levels (U.S. Department of Transportation, 2007). Although capital investment and revenues are very high, rail industry lags behind most industries in terms of profitability (Association of American Railroads, 2008b). Considering the potential savings and performance improvements, allocating and utilizing resources in an efficient and timely manner can help improve the profitability. Possible network disruptions due to inefficient maintenance scheduling of railway tracks may result in tens of millions of dollars in lost revenues. Maintenance scheduling planners in a major North American railroad company describe the curfew planning problem (CPP) as one of the most important and difficult problems. They also mention that the CPP is currently being solved each year manually by a group of five to six maintenance scheduling planners. Building a schedule from scratch takes planners about two weeks. Railroads need a decision support system that facilitates the creation of an annual maintenance timetable for their teams. Maintenance scheduling problem also arises in a variety of industries. Typical application areas include aircraft, vehicle fleet, power generation, pavement, highway, refinery and production facilities. A survey on maintenance scheduling literature can be found in Oke (2004).

There are a number of studies that consider the importance of minimizing rail traffic disruptions. The conflict between rail operations and rail infrastructure maintenance is emphasized in Lake & Ferreira (2002). They formulate a short-term maintenance

scheduling problem as a binary integer nonlinear programming problem and apply a two step heuristic technique to solve the formulation. A feasible solution is found in the first step and a heuristic (simulated annealing, local search, multiple local search or tabu search) is used to improve the feasible solution in the second step. The best results are obtained with simulated annealing. Higgins (1998) provides an integer programming model that aims to minimize the disruptions to train services and reduce maintenance costs. Their model is not applied on a whole network, it is applied on a 300 km track corridor with a four day planning horizon. The objective function is constructed to minimize the interference delays and prioritized finishing times of maintenance activities. Nonlinearity of the constraints in the model and size of the problem force the use of heuristics. They first find a feasible solution and improve it using tabu search. Higgins et al. (1999) use the same approach on a 89 km track corridor and obtain similar reductions in objective function value (about 7%) as compared to the schedule created manually. Another study that considers one rail link is by Budai et al. (2006). They discuss the preventive maintenance scheduling problem with the intent of minimizing the track possession costs and maintenance costs for one link in a rail network. Possession costs are determined by the time a track is occupied for maintenance and cannot be used for rail traffic. Their paper considers the grouping of preventive maintenance activities and gives four different heuristics for solving the problem.

A dynamic schedule generation technique for the rolling horizon is discussed by Cheung et al. (1999). Their study is based on real time data from Hong Kong subway rail system. For occasionally used tracks, which is the case in Australia and some European countries, Budai & Dekker (2004) show that the track possession is modeled in between operations. Budai et al. (2004) introduce a slightly different version of the problem where the objective is completing the project within the track's free time. They create a dynamic schedule for carrying out preventive maintenance activities and propose three heuristics for tackling the problem under several limitations. Of these three, the Max-to-Min

heuristic was found to be the most successful. Grimes & Barkan (2006) perform a study for measuring the cost-effectiveness of railway infrastructure renewal maintenance. Their results show that if railroads constrain the renewal maintenance to reduce the overall capital expenditures, increasing maintenance expenses that follow will more than offset the initial temporary reductions in capital spending.

Most of the existing studies work for a single-track on a short-term horizon and therefore can be used to modify a generated timetable after a network disruption occurs. Many of these approaches result in high computer running times and thus are not well-suited to handle the real-life complexities of a curfew planning problem which is defined on the whole network with a long-term horizon.

In this study, we focus on the maintenance scheduling of railway tracks and aim towards constructing a maintenance timetable that minimizes possible rail traffic disruptions. We propose four iterative algorithms that produce very good solutions within practical time limits. Proposed algorithms have the ability of handling some critical decisions implicitly. The algorithms are applied to the entire rail network and over a long-term horizon. We provide flexible integer programming formulations that are designed to run with dynamically changing data sets throughout iterations. The algorithms developed consider all of the real-life constraints affecting the performance of the resulting maintenance plan. In addition to the introduction, this chapter is organized into four other sections: Section 2.2 provides the problem description and explains the problem inputs and constraints. Section 2.3 presents our solution approaches. Section 2.4 presents the computational tests and Section 2.5 contains our concluding remarks.

2.2 Problem Description

Railway track maintenance scheduling problem is brought to our notice by a major US railroad company. This problem is called as the curfew planning problem (CPP) by the railroad company. Note that the rest of the terminology used throughout this chapter also comes from the railroad.

For a major railroad company, each year around 2500 repair jobs have to be completed on railway track network. Seven types of maintenance jobs are performed on the tracks: Capacity jobs, curve patching, concrete repair, gauging, out-of-face new, out-of-face repair and tie-surfacing. These job types are classified under two main categories as rail jobs and tie-surfacing jobs in which the first six types fall into rail job category. The jobs are partitioned into about 300 projects by their type and geographical proximity, i.e. rail jobs in a specific region are grouped into a rail project and similarly tie-surfacing jobs in this region are grouped into a tie-surfacing project. The duration of a rail project depends on the track distance being repaired or replaced, while that of a tie-surfacing project is calculated by adding the number of ties being processed. Duration also depends on the size of crew working on the project. Around 800 crews are grouped into 18 to 19 teams, which are supposed to complete the projects during the year and at the same time meet various business requirements. The work is scheduled at the project level. The timetable has to specify when and by which team each project should be started.

A project is "active" in a week if it is scheduled in that week. Some projects are located on the railroad yards and some are on the mainline. If a project is on the railroad yard, it does not cause a disruption in rail traffic. Mainlines can be single-tracked or double-tracked. If a project is located on a single-tracked mainline then it cause a complete blockage of the rail traffic and the region where the project is scheduled is called to be "under absolute curfew". The projects which are not on single-tracked mainline require "normal curfew". For the railroads absolute curfew is much more crucial and vital compared to the normal curfew. The projects requiring absolute curfew should be scheduled carefully in order to minimize the possible network disruptions in rail traffic.

The railroad network is divided into "subdivisions", each containing a set of projects. Subdivisions may involve more than one track segment and may require both rail work and tie-surfacing work. A subdivision is considered to be under absolute curfew in a week

if at least one of the absolute curfew requiring projects inside the subdivision is active in that week. In order to control possible network disruptions, some business requirements may be enforced on the number of subdivisions that are under absolute curfew at any week. Inside railroad network, there are usually 10 to 12 "service corridors" which are composed of several subdivisions. Each service corridor consists of specified track segments in the railroad, but some service corridors have overlapping track segments. Therefore, the same subdivision can be in more than one service corridor. A large portion of freight transportation is carried over service corridors. In order to prevent disruptions that may affect movement of high volumes of freight, each service corridor may have at most one of its subdivisions under absolute curfew at any week.

There are four types of teams: small and large rail teams (SR and LR); and small and large tie-surfacing teams (ST and LT). The project completion times are given in terms of weeks. A large team can finish a project roughly twice as fast as a small team. After a project is finished, teams relocate over the weekends. Combining two teams to finish a project in a shorter time is called "splitting". Splitting may also be applied to satisfy some business requirements that must be enforced in each week.

Each year there are some specific weeks where tracks are not used due to coal mine closures or special vacations. These free weeks are called "jamboree weeks" (and usually consist of two consecutive weeks each year). The set of projects that has to be completed during jamboree weeks are called "jamboree projects". The main priority of these special weeks is completing the jamboree projects, so most of the business requirements are relaxed. When jamboree weeks start, teams may have to interrupt their ongoing projects and travel long distances to start working on the jamboree projects.

2.2.1 Problem Inputs

- **Projects:** The list of project names, the type (rail or tie-surfacing) and the subdivision for each project, whether a project requires absolute curfew, and the number of weeks required for completing the project by a small team and by a large team.

- **Teams:** The list of teams and their types (SR, LR, ST or LT).
- **Subdivisions:** The list of subdivision names and the list of all adjoining subdivision pairs.
- **Service Corridors:** The list of service corridor names.
- **Subdivision-Service Corridor Mapping:** The list of the subdivisions in each service corridor.
- **Time Window:** Each project may have specific start and end weeks that show the interval in which the project is allowed to be active. These restrictions are caused by various reasons. For instance, maintenance teams may avoid working in the north during winter or they may avoid working at a region if a major sports event takes place at that time. Time windows is useful to reduce the problem size but they also reduce the feasible region.

2.2.2 Problem Constraints

Two groups of constraints must be satisfied:

- **Performance Constraints** are required by the railroad. They determine the implementability of the schedule generated.
- **Feasibility Constraints** are due to problem characteristics.

2.2.2.1 Performance constraints

1. **Absolute Curfew Constraints:** There may be at most μ subdivisions that are under absolute curfew per week. The value of parameter μ is specified by the railroad. As we stated earlier, all absolute curfew requiring projects in a subdivision are regarded as only one absolute curfew if they are active in the same week.
2. **Service Corridor At-Most Constraints:** At most one subdivision within a service corridor may be under absolute curfew in any week.
3. **Mutually Exclusive Subdivision Constraints:** Any pair of adjoining subdivisions in the list of such pairs should not be under absolute curfew simultaneously.
4. **Time Window Constraints:** Time windows of jamboree projects must be honored as hard constraints. The time windows of other projects are honored as much as possible.
5. **Distance Constraints:** Once a project is finished, all of the resources (heavy equipments, team members, etc.) should be moved to the next project over the weekend. The travel distance limit between projects is specified as 400 miles.

2.2.2.2 Feasibility constraints

1. Each team may work on only one project at a time.
2. Each project should be completed.
3. Rail projects must be done by the rail teams, and tie-surfacing projects by the tie-surfacing teams.
4. No team should be idle during the year. Breaks in the schedule of a team can be at the end of the year, if required.
5. The first week of the year is a holiday.

The second feasibility constraints may suggest that the problem of arranging schedules for rail teams and tie-surfacing teams is separable. However, the decomposition of the problem is not possible, because performance constraints such as absolute curfew, service corridor at-most, and mutually exclusive subdivision constraints have to be applied to both types of teams. For instance, a subdivision can be under absolute curfew due to both rail projects and tie-surfacing projects.

We designed four iterative algorithms to solve the CPP. In order to present our thought process better, we present the algorithms in the order they are developed. In the first two approaches, we solve 1-weekly and k-weekly integer formulations iteratively, and for the others we extend these approaches by using backtracking idea. The objective function of the CPP is to minimize the amount of violations in the performance constraints.

2.3 Solution Approaches

We present four algorithms we have developed in order to solve the CPP. In all algorithms, there are two common strategies applied. The first involves partitioning the set of rail projects and tie-surfacing projects among small and large teams and solving weekly or k-weekly models given the partition at hand. The second strategy starts by solving for jamboree weeks and assigning the jamboree projects and finishes by completing the rest of the maintenance schedule in a regular way, starting from the second week of the

year. We now present these common points in detail and then explain the four algorithms developed.

2.3.1 Partitioning the Projects

For partitioning the projects an integer programming (IP) formulation is solved separately for both rail and tie-surfacing projects. It is used to partition the projects among small and large teams. By the partitioning formulation, we ensure that the total time of projects that are assigned to a team type does not exceed the available number of working weeks for that team type. The available number of working weeks for a team type can be found by multiplying the number of teams of that type by the number of weeks in a year. It is the upper bound of total time of projects that can be assigned to a team type. We also enforce a lower bound limit on the total time of projects assigned. The limits applied to rail and tie-surfacing teams are the same. By this way, the average number of working weeks for small or large rail and tie-surfacing teams is kept closer. We create a couple of partitions by changing the limits of working weeks assigned to a team type. For these purposes we use the following straightforward constraints:

1. Total time of rail projects that are assigned to SR teams must be less than the available number of working weeks for SR teams and greater than the lower bound limit determined on the working weeks.

2. Total time of rail projects that are assigned to LR teams must be less than the available number of working weeks for LR teams and greater than the lower bound limit determined on the working weeks.

These constraints are applied similarly to the tie-surfacing projects.

2.3.2 Scheduling Jamboree Projects

Before assigning any other project, we schedule the projects that have to be done within jamboree weeks. First, jamboree projects are partitioned among small and large teams so that the sum of the project durations assigned to a particular team type does not exceed the available working weeks. Then, we solve an assignment model that assigns each jamboree project to one or two teams depending on their duration. Note that jamboree

projects have priority during jamboree weeks. While approaching jamboree weeks, if a project's duration crosses a jamboree project that is already assigned then the project is interrupted until the jamboree project is completed.

2.3.3 1-Weekly Algorithm

In order to form an annual maintenance schedule, we solve a linear integer programming formulation for each week iteratively. Since the first week of each year is a holiday, we start by solving for the second week. Once the model is solved for week w, some slots for the following weeks are also filled since each project may take more than a week. Hence when the model is solved for week $w + 1$, some of the slots may already be occupied due to the assignments made on previous weeks. The schedule that is generated iteratively is kept in the M matrix. The M_{wt} entry in this matrix shows the project assigned to team t in week w. Each time before solving the model for a week w, we record two sets using the matrix M: The first is the set of unoccupied team slots (E). The set E is utilized while forming the constraints of the 1-weekly formulation. The sets E_{SR}, E_{LR}, E_{ST} and E_{LT} are the disjoint subsets of the set E and they define the unoccupied team slots available for a particular team type in week w. The second set recorded is P_{split} which includes the projects that are to be split. A project is necessarily added to this set if the following conditions are satisfied:

1. The project's duration is more than the remaining weeks of the planning period (a year).
2. There are at least two available slots for the team type to which the project is assigned.

If a project's duration is more than the remaining weeks but if the second condition is not satisfied, then we add the project to the set P_{na} which shows that the project should not be assigned in this week. Note that P_{na} is a temporary list for the current week's iteration in the 1-weekly algorithm. We also maintain the set of available projects ($\check{P}$) throughout the algorithm, and this set keeps decreasing as we iterate through the weeks.

Initially the set $\check{P}$ consists of all the projects to be assigned. As the projects are assigned to the teams, they are removed from the set $\check{P}$.

Before presenting the 1-weekly formulation and the algorithm in detail, we show the schematic representation of the first few weeks to help with understanding how the algorithm flows. In Table 2-1, none of the projects initially are assigned to a team and all of the slots are empty. After 1-weekly model is solved for the second week, the empty slots that will be filled in the third week become definite. The process continues in a similar fashion on a weekly basis until no projects remain for assignment. Note that the solution of the 1-weekly model in a week w depends on the sets $\check{P}$ and E which are both dependent upon the decisions made in the previous weeks.

1-Weekly Integer Programming Formulation. The formulation is solved for each week w, starting from the second week to the last week of the year. We now present the linear integer formulation. Considering that the sets of available projects and slots ($\check{P}$ and E) change dynamically for each week during the flow of the algorithm, the model given assumes that we are in week w.

Indices:

w : Week

p : Project

u : Service corridor

s : Subdivision

a : Adjoining subdivision pair

e : Team type

t : Team number

Sets:

W : Set of all weeks in the planning horizon

W_J : Set of jamboree weeks

P_R : Set of rail projects

P_{TS}	:	Set of tie-surfacing projects
P	:	Set of all projects, $P = P_R \cup P_{TS}$
$\check{P}$	:	Set of available projects
P_{split}	:	Set of projects that have to be split
K	:	Set of team types: $K = \{SR, LR, ST, LT\}$
E_e	:	Set of unoccupied team slots for team type e for the current week w
E	:	Set of all unoccupied team slots for the current week w, $E = \bigcup_{e \in K} E_e$
P_s	:	Set of absolute curfew projects under subdivision s
J	:	Set of jamboree projects, $J \subset P$
T_e	:	Set of teams of type $e \in K$
T	:	Set of all teams, $T = \bigcup_{e \in K} T_e$
S	:	Set of subdivisions
U	:	Set of service corridors
S^{adj}	:	Set of pairs of adjoining subdivisions
P_e	:	Set of projects assigned to team type e in current partition

Parameters:

α_e	:	Number of available teams of type e
γ_p	:	1 if project p requires absolute curfew, 0 otherwise
μ	:	Maximum number of absolute curfews allowed per week
m_s	:	Total number of projects that require absolute curfew in subdivision s
λ_u^s	:	1 if subdivision s is in service corridor u
n_u	:	Number of subdivisions with $m_s * \lambda_u^s > 0$ in service corridor u
d_{ij}	:	Travel distance between projects i and j; $i, j \in P$
TW_{pw}	:	Number of time windows violated weeks if p is assigned to any teams in week w
$dlim$	:	Travel distance limit between two consecutive projects
r_{pt}	:	1 if a distance violation occurs in case project p is assigned to team t due to previous week's assignment ($d_{p`p} > d_{lim}$, $M_{w-1t} = p`$)

c_{va} : Per unit cost of absolute curfew constraint violation

c_{vm} : Per unit cost of mutually exclusive subdivision constraint violation

c_{vsc} : Per unit cost of service corridor at-most constraint violation

c_{tw} : Per unit cost of time windows constraint violation

c_{vd} : Per unit cost of distance constraint violation

Decision Variables:

x_{pt} : 1 if project $p \in \check{P}$ is started by team $t \in E$ (in the current 1-weekly period), 0 otherwise

y_s : 1 if subdivision $s \in S$ is under absolute curfew in current week w, 0 otherwise

va : Variable to convert curfew constraint to a soft constraint

vm_a : Variable to convert mutually exclusive constraint to a soft constraint; $a = (s_1, s_2) \in S^{adj}$

vsc_u : Variable to convert service corridor at-most constraint to a soft constraint; $u \in U$

The objective is to minimize the amount of violations in performance constraints (absolute curfew, mutually exclusive subdivisions, service corridor at-most, time windows and distance limit constraints). For example, if total number of subdivisions under absolute curfew in a week is 18, while the limit μ on absolute curfew is 15, then we count the amount of violation for absolute curfew constraint in this week as 3. The quality of the provided solution is determined by the amount of violations in performance constraints.

