Designing Robust Railroad Blocking Plans

by

Hong Jin

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Transportation and Logistics Systems

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1998

© Massachusetts Institute of Technology 1998. All rights reserved.

Author

Department of Civil and Environmental Engineering

June 30, 1998

Certified by

Cynthia Barnhart
Associate Professor of Civil and Environmental Engineering
Thesis Supervisor

Accepted by

Joseph Sussman
Chairman, Department Committee on Graduate Students
Designing Robust Railroad Blocking Plans

by

Hong Jin

Submitted to the Department of Civil and Environmental Engineering on June 30, 1998, in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Transportation and Logistics Systems

Abstract

On major domestic railroads, a typical general merchandise shipment, or commodity, may pass through many classification yards on its route from origin to destination. At these yards, the incoming traffic, which may consist of several shipments, is reclassified (sorted and grouped together) to be placed on outgoing trains. On average, each reclassification results in an one day delay for the shipment. In addition, the classification process is labor and capital intensive. To prevent shipments from being reclassified at every yard they pass through, several shipments may be grouped together to form a block. The blocking problem consists of choosing the set of blocks to be built at each terminal (the blocking plan) and assigning each commodity to a series of blocks that will take it from origin to destination. It is one of the most important problems in railroad freight transportation since a good blocking plan can reduce the number of reclassifications of the shipments, thus reducing operating costs and delays associated with excess reclassifications.

We provide a variety of model formulations that attain the minimum costs for different problem instances. The deterministic model identifies the blocking plan for the problems with certainty in problem inputs. Static stochastic models provide blocking plans that are feasible for all possible realizations of uncertainties in demand and supply. Dynamic stochastic models generate blocking plans that balance flow costs and plan change costs for possible realizations of uncertainties.

We adopt Lagrangian relaxation techniques to decompose the resulting huge mixed integer programming models into two smaller subproblems. This reduces storage requirements and computational efforts to solve these huge problems. We propose other enhancements to reduce computational burden, such as adding a set of valid inequalities and using advanced start dual solutions. These enhancements help tighten the lower bounds and facilitate the generation of high quality feasible solutions.

We test the proposed models and solution approaches using the data from a major railroad. Compared to current blocking plans, the solutions from our model reduce the total number of classifications significantly, leading to potential savings of millions of dollars annually. We also investigate various problem aggregation techniques to determine the appropriate ways of generating satisfactory blocking plans with differ-
ent levels of computational resources. We illustrate the benefits of robust planning by comparing the total costs of our robust plans with those of our deterministic plans. The experiments show that the realized costs can be reduced by around 50% using robust blocking plans.

Thesis Supervisor: Cynthia Barnhart
Title: Associate Professor of Civil and Environmental Engineering
Acknowledgments

Firstly, I would like to express my great appreciation for my advisor, Professor Cynthia Barnhart. Her constant guidance, encouragement and support are so valuable that the thesis is not possible without her direction.

Deep appreciation is extended to my committee members. Professors Joseph Sussman and Yosef Sheffi provide insightful discussions and suggestions that inspire many topics in my thesis. Professor Pamela Vance from Auburn University tirelessly travels to participate in the committee meetings, her suggestions and comments greatly help in solution algorithm development.

Also, I would like to take this opportunity to thank Carl Martland, my advisor when I was in the MIT Rail Group. His direction, encouragement lead me from a student to a researcher.

I thank UPS foundation and CSX Transportation for sponsoring my research. Raja Kasilingam, Adriene Bailey, Dharma Acharya, Yen Shan and James Lyvers from CSXT provide constant support and information on railroad operations. Harry Newton from Air Force Academy gives me a big help at the beginning of this study.

I want to thank my friends and office-mates in 5-012, Niranjan Krishnan, Francisco Jauffred, Andy Armacost and Daeki Kim for their useful discussions and unselfish help. I acknowledge my fellow students in CTS, Qi Yang, John Bowman, Yan Dong, Chris Capalice, Owen Chen, Joan Walker, Scott Ramming, Jon Bottom, Kazi Ahmed, Winston Guo, Zhaopu Si, Qian Ma, William Robert, Xing Yang, Julie Wei, Fang Lu and Yiyi He for their friendship.

Last but not least, I would like to express gratitude to my family, my parents for their affection and support, my wife, Dongyan Ma, her understanding, patience and support.
## Contents

1 Introduction ................................................................. 10
   1.1 Railroad Tactical Planning ........................................... 10
   1.2 The Railroad Blocking Problem ...................................... 12
      1.2.1 Problem Introduction ........................................... 12
      1.2.2 Significance of Blocking Problem ............................. 13
      1.2.3 Previous Blocking Research .................................... 14
      1.2.4 The Blocking Problem with Uncertainty ..................... 15
   1.3 Review of Related Literature ........................................ 18
      1.3.1 Network Design .................................................. 18
      1.3.2 Stochastic Optimization ........................................ 19
   1.4 Thesis Outline ....................................................... 21

2 Railroad Blocking with Deterministic Data .......................... 22
   2.1 Network Design Models ............................................... 22
      2.1.1 A Generalized Node-Arc Formulation .......................... 22
      2.1.2 A Path Formulation for Budget Network Design ............. 25
      2.1.3 Forcing Constraints ............................................ 27
   2.2 A Blocking Model with Deterministic Parameters ................. 27
      2.2.1 Rail Blocking Network ........................................... 28
      2.2.2 Model Formulation ............................................... 29
   2.3 Solution Approach ................................................... 30
      2.3.1 Lagrangian Relaxation .......................................... 30
      2.3.2 Solution Algorithms ............................................ 34
3 Railroad Blocking with Deterministic Data: A Case Study

3.1 Introduction

3.2 Problem Data

3.3 Problem Aggregation Schemes
  3.3.1 Node-Based Aggregation
  3.3.2 Route-Based Aggregation
  3.3.3 Commodity-Based Aggregation
  3.3.4 Comparison of Aggregation Schemes

3.4 Problem Aggregation Case Studies
  3.4.1 Implementation Details
  3.4.2 Node-Based Aggregation Case Study
  3.4.3 Route-Based Aggregation Case Study
  3.4.4 Commodity-Based Aggregation Case Study
  3.4.5 Computer Storage Requirements
  3.4.6 Comparison to Current Operations

3.5 Real Time Blocking Plan Generation

3.6 Efficiency Enhancements

3.7 Volume Constraints in the Blocking Problem

3.8 Summary

4 The Railroad Blocking Problem with Stochastic Inputs

4.1 Uncertainties in the Railroad Blocking Problem

4.2 Robustness and Robust Planning
  4.2.1 Robustness Definition
  4.2.2 Mathematical Formulations for Robustness
  4.2.3 Static vs. Dynamic Plans
4.2.4 Literature Review ........................................ 82
4.3 Static Blocking Models ...................................... 83
  4.3.1 Static Worst-Case Model ................................. 84
  4.3.2 Static Expected Performance Model ..................... 87
4.4 Dynamic Expected Performance Blocking Models .......... 88
  4.4.1 Quadratic Dynamic Model ............................... 89
  4.4.2 Linear Dynamic Model .................................. 92
4.5 Summary ...................................................... 96

5 Robust Railroad Blocking: A Case Study 97
  5.1 Alternative Blocking Plans ................................ 97
    5.1.1 Static Blocking Plans ................................ 97
    5.1.2 Dynamic Blocking Plans ............................... 98
  5.2 Implementation Issues .................................... 99
    5.2.1 Daily Variations in Demand ......................... 99
    5.2.2 Plan Change Costs .................................... 100
    5.2.3 Convergence Criteria ................................. 100
  5.3 Computational Results .................................... 101
    5.3.1 Single-Scenario Blocking Problems ................. 101
    5.3.2 Single-Scenario Blocking Plan Evaluation .......... 101
    5.3.3 Robust Planning Improvements ....................... 103
    5.3.4 Robust Planning with Limited Scenarios ............ 108
    5.3.5 Simulation Case Study ............................... 112
  5.4 Summary .................................................... 117

6 Contributions and Future Directions 118
  6.1 Contributions ............................................. 118
  6.2 Future Directions .......................................... 119
    6.2.1 Model Development ................................... 119
    6.2.2 Algorithmic Development ............................ 120
    6.2.3 Application Context ................................. 120
List of Figures

1-1 An Example of Blocking Path vs. Physical Route .................. 13
2-1 Column Generation Procedure for FLOW .......................... 33
2-2 Branch-and-Cut Procedure for BLOCK ............................. 35
2-3 Lagrangian Relaxation Approach for Blocking Problem ........... 36
3-1 Best Feasible Solutions under Different Run Time Limits .......... 58
3-2 Best Solution Profiles for Different Node Aggregation Levels ...... 60
3-3 Solution Values under Storage Space Constraints .................. 67
5-1 Realized Costs vs. Planned Costs for Static Plans Based on Single
     Scenarios .......................................................... 103
5-2 Cost Comparisons for Different Plans ............................... 105
5-3 Impacts of Scenario Selection on Realized Costs ................... 106
5-4 Realized Costs Comparisons for Static Plans Based on Single Scenarios 110
5-5 Realized Cost Comparisons for Dynamic Plans Based on Single Scenarios 111
5-6 Realized Cost Comparisons for Plans with Different Scenario Selection 113
# List of Tables