$$\min \quad c_{va} va + c_{vm} \sum_{a \in S^{adj}} vm_a + c_{vsc} \sum_{u \in U} vsc_u + c_{tw} TW_{pw} \sum_{p \in \check{P}} \sum_{t \in E} x_{pt} + c_{vd} \sum_{p \in \check{P}} \sum_{t \in E} r_{pt} x_{pt} \quad (2\text{–}1)$$

Apart from the basic objective function (2–1), we make the following modification to the objective during the flow of the algorithm: We try to get teams closer to the locations of the jamboree projects while moving toward the jamboree weeks. Six weeks prior to the jamboree weeks, we start to put distance violation costs for some of the available projects by considering their distance to the jamboree project ahead. By the aid of this

Table 2-1. 1-weekly iterative steps

Initial Tableau							
	Team 1	**Team 2**	**Team 3**	**...**	**...**	**...**	**Team K**
Week 1	-	-	-	-	-	-	-
Week 2	-	-	-	-	-	-	-
Week 3	-	-	-	-	-	-	-
...	-	-	-	-	-	-	-
...	-	-	-	-	-	-	-
Week 52	-	-	-	-	-	-	-
Before solving for week 3							
	Team 1	**Team 2**	**Team 3**	**...**	**...**	**...**	**Team K**
Week 1	-	-	-	-	-	-	-
Week 2	137	218	157	275	121	75	160
Week 3	-	-	157	275	121	-	160
Week 4	-	-	157	275	-	-	-
Week 5	-	-	-	275	-	-	-
Week 6	-	-	-	-	-	-	-
...	-	-	-	-	-	-	-
...	-	-	-	-	-	-	-
Week 52	-	-	-	-	-	-	-
Before solving for week 4							
	Team 1	**Team 2**	**Team 3**	**...**	**...**	**...**	**Team K**
Week 1	-	-	-	-	-	-	-
Week 2	137	218	157	275	121	75	160
Week 3	264	269	157	275	121	282	160
Week 4	-	269	157	275	-	282	-
Week 5	-	-	-	275	-	-	-
Week 6	-	-	-	-	-	-	-
...	-	-	-	-	-	-	-
...	-	-	-	-	-	-	-
Week 52	-	-	-	-	-	-	-
Before solving for week 5							
	Team 1	**Team 2**	**Team 3**	**...**	**...**	**...**	**Team K**
Week 1	-	-	-	-	-	-	-
Week 2	137	218	157	275	121	75	160
Week 3	264	269	157	275	121	282	160
Week 4	258	269	157	275	91	282	278
Week 5	-	-	-	275	91	-	278
Week 6	-	-	-	-	91	-	278
Week 7	-	-	-	-	91	-	-
Week 8	-	-	-	-	91	-	-
Week 9	-	-	-	-	-	-	-
...	-	-	-	-	-	-	-
...	-	-	-	-	-	-	-
Week 52	-	-	-	-	-	-	-

modification, the model is inclined to select projects that are closer to the jamboree projects' locations.

Constraints:

1. Determine if subdivision s is under absolute curfew or not. If project p is in subdivision s and requires absolute curfew, then $y_s \geq x_{pt}$ constraint assures that y_s equals 1 if project p is started by team t in the current week.

$$y_s \;\geq\; x_{pt} \qquad \forall \left\{ s \in S,\, p \in \check{P}_s,\, t \in E \right\} \tag{2–2}$$

2. Due to the assignments in previous weeks, some teams may already be occupied. If there is an active project in subdivision s which requires absolute curfew then subdivision s is set to be under absolute curfew.

$$y_s \;=\; 1 \qquad \forall \{ s \in S :\; M_{wt} = p,\; p \in P_s,\; t \notin E \} \tag{2–3}$$

3. If there is no project that requires absolute curfew in subdivision s, then subdivision s cannot be under absolute curfew.

$$y_s \;=\; 0 \qquad \forall \{ s \in S : m_s = 0 \} \tag{2–4}$$

4. Each project can only be assigned to a team that is qualified to do that project

$$x_{pt} \;=\; 0 \qquad \forall \left\{ p \in \check{P}_e,\, t \notin E_e,\, e \in K \right\} \tag{2–5}$$

5. Jamboree projects can not be assigned if the current week w is not a jamboree week.

$$x_{pt} \;=\; 0 \qquad \forall \{ p \in J,\, t \in T,\, w \notin W_J \} \tag{2–6}$$

6. At most one project can be assigned to each available team in each week. This means that a team can handle one project at a time.

$$\sum_{p \in \check{P}} x_{pt} \;\leq\; 1 \qquad \forall t \in E \tag{2–7}$$

7. If a project is to be split ($p \in P_{split}$), then it is assigned to two teams in week w.

$$\sum_{t \in E_e} x_{pt} \;=\; 2 \qquad \forall \left\{ p \in P_{split},\, p \in \check{P}_e,\, e \in K \right\} \tag{2–8}$$

8. If the project's duration is larger than the remaining weeks but it cannot be split in the current week ($p \in P_{na}$), we do not assign the project in the given week.

$$x_{pt} \;=\; 0 \qquad \forall \{ p \in P_{na},\, t \in E \} \tag{2–9}$$

9. Each project can be started by at most two teams of the same type.

$$\sum_{t \in E_e} x_{pt} \;\leq\; 2 \qquad \forall p \in \check{P}_e \tag{2–10}$$

10. For the current week, if number of available projects of type e is greater than or equal to unoccupied teams of the same type, a project will be assigned to each of these unoccupied teams.

$$\sum_{p \in \check{P}_e} x_{pt} \;=\; 1 \qquad \left\{\forall t \in E_e : \left|\check{P}_e\right| \geq E_e,\ \forall e \in K\right\} \tag{2–11}$$

11. For the current week, if the number of available unoccupied teams of type e is larger than the available projects of the same type ($\check{P}_e \notin P_{split}$), assign each of these projects to at least one of the available teams. Note that projects that are in set P_{split} are not part of this constraint since they reserve two unoccupied teams as stated in constraint (2–8).

$$\sum_{t \in E_e} x_{pt} \geq \;1 \qquad \left\{\forall p \in \check{P}_e \notin P_{split} : \left|\check{P}_e / P_{split}\right| < |E_e| - 2 * |P_{split}|\,,\ \forall e \in K\right\} \tag{2–12}$$

12. Sum of subdivisions that are under absolute curfew must be less than or equal to the given absolute curfew limit (This constraint is applied as a hard or soft constraint in specific places of the algorithm).

$$\sum_{s \in S} y_s \;\leq\; \mu + va \tag{2–13}$$

13. Adjoining subdivision pairs are mutually exclusive. They cannot be under absolute curfew in the same week simultaneously (applied as a hard or soft constraint interchangeably).

$$y_{s_1} + y_{s_2} \;\leq\; 1 + vm_a \qquad \forall\, a = (s_1, s_2) \in S^{adj} \tag{2–14}$$

14. Service Corridor At-Most Constraints: At most one subdivision within a service corridor may be under absolute curfew in any week.

$$\sum_{s \in S} y_s \lambda_u^s \;\leq\; 1 + vsc_u \qquad \forall u \in U \tag{2–15}$$

15. Bounds on integer variables.

$$x_{pt} \;\in\; \{0, 1\} \qquad \forall p \in \check{P},\ \forall t \in E \tag{2–16}$$

$$y_s \;\in\; \{0, 1\} \qquad \forall s \in S \tag{2–17}$$

16. Bounds on violation variables. Note that enforcing integrality for these variables is not required due to the construction of defined constraints. The amount of violation that can occur each week differ for each type of violation variables. That is why, the upper bounds of these variables are different. For instance, the lower bound for va variables is 0 and upper bound is number of teams minus the weekly limit on the number of subdivisions under absolute curfew. For instance, if there are 19 teams and if the weekly limit on maximum number of subdivisions under absolute curfew is 15, then violation amount in any week can be at most 4. This can happen in the worst case when 19 different projects are assigned to 19 teams in this week and all these 19 projects are located in different subdivisions and if they all require absolute curfew.

$$va \in [0, |T| - \mu] \tag{2–18}$$
$$vm_a \in [0, 1] \quad \forall\, a \in S^{adj} \tag{2–19}$$
$$vsc_u \in [0, n_u - 1] \quad \forall\, u \in U \tag{2–20}$$

Algorithm Details. The outline of our 1-weekly algorithm is shown in Figure 2-1. Each iteration of the algorithm constitutes finding the project-team matchings for a given week w. While assigning projects to the teams, we solve a linear integer formulation several times. First we try to honor all of the absolute curfew, mutually exclusive subdivisions, and service corridor at-most constraints simultaneously. This is relatively easy for the first several weeks, because there are many available projects to choose from and thus more flexibility. However, as we move toward the future, it may not be always possible to satisfy all of the performance related constraints at the same time as hard constraints. In such cases, we introduce the performance constraints as soft constraints and penalize the violations in the objective function. Relaxation of hard constraints may be done in a multi step approach by relaxing them one by one and in twos until we find a feasible solution. This approach is useful when the relative importance of constraints differs and in a situation, for instance, where you do not want a violation of a specific constraint for as long as possible and can sacrifice having more violations in other constraints.

The 1-weekly formulation that we have provided has the ability to make splitting decisions. The model makes the split if it finds it desirable. Note that this should not

be confused with the usage of the input set P_{split} which includes the projects that are split necessarily. A project is in set P_{split} if there is not enough time to do this project in the remaining weeks of the yearly planning period by considering the project's duration. Splitting a project between two teams can help in the following way: Imagine that two available teams start working on two different projects and assume that these projects both require absolute curfew and are located in different subdivisions. In this case, each of these teams will cause an increase in the number of subdivisions which are under absolute curfew. In weeks where absolute curfew constraint cannot be satisfied, letting two teams handle the same project at the same time may decrease the number of subdivisions that are under absolute curfew, and thus may allow the constraint to be held. This action will also shorten the handling time of the project by half. Therefore splitting a lengthy project may also decrease the chance of a continuing violation for long periods of time. Our observations show that the splitting ability makes it easier to find feasible solutions.

Steps of the algorithm
Input: Projects, teams, time windows of projects, subdivisions, and service corridors
Output: Maintenance timetable for the teams
Step 1. Initialization:
- **1.1** Get the inputs and form the necessary sets and arrays
- **1.2** Partition the projects
  - Partition the set of rail projects among small and large rail teams
  - Partition the set of tie-surfacing projects among small and large tie-surfacing teams
- **1.3** Construct and solve an IP for jamboree weeks to assign the jamboree projects

Step 2. For each partition obtained, find a yearly maintenance schedule:
- Set w=2
- **2.1** Construct and solve 1-weekly formulation for week w with absolute curfew, mutually exclusive and service corridor constraints as hard constraints
  - If model is infeasible, solve the formulation by relaxing all performance constraints
- **2.2** Implement week w's schedule. Record the number of performance violations in week w.
  - Set $w = w + 1$ and go to 2.1

Figure 2-1. 1-weekly algorithm

2.3.4 k-Weekly Algorithm

The promising results obtained by using the 1-weekly algorithm led us to develop the idea further. For each iteration, instead of solving for one week's assignment, we solve k-weekly IP models. The algorithm solves for a k-weekly period and implements only the

first week of this period. Then it continues solving for the next k-weekly period which starts with the week after the implemented week. The purpose of the k-weekly algorithm is to improve the optimization by solving the IP model over a longer time horizon. Focusing on assigning projects for only one week allows the model to assign projects to teams without considering the upcoming weeks. Hence, the model may not foresee possible future violations. As the value of the k gets bigger, the computational time of the model gets higher due to an increasing combinatorial effect. Therefore, we set the value of k as high as possible while trying to keep an iteration's solution time within practical limits.

k-Weekly Integer Programming Formulation. Our linear integer programing formulation used in finding a solution for a k-weekly period is shown below. We start by presenting only the additional sets and parameters used for this formulation.

Sets:

$W`$	:	Set of weeks included in the current k-weekly period
E_e^w	:	Set of unoccupied team slots for team type e in week w
E^w	:	Set of all unoccupied team slots in week w
W_w^p	:	Set of potential starting weeks for project p that passes through week w: $\{w` \in W_w^p : \ w \geq w` \geq w - \lceil t_{pe}/2 \rceil + 1, \ p \in P_e, \ w` \in W`\}$
$\bar{W}_w^p$	:	Set of potential starting weeks for project p that passes through week w if it is not split: $\left\{w` \in \bar{W}_w^p : \ w - \lceil t_{pe}/2 \rceil \geq w` \geq w - t_{pe} + 1, \ p \in P_e, \ w` \in W`\right\}$

Parameters:

ϑ	:	Total duration of available projects
w_1	:	First week of the current k-weekly period
w_L	:	Last week of the current k-weekly period
t_{pe}	:	Number of weeks to complete project p given that it is assigned to team type e
$r`_{pt}$	:	1 if a distance violation occurs in case project p is assigned to team t in week w_1 due to previous week's assignment ($d_{p`p} > d_{lim}$, $M_{w_1-1t} = p`$)

Decision Variables:

x_{ptw}	:	1 if project p is started by team t in week w, 0 otherwise
y_{sw}	:	1 if subdivision s is under absolute curfew in week w, 0 otherwise
va_w	:	Variable to convert an absolute curfew constraint to a soft constraint
vm_{aw}	:	Variable to convert a mutually exclusive constraint to a soft constraint
vsc_{uw}	:	Variable to convert a service corridor at-most constraint to a soft constraint

$$\min \sum_{w_1}^{w_L} \left\{ c_{va} va_w + c_{vm} \sum_{a \in S^{adj}} vm_{aw} + c_{vsc} \sum_{u \in U} vsc_{uw} + c_{tw} TW_{pw} \sum_{p \in P} \sum_{t \in T} x_{ptw} \right\} \tag{2–21}$$
$$+ c_{vd} \sum_{p \in P} \sum_{t \in T} r`_{pt} x_{ptw_1}$$

subject to

$$y_{sw} \geq x_{pt\bar{w}} \quad \forall \left\{ s \in S,\ w \in W`,\ p \in \check{P}_s,\ t \in E^w,\ \bar{w} \in W_w^p \right\} \tag{2–22}$$

$$y_{sw} \geq x_{pt\bar{w}} - \sum_{t` \neq t} x_{pt\bar{w}} \quad \forall \left\{ s \in S,\ w \in W`,\ p \in \check{P}_s,\ t \in E^w,\ \bar{w} \in \bar{W}_w^p \right\} \tag{2–23}$$

$$y_{sw} = 1 \quad \forall \{ s \in S,\ w \in W` :\ M_{wt} = p,\ p \in P_s,\ t \notin E^w \} \tag{2–24}$$

$$y_{sw} = 0 \quad \forall \{ s \in S,\ w \in W` : m_s = 0 \} \tag{2–25}$$

$$x_{ptw} = 0 \quad \forall \{ p \in P_e,\ t \notin E_e^w,\ w \in W`,\ e \in K \} \tag{2–26}$$

$$x_{ptw} = 0 \quad \forall \{ p \in J,\ w \notin W_J,\ t \in E^w \} \tag{2–27}$$

$$x_{ptw} = 0 \quad \forall \left\{ p \in \check{P}_e,\ t \in E_e^w,\ w \in W` : w + \lceil t_{pe}/2 \rceil - 1 > |W| \right\} \tag{2–28}$$

$$\sum_{t` \neq t} x_{pt`w} \geq x_{ptw} \quad \forall \left\{ p \in \check{P}_e,\ t \in E_e^w,\ w \in W` : w + t_{pe} - 1 > |W| \right\} \tag{2–29}$$

$$\sum_{p \in \check{P}} x_{ptw} \leq 1 \quad \forall \{ t \in E^w,\ w \in W` \} \tag{2–30}$$

$$\sum_{t \in E_e^w} x_{ptw} \leq 2 \quad \forall \{ p \in P_e,\ w \in W`,\ e \in K \} \tag{2–31}$$

$$\sum_{t \in E_e^w} \sum_{p \in \check{P}_e} \sum_{\bar{w} \in \bar{W}_w^p} x_{ptw} = |E_e^w| \quad \left\{ \forall w \in W` : \vartheta_e \geq \sum_{w_1}^{w} |E_e^w|,\ \forall e \in K \right\} \tag{2–32}$$

$$\sum_{\bar{p}\in\check{P}}\sum_{\bar{w}=F_1}^{T_1} x_{\bar{p}t\bar{w}} \leq (1-x_{ptw})(T_1-F_1+1),\ \forall p\in\check{P}_e,\ w\in W`\setminus w_L,\ t\in E^w \tag{2–33}$$

$$\sum_{\bar{p}\in\check{P}}\sum_{\bar{w}=F_2}^{T_2} x_{\bar{p}t\bar{w}} \leq (1-x_{ptw}+\sum_{t`\neq t} x_{pt`w})(T_2-F_2+1),\ \forall p\in\check{P}_e,\ w\in W`\setminus w_L,\ t\in E^w \tag{2–34}$$

$$\sum_{s\in S} y_{sw} \leq \mu + va_w \qquad \forall w\in W` \tag{2–35}$$

$$y_{s_1w}+y_{s_2w}\leq 1+vm_{aw} \qquad \forall\, a=(s_1,s_2)\in S^{adj},\ w\in W` \tag{2–36}$$

$$\sum_{s\in S} y_{sw}\lambda_u^s \leq 1+vsc_{uw} \qquad \forall u\in U,\ w\in W` \tag{2–37}$$

$$x_{ptw}\in\{0,1\} \qquad \forall p\in\check{P},\ \forall t\in E^w,\ \forall w\in W` \tag{2–38}$$

$$y_{sw}\in\{0,1\} \qquad \forall s\in S,\ \forall w\in W` \tag{2–39}$$

$$va_w\in[0,|T|-\mu] \qquad \forall w\in W` \tag{2–40}$$

$$vm_{aw}\in[0,1] \qquad \forall\, a\in S^{adj},\ \forall w\in W` \tag{2–41}$$

$$vsc_{uw}\in[0,n_u-1] \qquad \forall\, u\in U,\ \forall w\in W` \tag{2–42}$$

where
$$\begin{aligned} F_1 &= w+1 \\ T_1 &= \min\{w_L, w+\lceil t_{pe}/2\rceil-1\} \\ T_2 &= \min\{w_L, w+t_{pe}-1\} \\ F_2 &= w+\lceil t_{pe}/2\rceil \end{aligned}$$

Most of the constraints resemble the 1-weekly formulation except that they are defined for all weeks in the k-weekly period. Constraints (2–33) and (2–34) are the most important constraints in the k-weekly model. If an available project p is started at week w by a team t, then these constraints ensure that no other projects can be assigned to team t until this project ends. Note that the k-weekly model fills a portion ($k\times|T|$) of the M matrix, then inside the flow of the algorithm the schedule matrix M is completed taking into account the projects whose duration exceeds the period length. Number of variables and constraints in the k-weekly model are O($k\,[|P|\,|T|+|S|+\left|S^{adj}\right|+|U|]$) and

$O(k\,[|S|\,|T|+|T|+|P|+|S^{adj}|+|U|])$ respectively. They are k times the number of variables and constraints of the 1-weekly model.

Algorithm Details. The outline of our k-weekly algorithm is shown in Figure 2-2.

Steps of the algorithm
Input: Projects, teams, time windows of projects, subdivisions, and service corridors
Output: Maintenance timetable for the teams
Step 1. Initialization:
- **1.1** Get the inputs and form the necessary sets and arrays
- **1.2** Partition the projects
  - Partition the set of rail projects among small and large rail teams
  - Partition the set of tie-surfacing projects among small and large tie-surfacing teams
- **1.3** Construct and solve an IP for jamboree weeks to assign the jamboree projects

Step 2. For each partition obtained, find a yearly maintenance schedule:
Start with the first k-weekly period
- **2.1** Construct and solve the k-weekly formulation by setting performance constraints as hard constraints
  - If model is infeasible, solve by relaxing all performance constraints
- **2.2** Implement week w_1's schedule. Record the number of performance violations in week w_1. Continue to the next k-weekly period that starts with $w_1 + 1$, then go to 2.1

Figure 2-2. k-weekly algorithm

By applying a k-weekly formulation in each iteration, we minimize possible violations that may occur in a longer time horizon. The goal is to better predict the future. For this reason, we solve for k weeks and implement the first week of these k weeks iteratively to the end of the year.

2.3.5 Backtracking

In order to better manage possible future violations, we delayed implementing the solution of 1-weekly or k-weekly formulations and continued to solve for a specified portion of the remaining time horizon. If a violation is observed then the solution of the current period is changed by introducing a new constraint. After obtaining an alternative solution, we again solve for the upcoming periods and check whether any violations appear. The process of changing the current period's assignments continues until no future violations are seen. If future violations cannot be prevented in any of the attempts then we solve by relaxing the performance constraints and continue with the following period. We have named the process of checking possible future violations and then going back and changing

the current assignment in case of foreseen violations as "backtracking". We will now explain the idea for the 1-weekly and k-weekly algorithms.

2.3.5.1 1-weekly algorithm with backtracking

Each iteration of the algorithm includes backtracking. After we find a solution for the current week w, we don't implement the schedule immediately. First, the projects that are assigned to the empty slots are saved. Then, we solve for a number of weeks in the future and each time we find a violation we alter the assignment of week w. For this purpose, a constraint that accepts at most ρ of the previously assigned projects in week w is added to the original formulation. The constraint is given in (2–43). The set P" is the set of projects previously assigned to the empty slots. The value of ρ equals $|E| - 1$ initially and decreases by 1 each time a future violation is detected for the current set of assigned projects.

$$\sum_{p \in P"} \sum_{t \in E} x_{pt} \quad \leq \quad \rho \tag{2–43}$$

Algorithm Details. The outline of our 1-weekly algorithm with backtracking is displayed in Figure 2-3.

In step 2.1a, we move forward at least two weeks depending on the current week w. If we cannot find a violation in the upcoming weeks, we accept the current assignments of projects to the teams. Otherwise, we try to change the current solution and recheck for possible future violations. In the worst case, we force the model to change all of the projects assigned and find a completely different solution. We then recheck for the upcoming weeks. After this point the value of ρ decreases below zero. If we cannot find a solution that lets us avoid future violations during the last attempt, we simply accept the solution at hand and continue with the next week.

Steps of the algorithm
Input: Projects, teams, time windows of projects, subdivisions, and service corridors
Output: Maintenance timetable for the teams
Step 1. Initialization:
1.1 Get the inputs and form the necessary sets and arrays
1.2 Partition the projects
Partition the set of rail projects among small and large rail teams
Partition the set of tie-surfacing projects among small and large tie-surfacing teams
1.3 Construct and solve an IP for jamboree weeks to assign jamboree projects
Step 2. For each partition obtained, find a yearly maintenance schedule:
Set $w = 2$
2.1 Construct and solve 1-weekly formulation for week w with absolute curfew, mutually exclusive and service corridor constraints as hard constraints
2.1a If model is satisfied, solve from week $w + 1$ to $\max\{w + (|W| - w)/5, w + 2\}$
If none of the weeks cause a violation, go to step 2.2
Else if $\rho \geq 0$, add constraint (2–43) that accepts at most ρ of the previously assigned projects in week w. Set $\rho = \rho - 1$ and return to step 2.1
Else ($\rho < 0$), go to step 2.2
2.1b If model is infeasible, solve by relaxing all performance constraints
2.2 Implement week w's schedule. Record the number of performance violations in week w. Set $w = w + 1$, $\rho = |E| - 1$ and go to 2.1

Figure 2-3. 1-weekly algorithm with backtracking

2.3.5.2 k-weekly algorithm with backtracking

The k-weekly algorithm by itself considers k weeks and implements the first week. In the k-weekly algorithm with backtracking, we extend the idea by solving three more k-weekly periods to decide whether to accept the solution for week w.

Algorithm Details. The process is similar to 1-weekly algorithm with backtracking. Note that in each iteration of the algorithm, model determines the schedule of a k-weekly period, hence we record the unoccupied team slots for all of the k-weeks to form the sets E^w. Once a model is solved, we save the projects assigned to these slots. In order to check future violations, we prefer to solve for three k-weekly periods. If we do not find a violation in these three k-weekly periods, we accept the solution for w_1, otherwise we add a constraint of type (2–43) to change the set of assigned projects assigned to the empty slots.

2.4 Computational Tests

We implemented our algorithms in Java using CPLEX 11. Computational experiments were run on a 2.5 GHz PC with 2 GB of memory.

2.4.1 Testing with Real Life Instances

We have tested our algorithms on two real life instances provided by the railroad company. These two instances belong to years 2007 and 2008 respectively. The size of these instances is given in Table 2-2.

Table 2-2. Size of the 2007 and 2008 instances

Parameters	Year 2007	Year 2008
Number of Projects (Rail/Tie)	254 (139/115)	265 (150/115)
Number of Teams	19	19
Number of Rail Teams (Small/Large)	10 (9/1)	10 (7/3)
Number of Tie-surfacing Teams (Small/Large)	9 (2/7)	9 (2/7)
Number of Weeks	51	51
Number of Service Corridors	10	10
Number of Subdivisions	282	282

For 2007 instance, minimum project duration is 1 week and maximum duration is 36 weeks. Average project duration is 5.19 weeks with a standard deviation of 5.04 weeks. For 2008 instance, minimum and maximum project durations are 1 week and 28 weeks respectively. Average and standard deviation of project durations are 5.07 and 4.78 weeks. For both of these instances, proportion of total duration of projects to the available working duration is over 0.95. We assume equal priority between performance constraints; however, railroad planners can use different cost parameters and hence give different weights to the performance constraint violations based upon their business priority. The violations in the solutions implemented by the railroad company in 2007 and 2008 are shown in Table 2-3.

Table 2-3. Number of violations detected in the solutions implemented by the railroad

Violations	2007 Instance	2008 Instance
Time-Window	154	7
Distance Constraints	59	30
Absolute Curfew	2	0
Mutually Exclusive Subdivisions	29	43
Service Corridor	24	52
Total Violations	268	132

2.4.2 Comparison of Algorithms

No backtracking, 1 versus k weekly. The number of violations in performance constraints for 1-weekly and k-weekly algorithms are presented in Tables 2-4 and 2-5. As we expected, the k-weekly algorithm performs better than the 1-weekly algorithm. Note that in the k-weekly algorithm we solve for k weeks and implement the first of these weeks in each iteration. In the k-weekly algorithm, at the stage where we may have to relax all performance constraints, a solution with more violations in the first week may be accepted, but when total number of violations in all k weeks are considered k-weekly approach is more holistic and has the advantage of making better project assignments. Hence, it may result in a lower number of overall violations. We tested k-weekly algorithm by setting value of k as 3, 4, 5 and 6. Using these values, we observed that iterative model sizes and running time of the algorithm was in the practical limits.

Table 2-4. Comparison of violations: no backtracking, 1 versus k weekly (2007)

Violations (2007 instance)	1-weekly w/o backtracking	k-weekly w/o backtracking
Time-Window	71	92
Distance Constraints	60	42
Absolute Curfew	0	1
Mutually Exclusive Subdivisions	5	5
Service Corridor	21	11
Total Violations	157	151

Table 2-5. Comparison of violations: no backtracking, 1 versus k weekly (2008)

Violations (2008 instance)	1-weekly w/o backtracking	k-weekly w/o backtracking
Time-Window	0	0
Distance Constraints	30	32
Absolute Curfew	0	0
Mutually Exclusive Subdivisions	20	16
Service Corridor	45	25
Total Violations	95	73

With backtracking, 1 versus k weekly. We observe that backtracking improves the solutions of both 1-weekly and k-weekly algorithms when compared to the case

where no backtracking exists. k-Weekly with backtracking performs better than 1-weekly with backtracking as we expect. Comparisons between 1-weekly with backtracking and k-weekly with backtracking algorithms are presented in Tables 2-6 and 2-7.

Table 2-6. Comparison of violations: with backtracking, 1 versus k weekly (2007)

Violations (2007 instance)	1-weekly w/ backtracking	k-weekly w/ backtracking
Time-Window	96	78
Distance Constraints	52	42
Absolute Curfew	0	0
Mutually Exclusive Subdivisions	0	1
Service Corridor	5	11
Total Violations	153	132

Table 2-7. Comparison of violations: with backtracking, 1 versus k weekly (2008)

Violations (2008 instance)	1-weekly w/ backtracking	k-weekly w/ backtracking
Time-Window	0	0
Distance Constraints	42	22
Absolute Curfew	0	0
Mutually Exclusive Subdivisions	4	6
Service Corridor	15	26
Total Violations	61	54

Comparison of all algorithms. All of the four algorithms developed provides consistently better solutions in almost all types of performance constraints compared to the solutions implemented by the railroad (Table 2-3). Feedback from the railroad company about the obtained solutions was very positive. They are considering our iterative algorithms for deploying in the upcoming years. As per our observations, the success of the iterative algorithms depends on their ability to handle some critical decisions involved in the overall problem implicitly. These critical decisions are distance constraints which affect the order of projects handled by a team and the decisions of assigning projects to a large or a small team. Computation times for the provided solutions are given in Table 2-8. We observe that best solutions are obtained by using k-weekly with backtracking algorithms. As we expect, due to the size of the integer

formulations solved in each iteration 1-weekly algorithm without backtracking runs faster compared to the k-weekly algorithm without backtracking. When we compare 1-weekly without backtracking and 1-weekly with backtracking algorithms, there is an expected increase in solution times due to the backtracking process. This pattern is the same if k-weekly without backtracking and k-weekly with backtracking algorithms are compared. Note that generating this solution manually takes two weeks for the railroad planners.

Table 2-8. Average runtimes of the top five solutions with respect to total violations

	Runtime (secs)
1-weekly w/o backtracking	65.13
k-weekly w/o backtracking	512.11
1-weekly w/ backtracking	304.30
k-weekly w/ backtracking	920.57

2.4.3 Testing with Small Instances

We have modified the k-weekly formulation to obtain an exact formulation for the curfew planning problem. In particular, we removed partitioning of projects among small and large teams and added distance variables explicitly to the k-weekly model. Note that by using iterative algorithms we have the advantage of enforcing the distance constraints implicitly. Iterative algorithms check the previous slot's or week's (in the 1-weekly case) assignments in order to determine the assignments of future slots or weeks. In the analysis made for the original instance, we have seen that this approach helps identify good quality solutions. Furthermore the following results demonstrate that the exact models with explicit distance variables are not easy to solve within practical time limits. We have defined the following additional variables and constraints to form the exact formulation:

Additional Decision Variables and Constraints.

ls_p : 1 if project p is assigned to a large team, 0 otherwise; $p \in P$,

vd_{pw} : 1 if project p violates distance constraint in week w, 0 otherwise; $p \in P,\ w \in W$

$$\min \quad \sum_{w_1}^{w_L} \left\{ c_{va} va_w + c_{vm} \sum_{a \in S^{adj}} vm_{aw} + c_{vsc} \sum_{u \in U} vsc_{uw} + c_{tw} TW_{pw} \sum_{p \in P} \sum_{t \in T} x_{ptw} \right\} \tag{2–44}$$

$$+ c_{vd} \sum_{p \in P} \sum_{w \in W} vd_{pw}$$

subject to

$$x_{ptw} - \sum_{\substack{p' \neq p \\ d_{pp'} \leq d_{lim}}} x_{p't(w+t_{pe})} \leq vd_{pw} + \sum_{t' \neq t} x_{ptw} \quad \forall \{p \in P_e,\ t \in E_e^w,\ w \in W,\ e \in K\} \tag{2–45}$$

$$x_{ptw} - \sum_{\substack{p' \neq p \\ d_{pp'} \leq d_{lim}}} x_{p't(w+\lceil t_{pe}/2 \rceil)} \leq vd_{pw} + 1 - \sum_{t' \neq t} x_{ptw} \quad \forall \{p \in P_e,\ t \in E_e^w,\ w \in W,\ e \in K\} \tag{2–46}$$

$$ls_p \geq x_{ptw}, \quad \forall p \in P_R, \forall t \in T_2,\ \forall w \in W, \tag{2–47}$$

$$\sum_{w \in W} \sum_{t \in T_1} x_{ptw} \leq (1 - ls_p)2, \quad \forall p \in P_R, \tag{2–48}$$

$$\sum_{w \in W} \sum_{t \in T_1} x_{ptw} \geq 1 - ls_p, \quad \forall p \in P_R, \tag{2–49}$$

$$\sum_{w \in W} \sum_{t \in T_2} x_{ptw} \geq ls_p, \quad \forall p \in P_R, \tag{2–50}$$

$$\sum_{w \in W} \sum_{t \in T_2} x_{ptw} \leq 2ls_p, \quad \forall p \in P_R, \tag{2–51}$$

$$ls_p \geq x_{ptw}, \quad \forall p \in P_{TS}, \forall t \in T_4,\ \forall w \in W, \tag{2–52}$$

$$\sum_{w \in W} \sum_{t \in T_3} x_{ptw} \leq (1 - ls_p)2, \quad \forall p \in P_{TS}, \tag{2–53}$$

$$\sum_{w \in W} \sum_{t \in T_3} x_{ptw} \geq 1 - ls_p, \quad \forall p \in P_{TS}, \tag{2–54}$$

$$\sum_{w \in W} \sum_{t \in T_4} x_{ptw} \geq ls_p, \quad \forall p \in P_{TS} \tag{2–55}$$

$$\sum_{w \in W} \sum_{t \in T_4} x_{ptw} \leq 2ls_p, \quad \forall p \in P_{TS} \tag{2–56}$$

Note that w_L corresponds to the last week of planning horizon for the exact formulation. Constraints (2–45) and (2–46) enforce the distance constraints. If project p is assigned to two teams then the term $\sum_{t' \neq t} x_{ptw}$ equals one and constraint (2–45) becomes redundant since it would take $\lceil t_{pe}/2 \rceil$ weeks to complete the project, otherwise the term $\sum_{t' \neq t} x_{ptw}$ equals zero and constraint (2–46) is redundant as expected. Constraints (2–47) to (2–51) determine whether a rail project is assigned to a small or large rail team and take the necessary action. If a rail project is assigned to a small (large) rail team then constraints (2–48) to (2–51) assure that the project is not done by the other team type and it is assigned to at least one and at most two small (large) rail teams. Constraints (2–52) to (2–56) perform the same tasks for tie projects.

We formed 20 random instances. In order to form instances having similar characteristics as the real ones, we keep following statistics same as the real life instances: the ratio of the number of weeks required to complete the given projects to the number of available working weeks, the percentage of absolute curfew requiring projects and the percentage of mutually exclusive subdivision pairs. For the first 10 instances, the planning horizon's length is 5 weeks. For the remaining 10 instances, the length of the planning horizon is 10 weeks. The running time limit for the exact formulation is set to 8 hours. We check the best integer solutions detected by the exact model after 3, 5, 10, and 15 minutes and after 1 hour and 8 hours. The goal is to observe how the quality of the solutions found by the exact optimization model evolves over time. In Table 2-9, the second and third columns show the percentage gap and absolute gap between the best solution from iterative algorithms and the best solution from the exact model. The following (H-Time and E-time columns) show the times required to find these solutions by heuristic iterative algorithms and by exact optimization model respectively.