3.1 Levels of Node-Based Aggregation and Problem Sizes ............................................. 56
3.2 Computational Results for Node-Based Aggregation ................................................. 57
3.3 Computational Results for Route-Based Aggregation ............................................... 62
3.4 Appropriate Level of Aggregation and Solution ....................................................... 64
3.5 Computational Results for Commodity-Based Aggregation ....................................... 65
3.6 Appropriate Level of Aggregation and Solution Value with Storage Constraints ............. 67
3.7 Model Solution vs. Current Practice .......................................................................... 69
3.8 “One-Shot” Feasible Plan Generation Comparisons .................................................... 71
3.9 Computational Results Comparisons on Medium Level Aggregate Network ................. 73
3.10 Comparisons With and Without Car Volume Constraints .......................................... 74

5.1 Daily Demand Variations in a Representative Week .................................................. 100
5.2 Network Design Solutions Under a Single-Scenario ................................................ 101
5.3 Plan Change Cost Assumptions in Robust Blocking Plans .......................................... 107
5.4 Robust Plans under Different Change Cost Assumptions .......................................... 107
5.5 Realized vs. Planned Costs for Daily Demand Scenarios .......................................... 113
5.6 Comparisons of Realized Costs with Different Static Blocking Plans ...................... 115
5.7 Comparisons of Realized Costs with Different Dynamic Blocking Plans .................... 117
Chapter 1

Introduction

1.1 Railroad Tactical Planning

Railroad operations planning can be categorized according to Assad (1980) as Strategic, Tactical, or Operational. Strategic planning deals with long-term (usually in years) decisions on investments, marketing and network design including train services and schedules, capacity expansion, track abandonment and facility location. Tactical planning focuses on effective allocation of existing resources over medium-term planning horizons ranging from one month to one year. Operational planning, in a dynamic environment, manages the day-to-day activities, such as train timetable, empty car distribution, track scheduling and yard receiving/dispatching policies.

In strategic planning, railroads make up long-term business strategies, which usually involve major capital investments. These strategies include:

- **Network Design and Improvement** laying out the network coverage and service range;

- **Terminal Location and Capacity** selecting the terminals and their functionality on the network, and equipping the functional yards with adequate resources;

- **Service Planning and Differentiation** setting the target market and service standards for various traffic with different priority; and
- **Merger and Acquisition** expanding network coverage and business by combined effort and cooperation from other companies.

In tactical planning, railroads determine how to move the traffic from origins to destinations using available system resources. The results of tactical planning are operating plans, including

- **Blocking Plans** dictating which blocks (i.e. groups of shipments to be classified as units) should be built at each yard and which traffic should be assigned to each block;

- **Maintenance of Way Plans** scheduling maintenance for network facilities, including tracks and trains;

- **Train Schedule Plans** specifying blocks assigned to each train, and train routes and arrival/departure times at yards;

- **Crew Schedules** assigning crews to trains; and

- **Power Schedules** assigning locomotives to trains.

In operational planning, railroads specify in great detail the daily activities, an implementation plan, and an execution schedule. Examples include

- **Train Timetables** determining arrival and departure times of each train at stations in its itinerary; and

- **Empty car distribution** specifying the time, location and route for empty car routing and scheduling on the network;

The focus of this research is the blocking problem, a medium-term tactical planning problem. The objective of the blocking problem is to find an effective blocking plan that reduces the costs of timely delivery of all traffic from their origins to their destinations, using a given set of resources.
1.2 The Railroad Blocking Problem

1.2.1 Problem Introduction

A railroad physical network consists of a set of functional yards connected by links. Certain yards are classification yards where blocking operations (i.e., the grouping of incoming cars for connection with outgoing trains) are performed. In railroad freight transportation, a shipment, which consists of one or more cars with the same origin and destination (OD), may pass through several classification yards on its journey. The classification process might cause a one-day delay for the shipment, making it a major source of delay and unreliable service. Instead of classifying the shipment at every yard along its route, railroads group several incoming and originating shipments together to form a block. A block is defined by an OD pair that may be different from the OD pairs of individual shipments in the block. Once a shipment is placed in a block, it will not be classified again until it reaches the destination of that block. Ideally, each shipment would be assigned to a direct block, whose OD is the same as that of the shipment, to avoid unnecessary classifications and delays. However, blocking capacity at each yard, determined by available yard resources (working crews, the number of classification tracks and switching engines), limits the maximum number of blocks and maximum car volume that each yard can handle, preventing railroads from assigning direct blocks for all shipments.

Aiming at delivering the total traffic (i.e., the set of all shipments) with the fewest possible classifications, railroads develop blocking plans instructing the crews which blocks should be built at each yard and what shipments should be placed in each block. The sequence of blocks, to which one shipment is assigned along its route, form a blocking path for the shipment. It is worth noting that, for a given shipment, the blocking path may be different from its physical route. For example, consider the physical route O-A-B-C-D for a shipment from location O to location D, passing through locations A, B and C, as shown in Figure 1-1. Blocks might be built only from O to B and from B to D. Then, the blocking path for the traffic is O-B-D, which is a subsequence of its physical route. In general, each sequence of the terminals on a
shipment’s physical route corresponds to a possible blocking path for that commodity.

On a physical rail network, there might be a large number of routes connecting a given OD pair. However, most circuitous routes are eliminated due to excess distance. Using only the remaining routes, railroads attempt to design blocking plans to transport all the traffic while minimizing the number of classifications and satisfying blocking capacity constraints.

1.2.2 Significance of Blocking Problem

The impacts of blocking plans on railroad operations can be summarized as follows:

- An efficient blocking plan can reduce total operating costs in railroad operations. Classifications at yards are labor and capital intense, consisting of 10% of railroad total operating costs on average.

- The blocking plan has ripple effects on subsequent plans, including train scheduling, crew and power assignment that are developed based on given blocking plans. Therefore, the savings from blocking activities can be amplified through cost reductions in train, crew and power scheduling.
• A good blocking plan has the potential to improve railroad service levels. Through reductions in the number of classifications, a good blocking plan can decrease the potential delays occurring in classification yards, thereby enhancing service quality and, in turn, improving the ability of the railroad to compete with other freight transportation modes, such as trucking and airlines.

Currently, most railroads develop a blocking plan by incrementally refining an existing plan. These refinements are inherently local in nature and may fail to recognize the opportunities for improvements that require more global changes to the plan. In this research, we provide optimization models and solution approaches that are capable of solving real world applications.

1.2.3 Previous Blocking Research

Over the years, there have been several attempts to model and solve the blocking problem using optimization-based approaches.

Bodin, et al. (1980) formulate the blocking problem as an arc-based mixed integer multicommodity flow problem with a piece-wise linear objective function to capture queueing delays. This formulation includes capacity constraints at each yard in terms of the maximum number of blocks and the maximum car volume that can be handled. In the formulation, there is one binary variable for each possible block-routing combination. The number of binary variables is so large that most of them have to be set heuristically in order to find feasible solutions to the problem.

The Automatic Blocking Model (ABM) developed by Van Dyke (1986) applies a shortest path heuristic to assign traffic to a pre-specified set of blocks. He uses a greedy heuristic to add and drop blocks based on existing blocking plans to search for a better blocking scheme that satisfies yard capacity constraints.

Other studies include blocking in more comprehensive rail planning models, for example Crainic, et al. (1984) and Keaton (1989, 1992). In these models, the problem of developing a railroad operating plan, is formulated as a fixed charge network design problem with a nonlinear objective function to capture the congestion effects at the
yards. An operating plan includes a blocking plan, the assignment of blocks to trains, and train frequency decisions.

Recently, Newton, et al. (1997) and Newton (1996) propose a path-based network design formulation for the blocking problem where, as in Bodin, blocking capacity at the yards is explicitly specified by a maximum number of blocks and maximum car volume. A set of disaggregate forcing constraints are added to tighten the linear programming relaxation lower bound. They test the model on a strategic network, the size of which is among the largest blocking problems reported in the literature. Their approach adopts the state-of-the-art branch-and-price algorithm proposed by Barnhart, et al. (1996).