Given 8 hours running time, only 6 of the small instances (indicated with an asterisk) are solved to optimality by the exact model. The average running time for these 6

Table 2-9. Computational results with small instances: planning horizon 5 weeks

Instances	%Gap	Abs. Gap	H-Time (secs)	E-Time (secs)
1	0.00%	0	52.41	28,800
2	5.77%	3	15.53	28,800
3	3.70%	2	41.69	28,800
4*	11.54%	3	13.91	587.81
5*	1.96%	1	15.38	2,964.02
6*	0.00%	0	5.02	4,083.74
7*	5.13%	2	21.78	5,397.77
8*	4.55%	2	69.99	11,682.17
9	8.51%	4	63.13	28,800
10*	2.86%	1	30.36	1,455.69
Average	4.40%	1.8	32.92	

instances is 4,361.86 secs. For the same set of instances the average running times of iterative algorithms is 26.07 secs, with an average absolute gap of 1.5 violations.

Table 2-10 presents the progress of exact optimization model. The second and third columns show the time and gap of first feasible solutions found by exact model. The remaining six columns present how the solutions found by exact formulation evolved over time. Percentage gaps given are the optimality gaps reported by CPLEX after specified amount of time. For 4 of the 10 instances, exact model ended with only feasible solutions after 8 hours. It is observed that exact model could find an initial starting solution with an average gap of 60.40% in average of 422.77 secs over 10 instances.

This observation coupled with the evaluation of solutions found in discrete time intervals shows that exact model cannot find good solutions within practical time limits even in the case of small instances. This behavior is clearer when we solve for instances with a planning horizon of 10 weeks. In all 10 of these instances, exact model could not report a feasible solution within 8 hours of running time. The difficulty in solving exact formulation is due to two main reasons: Exact formulation requires adding decision variables to determine whether a project should be assigned to a small or a large team. Secondly, it also requires adding distance limit constraints to the formulation explicitly. On the other hand, iterative algorithms deal with distance constraints implicitly due to

Table 2-10. Progress of exact optimization model

Instances	$Time_1$ (secs)	$\%Gap_1$	%Gap in Exact Optimization Model 3 mins	5 mins	10 mins	15 mins	1 hr	8 hrs
1	536.58	55.36%	-	-	21.43%	11.29%	5.08%	3.45%
2	506.60	55.02%	-	-	38.14%	38.14%	18.92%	7.69%
3	293.28	56.17%	34.41%	34.41%	12.86%	12.86%	6.15%	3.17%
4	251.02	59.08%	-	12.82%	0.00%			
5	161.95	75.89%	67.55%	36.46%	36.46%	19.74%	0.00%	
6	576.84	32.01%	-	-	23.88%	23.88%	1.92%	0.00%
7	208.62	65.95%	-	57.32%	44.57%	17.74%	1.92%	0.00%
8	709.60	93.52%	-	-	-	87.54%	13.60%	0.00%
9	536.50	68.60%	-	-	51.42%	51.42%	17.81%	5.51%
10	446.70	42.42%	-	-	23.89%	16.50%	0.00%	
Average	422.77	60.40%						

their construction. Furthermore, iterative algorithms partition the projects among small and large teams before the iterative models are solved. Iterative models are based on sets of projects which are pre-partitioned among small and large teams.

2.5 Summary and Conclusions

The curfew planning problem (CPP) is an important real-life problem faced by railroads. The problem deals with scheduling rail track maintenance for the whole train network over a long-term horizon. Our study presents four different algorithms for finding the problem's solution with the objective of minimizing the total number of violations in the performance constraints. The algorithms are tested using real-life instances provided by a major North American railroad company. Comparisons among algorithms are provided. The best performing algorithm is the k-weekly algorithm with backtracking. In general, backtracking helps improve both the 1-weekly and k-weekly algorithms. There are two main reasons why backtracking is beneficial: Firstly, the solution of the current week affects the empty slots of the following weeks. Due to the already occupied teams and ongoing projects, it may be more difficult to find a schedule which does not violate performance constraints in the following weeks. Secondly, in any iteration, many possible ways exist for assigning projects to the teams. Particularly when there are many available projects, alternative ways for making the assignments increase. However, using a project

in a later stage can sometimes be more beneficial because it may create a possibility to honor the performance constraints. Backtracking may help delay the assignment of a project by checking for possible future violations.

We also compare the performance violations of the solution implemented by the railroad company with our solutions. The solution quality as determined by the number of violations in the performance constraints is improved by around 50% compared to that of the railroad company. Moreover, the running times of the algorithms fall within practical levels and the applied methods are flexible. The integer programming formulations provided are designed to run with dynamically changing data sets. Hence, one can enter a partial schedule and the algorithm can complete the rest of the schedule. For instance, after half of the year passes and some projects are completed, railroad planners can feed the remaining projects into the algorithm and conduct a new run by using different weights for different performance criteria. Moreover, the planner can preassign a rail/tie-surfacing project to a small or a large rail/tie-surfacing team, modify a given partition or else provide a new partition that considers their usual business practices.

In a typical exact formulation, distance constraints and decisions of assigning any project to a small or a large team make the problem even more complex. In this study, we have provided a successful way of handling complicated decisions and constraints. Our iterative algorithms implicitly deal with these complex decisions and constraints by defining flexible formulations on short planning horizons and then solving them successively.

CHAPTER 3
A NOVEL GENERIC MODEL TO SOLVE COMBINED NETWORK DESIGN AND COMMODITY ROUTING PROBLEMS

3.1 Introduction

The Multicommodity Capacitated Network Design (MCND) problem has many applications in transportation, telecommunication, energy, computer, and production systems (Balakrishnan et al., 1997; Magnanti & Wong, 1984; Minoux, 1989). This class of problems is frequently used to construct and improve networks, build service routes and schedules.

In MCND, several distinct commodities with given demand are to be routed on a given network. Each commodity is defined by its unique origin-destination node pairs. Depending on the application a commodity might be data, products, electricity, etc. Flow of the commodities is achieved by installing facilities on the arcs of the network. Facilities are transportation mediums that are required to carry the corresponding commodities. Each facility has a capacity and facilities are installed in discrete amounts. There are two types of costs for each arc: Fixed cost for each facility installed and variable cost for routing one unit of each commodity on a facility installed. The objective is designing network and arranging the flow of commodities such that total cost is minimized while satisfying capacity constraints and demand requirements. In this generic version of the problem, flow of commodities can be fractional meaning that several paths can be used to satisfy demand of a commodity from its origin to its destination. Hence, MCND problem can be called as bifurcated or splittable MCND problem. Most of the studies in the literature focus on bifurcated network design problems. However, non-bifurcated problems, where each commodity should flow on a single path, arise in many applications such as computer networks (Gavish & Altinkemer, 1990), telecommunication networks (van Hoesel et al., 2002, 2003), express package delivery, and freight transportation in consolidation-based carriers.

MCND is a well-known NP-hard problem (Magnanti & Wong, 1984; Minoux, 1989). Hence, heuristic approaches are mostly used for the solution of the problem compared to exact approaches. Uncapacitated versions of the MCND are studied extensively in the literature and several efficient specialized algorithms had been developed (Balakrishnan et al., 1989; Holmberg & Hellstrand, 1998; Magnanti et al., 1986; Magnanti & Wong, 1984). Uncapacitated network design problem is also NP-Hard (Balakrishnan et al., 1989; Magnanti & Wong, 1984) as the capacitated ones. However, MCND problem is comparatively more difficult and it arises more often in real-life cases.

There are several variants of the generic bifurcated MCND problem. If at most one unit of facility can be installed on each arc, then the problem is called as fixed-charge multicommodity capacitated network design problem (see e.g. Crainic et al., 2000; Ghamlouche et al., 2003; Holmberg & Yuan, 2000). In this case, binary design variables are used for each arc instead of general integer variables.

For some network design applications, routing costs can be ignored and only fixed costs are important. In these applications, we are trying to install multiple facilities of same capacity on the edges of the network. This variant of the MCND is introduced as network loading problem (NLP) by Magnanti et al. (1993) and Magnanti et al. (1995). With respect to capacity usage Magnanti et al. (1993) and Magnanti et al. (1995) solve undirected version of the NLP where capacity of a facility installed on an edge is shared by the commodities flowing in both directions. Hence, in undirected models sum of flow in both directions should not exceed the capacity of the facility. In the bidirected (also called as directed) NLP, once a facility is installed, same capacity can be used separately by commodities flowing in different directions (see e.g. van Hoesel et al., 2002, 2003). In directed models, upper bound of flow for commodities flowing in different directions are constrained separately using two constraint sets by the capacity of the installed facility.

Several different mathematical formulations had been proposed in the literature in order to model generic bifurcated MCND problem and its variants (such as fixed-charge

MCND, network loading problem, non-bifurcated MCND). These formulations have some common characteristics when we consider types of variables used. They mostly utilize arc-based or path-based flow variables and arc-based (edge-based) design variables. For instance, Crainic et al. (2000) and Katayama et al. (2009) use path-based flow variables and arc-based design variables. Ghamlouche et al. (2003), Holmberg & Yuan (2000), Frangioni & Gendron (2009), Crainic et al. (2001), Crainic et al. (2004), and Ghamlouche et al. (2004) use arc-based flow variables and arc-based design variables. Berger et al. (2000) and Gendron et al. (2002) solve network loading problems and use path-based flow variables and edge-based design variables. Magnanti et al. (1995) and Bartolini & Mingozzi (2009) use arc-based flow variables and edge-based design variables. Note that edge-based design variables are mostly used for network loading problems which is more common in telecommunications and computer networks. The only exception we know is the paper by Bartolini & Mingozzi (2009) which uses edge-based design variables to solve a non-bifurcated network design problem.

One common characteristic of all the formulations is to use arc-based (or edge-based) design variables. An arc-based design variable is used to determine number of facilities to be built on an arc. In this study, we utilize binary path-based design variables to solve a non-bifurcated network design problem. In this problem, facilities are installed on the arcs and provide a capacity only in the direction of the arcs they are installed on. The usual practice to solve this kind of problems is first deciding on the facilities to be built then determining how the commodities will flow over the facilities in a sequential manner. We can differentiate between completely isolated and interactive iterative algorithms. In a completely isolated iterative algorithm, underlying subproblems are solved in complete isolation and result of one subproblem is used as an input for the other; whereas, in an interactive iterative algorithm two problems are solved by iterating between each another, hence, results of both problems affect each other. In a typical interactive iterative solution procedure, you first build a set of potential facilities on the links of the network, then

select a commodity based on some criteria and identify a set of potential paths which can carry this commodity over the network of installed facilities. For each potential path which is a sequence of installed facilities, you assign candidate commodity and other available commodities which can go from the origin to the destination of the potential path and calculate cost of paths with assigned blocks. Then you select the best path with added commodities on it, update unassigned commodities and list of potential paths, and repeat same steps until no commodities remain unassigned. Gorman (1998) and Barnhart et al. (2002) provide interactive iterative algorithms for railroad operating plan and express shipment delivery, respectively.

Network design and commodity routing depends on each other and solving these problems sequentially may result into suboptimal solutions. Our motivation is to create a holistic approach that provides an integrated solution to network design and commodity routing problems. For this problem, we introduce a new formulation based on path-based flow variables and path-based design variables. Since we utilize path-based design variables, there are no fixed charge constraints in the introduced model. Proposed model enables us to solve network construction and commodity flow problems simultaneously in a more effective way compared to the sequential approach for a practical size instance of a major railroad company. Furthermore, the resultant holistic approach utilizing path-based design variables is flexible enough to handle many business constraints specific to the particular application context. The proposed generic model can be particularly beneficial for asset management consideration in various applications and can be adapted to solve non-bifurcated network design problems of other freight carriers such as less-than-truckload service providers and express package delivery companies.

3.2 Problem Description

In our problem, each commodity has to follow a single path from its origin to its destination. This version of the problem is known as non-bifurcated (or unsplittable, or binary) problem. Commodities have to share the capacity of the installed facilities on the

arcs of the network. Initially, there are no facilities installed on the arcs. We should decide at which arcs facilities are to be installed and how the commodities flow on the installed facilities without exceeding their capacity. The goal is to minimize fixed cost of installed facilities and variable routing costs while satisfying the transportation demand. We can categorize our problem as Non-bifurcated Multicommodity Capacitated Network Design (NMCND) problem which belongs to the general class of MCND problems. NMCND problem arises in many real life systems such as less-than-truckload trucking, express package delivery and railway freight routing.

Main inputs for the NMCND problem are the networks and commodities. Commodities can be any type of freight (blocks, a set of products, letters, messages, etc.) and are defined by distinct origin-destination pairs. We need to install facilities on arcs such that all commodities can be feasibly carried over the network. Facilities can be any type of transportation modes (cables, trucks, trains, planes, ships) depending on the application area (telecommunication, railways, trucking, etc.).

Let $G = (N, A)$ be the directed network where N is the set of nodes and A is the set of arcs. Let K denote the set of commodities. Each commodity $k \in K$ has demand of d^k which needs to be transported from its origin node $O(k)$ to its destination node $D(k)$. Using this notation, we will demonstrate previous models in the literature and the new model.

3.3 Traditional Bifurcated Multicommodity Network Design Models

Classical formulation for bifurcated MCND uses arc-based flow variables and arc-based design variables. This formulation is presented in Frangioni & Gendron (2009) as follows;

$$\min \sum_{k \in K} \sum_{(i,j) \in A} d^k c_{ij}^k x_{ij}^k + \sum_{(i,j) \in A} f_{ij} y_{ij} \tag{3–1}$$

subject to

$$\sum_{j \in N_i^+} x_{ij}^k - \sum_{j \in N_i^-} x_{ji}^k = \delta_i^k \qquad \forall i \in N,\ k \in K \tag{3–2}$$

$$\sum_{k \in K} d^k x_{ij}^k \leq u_{ij} y_{ij} \qquad \forall (i,j) \in A \tag{3–3}$$

$$0 \leq x_{ij}^k \leq 1 \qquad \forall (i,j) \in A,\ k \in K \tag{3–4}$$

$$y_{ij} \in Z^+ \qquad \forall (i,j) \in A \tag{3–5}$$

where $N_i^+ = \{j \in N \,|\, (i,j) \in A\}$, $N_i^- = \{j \in N \,|\, (j,i) \in A\}$ and $\delta_i^k = 1$ if $i = O(k)$, $\delta_i^k = -1$ if $i = D(k)$ and $\delta_i^k = 0$ otherwise. Continuous x_{ij}^k variables are the arc-based flow variables which indicate the fraction of commodity k flowing on arc $(i,j) \in A$ and integer y_{ij} variables are the arc-based design variables. c_{ij}^k is the variable cost of moving one unit of commodity k on arc (i,j) and f_{ij} is the fixed cost of installing one unit of facility on arc (i,j). d^k is the demand of commodity k and u_{ij} is the flow capacity provided by a facility installed on arc (i,j).

Another frequently used model to formulate MCND is the one with path-based flow variables and arc-based design variables. This formulation is can be presented as follows (see e.g. Crainic, 2000, for a similar representation of this model);

$$\min \quad \sum_{p \in P} d^p c_p x_p + \sum_{(i,j) \in A} f_{ij} y_{ij} \tag{3–6}$$

subject to

$$\sum_{p \in P^k} x_p = 1 \qquad \forall k \in K \tag{3–7}$$

$$\sum_{p \in P_{(i,j)}} d^p x_p \leq u_{ij} y_{ij} \qquad \forall (i,j) \in A \tag{3–8}$$

$$0 \leq x_p \leq 1 \qquad \forall p \in P \tag{3–9}$$

$$y_{ij} \in Z^+ \qquad \forall (i,j) \in A \tag{3–10}$$

where P^k is the set of paths for commodity k and $P = \cup_{k=1}^{K} P^k$. $P_{(i,j)}$ is the set of paths that flows through arc (i,j). We set demand of a path equal to the demand of its corresponding commodity ($d^p = d^k$ for all $p \in P^k$). Continuous x_p variables are the path-based flow variables which indicate the fraction of commodity k that flows on path $p \in P^k$.

3.4 Non-bifurcated Network Design Problem

In order to convert generic bifurcated MCND formulation to the non-bifurcated MCND problem, we can simply define flow variables as binary variables in the above two formulations. Note that requiring flow variables to take binary values considerably increases the difficulty of the non-bifurcated network design problem compared to the bifurcated case. The resulting NMCND formulation with arc-based flow variables and arc-based design variables is as follows:

$$\min \sum_{k \in K} \sum_{(i,j) \in A} d^k c_{ij}^k x_{ij}^k + \sum_{(i,j) \in A} f_{ij} y_{ij} \tag{3–11}$$

subject to

$$\sum_{j \in N_i^+} x_{ij}^k - \sum_{j \in N_i^-} x_{ji}^k = \delta_i^k \qquad \forall i \in N,\ k \in K \tag{3–12}$$

$$\sum_{k \in K} d^k x_{ij}^k \leq u_{ij} y_{ij} \qquad \forall (i,j) \in A \tag{3–13}$$

$$x_{ij}^k \in \{0,1\} \qquad \forall (i,j) \in A,\ k \in K \tag{3–14}$$

$$y_{ij} \in Z^+ \qquad \forall (i,j) \in A \tag{3–15}$$

NMCND with path-based flow variables and arc-based design variables is formulated as follows:

$$\min \sum_{p \in P} d^p c_p x_p + \sum_{(i,j) \in A} f_{ij} y_{ij} \tag{3–16}$$

subject to

$$\sum_{p \in P^k} x_p = 1 \qquad \forall k \in K \tag{3–17}$$

$$\sum_{p \in P_{(i,j)}} d^p x_p \leq u_{ij} y_{ij} \qquad \forall (i,j) \in A \tag{3–18}$$

$$x_p \in \{0,1\} \qquad \forall p \in P \tag{3–19}$$

$$y_{ij} \in Z^+ \qquad \forall (i,j) \in A \tag{3–20}$$

3.5 Proposed Non-bifurcated Network Design Model

In all of the four formulations defined for bifurcated and non-bifurcated network design problems, a major problem is the existence of fixed costs of installing facilities. In all formulations, while defining capacity constraints, we multiply arc-based design variables (or edge-based design variables especially in case of network loading problems) with the capacity of the installed facilities in the right-hand side of the constraint. In order to remove these fixed-charge constraints, we present a novel formulation which use path-based design variables and path-based flow variables. In this new model, selecting a path-based design variable corresponds selecting all the arcs of this path in the traditional arc-based design formulation. Once paths for design variables are enumerated, we use what we call facility-arcs instead of traditional arcs in the formulation. A traditional arc is a regular arc of the network and in arc-based design formulations decision to be taken is the number of facilities to install on the arc. A facility-arc resembles a facility installed on a specific arc, hence it is an (arc, facility) pair. Proposed model decides to use this facility if at least one commodity path that flows on the path of this facility-arc is selected. A facility-arc has the capacity of the facility that is planned to install on that arc. In the new formulation, we can construct our capacity constraints on the facility-arcs instead of traditional arcs. With the help of facility arcs, fixed-charge constraints used in the formulations utilizing arc-based design variables are removed. Instead of defining an integer variable for design variables, we are able to use binary variables to determine

whether we use a design path or not. Hence, in the model, binary variables are used for both design and flow decisions. The resulting formulation has a structure that allows less fractional values in its LP relaxation. The proposed formulation is as follows:

$$\min \quad \sum_{p \in P} d^p c_p x_p + \sum_{\bar{p} \in \bar{P}} f_{\bar{p}} y_{\bar{p}} \tag{3–21}$$

subject to

$$\sum_{p \in P^k} x_p = 1 \qquad \forall k \in K \tag{3–22}$$

$$\sum_{p \in P^{\bar{p}}_{(i,j)}} d^p x_p \leq u^{\bar{p}}_{ij} \qquad \forall \bar{p} \in \bar{P},\ (i,j) \in A_{\bar{p}} \tag{3–23}$$

$$y_{\bar{p}} \geq x_p \qquad \forall \bar{p} \in \bar{P},\ p \in P^{\bar{p}} \tag{3–24}$$

$$x_p \in \{0,1\} \qquad \forall p \in P \tag{3–25}$$

$$y_{\bar{p}} \in \{0,1\} \qquad \forall \bar{p} \in \bar{P} \tag{3–26}$$

where $\bar{P}$ is the set of all design paths. Note that each commodity path $p \in P$ is defined by a sequence of facility-arcs. $P^{\bar{p}}$ represents the set of all commodity paths that flows through design path $\bar{p}$. Hence, $P^{\bar{p}}$ includes any commodity path p that flows on at least one of the facility-arcs of this design path $\bar{p}$. $P^{\bar{p}}_{(i,j)}$ shows the set of commodity paths that flow on design path $\bar{p}$ which is passing through arc (i,j).

Proposed formulation can be applied in many non-bifurcated network design applications. It might be especially useful when the facilities installed are moving objects like trucks, trains, aircrafts, etc. In these applications, path-based design variables can easily help to define asset management related business constraints. For instance, a path-based design variable can be thought as a truck's route. A flow path variable can represent a commodity traveling on a segment of a given truck's path. Note that a commodity can also use several design paths (e.g. several truck paths) on its way from its origin to its destination.

3.6 Summary and Conclusions

We focused on a non-bifurcated network design problem where each commodity flows on a single path from its origin to its destination. In this problem, facilities are installed on the arcs and each facility provides a capacity in the direction of the arc on which it is installed. Initially, there are no facilities installed on the arcs. Such non-bifurcated network design problems may arise for various freight carriers such as less-than-truckload service providers, rail freight carriers, express package delivery companies, etc. We propose a novel model with the goal of solving network design and commodity routing problems simultaneously. For this purpose, we use path-based flow variables and path-based design variables which are both binary variables. As far as we know, path-based binary design variables had not been used in the literature for the solution of non-bifurcated network design problems. Furthermore, these two highly related problems were mostly solved using sequential approaches. Proposed model provides a holistic approach and is flexible for incorporating asset management considerations. In the next chapter, we adapt the proposed model to solve a practical size instance of a major railroad company.

CHAPTER 4
INTEGRATION OF TRAIN ROUTING AND BLOCK-TO-TRAIN ASSIGNMENT PROBLEMS: AN APPLICATION OF THE PROPOSED NETWORK DESIGN MODEL

4.1 Introduction

Operating plan of a railroad determines movement of rail car loads, locomotives and crews over rail network. Developing an operating plan requires to solve blocking, train routing, block-to-train assignment, empty car distribution and train timetabling problems. These problems are often solved separately due to size and difficulty of the overall problem. As an example, a major North American railroad company typically operates around 400 trains daily to move 1300 blocks on a network of 12000 nodes and 2000 nodes.

The objective in solving train routing and block-to-train assignment problems is to construct a set of trains such that all blocks can be carried feasibly over the train network. Hence, the main decisions to be given are trains (origin, destination and route of each train) and block-to-train assignments. These highly interrelated problems are usually solved in a sequential manner. Candidate trains are constructed first and then blocks are assigned to candidate trains using a cost function and best candidate train is selected to be built. This process continues in an iterative way until no blocks remain to be routed. Railroads need a more holistic approach for creating multiple trains at once while feasibly routing blocks over the set of trains formed.

Cordeau et al. (1998) make a very good review of optimization models for the train routing and scheduling problems. The first part of their review surveys train routing problems in the context of rail freight transportation and the second part reviews optimization models for train scheduling in both freight and passenger transportation. Jha et al. (2008) solve block-to-train assignment problem and assume that blocking plan, train routings and their schedule are given.

Different approaches have been introduced to solve interrelated railroad operating plan problems simultaneously. Gorman (1998) and Ahuja et al. (2005) confirm that

most papers in the literature solve train routing problem and block-to-train assignment problem separately. An iterative scheme is used to solve these two problems successively (see e.g. Crainic et al., 1984; Crainic & Rousseau, 1986; Haghani, 1989; Keaton, 1989, 1991). Gorman (1998) also decomposes the problem similarly. The author first generates candidate train schedules by using tabu-enhanced genetic search and then checks cost of each schedule by routing blocks over the schedule in an iterative way. Gorman (1998)'s main contribution is to produce a detailed weekly operating plan which provides which train runs which day.

In this study, we address the train routing and block-to-train assignment problems and adapt our generic non-bifurcated model to solve these problems simultaneously. We assume that blocking plan is available. We incorporate locomotive and crew considerations into this single model. The resulting model is successful in finding a good solution in reasonable time limits for a real world instance of a major North American railroad company. The proposed model is also suitable for obtaining an incremental solution in case some train paths or block paths are favorable and selected in advance by the railroad.

4.2 Problem Description

Main inputs of the problem are rail network and blocks. While building trains and assigning blocks to trains, we need to satisfy some feasibility constraints and also many business constraints. Feasibility constraints are the ones that already exist in the generic model. The definitions of these constraints using the terminology adapted from this problem are as follows:

1. Exactly one block path is selected for each block.
2. Relation between block path and train path variables: If a block path that flows on a train path t is selected then train path t should also be selected.

Business constraints incorporated into the proposed generic optimization model are as follows:

1. Train load capacities at links
   This constraint restricts the maximum volume (measured by number of cars) on a train while passing through an arc.

2. Train capacities at nodes
   Number of trains starting and terminating at a node is limited.

3. Train capacities at links
   Number of trains that can pass through a link is limited.

4. Block swaps at nodes
   Number of block swaps at a node is limited.

5. Locomotive imbalance at nodes
   In order to determine locomotive imbalance, we first need to determine locomotive requirements for a train. For this purpose, we introduce locomotive requirement variables that show the minimum locomotive requirement for each train. Our assumption is that minimum number of locomotives required to pull a train depends on the maximum locomotive requirement over all the segments of the train route. In order to find the imbalance, we need the actual flow of locomotives originating and terminating at each station. For this purpose, we define another set of locomotive flow variables, which is equal to the number of active locomotives if there is no locomotives deadheaded on the train. We then penalize the imbalance between the number of locomotives attached to originating trains and terminating trains at each station.

6. Crew imbalance on crew districts
   We define crew imbalance variables to determine the imbalance of trains running in opposite directions on a crew district and then penalize crew imbalance in the objective function.

7. Work-event capacity
   The number of trains that can stop at a node is limited. A selected block path generates work-events at all its switching nodes and a selected train path generates work-events at all its crew-changing nodes.

8. Crew work time limit
   Maximum work time (on track time plus stop time) in a crew district is limited.

Apart from the usual fixed and variable costs mentioned in the generic model, we also needed to incorporate many other objectives which are specific to this problem. These objective terms can be listed as follows:

Train Costs

1. Train Start Cost
   Fixed cost of running a potential train

2. Train Miles Cost
   Train distance cost for each potential train

3. Train Work Event Costs
   Cost incurred due to work events when a train stops at an intermediate station on its route. The intermediate work events occur due to pickups and setoffs of blocks or crew changes.

Car Costs

1. Block Swaps Cost
   Block swap costs for each block path. Individual costs depend on the number of block swaps performed in the corresponding block path.

2. Car Movement Cost
   Car movement cost is calculated for each block path by multiplying corresponding block's volume with block path's distance.

3. Car Hours Cost
   Car travel time cost depends on travel time of cars on selected block paths.

Locomotive costs

1. Locomotive Light Travel Cost
   Light travel is due to imbalance of locomotive flows over the network. If there is an imbalance, locomotives have to without pulling a train which is called as light travel.

2. Locomotive Ownership Cost
   This is the due to per hour cost of owning a locomotive.

3. Locomotive Active Pulling Cost
   This is the per hour operating cost of a locomotive actively pulling a train.

4. Locomotive Deadheading Cost
   This is the deadheading cost of a locomotive per hour

Crew costs

1. Active Crew Cost
   For each crew link on the path of a potential train, we calculate crew start cost, crew wages and trip cost. Crew wages depend on the travel time of train on the given crew link.

2. Crew Imbalance Cost
   Imbalance in number of trains running in opposite directions on a crew district is multiplied with the cost of a crew deadheading in that crew district.

4.3 Solution Approach

4.3.1 Generic Model Tailored for the Integrated Problem

Path based flow and design variables form the core of the proposed non-bifurcated capacitated network design model. In this specific problem, commodities are the given blocks and each commodity is to be carried by trains from their origin to their destination. In a model with arc based design variables, one can assume installing transportation mediums with sufficient capacity on the arcs of the network. However, actual route of a train cannot be determined by this approach. Since trains are moving over the network, using a path-based model would be beneficial to determine train routes and incorporate many asset management related objectives into the model which would otherwise not be possible unless a sequential approach is used. Block path variables correspond to path based flow variables and train path variables correspond to path based design variables in the proposed model. The proposed NMCND formulation can be adapted to formulate integrated train routing and block-to-train assignment by defining the following sets, parameters and decision variables.

Sets:

N	:	Set of nodes
A	:	Set of arcs
A_t	:	Set of arcs of potential train path t
T	:	Set of all potential train paths
T_i^+	:	Set of train paths originating at node i
T_i^-	:	Set of train paths terminating at node i
B	:	Set of all blocks
β	:	Set of all potential block paths
β^b	:	Set of potential block paths for block b

β^t	:	Set of potential block paths that flow on train path t
$\beta^t_{(i,j)}$	:	Set of potential block paths that use arc (i,j) and flow on train path t
$T_{(i,j)}$	:	Set of trains that pass through arc (i,j)
β^i	:	Set of potential block paths that makes a block swap at node i
W_{ti}	:	Set of block paths that cause a work event for train t at node i
W^U_{ti}	:	Set of block paths that cause a pick-up work event for train t at node i
W^O_{ti}	:	Set of block paths that cause a set-off work event for train t at node i
Z_i	:	Set of train paths that stops at crew-change node i
L	:	Set of all crew links
L_t	:	Set of crew links of train path t
$\overleftarrow{T}_l$, $\overrightarrow{T}_l$	:	Set of train paths that goes in opposite directions on crew link l
N_t	:	Set of nodes of train path t
N^l_t	:	Set of nodes of crew link l which are used by train t

Parameters:

d^p	:	Number of cars in the block associated with block path p, (note that $d^p = d^b \ \forall p \in \beta^b$)
$u^t_{(i,j)}$	:	Capacity of train t on link (i,j) in terms of number of cars it can carry
O^i	:	Maximum number of trains that can originate at node i
E^i	:	Maximum number of trains that can terminate at node i
F	:	Maximum number of locomotives (active and deadheading) that can be attached to a train
∂	:	Minimum number of active locomotives required to pull a train
$\wp$	:	Maximum number of active locomotives that can be used to pull a train
$H_{(i,j)}$	:	Horse power requirement per unit weight on arc (i,j)
H_{std}	:	Horse power of a standard locomotive
w_p	:	Weight of the block which is carried by block path p (Note that $w_p = w_b \ \forall p \in \beta^b$)

$\ell_{(i,j)}$	:	Maximum number of trains that can pass through arc (i, j)
∇^i	:	Block swap capacity at node i
C_i	:	Work-event capacity of node i
ω_t^l	:	Travel time of train t on crew link l
u_i	:	Stop time for a pick-up work event at node i
o_i	:	Stop time for a set-off work event at node i
$\Re$	:	Maximum crew work time in a crew link
f_t	:	Fixed cost of running potential train path t
m_t	:	Train miles for potential train t
c_1	:	Cost of moving a train per mile
c_2	:	Intermediate work events cost per stop
n_p	:	Number of block swaps for block path p
θ_p	:	Block swap cost depending on the volume of the corresponding block for block path p
v_p	:	Cost of moving cars of block path p
ς_p	:	Car travel time cost for block path p depending on volume of the corresponding block and travel time of the cars along the block path
σ_l	:	Cost paid to active crews on crew link l
κ_l	:	Crew deadheading cost for the crew link l
γ	:	Light travel penalty for a locomotive
c_3	:	Cost of actively pulling a train per hour
c_4	:	Cost of deadheading a locomotive per hour
c_5	:	Cost of owning a locomotive per hour
ρ_t	:	Travel time of potential train path t in hours

Decision Variables:

x_p	:	1 if candidate block path $p \in \beta$ is selected, 0 otherwise
y_t	:	1 if potential train path $t \in T$ is selected, 0 otherwise

$\tilde{v}_i$	:	Locomotive flow imbalance at node i
$\bar{v}_l$	:	Crew imbalance at crew link l
$\Im_i$	:	Amount of violation of work-event capacity at node i
w_{ti}	:	1 if train t stops at node i for a work-event, 0 otherwise
r_t	:	Active locomotive requirement for train path t
z_t	:	Number of locomotives attached to train t
		(active plus deadheading locomotives)

Note that model lets us to define different capacities on each link for a given train. This is more realistic since railroads may require the load limits to be changed according to geographical conditions on that arc of the rail network. However, one can also use a constant capacity, in that case $u^t_{(i,j)} = u^t$ for all arcs (i,j). Using these notations the model is formulated as follows:

$$\min \quad \sum_{t\in T} f_t y_t + c_1 \sum_{t\in T} m_t y_t + c_2 \sum_{t\in T}\sum_{i\in N_t} w_{ti} + \sum_{p\in\beta} \theta_p n_p x_p + \sum_{p\in\beta} v_p x_p + \sum_{p\in\beta} \varsigma_p x_p \tag{4–1}$$
$$+ \sum_{t\in T}\sum_{l\in L_t} \sigma_l y_t + \sum_{l\in L} \kappa_l \bar{v}_l + \gamma \sum_{i\in N} \tilde{v}_i + c_3 \sum_{t\in T} \rho_t r_t + c_4 \sum_{t\in T} \rho_t (z_t - r_t) + c_5 \sum_{t\in T} \rho_t z_t$$

subject to

$$\sum_{p\in\beta^b} x_p = 1 \qquad \forall b \in B \tag{4–2}$$

$$\sum_{p\in\beta^t_{(i,j)}} d^p x_p \le u^t_{ij} \qquad \forall t \in T,\ (i,j) \in A_t \tag{4–3}$$

$$y_t \ge x_p \qquad \forall t \in T,\ p \in \beta^t \tag{4–4}$$

$$\sum_{t\in T_i^+} y_t \le O^i \qquad \forall i \in N \tag{4–5}$$

$$\sum_{t\in T_i^-} y_t \le E^i \qquad \forall i \in N \tag{4–6}$$

$$\sum_{p\in\beta^i} x_p \le \nabla^i \qquad \forall i \in N \tag{4–7}$$

$$\sum_{t \in T_{(i,j)}} y_t \leq \ell_{(i,j)} \qquad \forall (i,j) \in A \tag{4–8}$$

$$w_{ti} \geq x_p \qquad \forall t \in T,\ i \in N_t,\ p \in W_{ti} \tag{4–9}$$

$$w_{ti} \geq y_t \qquad \forall t \in T,\ i \in N_t :\ t \in Z_i \tag{4–10}$$

$$\sum_{t \in T} w_{ti} \leq C_i + \Im_i \qquad \forall i \in N \tag{4–11}$$

$$\bar{v}_l \geq \sum_{t \in \overleftarrow{T}_l} y_t - \sum_{t \in \overrightarrow{T}_l} y_t \qquad \forall l \in L \tag{4–12}$$

$$\bar{v}_l \geq \sum_{t \in \overrightarrow{T}_l} y_t - \sum_{t \in \overleftarrow{T}_l} y_t \qquad \forall l \in L \tag{4–13}$$