1.2.4 The Blocking Problem with Uncertainty

In railroad tactical planning, both demand and resources in rail systems are typically assumed to be known and deterministic, and tactical plans are developed accordingly. However, the medium-term planning horizons of tactical planning involve multiple periods, where variations in demand and supply are inevitable. The rift between the current planning philosophy and actual operating environments yields static or fixed operating plans that sometimes result in chaos in real time operations and high unexpected costs. For example, operating plans based on a deterministic demand or supply may not be feasible for certain realized demands in busy periods. On the other hand, operating plans based on high demands in busy periods might not be efficient for low demands in slack periods. Dong (1997) develops a simulation model to test the effects of adhering to a set of fixed schedules in rail yard operations, and he finds that such a strategy demands extra resources, resulting in large increases in actual operating costs. Similar problems of static plans not fitting dynamic operating environments are also observed in airline operations. Clarke (1997) reports that realized costs in actual operations are much higher than the planned costs, and he concludes that the realized costs, not the planned costs, should be used in evaluating alternative plans. To estimate realized costs, we have to evaluate operating plans against all possible realizations of uncertainties.
Robust planning involves generating plans that perform well under all or most potential realizations of uncertainties. Depending on specific planning objectives, a robust plan might be designed to yield the minimum cost for the worst case realization; to achieve the minimum expected cost over all realizations; to satisfy a certain level of service requirements; or to require minimal adjustments to be optimal for all realizations.

Robust planning is important in various applications, including:

- **Ground Holding Problem**: an air traffic flow management application (see Richetta and Odoni [1993] and Vranas, et al. [1994]) involving the determination of aircraft ground holds when landing capacity at airports is affected by adverse weather conditions, etc.;

- **Portfolio Management**: (Zenios 1992) incorporates risk into financial investment decisions and determines the investment portfolio to maximize expected return.

- **Survivable Network Design**: the design of telecommunications networks (Stoer [1992] and Goemans and Bertsimas [1993]) that are redundant in that the networks remain operational when some links are damaged.

These applications are similar in that changes to specific decisions might be impossible or costly. For example, in the portfolio management problem, changing the current portfolio involves significant transaction fees. In the survivable network design problem, if the network crashes due to a facility failure, it could take a long time to replace the failed facility, causing the system to halt for a while and incur large costs and severe losses. This inability to change decisions freely explains the significance of robust planning, where performance of a robust plan or design is insensitive to the particular realization of uncertainties. The objective of robust planning is to achieve the most desirable performance under different realizations of uncertainties, through intelligent design with certain constraints and random factors.
Despite the existence of uncertainty, to our knowledge, there is no literature on stochastic blocking optimization. The lack of research in this area is attributed to the following:

- Robust planning involves multiple objectives. In general, robust planning has to deal with the trade-off between maintaining plan constancy and adapting the plan to the realizations of uncertainties, e.g., variations in demand and/or supply. The expected total costs could be high under a constant plan, and lower operating costs could be achieved by allowing these blocking plans to be adjusted to better match specific realizations of uncertainties. However, due to ripple effects associated with changing blocking plans, blocking plan adjustments are usually costly. The goal of robust planning is to balance these competing objectives by determining an optimal plan, sometimes defined as the plan with the lowest expected total costs, including both the original planned costs and the costs resulting from plan adjustments.

- Robustness is difficult to define and model. Depending on specific problem instances, a robust plan, under all realizations of uncertainties, could be a plan that yields the minimum cost for the worst case, or a plan that achieves the minimum expected cost, or a plan that satisfies a certain level of service requirements, or a plan that requires minimal adjustments, if any, to be optimal for most cases.

- Optimization models capturing stochasticity are often computationally intractable due to their large size. In fact, deterministic optimization models for actual blocking applications are challenging to solve. The additional complexity and problem size when stochasticity is considered leads to issues of tractability, even for relatively small networks, especially when the problem needs to be solved in a selectively short period of time.

In this research, we establish a framework for modeling and solving blocking problems with uncertain data to generate robust blocking plans. We will survey robustness
definitions and evaluate their applicability to the railroad blocking problem. We illustrate the benefits of our approach using blocking data provided by a major railroad.

1.3 Review of Related Literature

In this research, we model the railroad blocking problem as a special case of the network design problem and try to achieve robust blocking plans by considering variability in input parameters. In this section, we review literature on network design and stochastic optimization.

1.3.1 Network Design

Network design is a general class of problems involving the selection of a set of locations and/or the set of movements to include in a network in order to flow commodities from their origins to their destinations and achieve maximum profits, while satisfying level of service requirements. Level of service requirements might include requirements to move commodities from origins to destinations within certain time frames or distances, or requirements to maintain a certain level of network connectivity (for example, a minimum number of disjoint paths might be required for certain commodities). Network design problems arise in numerous applications in the transportation, logistics and the telecommunications industries. Examples include:

- **Airline Network Design**, discussed in Knicker (1998), involves the selection of an optimal set of routes and schedules for the airline's aircraft fleet such that profits are maximized.

- **Express Shipment Network Design**, discussed by Barnhart and Schneur (1996), Barnhart et al. (1997) and Krishnan, et al. (1998), jointly determines aircraft flights, ground vehicle and package routes and schedules to transport the packages from their origins to their destinations within certain time windows.

- **Telecommunications Network Design**, discussed in Stoer (1992), Jan (1993) and Balakrishnan, et al. (1998), determines a set of offices and fiber links to
deliver the messages from origins to destinations with minimum cost and certain levels of reliability.

- **Logistics Network Design**, discussed in Ballou (1995) and Sheffi (1997), decides the locations of facilities and routings for raw materials and final products to achieve minimum logistics costs.

- **Less-Than-Truckload Network Design**, discussed in Lamar, et al. (1990), Powell (1994) and Farvolden and Powell (1986), determines minimum cost routes and schedules for tractors and trailers to convey freight from origins to destinations with the available fleet and facilities.

- **Multimode Freight Network Design**, discussed in Crainic and Rousseau (1986), determines an uncapacitated service network design using decomposition heuristics and column generation to minimize the costs of delivering multiple commodities.

### 1.3.2 Stochastic Optimization

Stochastic optimization is an important approach of solving stochastic problems with multiple data instances that might be potentially realized in the future. This approach incorporates the multiple data instances to achieve robust solutions. Birge (1995) provides a detailed survey of stochastic optimization models and solution approaches. We summarize some recent work as follows:

- The two-stage and multi-stage stochastic linear programs. The *two-stage* linear program is originated by Dantzig (1955) and Beale (1955), which decomposes decision processes under uncertainty into two stages. In the first stage, decisions are made for the activities that cannot be postponed. The remaining decisions are made in the second stage until better information becomes available. When realizations of uncertainty are revealed sequentially over time, such a decision making process naturally becomes a *multi-stage* programming problem. Kouvelis and Yu (1997) and Gutierrez, et al. (1996) apply the two-stage
stochastic programming method to solve a robust network design problem with
the objective of minimizing the cost for the worst case realization of uncertain-
ries. Cheung and Powell (1994) discuss two-stage and multi-stage planning for
distribution problems involving the movement of inventory from plants to ware-
program to model the vehicle routing problem with uncertain demands.

- Probabilistic constraint (also known as chance constrained) models (Charnes
  and Cooper [1959, 1963]) result in one plan that guarantees a minimum level
  of service some percentage of time assuming that the plan cannot be changed
  for any realization of uncertainties. For example, a major railroad might design
  its network to guarantee at least 98% of the high priority traffic, such as auto-
mobiles, to be delivered within 4 days. There is no application of probabilistic
  constraint models to railroad blocking problems or network design problems.
  Charnes and Cooper (1959, 1963), Kibzun and Kurbakovskiy (1991), Kibzun
  and Kan (1996) and Birge and Louveaux (1997) describe other types of appli-
cations of this model.

- Richetta and Odoni (1993), and Vranas, et al. (1994) discuss robust planning in
  air traffic flow management. They develop schedules for holding aircraft on the
  ground in order to minimize their time in the air. In these models, a static, fixed
  plan is generated that is feasible for many possible realizations of uncertainties.

- Mulvey, et al. (1995) propose a robust optimization approach for stochastic
  models with two distinct components: a structural component that is fixed and
  free of any variation in input data, and a control component that is subject to the
  variations. This approach generates a series of solutions that are progressively
  less sensitive to realizations of the model data.

- Stoer (1992) discusses models and solution approaches for designing survivable
  communication networks. Network survivability is achieved through redundant
  designs that possess multiple disjoint routes between pairs of nodes to allow
continued operation of a network even when failure occurs somewhere. The idea of survivability is also adopted in internet computer network design (Hafner and Lyon [1996]).

- Most recently, Jauffred (1997) proposes a stochastic model that generates an average plan that is closer to the solution of high probability events, than to infrequent events. Modifications to the average plan are allowed as uncertainties are realized, but these changes incur higher costs the greater the deviation.

1.4 Thesis Outline

In Chapter 2, we present a mixed-integer-program and algorithms for the blocking problems with deterministic parameters. In Chapter 3, we illustrate the application of our deterministic model and solution algorithms with case studies using data from a major railroad. In the case studies, we analyze the trade-off between model size and solution quality using different problem aggregation schemes. In Chapter 4, we survey various definitions of robustness and present blocking models capturing demand variations. We solve these models using a variant of the algorithm presented in Chapter 2. In Chapter 5, we evaluate different schemes for generating blocking plans considering variations in daily demands. We compare different planning philosophies to demonstrate the effects on system costs. Finally, in Chapter 6, we summarize our contributions and describe directions for future research.
Chapter 2

Railroad Blocking with Deterministic Data

2.1 Network Design Models

We begin by presenting various formulations for the network design problem.