$$y_t \omega_t^l + \sum_{i \in N_t^l} \Big(\sum_{p \in W_{ti}^U} x_p u_i + \sum_{p \in W_{ti}^O} x_p o_i \Big) \leq \Re \qquad \forall t \in T,\ l \in L_t,\ i \in N_t^l \tag{4–14}$$

$$r_t \geq \sum_{\beta_{(i,j)}^t} x_p w_p H_{(i,j)} / H_{std} \qquad \forall t \in T,\ (i,j) \in A_t \tag{4–15}$$

$$\partial y_t \leq r_t \leq \wp y_t \qquad \forall t \in T \tag{4–16}$$

$$r_t \leq z_t \leq F y_t \qquad \forall t \in T \tag{4–17}$$

$$\tilde{v}_i \geq \sum_{t \in T_i^+} z_t - \sum_{t \in T_i^-} z_t \qquad \forall i \in N \tag{4–18}$$

$$\tilde{v}_i \geq \sum_{t \in T_i^-} z_t - \sum_{t \in T_i^+} z_t \qquad \forall i \in N \tag{4–19}$$

$$x_p \in \{0,1\} \qquad \forall p \in \beta \tag{4–20}$$

$$y_t \in \{0,1\} \qquad \forall t \in T \tag{4–21}$$

$$w_{ti} \in \{0,1\} \qquad \forall t \in T,\ i \in N_t \tag{4–22}$$

$$v_l,\ v_i \geq 0 \qquad \forall l \in L,\ i \in N \tag{4–23}$$

Constraints (4–2) ensure that exactly one block path is selected for each block. Constraints (4–3) restrict the sum of block volumes assigned to a train on an arc. Constraints (4–4) form the relation between flow and design variables. It requires that a train path is selected if a block path flows on it and that block path is selected in

the solution. Constraints (4–5) and (4–6) enforce the originating and terminating train capacities at nodes. Constraints (4–7) limit the number of block swaps occurring at each node. Constraint (4–8) restricts the total number of trains that are passing through each arc. Constraints (4–9), (4–10) and (4–11) are used to limit the number of trains that can stop at a node (work event capacity). Constraints (4–12) and (4–13) are used to calculate the difference of trains running in opposite directions on a crew district. Constraints (4–14) limit the maximum crew work time (travel time plus stop time) in a crew district. Constraints (4–15) and (4–16) determine the active locomotives required to pull each train. Constraints (4–17) are used to calculate actual number of locomotives (active and deadheading) attached to a train. Constraints (4–18) and (4–19) determine the imbalance of locomotives at each node. Objective function (4–1) minimizes the costs related to trains (train starts, train miles, train work events), cars (block swaps, car movement, car hours), locomotives (light travel, active pulling, deadheading, locomotive ownership) and crews (active crew cost, crew imbalance).

4.3.2 Iterative Train Construction Approach

Iterative approach builds trains one by one until for routing all unassigned blocks. The pseudo code of this sequential method is as follows:

- Enumerate potential train paths
- Construct trains iteratively as follows:
  - Select a candidate block for which a train is built. The candidate block is selected based on daily volume and number of potential trains. A scaling value is used to determine minimum block volume. As model iterates, the scaling factor is reduced by the factor of two.
  - For the candidate block, identify a set of potential trains which can carry this block.
  - Assign blocks to each potential train using a greedy heuristic.
    - For each potential train, assign candidate block and other blocks going from the origin to the destination of the train.

    - Compute cost impact: In an iterative loop, assess the objective function impact of adding a block to the train and check the feasibility of adding the block. Add the best feasible block to the potential train and iterate until no more blocks can be added.

  - Construct one train to be added in the solution: Among the potential trains with assigned blocks, select the one with best objective function value if it is feasible from

    - crew route and train load perspective
    - nodes and links capacity perspective

  - Update unassigned blocks, list of potential trains and all the other statistics. Repeat same steps, by reducing the minimum block volume which can be candidate in new train construction by half until all blocks are routed.

4.4 Computational Experience

We try our solution method on a real instance of a major North American Railroad. In this instance, 1200 blocks are to be routed from their origin to their destination. We compare results we get using our integrated model with the solution implemented by the railroad and also with the results we get using the iterative approach.

In Table 4-1, we present percent improvements of iterative approach and proposed holistic approach over railroad solution. Overall cost improvements for the holistic and iterative approaches are 15.84% and 4.46% successively. Except for the number of block swaps, holistic approach consistently has better improvements for all cost factors compared to the iterative approach. Increase in number of block swaps is acceptable since total number of trains and average length of the trains decreases in the holistic approach. This leads to more block swaps and better utilization of train capacities. Average number of cars on a train arc is 82.84 for the holistic approach. For the railroad solution and iterative approach, average number of cars in a train arc is 72.01 and 75.16 consecutively. Computer runtimes of the iterative approach and proposed model is given in Table 4-2.

4.5 Summary and Conclusions

We adapted our generic non-bifurcated model for the solution of train routing and block-to-train assignment problems. Using the generic non-bifurcated model which

Table 4-1. Percentage improvements of proposed model and iterative approach over the railroad solution

	Iterative Approach	**Holistic Approach**
Train Cost Savings		
Train Starts	1.90	9.48
Train Miles	-0.83	12.59
Train Work Events	0.49	7.54
Car Cost Savings		
Total Car-Hours	1.44	4.62
Car Movement Cost	0.23	0.28
Total Block Swaps	-4.82	-13.25
Locomotive Cost Savings		
Total locomotive requirement	13.60	23.84
Ownership Cost	2.68	10.01
Active Pulling Cost	1.10	3.94
Deadheading Cost	2.16	10.32
Light Travel Cost	9.46	21.43
Crew Cost Savings		
Total Crew Requirements	-0.61	13.11
Active Crew Cost	1.23	10.76
Crew Imbalance Cost	31.67	64.76
Total Operating Plan Cost	4.46	15.84

Table 4-2. Comparison of runtimes for proposed model and iterative approach

	Iterative Approach	**Holistic Approach**
Computational time (secs)	302.35	395.56

utilizes binary path based design and flow variables has led us incorporate many asset related business constraints. Note that this formulation is also suitable for applying an incremental approach in case some train paths or block paths are favorable for the railroad. We applied our solution approach on a real life instance of a major North American railroad company and obtained a good quality solution which improves the solution of the railroad and iterative approach in a few minutes.

CHAPTER 5
A DECOMPOSITION APPROACH FOR SERVICE NETWORK DESIGN PROBLEMS ON HUB-AND-SPOKE NETWORKS

5.1 Introduction

Traditional network design problems are taken into a new level with the addition of time dimension into the design. Carriers have to provide strict delivery times (usually called as service commitments) for their services they announce between different origin-destination terminal pairs over their network. Designing a reliable and cost efficient service network is critical in the operation of all service providers. A time and cost efficient plan consists of an optimal load schedule and shipment routes that bring all shipments to their destinations on time at minimal operating cost. A load corresponds to a capacitated facility installed on a link if we consider capacitated multicommodity network design problems. In service network design context, a load can be better defined as a consolidation of different shipments traveling together on a link from one terminal to another.

Crainic (2000) reviews network design models, relevant solution approaches and service network design formulations in freight transportation. In a more recent paper Wieberneit (2008) reviews different formulations and solution frameworks for service network design problems. Author presents five different practical service network design problems in literature arising in express shipment delivery, letter mail delivery on flight networks, and less-than-truckload operations in Europe and North America. Service network design reviews which are specialized on a specific transportation mode, long-haul or intermodal operations include Christiansen et al. (2007) for maritime transportation, Assad (1980) and Cordeau et al. (1998) for rail transportation, Crainic (2003) for long-haul transportation, and Crainic & Kim (2007) for intermodal transportation.

Formulations for service network design problems can be classified as static (frequency) or time-dependent (dynamic). This classification is mostly due to Crainic (2000) and Crainic & Kim (2007). In static models, time dimension is implicitly

considered and services and their frequencies are determined. It is assumed that demand does not vary during the planning horizon. Some examples of static formulations have been presented by Crainic & Rousseau (1986) and Crainic & Roy (1988) for multimodal transportation, Powell & Sheffi (1983), Powell (1986) and Powell & Sheffi (1989) for LTL motor carriers, Keaton (1992) and Newton et al. (1998) for rail transportation, Barnhart & Schneur (1996), Armacost et al. (2002), Kim et al. (1999) for express shipment services, Christiansen et al. (2004) for maritime transportation. In time-dependent models, movement of resources is represented in time and decisions involving services and their detailed schedules are made. Since time dimension is explicitly considered, formulations are based on time-space networks and resultant networks and formulations are larger in size. Time-dependent formulation examples are studied by Andersen et al. (2009a), Andersen et al. (2009b), and Pedersen et al. (2009) for rail intermodal operations, Haghani (1989) and Gorman (1998) for rail transportation, Smilowitz et al. (2003) for express shipment services, Farvolden & Powell (1994) for LTL carrier services.

Andersen et al. (2009a) integrate asset management considerations into service network design models for consolidation-based freight carriers and compare four different formulations for service network design with asset management. The four formulations combine arc and cycle design variables with arc and path flow variables. Authors generate 21 instances and compare the performance of formulations using these instances. They also test impact of asset management constraints on the solution quality and solution time by gradually removing asset related constraints from the original arc-arc formulation of service network design with asset management. Andersen et al. (2009b) present a new model for service network design with asset management and multiple fleet coordination. The model addresses intermodal transportation operations and aims to determine service departure times such that demand throughput time and fixed cost of operating the fleets are minimized. Authors test the model on an actual rail intermodal application. Other

recent contributions incorporating asset management into service network design include Andersen & Christiansen (2009), Pedersen et al. (2009), and Teypaz et al. (2010).

Lium et al. (2009) and Hoff et al. (2009) address service network design problems with stochastic demand. Lium et al. (2009) study the importance of introducing stochasticity in service network design problems using a small size, basic problem. Hoff et al. (2009) proposes a model inspired by Lium et al. (2009) and use a metaheuristic method to solve large instances of service network design problem with stochastic demand.

Hub-and-spoke networks are frequently utilized to solve consolidation problems, where, instead of sending each shipment directly to its destination, shipments are combined into loads and routed through hubs. Hub-and-spoke based networks are composed of end of line terminals and break-bulks (hub locations). Break-bulk locations are used to consolidate demand coming from different destinations. Jarrah et al. (2009) solves a real-life, large-scale instance of the service network design problem in the context of the LTL industry on a hub-and-spoke system. They present a novel network design model and decompose this massive model into a set of efficient integer programming models for each destination terminal along with a coordinating master network design problem. In each subproblem they generate a load planning tree which defines the feasible freight flow to a destination terminal from all other terminals in the network. Load planning trees is introduced as decision variables in the master network design formulation. Linear programming (LP) relaxation of the master network design model is solved using a slope-scaling heuristic which was first introduced by Kim & Pardalos (1999). LP relaxations in each iteration of the slope-scaling heuristic are solved using column generation with each column corresponding to a feasible load-planning tree. Authors use a modified slope scaling heuristic that uses a gradual costing strategy to slow down the convergence of the heuristic. They generate major potential cost savings for the target LTL carrier in about two hours for each run.

We study a service network design (SND) problem on a hub-and-spoke network and propose network shrinking based decomposition scheme for its solution. Network shrinking idea allows the generation of smaller time-space networks that can be solved within reasonable time limits. The approach can be utilized by all consolidation based carriers, such as express package delivery, less-than-truckload service providers, freight rail carriers, etc. In the first phase of this algorithm we solve an integer programming model by using slope scaling heuristic and propose improvement steps which significantly decreased the costs of the solution obtained by slope-scaling heuristic. The second phase solves a model on a time-space network which is composed of only hub locations. The third phase improves the non-hub to hub and hub to non-hub connections of the first phase. We applied our algorithm on a real life instance for a less-than-truckload motor carrier and obtained considerable improvements in transportation costs and load capacity utilizations in a reasonable time limit.

5.2 Problem Description

The goal of the problem is to install loads on the links of the given the network and decide on routes for each shipment such that total transportation costs (total mileage costs) are minimized. Each route is a sequence of loads a shipment should take to travel from its origin to destination. We assume that shipments' demand quantities are not big enough to fill load capacities. Once a shipment is loaded from its origin, it travels hub locations and finally reaches its destination terminal. Note that usually two hubs are visited on the route of the shipment. However, we allow more than two hubs to be visited. We assume that intermediate handling (sorting and regrouping) of shipments can take place only at hubs. At non-hub locations, only pick-up and set-off are allowed. In order to make better consolidation at hubs, shipments are allowed to be hold at hubs. Holding is useful to aggregate different shipments into a single load and utilize loads' capacities better. Holding of shipments is valid as long as shipments satisfy their corresponding service commitments. All the shipments follow a single path from their origin to their

destination. Hence, we solve a non-bifurcated service network design problem. Inputs, constraints and outputs of our service network design problem are given as follows:

5.2.1 Problem Inputs

- **Locations:** There are multiple locations either shipments originate or terminate. There are two types of locations hubs and non-hubs (also called as end-of-line terminals). We are given set of non-hub locations and set of hub locations.
- **Links with their distance and travel time:** Traveling time is obtained by using average speed of transportation medium. We assume that almost all locations are connected to each other directly. This is especially true for road networks.
- **Service:** Each service is defined by an origin, destination location pair.
- **Service commitments:** For each service and for each specific day of the week, carrier provides corresponding service commitments. Each service commitment is defined by its corresponding service origin location, service destination location, cutoff day, cutoff time, recovery day, recovery time and planned weight. Planned weights are usually constructed by the carrier looking at their historical demand data between origin-destination terminal pairs. For example, if a shipment is ready at location A (origin location) on Monday (cutoff day) by 9 pm (cutoff time) than the costumer can pick it up from location B (destination location) at 8 am (recovery time) on Tuesday (recovery day). Services and corresponding service commitments are provided from almost all locations to all other locations.
- **Load capacity:** Capacity of transportation medium.
- **Transportation cost per mile**

5.2.2 Problem Constraints

- **Service commitment time limits:** Durations of shipment routes should satisfy their corresponding service commitment time limit, i.e., all shipments must arrive to their destinations before their recovery day and time.
- **Load capacity limit:** Total shipment weight traveling on a load should not exceed its capacity.
- **Shipment Flow rule:** Flows dictate next via location from a particular location on a particular day. At each location on a given day, all shipments with the same actual destination should be routed to the same next via location.

5.2.3 Problem Outputs

- **Shipment Routing Plan:** For each possible destination from an origin, we find the route to follow. Note that shipment routes should obey the flow rules.
- **Loads and Load schedule:** Identify loads to be used and determine departure time and arrival time of the load. There can be multiple loads departing from a location on the same day.

The objective of the problem is to create a load plan and shipment routing plan which minimizes cost. We assume that major cost is transportation cost per mile. Handling cost at the terminals is relatively small compared to the cost of moving packages between terminals. Main goal is reducing the mileage while maintaining the service.

5.3 Decomposition-Based Solution Approach

We decompose the problem into three parts. In the first phase, we determine flows of shipments using a space network composed of all locations. This phase performs facility consolidation by deciding on how inbound shipments are sorted and consolidated for their next via locations. In the second phase, we determine loads and load departure times on a shrunk space-time network which is formed using only hub locations. This phase performs a temporal consolidation by holding shipments over time in order to form larger shipments. Third phase is used for pick-up and delivery routes. We improve direct non-hub to hub and hub to non-hub connections by performing multi-stop consolidation. In this phase several non-hub locations assigned to a particular hub are served together on a single route.

5.3.1 Phase 1: Facility Consolidation

In the first phase, using the overall network and shipment weights, we are trying to identify shipment routes that satisfy flow rules. While routing all shipments, the model also takes into account load capacities. Volume of shipment routes passing through a specific link should not exceed the total capacity of loads installed on the link and route travel times should satisfy commitment time limits. We solve phase 1 problem on a space network.

Let N denote the set of nodes in the network. Each node can be origin, destination or hub location. A denote the set of arcs in the network. c_{ij} is the cost incurred by each load used on arc (i, j). u_{ij} is the capacity of each potential load on arc (i, j). Decision variable y_{ij} equals 0 if no load is made on arc (i, j), otherwise y_{ij} denotes the number of loads made on arc (i, j). Planned service weights need to be routed for each service ($S = \{(u, v) : u \in N, v \in N\}$, we index each service by s). Planned service weights (w_s, $s \in S$) equal to the total planned weight for the corresponding service during the whole planning week. Note that some portion of this planned weekly weight can be realized demand which is preordered before the week starts and some portion can be due to the estimate of the carrier using historical data. α_s denotes the maximum number of hours the service from node u to node v can take. We enumerate directed shipment routes for each service. The set P denotes the set of all potential shipment routes and the set $P^s \subset P$ denote the set of potential shipment routes for service $s \in S$. We use index p to denote a shipment route. We set w_p equal to w_s for all $p \in P^s$ and define a binary decision variable x_p for each potential shipment route $p \in P$. x_p equals 1 when service s uses shipment route $p \in P^s$, and it is zero otherwise. To be able to set up the model, for each arc $(i, j) \in A$ we determine all the potential shipment routes that contain the arc and denote this set by P_{ij}. We also prepare the sets Q_{ijks} and W_{ik} considering the set of potential shipment routes enumerated. These two sets are used to define shipment flow rule constraints. Q_{ijks} is the set of all shipment routes for service s with destination k which flow on arc (i, j). W_{ik} is the set of all next via locations visited after leaving node i by the potential shipment routes having destination k. Variable z_{ijk} takes the value of 1 if at least one service with destination k uses arc (i, j). We add a z_{ijk} variable to the model for nodes $\{i, j, k\}$ only if corresponding set W_{ik} has more than 1 element.

$$\min \sum_{(i,j)\in A} c_{ij} y_{ij} \tag{5–1}$$

subject to

$$\sum_{p \in P^s} x_p = 1 \qquad \forall s \in S \tag{5–2}$$

$$\sum_{p \in P_{ij}} w_p x_p \leq u_{ij} y_{ij} \qquad \forall (i,j) \in A \tag{5–3}$$

$$z_{ijk} \geq \sum_{p \in Q_{ijks}} x_p \qquad \forall i,\ k,\ j,\ s : |Q_{ijks}| > 0 \tag{5–4}$$

$$\sum_{j \in W_{ik}} z_{ijk} \leq 1 \qquad \forall i,\ k : |W_{ik}| > 1 \tag{5–5}$$

$$x_p \in \{0, 1\} \qquad \forall p \in P \tag{5–6}$$

$$y_{ij} \in Z^+ \qquad \forall (i,j) \in A \tag{5–7}$$

$$0 \leq z_{ijk} \leq 1 \qquad \forall i,\ j,\ k \in N \tag{5–8}$$

Constraints (5–2) state that exactly one shipment route is selected for each service. Capacity constraints (5–3) require that maximum weight flowing on an arc does not exceed the capacity installed on the arc. Constraints (5–4) and (5–5) ensure that all shipments with the same destination are routed to the same next via location after passing through a given location.