2.1.1 A Generalized Node-Arc Formulation

To facilitate the discussion, we introduce the following notations.

* Parameters:
  - $G(N, A)$ is the graph with node set $N$ and arc set $A$
  - $K$ is the set of commodities $k$
  - $v^k$ is the volume of commodity $k$, $\forall k \in K$
  - $O(k)$ is the origin of commodity $k$, $\forall k \in K$
  - $D(k)$ is the destination of commodity $k$, $\forall k \in K$
  - $c^k_\alpha$ is the per unit cost of flow on arc $\alpha$ for commodity $k$, $\forall \alpha \in A$, $\forall k \in K$
  - $f_\alpha$ is the cost of including arc $\alpha$ in the network, $\forall \alpha \in A$
u_a is the flow capacity on arc a, \( \forall a \in A \)

\( \xi_i^a \) is the incidence indicator that equals 1 if \( i \) is the origin of arc \( a \) and 0 otherwise, \( \forall i \in N, \forall a \in A \)

\( \rho_j^a \) is the incidence indicator that equals 1 if \( j \) is the destination of arc \( a \) and 0 otherwise, \( \forall j \in N, \forall a \in A \)

d_a is the cost of including arc \( a \) in the network, \( \forall a \in A \)

\( B \) is the design budget for the network

\( B(i) \) is the design budget at node \( i \), \( \forall i \in N \)

e_k^a is the per unit cost of handling commodity \( k \) on arc \( a \), \( \forall k \in K, \forall a \in A \)

\( V \) is the flow budget for the network

\( V(i) \) is the flow budget at node \( i \), \( \forall i \in N \)

**Decision Variables:**

\( x_a^k \) is the fraction of \( v^k \) on arc \( a \), \( \forall a \in A, \forall k \in K \)

\( y_a \) is the binary design variable, where \( y_a = 1 \) if an arc \( a \) is chosen and \( y_a = 0 \) otherwise, \( \forall a \in A \)

The node-arc formulation for the network design problem is:

\[
\text{(NODE) \quad Min} \quad \sum_{k \in K} \sum_{a \in A} c_a v^k x_a^k + \sum_{a \in A} f_a y_a
\]  

s.t.

\[
\sum_{k \in K} x_a^k v^k \leq u_a y_a \quad \forall a \in A
\]  

\[
\sum_{a \in A} x_a^k \xi_i^a - \sum_{a \in A} x_a^k \rho_i^a = \begin{cases} 
1 & \text{if } i = O(k) \\
-1 & \text{if } i = D(k) \quad \forall i \in N, \forall k \in K \\
0 & \text{otherwise}
\end{cases}
\]
\[
\sum_{a \in A} d_ay_a \leq B \quad (2.4)
\]
\[
\sum_{a \in A} d_ay_a \xi_{ai}^a \leq B(i) \quad \forall i \in \mathcal{N} \quad (2.5)
\]
\[
\sum_{k \in \mathcal{K}} \sum_{a \in A} e_a^k v_k x_a^k \leq V \quad (2.6)
\]
\[
\sum_{k \in \mathcal{K}} \sum_{a \in A} e_a^k v_k x_a^k \xi_{ai}^a \leq V(i) \quad \forall i \in \mathcal{N} \quad (2.7)
\]
\[
y_a \in \{0, 1\} \quad \forall a \in A \quad (2.8)
\]
\[
x_a^k \geq 0 \quad \forall a \in A, \ \forall k \in \mathcal{K} \quad (2.9)
\]

Constraints (2.2) are the forcing constraints requiring that no flow can be sent on arcs unless the arcs are included in the network and that the maximum flow on arc \(a\) cannot exceed \(u_a\). Equalities (2.3) are the network flow conservation constraints, ensuring that all commodities are shipped from their origins to their destinations, respectively. Inequality (2.4) is the arc building budget constraint on the entire network and inequalities (2.5) are the arc building budget constraints for individual nodes, requiring that the budgets must be satisfied. Similarly, the inequalities (2.6) and (2.7) are the flow processing budget constraints, limiting the maximum cost of flow processing over the network and at individual nodes, respectively. The objective (2.1) is to minimize the total costs of shipping flows over the network and building the network.

**Fixed Charge vs. Budget Network Design**

If the budget constraints on arc building and flow processing (2.4-2.7) are absent or not binding, then the problem **NODE** becomes a fixed charge network design problem. Fixed charge network design problems trade-off design expenditures (the second term in 2.1) and improved network operations in the form of lower operating costs (the first term in 2.1).

In contrast, the budget network design problem removes the design expenditures from its objective. Instead, limits on the selection of design variables are imposed through budget constraints (2.4-2.7) at individual nodes and/or in the whole network. These budget constraints might apply to the binary design variables and/or
the continuous flow variables.

Despite the differences in the problem structure, most solution approaches can be generalized and applied to both types of network design problems. In our research, both deterministic and stochastic solution approaches can be used to solve either class of network design problems. Since the railroad blocking problem is a special case of the budget network design problem, we will focus our discussion on it.

Uncapacitated vs. Capacitated Network Design

In some network design problems, there is no limit on the amount of flow on an arc, i.e., $u_a$ in the forcing constraint (2.2) is a sufficiently large number for every arc $a \in A$. We refer to this type of problems as uncapacitated network design problems. However, in some other instances, the capacity $u_a$ is some fixed number that limits the maximum flows on arc $a$ if it is selected. We refer to these problems as capacitated network design problems. In the railroad blocking problem, there is no limit on the amount of flow assigned to a block, therefore, it is an uncapacitated network design problem.

2.1.2 A Path Formulation for Budget Network Design

By the flow decomposition theorem (Ahuja, et al. [1993]), we know that for the node-arc formulation there exists an equivalent path formulation for network design. In addition to the notations used in the node-arc formulation, we introduce the following for the path formulation.

Parameters:

- $Q(k)$ is the set of potential paths $q$ for commodity $k$, $\forall k \in K$

- $\delta_a^q$ is the incidence indicator that equals 1 if arc $a$ is on path $q$, $\forall a \in A$, $\forall q \in Q(k)$

- $PC_q^k$ is the per unit flow cost traversing path $q$, $\forall q \in Q(k)$, $\forall k \in K$
Decision Variables:

- $f^k_q$ is the proportion of commodity $k$ on path $q$, $\forall q \in Q(k), \forall k \in K$

Then, the path formulation for network design is written as:

\[
\text{(PATH)} \quad \min \sum_{k \in K} \sum_{q \in Q(k)} PC^k_q v^k f^k_q \\
\text{s.t.}
\]

\[
\sum_{q \in Q(k)} f^k_q \delta^q_a \leq y_a, \quad \forall k \in K, \forall a \in A \tag{2.11}
\]

\[
\sum_{q \in Q(k)} f^k_q = 1, \quad \forall k \in K \tag{2.12}
\]

\[
\sum_{a \in A} d_a y_a \leq B \tag{2.13}
\]

\[
\sum_{a \in A} d_a y_a c^a_\delta \leq B(i), \quad \forall i \in N \tag{2.14}
\]

\[
\sum_{k \in K} \sum_{q \in Q(k)} \sum_{a \in A} e_a v^k f^k_q \delta^q_a \leq V \tag{2.15}
\]

\[
\sum_{k \in K} \sum_{q \in Q(k)} \sum_{a \in A} e_a v^k f^k_q \delta^q_a \leq V(i), \quad \forall i \in N \tag{2.16}
\]

\[
y_a \in \{0, 1\}, \quad \forall a \in A \tag{2.17}
\]

\[
f^k_q \geq 0, \quad \forall q \in Q(k), \forall k \in K \tag{2.18}
\]

The path formulation is the result of a variable substitution in the node-arc formulation, therefore, the interpretation of the formulation remains the same. Constraints (2.11) are the forcing constraints requiring that no path is used unless all arcs on the path are selected. Equalities (2.12) are the convexity constraints that ensure all commodities are shipped from their origins to their destinations. Inequalities (2.13), (2.14), (2.15) and (2.16) are budget constraints on design variables and flow variables in the entire network and at individual network nodes, respectively. Equations (2.17) and (2.18) are binary integrality and non-negativity constraints for design variables and flow variables, respectively.
2.1.3 Forcing Constraints

In network design, we have two types of decision variables—the binary design variables \((y_a)\) and the continuous flow variables \((f^k_q)\). The forcing constraints (2.11) are the only constraints involving both types of variables. Without these constraints, the problem decomposes into two separate subproblems, one in the binary design variables and the other in the continuous flow variables. This observation is the underlying motivation for various solution algorithms, such as Lagrangian relaxation, which will be discussed in a later chapter.

There is an alternative formulation to the path formulation where the forcing constraints (2.11) are aggregated over all commodities, yielding

\[
\sum_{k \in \mathcal{K}} \sum_{q \in Q(k)} f^k_q \delta^a_q \leq |\mathcal{K}| y_a, \quad \forall a \in \mathcal{A} \quad (2.11'),
\]

where \(|\mathcal{K}|\) is the cardinality of the arc set \(\mathcal{K}\).

This alternative formulation is much more compact. For example, in one of test problems in Chapter 3, the disaggregated formulation contains over 65 million forcing constraints (2.11) while the aggregated version contains only 9,066 forcing constraints (2.11'). However, the disaggregated formulation is computationally advantageous because it provides tighter lower bounds. Hence, we will use the disaggregated model in our work.

2.2 A Blocking Model with Deterministic Parameters

A railroad blocking problem is a budget network design problem in which the nodes represent yards at which the commodities can be classified and the arcs represent potential blocks. The objective is to minimize the total operating costs of delivering all traffic over the network while satisfying capacity constraints at the yards. The capacities include only limits on the maximum number of blocks that can be built at
each yard and the maximum number of cars or *car volume* that can be handled at each yard.

### 2.2.1 Rail Blocking Network

We adopt the notations introduced for the general network design problem as follows:

* **Parameters:**

  - $G(\mathcal{N}, \mathcal{A})$ is the railroad network graph with classification yard set $\mathcal{N}$ and candidate block set $\mathcal{A}$
  - $\mathcal{K}$ is the set of commodities, that is, origin-destination specific shipment on the railroad
  - $v^k$ is the car volume of commodity $k$, measured in the number of cars per day
  - $\delta_a^q$ is the incidence indicator that equals 1 if block $a$ is on blocking path $q$ and 0 otherwise
  - $\xi^a_i$ is the incidence indicator that equals 1 if yard $i$ is the origin of block $a$ and 0 otherwise
  - $Q(k)$ is the set of all candidate blocking paths for commodity $k$
  - $C_a \geq 0$ is the per unit cost of flow on arc $a$
  - $B(i)$ is the maximum number of blocks that can be built at yard $i$
  - $V(i)$ is the maximum car volume that can be classified at yard $i$
  - $PC_q^k$ is the per unit path cost of flow for commodity $k$ on blocking path $q$

* **Decision Variables:**

  - $f_q^k$ is the proportion of commodity $k$ on path $q$
  - $y_a = 1$ if block $a$ is built and 0 otherwise
2.2.2 Model Formulation

A path-based NDP formulation of the railroad blocking problem is as follows

$$\begin{align*}
\text{(B\_PATH)} & \quad \min \sum_{k \in \mathcal{K}} \sum_{q \in \mathcal{Q}(k)} PC_q^k v^k f_q^k \\
\text{s.t.} & \quad \sum_{q \in \mathcal{Q}(k)} f_q^k q_a^k \leq y_a, \quad \forall k \in \mathcal{K}, \forall a \in \mathcal{A} \\
& \quad \sum_{q \in \mathcal{Q}(k)} f_q^k = 1, \quad \forall k \in \mathcal{K} \\
& \quad \sum_{a \in \mathcal{A}} y_a q_a^k \leq B(i), \quad \forall i \in \mathcal{N} \\
& \quad \sum_{k \in \mathcal{K}} \sum_{q \in \mathcal{Q}(k)} \sum_{a \in \mathcal{A}} v^k f_q^k q_a^k \leq V(i), \quad \forall i \in \mathcal{N} \\
& \quad f_q^k \geq 0, \quad \forall q \in \mathcal{Q}(k), \forall k \in \mathcal{K} \\
& \quad y_a \in \{0,1\}, \quad \forall a \in \mathcal{A}
\end{align*}$$

(2.19)

(2.20)

(2.21)

(2.22)

(2.23)

(2.24)

(2.25)

where $PC_q^k = \sum_{a \in \mathcal{A}} C_a q_a^k$, $\forall q \in \mathcal{Q}(k)$, $\forall k \in \mathcal{K}$.

This model formulation is similar to the one proposed in Newton, et al. (1997). Constraints (2.20) are the forcing constraints requiring that no path is used unless all blocks on the path are selected. Equalities (2.21) are the convexity constraints that guarantee all commodities are shipped from origin to destination. Inequalities (2.22) are the block building capacity constraints at individual yards and constraints (2.23) enforce the maximum car volume that can be handled at each yard. The objective is to minimize the total cost of shipment flows over the network.

Compared to the generic budget network design (PATH), the railroad blocking problem is a special case in that the design (2.13) and flow (2.15) budget constraints in the entire network are relaxed and the corresponding design (2.14) and flow (2.16) budget constraints at individual yards are specialized to limit their maximum outdegree (2.22) and maximum flow (2.23).
2.3 Solution Approach

2.3.1 Lagrangian Relaxation

The challenges of solving railroad blocking problems include:

- **The forcing constraints are difficult.** The forcing constraints (2.20) are the only constraints involving both types of variables — binary design variables \((y_a)\) and continuous flow variables \((f^k_q)\). Without the forcing constraints, the problem decomposes into two separate subproblems, one in the binary variables and the other in the continuous variables. Also, the number of the forcing constraints is very large, which is the product of numbers of shipments and potential blocks.

- **Railroad blocking problems are usually very large.** For major railroads, their blocking problems contain hundreds or thousands of nodes, and millions or billions of variables and constraints. These problems are much larger than any network design problems in the literature, requiring extremely large amount of storage space to load and solve. In fact, direct solution of these problems with workstation class computers is not practical; decomposition approaches are necessary.

The above observations motivate us to apply *Lagrangian Relaxation* (Fisher [1981]) to relax the forcing constraints and decompose the problem. Let \(\lambda^k_a\) denote the dual variable for the forcing constraint (2.20) corresponding to arc \(a\) and commodity \(k\) in the problem **B_PATH**. Then, the following Lagrangian relaxation can be obtained:

\[
\mathcal{L}(\lambda^k_a) = \min \sum_{k \in \mathcal{K}} \sum_{q \in \mathcal{Q}(k)} PC^k_q \nu^k_q f^k_q + \sum_{k \in \mathcal{K}} \sum_{a \in \mathcal{A}} \lambda^k_a \left( \sum_{q \in \mathcal{Q}(k)} f^k_q \delta^k_a - y_a \right) \tag{2.26}
\]

\[
= \min \sum_{k \in \mathcal{K}} \sum_{a \in \mathcal{A}} \left[ \sum_{q \in \mathcal{Q}(k)} (C^k_a \nu^k_q + \lambda^k_a) \delta^k_a \right] f^k_q - \sum_{a \in \mathcal{A}} \left( \sum_{k \in \mathcal{K}} \lambda^k_a \right) y_a \tag{2.27}
\]

\[
s.t. \quad \sum_{q \in \mathcal{Q}(k)} f^k_q = 1, \quad \forall k \in \mathcal{K} \tag{2.28}
\]
\[
\sum_{a \in A} y_a \xi_a^a \leq B(i), \quad \forall i \in \mathcal{N} \tag{2.29}
\]
\[
\sum_{k \in \mathcal{K}} \sum_{q \in \mathcal{Q}(k)} \sum_{a \in A} v^k f_q^k \delta_{a^q} \xi_a^a \leq V(i), \quad \forall i \in \mathcal{N} \tag{2.30}
\]
\[
f_q^k \geq 0, \quad \forall q \in \mathcal{Q}(k), \forall k \in \mathcal{K} \tag{2.31}
\]
\[
y_a \in \{0, 1\}, \quad \forall a \in A. \tag{2.32}
\]

Note in the Lagrangian relaxation problem that constraints (2.28), (2.30) and (2.31) involve only the flow variables \((f_q^k)\), while constraints (2.29) and (2.32) include only the design variable \((y_a)\). Similarly, the objective function (2.27) consists of two additive parts, each covering only one set of decision variables. Hence, the Lagrangian relaxation problem can be decomposed into two separate subproblems, a flow subproblem and a block subproblem.

To find the set of multipliers that attain the tightest lower bound, we solve the Lagrangian Dual Problem, expressed as
\[
\mathcal{L}^* = \max_{\lambda^k_{a^q} \geq 0} \mathcal{L}(\lambda^k_{a^q}). \tag{2.33}
\]

The Lagrangian Dual problem can be solved by applying subgradient optimization or other approaches for nondifferentiable optimization. In this study, we adopt the subgradient optimization approach as discussed in Fisher (1981) and Ahuja, et al. (1993).