In order to solve this problem, we adopt slope scaling heuristic technique due to Kim & Pardalos (1999) and improved its solution by postprocessing steps. In this iterative solution approach, we solve the LP relaxation of the model, and we update fixed charge costs for links and then solve the LP relaxation again until no improvement is found for a certain number of iterations. Fixed charge costs for each link is updated considering the load capacity utilization factors, which is found by dividing fractional number of loads from the LP relaxation solution with the rounded number of loads on this link. Hence, load capacity utilization factor on arc (i,j) is calculated by $\varphi_{ij} = y_{ij}/\lceil y_{ij} \rceil$. Fixed charge cost of a link is updated by dividing with load capacity utilization factor of the link. Hence, if load capacity utilization factor is low, fixed charge cost of the link increases, this in turn makes it more probable for the corresponding y_{ij} value to get closer to its rounded

down integer value. In order to prevent slope scaling heuristic getting stuck in local optima, we apply gradual costing of the links by updating fixed cost of only a specified number of links.

Improvement approach for the slope-scaling solution. In order to improve the solution obtained by slope scaling heuristic, we apply following steps. Let y_{int} be the integer solution we get after rounding up the continuous y variables at the best iteration of the slope scaling heuristic.

Step 1: We solve an integer programming model (IP-1) where cost coefficients in the objective function are the updated fixed costs at the best iteration of the heuristic. We give y_{int} as the starting solution to IP-1 model. This model is solved very quickly since we use the updated costs at the best iteration of the heuristic; it takes about 2 minutes to solve it to optimality. Solution of IP-1 improves the solution of the slope scaling heuristic around 17%.

Step 2: We get the integer solution from step 1. We calculate total weight flowing on each arc by checking the selected shipment routes and then find actual load utilization of each arc's capacity by dividing total flow on the arc by capacity of a load. We solve a new integer programming model (IP-2) where we replace y variables with integer part of fractional load utilization factor plus a binary variable. Note that load utilization can be viewed as the fractional number of loads required on each arc. In order to find integer part, we round down fractional load utilization of each arc; i.e., if load utilization is 5.6 for an arc, then in IP-2 model, we replace y with 5 plus a binary variable. In IP-2, we use actual fixed costs for each arc and give the solution in step 1 as the starting solution for the new IP model. Solution of IP-2 improves the solution of IP-1 around 7%.

5.3.2 Phase 2: Temporal Consolidation

In the second phase, time component is added to the space network, which helps to determine loads and their arrival and departure times. For each service and each day of week, planned service commitment weights are routed over the space-time network.

The solution of the phase 1 is used as an input and shipment holding structure and resulting load timings are found using a space-time network. We utilize the outputs of first phase in the second phase. The arcs selected and flows found for each service are used as an input. Since flow rules are given, enumeration of possible paths will be restricted and less time consuming. For the same service departing on different days of the week, sequence of locations of the selected routes is going to be the same but shipment holding structure may differ.

5.3.2.1 Network shrinking approach

Since flows are already given, we expect to solve an easier integer programming model in the second phase however, size of the network is considerably larger compared to the space network. To be able to solve the problem model on the space-time network, we use network shrinking idea. Shrunk network is formed by using only hub locations. In the shrunk network, all hub locations are fully connected. Shrunk network results into a space time model which has considerably smaller size compared to the original network.

Following approach is used to form hub to hub shrunk network: We use the location sequences formed for all services in the first phase. Most of the location sequences have a similar structure of starting with a non-hub location and traveling several hub locations and terminating at a non-hub destination location. For instance, one possible shipment route might traverse following locations: non-hub origin, hub 1, hub 2, non-hub destination. We shrink this potential shipment route so that it starts at hub 1 and ends with hub 2. We then set service commitment earliest departure time (cut-off time) from hub 1 as the earliest time shipment can reach hub 1 location from its non-hub origin location. Similarly, we determine latest arrival time (recovery time) to hub 2 location. Following this routine, we modify cut-off and recovery times of all service commitments such that they all go from hub to hub locations. We then create dated arcs between hubs. For all arcs suggested by the space network, we determine the ones whose tail and head

nodes are hub nodes and create twelve dated arcs for each day of week. Departure times of dated arcs are distributed in equal intervals.

For 7-day shrunk space time network, we slightly change the notation to represent dated components. $\hat{N}$ denotes the set of nodes in the network. Each node represents a specific location on a specific day of week. In the second phase, we use the arcs selected in the first phase, however, we replace each arc by a set of dated arcs between same pair of tail and head nodes with different departure times. The set of dated arcs is shown by $\hat{A}$. Each dated arc is defined by an arc ID and a specific departure time; hence, a dated arc represents a single load departing at a certain time. Planned freights for each service commitment need to be routed over the space-time network. We denote the set of service commitments by C.

Path enumeration for Space-Time Network. For each service commitment $c \in C$, we enumerate set of potential shipment routes (P^c). The path enumeration is exactly the same as for space network with following minor changes: Only those paths are considered which have the same location sequence as suggested by space network solution. At any location shipment is allowed to stay to depart on upcoming days if the service commitment recovery time permits. The cost of a path is considered to be the time taken to reach the destination.

5.3.2.2 Space time model for the shrunk network

Note that we are using dated arcs in the space time network and each dated arc represents a single load with a specific departure time. Dated arcs resemble the facility-arcs introduced in the chapter 2. That is why, we utilize the non-bifurcated network design model proposed to solve combined network design and commodity routing problem. Hence, while defining capacity constraints, we don't have to use fixed-charge constraints. This results into the following model:

$$\min \sum_{(i,j)\in\hat{A}} c_{ij} y_{ij} \tag{5–9}$$

subject to

$$\sum_{p \in P^c} x_p = 1 \qquad \forall c \in C \tag{5–10}$$

$$\sum_{p \in P_{ij}} w_p x_p \leq u_{ij} \qquad \forall (i,j) \in \hat{A} \tag{5–11}$$

$$y_{ij} \geq \sum_{p \in P^c_{ij}} x_p \qquad \forall (i,j) \in \hat{A},\ c \in C \tag{5–12}$$

$$x_p \in \{0,1\} \qquad \forall p \in P \tag{5–13}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in \hat{A} \tag{5–14}$$

Objective of the space-time model is the same as the space network model. Constraints (5–10) are very similar to the one used in the space network model. In the space model, we were selecting exactly one shipment route for each service. In the second phase, exactly one shipment route is selected for each service and for each day of the week service is provided (hence for each service commitment). Constraints (5–11) are different, in the space network; we were determining number of loads to be used on each arc. For space-time network, between two locations we construct several dated arcs and each of them represents a load-arc defining an arc id and a load departing at a specified time. To be able to set up the model, we again form the set P_{ij} where (i,j) is a dated arc; that is, we keep the set of potential routes flowing on dated arc (i,j). Note that w_p denotes the planned service commitment weight for the corresponding service and day. We set w_p equal to w_c for all $p \in P^c$. Second constraint ensures that total weight of shipments assigned to a dated arc does not exceed load capacity. Constraints (5–12) are used to set up the relation between design variables and flow variables. They ensure that dated arc (i,j) is selected if at least one potential shipment route flowing on this arc is selected. Note that a dated arc is simply a load with given departure and arrival times on its tail and head nodes respectively. While defining constraint (5–12), we again used the tightened

version of the constraint by forming the set P_{ij}^{c} which include the set of potential routes for service commitment c flowing on dated arc (i, j).

5.3.3 Phase 3: Multi-stop Consolidation

After determining hub to hub flows, non-hub to hub and hub to non-hub assignments coming from the first phase is improved by connecting non hub locations between themselves. In this way, we can decrease transportation costs, make better consolidation and use less number of vehicles to pickup loads from non-hub locations and bring them to hub locations, similarly we use less vehicles to deliver shipments from a hub location to the non-hub locations it is connected.

As a result of phase 1, we know which non-hub locations are connected to a given hub and from phase 2, we also know the load departure times of each service commitment from hub locations. Objective in the third phase is to construct loads and their departure times such that several non-hub locations can be visited consecutively before arriving to the first hub location or after departing from the last hub location on the shipment route. In this phase, we allow pick-ups from other non-hub locations for non-hub to hub portion of the route. Similarly, set-off is permitted at other non-hub locations for hub to non-hub portion of the route. This results into better consolidation and decreased overall mileage.

Solution Approach for Phase 3. We solve third phase for each hub location separately. Each hub location along with all its associated non-hub locations forms a sub-problem. Non-hub to hub and hub to non-hub connections are improved in separate steps. First, we solve all sub-problems to improve non-hub to hub connections of associated shipment routes. Secondly, we solve sub-problems for improving hub to non-hub connections. We will first describe the process for non-hub to hub connections. Same approach is applied for improving hub to non-hub connections.

For the non-hub to hub connections, in a sub-problem for a given hub, we consider set of service commitments whose shipment routes pass through the hub location after leaving their non-hub origin locations. For each sub-problem, we form a space-time

network involving a single hub location and several non-hub locations. As in phase 2, we construct multiple dated arcs connecting same pair of nodes. For each non-hub location, we enumerate directed paths originating at non-hub locations and terminating at the hub location. Note that there is only one hub in each sub-problem. Each directed path represents a sequence of loads and describes a pick-up sequence. Among the set of directed paths, we always have the path going from non-hub origin to the hub location directly. There are also paths visiting multiple non-hub intermediate locations before arriving to the hub location.

For each sub-problem, we use the following notation to construct the integer programming model for non-hub to hub connections:

$\tilde{C}$: Set of service commitments originating from non-hub locations and going through same hub location.

a_c : Available time of the service commitment $c \in \tilde{C}$ at its origin location.

l_c : Latest time by which shipment route for service commitment c should reach the hub location. Notice that we know the departure time of the route for this service commitment from the hub location using second phase.

P : Set of directed paths enumerated

R^c : Set of flow paths for a service commitment c which can feasibly carry planned weight for the commitment to its hub location. Flow paths are subpaths of directed paths. We include a flow path to set R^c if the corresponding whole directed path picks up planned weight for service commitment c after time a_c and can carry it to hub location before time l_c.

R_p : Set of flow paths flowing on path p.

Q_{ijkc} : Set of all shipment routes for service commitment c with destination k which flow on arc (i, j). Notice that we use dated nodes to form this set. Hence, node i represent a specific location and a departure day.

W_{ik} : Set of all next via locations visited after leaving node i by the potential shipment routes having destination k.

Let x_r be a binary variable taking value of 1 if planned weight for service commitment c is assigned to partial directed path $r \in R^c$ and let y_p be a binary variable taking value of 1 if a directed path is selected. Then, the problem can be formulated as follows:

$$\min \sum_{p \in P} c_p y_p \tag{5–15}$$

subject to

$$\sum_{r \in R^c} x_r = 1 \qquad \forall c \in C \tag{5–16}$$

$$\sum_{r \in R_p} w_r x_r \leq u_p \qquad \forall p \in P \tag{5–17}$$

$$y_p \geq x_r \qquad \forall p \in P,\ r \in R_p \tag{5–18}$$

$$z_{ijk} \geq \sum_{r \in Q_{ijkc}} x_r \qquad \forall i,\ k,\ j,\ c : |Q_{ijkc}| > 0 \tag{5–19}$$

$$\sum_{j \in W_{ik}} z_{ijk} \leq 1 \qquad \forall i,\ k : |W_{ik}| > 1 \tag{5–20}$$

$$x_r \in \{0, 1\} \qquad \forall r \in R \tag{5–21}$$

$$y_p \in \{0, 1\} \qquad \forall p \in P \tag{5–22}$$

$$0 \leq z_{ijk} \leq 1 \qquad \forall i,\ j,\ k \in N \tag{5–23}$$

Constraints (5–16) state that that for each service commitment; exactly one partial directed path is selected. Constraints (5–17) ensure that total weight assigned to a directed path cannot exceed load capacity. Constraint (5–18) sets the relation between design paths and flow paths. If a flow path (a subpath of corresponding directed path) is selected for a service commitment then the corresponding directed path (design path) should be chosen by the model. Since we try to change non-hub to hub and hub to non-hub parts of shipment routes in the third phase, we need to check if shipment flow

rules are satisfied. Hence, constraints (5–19) and (5–20) are included in the model as in the first phase. Notice that there are no fixed charge constraints since we formulated the problem by using the proposed non-bifurcated network design model.

5.4 Computational Experience

The proposed decomposition approach for solving service network design problem on hub-and-spoke networks is needed by all consolidation based carriers, such as express package delivery, less-than-truckload service providers, freight rail carriers, etc. We applied our approach to solve a real life instance of a less-than-truckload carrier.

The case for a Less-than-Truckload Motor Carrier. A North American less-than-truckload carrier maintains around 60 terminals (end-of-line terminals and hubs). The company has time sensitive shipments with strict delivery dates. Shipments are handled at 10 hubs. In these locations arriving shipments are sorted. Then some packages are delivered to customers and others are again regrouped to send to other hubs. The handling cost of the company at the terminals is relatively small compared to the cost of moving packages between terminals; therefore, the main goal is to reduce the mileage cost while maintaining the service. Carrier pays to drivers per mile. If a load distance is less than a certain limit, they pay for the rate of single driver else they pay for the rate of team driver for this load. Service commitments are set by the carrier as an input considering their historical data. In each week, the carrier has around 21000 service commitments. A provided solution should give load schedule over the network and shipment routing plan for each service defined between origin destination terminals. The carrier requires an algorithm that can improve their current solution (incremental algorithm). They are also looking for a tool which has the ability to find a better solution from scratch (zero-based algorithm) compared to their current operating plan cost. Developed algorithm should select best routes to handle shipments and decide on using certain set of hubs among given candidate set. The company does not have sophisticated

planning system. Managers route packages based on the historical data and their previous experience.

In order to obtain an incremental solution, we get carrier's shipment routes for each service commitment and use set of locations visited by each shipment route as an input. Hence, phase 1 of our algorithm is not needed to be used in this case. We use phase 2 and change only the holding structure of shipments and keep the set of locations visited by each shipment route same as the given input. In phase 3, we have the chance of improving a given solution by visiting a set of non-hub locations instead of using direct non-hub to hub or direct hub to non-hub connections. In our experiments, we observe that both phase 2 and 3 improved the solutions for hub to hub and non-hub to hub (or vice versa) connections significantly. In Table 5-1, we present results we obtained using our zero-based algorithm and incremental approach and compare them with currently implemented solution by the carrier. The column "Proposed-1" shows the results when we use carrier's solution as a starting solution (incremental approach), the column titled "Proposed-2" presents the results when we find a solution from scratch (zero-based solution). "LTL Carrier" column represents the solution implemented by the carrier. We observe that the algorithm presented justify itself by improving a given starting solution by the carrier, also by forming a solution from scratch which results into best solution among these three solutions presented.

In Table 5-2, we present optimality gaps (%) and total computer running time in seconds for each phase of our proposed decompositon-based algorithm. Notice that in phase 3 an integer programming model is solved for each hub separately. For this phase, optimality gap reported is the average of optimality gaps obtained over all hubs. We observe that we can solve optimization models for most of the hubs (90%) to optimality in a matter of seconds. For a small portion of the hubs where shipment traffic is very high, optimization model tries to close the gap till the run time limit of 15 minutes. Hence, total solution time reported is 2254.06 seconds. In phase 1, slope scaling heuristic and improvement

Table 5-1. Comparison of incremental and zero-based solutions obtained by proposed algorithm with LTL carrier's solution

	Proposed-1	**Proposed-2**	**LTL Carrier**
Number of Loads			
Non-Hub to Hub	301	310	323
Hub to Non-Hub	385	410	462
Hub to Hub	600	556	710
Total	1,286	1,276	1,495
Cost			
Non-Hub to Hub	$ 91,574.96	$ 84,148.08	$ 94,242.14
Hub to Non-Hub	$ 102,477.84	$ 109,573.98	$ 118,466.58
Hub to Hub	$ 395,739.98	$ 367,977.98	$ 448,626.04
Total	$ 589,792.78	$ 561,700.04	$ 661,334.76
Avg. Load Utilization			
Non-Hub to Hub	50.90%	46.37%	48.79%
Hub to Non-Hub	41.42%	35.47%	36.06%
Hub to Hub	52.70%	54.27%	44.63%
Total	48.34%	45.37%	43.16%

steps run in less than thirty minutes. In order to report the optimality gap of the resulting solution, we input the resulting solution as a starting solution to the exact optimization model and report the gap after three minutes. For phase 2, we use a running time limit of 1 hour for the space time model. The optimality gap reported is the gap we obtained after the running time limit is exceeded.

Table 5-2. Optimality gap and total running times for each phase of proposed algorithm

	Optimality Gap	**Running Time (secs)**
Phase 1	5.97	1684.58
Phase 2	3.88	3600.00
Phase 3	0.79	2254.06

5.5 Summary and Conclusions

We proposed a decomposition-based approach for the solution of a service network design problem on a hub-and-spoke network. We applied our approach on a real life instance for a less-than-truckload carrier and obtained a good quality solution in a practical running time. Proposed decomposition includes an innovative network shrinking approach that separates the network into three partitions: non-hub to hub, hub to hub

and hub to non-hub network. Network shrinking results into smaller space time networks and corresponding space time models can be solved satisfactorily within running time limits. In the first phase of our algorithm, we adapt slope scaling heuristic to solve corresponding integer programming model on a space network and propose improvement steps for the resulting solution. We observe that improvement steps significantly decreased the costs. Heuristic solution of phase 1 is improved approximately by 23%. In phase 3, we improved the solutions for direct non-hub to hub and hub to non-hub connections. This is achieved by letting several intermediate non-hub locations to be visited consecutively before arriving the first hub location or after departing from the last hub location on the shipment route. We could decrease transportation costs by 8.64% on the average for non-hub to hub and hub to non-hub connections. Similarly, we could make better consolidation over the service network by increasing load capacity utilization by 3.15% on the average for non-hub to hub and hub to non-hub connections.

REFERENCES

Ahuja, R., Cunha, C., & Sahin, G. (2005). Network models in railroad planning and scheduling. In C. Smith (Ed.) *Tutorials in Operations Research*, chap. 3, (pp. 54–101). INFORMS.

Andersen, J., & Christiansen, M. (2009). Designing new European rail freight services. *Journal of the Operational Research Society*, *60*(3), 348–360.

Andersen, J., Crainic, T. G., & Christiansen, M. (2009a). Service network design with asset management: Formulations and comparative analyses. *Transportation Research Part C: Emerging Technologies*, *17*(2), 197 – 207.

Andersen, J., Crainic, T. G., & Christiansen, M. (2009b). Service network design with management and coordination of multiple fleets. *European Journal of Operational Research*, *193*(2), 377 – 389.

Armacost, A. P., Barnhart, C., & Ware, K. A. (2002). Composite variable formulations for express shipment service network design. *Transportation Science*, *36*(1), 1–20.

Assad, A. A. (1980). Models for rail transportation. *Transportation Research Part A: General*, *14*(3), 205 – 220.

Association of American Railroads (2008a). Overview of America's Freight Railroads. Available at `http://www.aar.org/InCongress/BackgroundPapers.aspx` as of September 2008.

Association of American Railroads (2008b). Rail industry structure and market behavior. `www.aar.org/IndustryInformation/~/media/AAR/BackgroundPapers/145.ashx` Accessed in September 2008.

Balakrishnan, A., Magnanti, T., & Mirchandani, P. (1997). Network design. In M. Dell'Amico, F. Maffioli, & S. Martello (Eds.) *Annotated Bibliographies in Combinatorial Optimization*, (pp. 311–334). John Wiley & Sons, New York.

Balakrishnan, A., Magnanti, T. L., & Wong, R. T. (1989). A Dual-Ascent Procedure for Large-Scale Uncapacitated Network Design. *Operations Research*, *37*(5), 716–740.

Barnhart, C., Krishnan, N., Kim, D., & Ware, K. (2002). Network design for express shipment delivery. *Comput. Optim. Appl.*, *21*(3), 239–262.

Barnhart, C., & Schneur, R. R. (1996). Air Network Design for Express Shipment Service. *Operations Research*, *44*(6), 852–863.

Bartolini, E., & Mingozzi, A. (2009). Algorithms for the non-bifurcated network design problem. *Journal of Heuristics*, *15*(3), 259–281.

Berger, D., Gendron, B., Potvin, J.-Y., Raghavan, S., & Soriano, P. (2000). Tabu search for a network loading problem with multiple facilities. *Journal of Heuristics*, *6*(2), 253–267.

Bog, S., Nemani, A., & Ahuja, R. (2010). Iterative algorithms for the curfew planning problem. *Journal of the Operational Research Society*. Advance online publication; doi: 10.1057/jors.2010.1.

Budai, G., & Dekker, R. (2004). A dynamic approach for planning preventive railway maintenance activities. In J. Allan, C. Brebbia, R. Hill, G. Sciutto, & S. Sone (Eds.) *Computers in Railways IX*, (pp. 323–332). WIT Press.

Budai, G., Huisman, D., & Dekker, R. (2004). Scheduling preventive railway maintenance activities. In *IEEE International Conference on Systems, Man and Cybernetics*, vol. 5, (pp. 4171–4176).

Budai, G., Huisman, D., & Dekker, R. (2006). Scheduling preventive railway maintenance activities. *Journal of the Operational Research Society*, *57*(9), 1035–1044.

Cheung, B., Chow, K., Hui, L., & Yong, A. (1999). Railway track possession assignment using constraint satisfaction. *Engineering Applications of Artificial Intelligence*, *12*, 599–611.

Christiansen, M., Fagerholt, K., Nygreen, B., & Ronen, D. (2007). Maritime transportation. In B. C., & L. G. (Eds.) *Handbooks in Operations Research and Management Science: Transportation*, vol. 14, (pp. 189–284). Elsevier, Amsterdam.

Christiansen, M., Fagerholt, K., & Ronen, D. (2004). Ship Routing and Scheduling: Status and Perspectives. *Transportation Science*, *38*(1), 1–18.

Cordeau, J.-F., Toth, P., & Vigo, D. (1998). A Survey of Optimization Models for Train Routing and Scheduling. *Transportation Science*, *32*(4), 380–404.

Crainic, T. (2003). Long-haul freight transportation. In R. W. Hall (Ed.) *Handbook of Transportation Science*, vol. 56, (pp. 451–516). Springer New York.

Crainic, T., Ferland, J.-A., & Rousseau, J.-M. (1984). A Tactical Planning Model for Rail Freight Transportation. *Transportation Science*, *18*(2), 165–184.

Crainic, T., & Kim, K. (2007). Intermodal transportation. In B. C., & L. G. (Eds.) *Handbooks in Operations Research and Management Science: Transportation*, vol. 14, (pp. 467–537). Elsevier, Amsterdam.

Crainic, T. G. (2000). Service network design in freight transportation. *European Journal of Operational Research*, *122*(2), 272 – 288.

Crainic, T. G., Frangioni, A., & Gendron, B. (2001). Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. *Discrete Appl. Math.*, *112*(1-3), 73–99.

Crainic, T. G., Gendreau, M., & Farvolden, J. M. (2000). A simplex-based tabu search method for capacitated network design. *INFORMS Journal on Computing*, *12*(3), 223–236.

Crainic, T. G., Gendron, B., & Hernu, G. (2004). A slope scaling/lagrangean perturbation heuristic with long-term memory for multicommodity capacitated fixed-charge network design. *Journal of Heuristics*, *10*(5), 525–545.

Crainic, T. G., & Laporte, G. (1997). Planning models for freight transportation. *European Journal of Operational Research*, *97*(3), 409 – 438.

Crainic, T. G., & Rousseau, J.-M. (1986). Multicommodity, multimode freight transportation: A general modeling and algorithmic framework for the service network design problem. *Transportation Research Part B: Methodological*, *20*(3), 225 – 242.

Crainic, T. G., & Roy, J. (1988). Or tools for tactical freight transportation planning. *European Journal of Operational Research*, *33*(3), 290 – 297.

CSCMP (2009). The Council of Supply Chain Management Professionals (CSCMP). 20th Annual State of Logistics Report. Riding out the recession.

Farvolden, J. M., & Powell, W. B. (1994). Subgradient Methods for the Service Network Design Problem. *Transportation Science*, *28*(3), 256–272.

Frangioni, A., & Gendron, B. (2009). 0-1 reformulations of the multicommodity capacitated network design problem. *Discrete Applied Mathematics*, *157*(6), 1229 – 1241.

Gavish, B., & Altinkemer, K. (1990). Backbone Network Design Tools with Economic Tradeoffs. *Informs Journal on Computing*, *2*(3), 236–252.

Gendron, B., Potvin, J.-Y., & Soriano, P. (2002). Diversification strategies in local search for a nonbifurcated network loading problem. *European Journal of Operational Research*, *142*(2), 231 – 241.

Ghamlouche, I., Crainic, T., & Gendreau, M. (2004). Path relinking, cycle-based neighbourhoods and capacitated multicommodity network design. *Annals of Operations Research*, *131*(1-4), 109–133.

Ghamlouche, I., Crainic, T. G., & Gendreau, M. (2003). Cycle-based neighbourhoods for fixed-charge capacitated multicommodity network design. *Operations Research*, *51*(4), 655–667.

Gorman, F. M. (1998). An application of genetic and tabu searches to the freight railroad operating plan problem. *Annals of Operations Research*, *78*, 51–69.

Grimes, G., & Barkan, C. (2006). Cost-effectiveness of railway infrastructure renewal maintenance. *Journal of Transportation Engineering*, *132*, 601–608.

Haghani, A. E. (1989). Formulation and solution of a combined train routing and makeup, and empty car distribution model. *Transportation Research Part B: Methodological*, *23*(6), 433 – 452.

Higgins, A. (1998). Scheduling of railway track maintenance activities and crews. *Journal of the Operational Research Society*, *49*, 1026–1033.

Higgins, A., Ferreira, L., & Lake, M. (1999). Scheduling rail track maintenance to minimize overall delays. In A. Ceder (Ed.) *Transportation and traffic theory in the 21st century: Proceedings of the 14th International Symposium on Transportation and Traffic Theory*, (pp. 779–796). Jerusalem, Israel: Pergamon.

Hoff, A., Lium, A.-G., Lkketangen, A., & Crainic, T. G. (2009). A metaheuristic for stochastic service network design. *Journal of Heuristics*. Doi: 10.1007/s10732-009-9112-8.

Holmberg, K., & Hellstrand, J. (1998). Solving the Uncapacitated Network Design Problem by a Lagrangean Heuristic and Branch-and-Bound. *Operations Research*, *46*(2), 247–259.

Holmberg, K., & Yuan, D. (2000). A Lagrangian Heuristic Based Branch-and-Bound Approach for the Capacitated Network Design Problem. *Operations Research*, *48*(3), 461–481.

Jarrah, A. I., Johnson, E., & Neubert, L. C. (2009). Large-scale, less-than-truckload service network design. *Operations Research*, *57*(3), 609–625.

Jha, K. C., Ahuja, R. K., & Şahin, G. (2008). New approaches for solving the block-to-train assignment problem. *Networks*, *51*(1), 48–62.

Katayama, N., Chen, M., & Kubo, M. (2009). A capacity scaling heuristic for the multicommodity capacitated network design problem. *J. Comput. Appl. Math.*, *232*(1), 90–101.

Keaton, M. H. (1989). Designing optimal railroad operating plans: Lagrangian relaxation and heuristic approaches. *Transportation Research Part B: Methodological*, *23*(6), 415 – 431.

Keaton, M. H. (1991). Service-cost tradeoffs for carload freight traffic in the u.s. rail industry. *Transportation Research Part A: General*, *25*(6), 363 – 374.

Keaton, M. H. (1992). Designing Railroad Operating Plans: A Dual Adjustment Method for Implementing Lagrangian Relaxation. *Transportation Science*, *26*(4), 263–279.

Kim, D., Barnhart, C., Ware, K., & Reinhardt, G. (1999). Multimodal Express Package Delivery: A Service Network Design Application. *Transportation Science*, *33*(4), 391–407.

Kim, D., & Pardalos, P. M. (1999). A solution approach to the fixed charge network flow problem using a dynamic slope scaling procedure. *Operations Research Letters*, *24*(4), 195 – 203.

Lake, M., & Ferreira, L. (2002). Minimising the conflict between rail operations and infrastructure maintenance. In M. Taylor (Ed.) *Transportation and traffic theory in the 21st century: Proceedings of the 15th International Symposium on Transportation and Traffic Theory*, (pp. 63–74). Adelaide, Australia: Elsevier.

Lium, A.-G., Crainic, T. G., & Wallace, S. W. (2009). A Study of Demand Stochasticity in Service Network Design. *Transportation Science*, *43*(2), 144–157.

Magnanti, T., Mirchandani, P., & Vachani, R. (1993). The convex hull of two core capacitated network design problems. *Mathematical Programming*, *60*(2), 233–250.

Magnanti, T. L., Mirchandani, P., & Vachani, R. (1995). Modeling and Solving the Two-Facility Capacitated Network Loading Problem. *Operations Research*, *43*(1), 142–157.

Magnanti, T. L., Mireault, P., & Wong, R. T. (1986). Tailoring benders decomposition for uncapacitated network design. *Mathematical Programming Study*, *26*, 112–154.

Magnanti, T. L., & Wong, R. T. (1984). Network Design and Transportation Planning: Models and Algorithms. *Transportation Science*, *18*(1), 1–55.

Minoux, M. (1989). Network synthesis and optimum network design problems: Models, solution methods and applications. *Networks*, *19*, 313–360.

Nemani, A., Bog, S., & Ahuja, R. (2010). Solving the curfew planning problem. *Transportation Science*. Accepted for publication.

Newton, H. N., Barnhart, C., & Vance, P. H. (1998). Constructing railroad blocking plans to minimize handling costs. *Transportation Science*, *32*(4), 330–345.

Oke, S. (2004). Maintenance scheduling: description, status, and future directions. *South African Journal of Industrial Engineering*, *15*(1), 101–117.

Pedersen, M. B., Crainic, T. G., & Madsen, O. B. G. (2009). Models and tabu search metaheuristics for service network design with asset-balance requirements. *Transportation Science*, *43*(2), 158–177.

Powell, W. B. (1986). A Local Improvement Heuristic for the Design of Less-than-Truckload Motor Carrier Networks. *Transportation Science*, *20*(4), 246–257.

Powell, W. B., & Sheffi, Y. (1983). The load planning problem of motor carriers: Problem description and a proposed solution approach. *Transportation Research Part A: General*, *17*(6), 471–480.

Powell, W. B., & Sheffi, Y. (1989). Design and implementation of an interactive optimization system for network design in the motor carrier industry. *Operations Research*, *37*(1), 12–29.

Smilowitz, K. R., Atamtrk, A., & Daganzo, C. F. (2003). Deferred item and vehicle routing within integrated networks. *Transportation Research Part E: Logistics and Transportation Review*, *39*(4), 305 – 323.

Teypaz, N., Schrenk, S., & Cung, V.-D. (2010). A decomposition scheme for large-scale service network design with asset management. *Transportation Research Part E: Logistics and Transportation Review*, *46*(1), 156 – 170.

U.S. Department of Transportation (2007). U.S. Department of Transportation, Federal Highway Administration, Office of Freight Management and Operations, Freight Analysis Framework, Table 2-1. `http://ops.fhwa.dot.gov/freight/freight_analysis/nat_freight_stats/` Accessed in September 2008.

USDOT RITA BTS (2010a). U.S. Department of Transportation (USDOT), Research and Innovative Technology Administration (RITA), Bureau of Transportation Statistics (BTS). Freight Transportation: Global Highlights 2010. Available at `http://www.bts.gov/publications/freight_transportation/` as of April 16, 2010.

USDOT RITA BTS (2010b). U.S. Department of Transportation (USDOT), Research and Innovative Technology Administration (RITA), Bureau of Transportation Statistics (BTS). National Transportation Statistics, Table 1-46b: U.S. Ton-Miles of Freight (BTS Special Tabulation). Available at `http://www.bts.gov/publications/national_transportation_statistics/` as of January 2010.

USDOT RITA BTS (2010c). U.S. Department of Transportation (USDOT), Research and Innovative Technology Administration (RITA), Bureau of Transportation Statistics (BTS). National Transportation Statistics, Table 3-19b. Available at `http://www.bts.gov/publications/national_transportation_statistics/` as of January 2010.

USDOT RITA BTS (2010d). U.S. Department of Transportation (USDOT), Research and Innovative Technology Administration (RITA), Bureau of Transportation Statistics (BTS). National Transportation Statistics, Table 3-5. Available at `http://www.bts.gov/publications/national_transportation_statistics/` as of January 2010.

van Hoesel, S., Koster, A., van de Leensel, R., & Savelsbergh, M. (2002). Polyhedral results for the edge capacity polytope. *Mathematical Programming*, *92*(2), 335–358.

van Hoesel, S., Koster, A., van de Leensel, R., & Savelsbergh, M. (2003). Bidirected and unidirected capacity installation in telecommunication networks. *Discrete Applied Mathematics*, *133*, 103–121.

Wieberneit, N. (2008). Service network design for freight transportation: a review. *OR Spectrum*, *30*(1), 77–112.

BIOGRAPHICAL SKETCH

Suat Boğ was born in Samsun, Turkey in 1981. He received his B.S. degree in industrial engineering from Boğaziçi University, Istanbul in 2004 and his M.S. degree in industrial engineering from Koç University, Istanbul in 2006. Since August 2006, he has been pursuing his doctoral degree in the Department of Industrial and Systems Engineering at the University of Florida. Suat's academic research is focused on solving large-scale discrete optimization problems. His research interests include integer programming, network optimization and heuristics. In his Ph.D. study, he worked with Dr. Ravindra K. Ahuja and conducted research on network design, routing and scheduling problems emerging in freight transportation industry. Specifically, he focused on three important areas related to freight transportation: integrated transportation planning for non-bifurcated network design problems, service network design on hub-and-spoke networks and transportation network disruption.