Note in the Lagrangian relaxation problem that constraints (2.28), (2.30) and (2.31) involve only the flow variables \((f_q^k)\) while constraints (2.29) and (2.32) include only the design variable \((y_a)\). Similarly, the objective function (2.27) consists of two additive parts, each covering only one set of decision variables. Hence, the Lagrangian relaxation problem can be decomposed into two separate subproblems, a flow subproblem and a block subproblem.
Flow Subproblem

The flow subproblem is

\[
\text{(FLOW)} \quad \min \sum_{k \in \mathcal{K}} \sum_{q \in \mathcal{Q}(k)} \left( \sum_{a \in \mathcal{A}} (C_{a} v^{k} + \lambda_{a}^{k}) \delta_{a}^{q} \right) f_{q}^{k} \tag{2.34}
\]

s. t.

\[
\sum_{q \in \mathcal{Q}(k)} f_{q}^{k} = 1, \quad \forall k \in \mathcal{K} \tag{2.35}
\]

\[
\sum_{k \in \mathcal{K}} \sum_{q \in \mathcal{Q}(k)} v^{k} f_{q}^{k} \delta_{a}^{q} \leq V(i), \quad \forall i \in \mathcal{N} \tag{2.36}
\]

\[
f_{q}^{k} \geq 0, \quad \forall q \in \mathcal{Q}(k), \forall k \in \mathcal{K}. \tag{2.37}
\]

This subproblem is a multicommodity flow problem with arc cost \( C_{a} v^{k} + \lambda_{a}^{k} \) for each arc \( a \) and commodity \( k \). Since the Lagrangian multipliers are commodity specific, the same arc may have different costs for different commodities. The objective of the subproblem is to minimize the cost of delivering the commodities while satisfying the car volume capacity constraints on the nodes.

Typically, the number of blocks is large as is the number of potential block sequences (or blocking paths) for the commodities. However, most of the blocking paths will not be used in the solution. These observations motivate the use of column generation (Barnhart, et al. [1995]) in solving the flow subproblem. The implementation details of column generation is summarized in Figure 2-1. Also, notice that the multicommodity flow problem becomes a set of \(|\mathcal{K}| \) separable shortest path problems if there are no car volume constraints (2.36).

Block Subproblem

The block subproblem is

\[
\text{(BLOCK)} \quad \min - \sum_{a \in \mathcal{A}} \left( \sum_{k \in \mathcal{K}} \lambda_{a}^{k} \right) y_{a} \tag{2.38}
\]
The solution to the block subproblem will be a set of blocks that satisfy the block building capacity constraints at each node. This problem can be solved easily by simply sorting the blocks that originate at each node in non-increasing order of $\lambda_i^a$ and choosing the first $B(i)$ blocks originating at each node $i$. However, for any choice of Lagrangian multipliers, this solution coupled with the optimal solution of the flow subproblem will not provide a lower bound any tighter than the bound from the linear programming (LP) relaxation of the original model. This follows from the fact that the block subproblem has the integrality property, i.e., all the extreme points of the LP relaxation of BLOCK are integral. We improve the value of the lower bound by adding a set of valid inequalities to the block subproblem.

\[ s.t. \quad \sum_{a \in A} y_a \xi_i^a \leq B(i), \quad \forall i \in \mathcal{N} \quad (2.39) \]
\[ y_a \in \{0, 1\}, \quad \forall a \in \mathcal{A}. \quad (2.40) \]
Enhanced Block Subproblem

In the block subproblem, there are no constraints imposed on the connectivity of the network. However, the blocks in any feasible solution to the original problem (B\_PATH) will not only meet the maximum block number constraints (2.39), they will also provide at least one origin-destination path for each commodity. The connectivity constraint is implied in the original path-based formulation by the forcing constraints (2.20), which were relaxed when we formed the Lagrangian relaxation. Therefore, the solution to the above block subproblem will not necessarily provide a path for each commodity. This results in a weak lower bound and causes difficulty in generating feasible solutions. To remedy this, we add the following connectivity constraints to the block subproblem.

\[ \sum_{a \in \text{cut}(k)} y_a \geq 1, \forall \text{cut}(k) \in [O(k), D(k)], \forall k \in \mathcal{K}, \]

where \text{cut}(k) is an element in \([O(k), D(k)]\), the set of all possible cuts in the blocking network between the origin and destination of commodity \(k\).

In general, there is a large number of cuts for each commodity, which is exponential with respect to network size. Hence, the inclusion of all these constraints is impractical, especially when the problem size is large. On the other hand, many of these constraints will be satisfied by a given solution to BLOCK. So, we use row generation, summarized in Figure 2-2, to add only the constraints that are violated by the current solution.

2.3.2 Solution Algorithms

The overall solution procedure for B\_PATH is summarized in the flowchart of Figure 2-3. The algorithm can be divided into two parts—an Inner Loop and an Outer Loop. Given a set of Lagrangian multipliers, the two subproblems are solved sequentially in the inner loop, giving a lower bound to the original problem B\_PATH. We also attempt to generate a feasible solution in the inner loop. Based on the solutions
to the subproblems \((f^k_q, y_a)\), the Lagrangian multipliers are updated in the outer loop. The goal of the overall solution procedure is to search for the sharpest lower bound to the original problem \(\text{(B\_PATH)}\), i.e., to solve the Lagrangian Dual \((2.33)\). The algorithm terminates when the gap between the lower bound and feasible solution is less than a user specified tolerance.

**Outer Loop**

The Lagrangian Dual problem is solved by using subgradient optimization. In each iteration \(t\), the Lagrangian multipliers are updated as follows:

\[
\lambda^k_a^t = [\lambda^k_a^{t-1} + \theta_t \left( \sum_{q \in Q(k)} f^k_q \delta^t_a - y_a \right)]^+, \tag{2.41}
\]

where the notation \([x]^+\) denotes the positive part of the vector \(x\). To guarantee convergence (Ahuja [1993]), we choose the step size \((\theta_t = c/t)\) where \(c\) is a constant and \(t\) is the iteration index to satisfy the following conditions:

\[
\lim_{t \to \infty} \theta_t \to 0
\]
Figure 2-3: Lagrangian Relaxation Approach for Blocking Problem
\[ \lim_{t \to \infty} \sum_{i=1}^{t} \theta_i \to \infty. \]

**Inner Loop**

Given the Lagrangian multipliers from the outer loop, we solve the two subproblems at each iteration. The combined solutions from the two subproblems provide a valid lower bound on the optimal solution value.

**Flow Subproblem**

As discussed earlier, the flow subproblem is a multicommodity flow problem. Because of the exponentially large number of potential blocking paths, column generation is used to solve this subproblem. Instead of enumerating all possible blocking paths, we first solve the flow subproblem \texttt{FLOW} over a subset of the possible path variables. We refer to the multicommodity flow problem with only a subset of path variables as the \textit{Restricted Master Problem} (RMP) of the flow subproblem \texttt{FLOW}. RMPs are solved repeatedly, with successive RMPs constructed by adding new path variables with negative reduced costs. The reduced cost calculations are based on the optimal solution to the current RMP. Let \( \sigma \) and \( \pi \) be the dual variables associated with the convexity constraints (2.35) and volume capacity constraints (2.36), respectively. Then, the reduced cost of path \( q \) can be expressed as

\[
C_q^{\sigma,\pi} = \sum_{a \in A} (C_a v^k + \lambda_a^k + v^k \sum_{i \in N} \pi_i \xi_i^q) \delta_a^q - \sigma^k. \tag{2.42}
\]

The RMP solution will be optimal if \( C_q^{\sigma,\pi} \) is nonnegative for all paths \( q \in \mathcal{Q}(k) \) and all commodities \( k \); otherwise, there will exist at least one path, the addition of which might improve the RMP solution. To efficiently determine potential paths, we can solve a shortest path problem with modified arc costs \( (C_a v^k + \lambda_a^k + v^k \sum_{i \in N} \pi_i \xi_i^q) \) for each arc \( a \) and commodity \( k \), and compare the minimum path cost with the value of the dual variable \( \sigma^k \).
**Block Subproblem**

In the enhanced block subproblem BLOCK, a set of connectivity constraints are added to make the solution to the subproblem contain at least one path for each commodity. These constraints are redundant in the original problem (B_PATH) because of the presence of the forcing constraints. However, they might be violated by solutions of the subproblem (BLOCK). Addition of these constraints will enhance the quality of the lower bound generated by the Lagrangian Relaxation. Also, as discussed later, they will improve our ability to generate feasible solutions.

Because of the large number of connectivity constraints, we use a dynamic row generation approach, called branch-and-cut, in solving this subproblem. That is, we solve it using branch-and-bound, with connectivity cuts (potentially) generated at each node of the branch-and-bound tree. We generate only the constraints that are violated by the current solution to the block subproblem and continue adding connectivity constraints until each commodity has at least one OD path in the BLOCK solution.

With the addition of the connectivity constraints, an integral solution is not guaranteed in solving the LP relaxation of BLOCK. We branch on the largest fractional variables, setting them to 1, and we search the nodes of the tree in depth-first order. This helps to generate good feasible solutions quickly. If the original problem is feasible, we will obtain a blocking plan that contains at least one OD path for each commodity.

### 2.3.3 Lower Bound

Since the BLOCK and FLOW subproblems are independent, the solutions to these subproblems also solve the Lagrangian Relaxation problem \( \mathcal{L}(\lambda^f) \). In particular, the sum of the objectives from the two subproblems provides a lower bound to the original problem B_PATH according to the Lagrangian lower bound theorem in Fisher (1981).

Similarly, the linear programming relaxation also provides a lower bound on the
original problem B_PATH. Due to addition of the connectivity constraints, which are implied in the original formulation (B_PATH), and are valid in the Lagrangian relaxation, the Lagrangian relaxation with added connectivity constraints potentially attains a lower bound at least as large as the bound obtained by the linear programming relaxation of B_PATH.

2.3.4 Upper Bound

Besides providing tight lower bounds, the Lagrangian relaxation approach can also generate high quality feasible solutions. Unlike existing dual-based approaches that rely on some external feasible solution generation procedure (mainly add-and-drop heuristic or branch-and-bound), Lagrangian relaxation can generate feasible solutions directly for blocking problems in which the car volume constraints are absent. A feasible solution to the original problem can be generated by solving the two subproblems sequentially: First, the block subproblem is solved to generate a set of blocks ($\{Y_a\}$) and second, the flow subproblem is solved given the blocks selected in the first step. The solution generated is feasible if car volume constraints are not present since the blocking subproblem selects at least one OD path for each commodity.

In the general railroad blocking problem, however, the above procedure might not always find a feasible solution because the car volume constraints could be violated. We propose two methods of resolving this issue. One simple approach is to discard solutions that violate the car volume constraints. As the Lagrangian multiplier values get close to the optimal dual solution, it is more likely that the sequential procedure will find a feasible solution. This heuristic is most effective when the car volume constraints are not particularly tight. Another approach is to introduce feasibility cuts by solving a feasibility problem based on the network configuration obtained in BLOCK:

$$\text{(FEAS)} \min w = \sum_{k \in K} \sum_{a \in A} r_a^k$$  \hspace{1cm} (2.43)
s.t.

\[ \sum_{q \in Q(k)} f_q^k \delta_{a}^k - r_a^k \leq \bar{y}_a, \quad \forall k \in \mathcal{K}, \forall a \in \mathcal{A} \]  
(2.44)

\[ \sum_{q \in Q(k)} f_q^k = 1, \quad \forall k \in \mathcal{K} \]  
(2.45)

\[ \sum_{k \in \mathcal{K}} \sum_{q \in Q(k)} \sum_{a \in \mathcal{A}} v^k f_q^k c_{a}^k \leq V(i), \quad \forall i \in \mathcal{N} \]  
(2.46)

\[ f_q^k \geq 0, \quad \forall q \in Q(k), \forall k \in \mathcal{K}. \]  
(2.47)

\[ r_a^k \geq 0 \quad \forall k \in \mathcal{K}, \forall a \in \mathcal{A} \]  
(2.48)

If the optimal objective value \( w = 0 \), then the flow subproblem is feasible. Otherwise, at least one of the flow budget constraints is violated, forcing some flows to be sent on the arcs not selected in the solution \( \{\bar{y}_a\} \) to the block subproblem. Then, a feasibility cut can be identified by the following theorem.

**Proposition 2.1** When the above feasibility problem has non-zero objective \( (w > 0) \) for a given blocking plan \( (\bar{y}_a) \), then, a feasibility cut of the form \( \sum_{a \in \mathcal{F}} y_a \geq 1 \) can be identified if the problem is feasible, where \( \mathcal{F} \) is a set of blocks not selected in the block subproblem (that is, \( \bar{y}_a = 0 \)) with associated positive dual values.

**Proof** Suppose that \( (-\theta_a^k), \sigma^k \), and \( (-\pi_i) \) are the duals associated with constraints 2.44, 2.45 and 2.46, respectively. The dual of the above feasibility problem is:

\[
(DUAL\_FEAS) \max z_{dual} = \sum_{a \in \mathcal{A}} \sum_{k \in \mathcal{K}} -\theta_a^k \bar{y}_a + \sum_{k \in \mathcal{K}} \sigma^k + \sum_{i \in \mathcal{N}} -\pi_i V(i) \]  
(2.49)

s.t.

\[ \sum_{a \in \mathcal{A}} -\theta_a^k \delta_{a}^q + \sigma^k - \sum_{i \in \mathcal{N}} \sum_{a \in \mathcal{A}} v^k \delta_{a}^q c_{a}^k \pi_i \leq 0, \quad \forall q \in Q(k), \forall k \in \mathcal{K} \]  
(2.50)

\[ \theta_a^k \leq 1, \quad \forall a \in \mathcal{A}, \forall k \in \mathcal{K} \]  
(2.51)

\[ \theta_a^k, \pi_i \geq 0, \quad \forall i \in \mathcal{N}, \forall a \in \mathcal{A}, \forall k \in \mathcal{K}. \]  
(2.52)

We can interpret the \( \theta_a^k \)'s as commodity specific non-negative tolls associated with using block \( a \) and \( \pi_i \) as non-commodity specific non-negative tolls associated with
being classified at node \( i \). Then constraint 2.44 requires that \( \sigma^k \) is less than or equal to the path cost plus the arc tolls \( \theta_a^k \) plus the node tolls \( \pi_i \) associated with each path for commodity \( k \). That is, \( \sigma^k \) is less than or equal to the shortest modified path cost for commodity \( k \). Since \( \sigma^k \) has a positive coefficient in the objective and modified path costs are non-negative by definition, it will always be equal to the shortest modified path cost in an optimal dual solution. Thereby, \( \sigma^k \) is always non-negative.

If a feasible solution exists, then, the optimal solution to the feasibility problem is 0. Therefore, the dual objective for an optimal \( \bar{y}_a \) is non-positive, i.e.,

\[
\sum_{a \in A} \sum_{k \in K} -\theta_a^k \bar{y}_a + \sum_{k \in K} \sigma^k + \sum_{i \in N} -\pi_i V(i) \leq 0. \tag{2.53}
\]

Since \( \theta_a^k, \sigma^k, \) and \( \pi_i \) is a feasible solution to the dual, any feasible blocking plan \( \{y_a\} \) satisfies the following:

\[
\sum_{a \in A} \sum_{k \in K} -\theta_a^k y_a + \sum_{k \in K} \sigma^k + \sum_{i \in N} -\pi_i V(i) \leq 0. \tag{2.54}
\]

The complementary slackness conditions of the problem are:

\[
\theta_a^k \left[ \sum_{q \in Q(k)} f_q^k \gamma_a^q - \tau_a^k \bar{y}_a \right] = 0, \forall k \in K, \forall a \in A \tag{2.55}
\]

\[
\pi_i \left[ \sum_{k \in K} \sum_{q \in Q(k)} \sum_{a \in A} v^k \gamma_a^q c_a - V(i) \right] = 0, \forall i \in N \tag{2.56}
\]

\[
f_q^k \left[ \sum_{a \in A} \theta_a^k \delta_a^q - \sum_{i \in N} \sum_{a \in A} v^k \gamma_a^q \pi_i \right] = 0, \forall q \in Q(k), \forall k \in K \tag{2.57}
\]

\[
\tau_a^k [1 - \theta_a^k] = 0, \forall a \in A, \forall k \in K. \tag{2.58}
\]

Condition 2.56 implies

\[
\sum_{k \in K} \sum_{q \in Q(k)} \sum_{a \in A} v^k f_q^k \gamma_a^q c_a = V(i), \forall i \in N, \tag{2.59}
\]

if \( \pi_i \) is not 0. Then, the dual objective for any feasible blocking plan can be rewritten
as:
\[
\sum_{a \in A} \sum_{k \in K} -\theta^k_y a + \sum_{k \in K} \sigma^k + \sum_{i \in N \setminus \pi_i > 0} \sum_{k \in K} \sum_{q \in Q(k)} \sum_{a, s} -\pi_i v^k f^k q a_s i.
\] (2.60)

Condition 2.57 implies
\[
f^k q^k \pi^k - \sum_{i \in N \setminus \pi_i > 0} \sum_{k \in K} \sum_{q \in Q(k)} \pi_i v^k f^k q a_s i = \sum_{a \in A} f^k q a i, \quad \forall q \in Q(k), \forall k \in K.
\] (2.61)

Summing over commodities and paths, we have
\[
\sum_{k \in K} \sigma^k - \sum_{i \in N \setminus \pi_i > 0} \sum_{k \in K} \sum_{q \in Q(k)} \pi_i v^k f^k q a_s i \geq \sum_{k \in K} \sum_{q \in Q(k)} \sum_{a \in A} f^k q a d q.
\] (2.62)

When the feasibility problem has a non-zero objective value, there exists at least a set of \( \tau^k_a \) and \( \theta^k_a \), whose values are positive from 2.58. Thus, it follows from 2.43, 2.44 and 2.62 that:
\[
\sum_{k \in K} \sigma^k - \sum_{i \in N \setminus \pi_i > 0} \sum_{k \in K} \sum_{q \in Q(k)} \pi_i v^k f^k q a_s i > 0.
\] (2.63)

Therefore, \( \sum_{a \in A} \sum_{k \in K} -\theta^k_y a \) has to be negative in order to satisfy 2.54. Since \( y_a \) is binary and \( \theta^k_a \) is non-negative, \( \sum_{a \in A} \sum_{k \in K} -\theta^k_y a \) \( \leq 0 \) implies \( \sum_{a \in A} \sum_{k \in K} \theta^k_y a \geq 1 \), i.e., at least one of the blocks whose associated dual values are positive has to be added. For those blocks with \( \overline{y}_a = 1 \), the corresponding \( \tau^k_a \) is always equal to 0 because \( \sum_{q \in Q(k)} f^k_q \) is not greater than 1. Therefore, discarding the constraints 2.44 with \( \overline{y}_a = 1 \) will not affect problem feasibility and \( \sum_{a \in A} \sum_{k \in K} \theta^k y a \geq 1 \) is still satisfied. Therefore, the blocks to be added are among the blocks not selected in the block subproblem that have positive \( \theta \) value.

To resolve infeasibility, at least one block in \( F \) has to be selected in the block subproblem. Therefore, we add the feasibility cut to the block subproblem and resolve it. In this way, after a finite number of iterations, we can locate a feasible solution if one exists. Clearly, this approach is very involved in that the feasibility problem and the block subproblem have to be solved several times to generate one feasible
solution.

The selection of the appropriate feasible solution generation depends on the problem characteristics. When the car volume constraints are not tight, the first simple heuristic is preferred; otherwise, the second procedure is more appropriate.

2.3.5 Dual Ascent Processing

To speed up the convergence of the subgradient optimization, we would like to start with multipliers that give a good lower bound. To that end, we have apply a dual-ascent approach for the blocking problem that is an extension of an approach developed by Balakrishnan, et al. (1989) for the uncapacitated fixed-charge network design problem (UFCND). Our approach generates an initial set of multipliers for the subgradient optimization. A detailed discussion of applying dual ascent in the blocking problem can be found in Barnhart, et al. (1997).

Letting \( \lambda^k_a, \sigma^k, \beta_i \) and \( \pi_i \) be the dual variables for constraints (2.20), (2.21), (2.22) and (2.23) of \texttt{B.PATH}, respectively, we obtain the following dual to its LP relaxation:

\[
\begin{align*}
\text{max} & \quad \sum_{k \in K} \sigma^k - \sum_{i \in N} (B(i)\beta_i + V(i)\pi_i) \\
\text{s. t.} & \quad \sigma^k - \sum_{a \in A} [\delta_a^k \lambda^k_a + v^k \delta_a^k \sum_{i \in N} \xi^a_i \beta_i] \leq PC^k_q v^k \quad \forall q \in Q(k), \forall k \in K \\
& \quad \sum_{k \in K} \lambda^k_a - \sum_{i \in N} \xi^a_i \beta_i \leq 0 \quad \forall a \in A \\
& \quad \lambda^k_a, \beta_i, \pi_i \geq 0
\end{align*}
\]

The intuitive interpretation of the dual variables are: \( \lambda^k_a \) is a commodity specific toll associated with using arc \( a \) for commodity \( k \); \( \pi_i \) is the toll associated with flow being classified at a yard \( i \); Similarly, \( \beta_i \) is the toll associated with building a block at yard \( i \). Then the first constraint set (2.65) requires that \( \sigma^k \) is no more than any path costs modified by adding various tolls for commodity \( k \). That is, \( \sigma^k \) is less than or equal to the shortest modified path cost for commodity \( k \). Since \( \sigma^k \) has a positive
coefficient in the objective, it will always be equal to the shortest modified path cost in an optimal dual solution. The second constraint set (2.66) requires that the sum of the tolls $\lambda_a^k$ charged for all commodities on arc $a$ must be less than or equal to the sum of the arc selection tolls at the originating yard $i$ of arc $a$.

Barnhart, et al. present an approach that begins with a dual feasible solution, then attempts to improve that solution by first adjusting the $\lambda_a^k$'s only and then by adjusting the $\lambda_a^k$'s and $\beta_i$'s simultaneously.

Barnhart, et al. prove that the following is a dual feasible solution to the blocking problem, B_PATH where $C_a = 1$.

$$\beta_i = vol_i \quad \forall i \in N$$  \hspace{1cm} (2.68)

$$\lambda_a^k = \sum_{i \in N} \xi_i^a \beta_i \quad \forall k \in K_h, a = a(k)$$  \hspace{1cm} (2.69)

$$\lambda_a^k = v^k \quad \forall k \in K_l, a = a(k)$$  \hspace{1cm} (2.70)

$$\lambda_a^k = 0 \quad \text{otherwise}$$  \hspace{1cm} (2.71)

$$\pi_i = 0 \quad \forall i \in N$$  \hspace{1cm} (2.72)

$$\sigma^k = v^k + \beta_o(k) \quad \forall k \in K_h$$  \hspace{1cm} (2.73)

$$\sigma^k = 2v^k \quad \forall k \in K_l,$$  \hspace{1cm} (2.74)

where $K_h$ and $K_l$ are the sets of high and low volume commodities defined in Barnhart, et al. (1997).

Dual solutions can be modified by applying the procedures described in Barnhart, et al. (1997). A summary of the procedure is as follows:

First we fix the $\beta_i$'s at their initial values and apply the algorithm of Balakrishnan, et al. to get ascent by adjusting the $\lambda_a^k$'s. Then we consider increasing the $\beta_i$ for each node $i$, one at a time. For node $i$, we attempt to construct a set $K'$ of $B(i) + 1$ commodities that originate at node $i$ and share no tight arcs out of $i$ on their shortest paths. To identify a candidate set, we use the following heuristic.

1. Let $K' = \emptyset$, let $A' = \emptyset$ be the set of all tight arcs originating at $i$ in a shortest path for some member of $K'$, consider the commodities originating at node $i$ in
non-increasing order by volume.

2. Select the next commodity \( k \). If no more commodities remain, STOP.

3. If for any of the shortest paths for \( k \) the arc leaving \( i \) is tight and is in \( A' \), go to 2.

4. Otherwise, add \( k \) to \( K' \). For every tight arc leaving \( i \) add it to \( A' \).

5. Go to 2.

We consider the higher volume commodities first, since they will likely have fewer shortest paths for a given value of \( \beta_i \). In fact, if \( v^k > \beta_i \) the only shortest path for the commodity is the direct OD arc. This is true since any path other than the direct OD arc has cost at least \( 2v^k \) while the direct OD arc has cost \( v^k + \beta_{\sigma(k)} < 2v^k \). We facilitate the check for tight arcs on shortest paths by maintaining a set of distance labels for each commodity and recording the amount of slack remaining in the constraint (2.66) for each arc.

If we identify a large enough set of commodities, we increase the value of \( \beta_i \), and \( \lambda^k_\alpha \) for each arc leaving \( i \) on a shortest path for \( k \in K' \). The duals are increased until a new shortest path is introduced for one of the commodities or the slack on a previously loose arc leaving \( i \) becomes zero. The latter case can occur when two or more commodities in \( K' \) have a common loose arc on one of their shortest paths.

We repeat the ascent for each node until we are no longer able to identify a large enough set of commodities.

2.4 Summary

In this chapter, we formulate the railroad blocking problem as a network design problem with maximum degree and flow constraints on the nodes and propose a heuristic Lagrangian Relaxation approach to solve the problem. The new approach decomposes the complicated mixed integer programming problem into two simple subproblems, enhancing the model capability of solving large-scale problems from
the real world. A set of inequalities are added to one subproblem to tighten the lower bounds and facilitate generating feasible solutions. Subgradient optimization is used to solve the Lagrangian Dual. An advanced dual feasible solution is generated to speed up the convergence of the subgradient method. In the next chapter, we will test our model and solution approach on blocking problems from a major railroad.
Chapter 3

Railroad Blocking with Deterministic Data: A Case Study

3.1 Introduction

In this chapter, we illustrate the applications of the model and algorithms discussed in Chapter 2 through solving the problems from a major railroad.

The major challenge in solving railroad blocking problems is the large size of real world problems. A typical rail network consists of thousands of yards, links and millions of potential blocks, over which thousands of commodities have to be delivered from the origins to the destinations. Solving the problem in full size would demand large storage space and a long solution time; therefore, certain levels of problem aggregation are pursued to generate low cost blocking plans in reasonable time and within available storage space.

Similar ideas of problem aggregation are widely adopted and researched in urban transportation planning and analysis. The detailed network for a moderate size city could be too large and complex for “sophisticated” models, such as the equilibrium traffic assignment model. Hearn (1978), Friesz (1985) and Hazelton and Polak (1994) provide surveys of network aggregation practices, and present related mathematical models. In order to reduce problem size, detailed city networks are aggregated to include much less detailed levels. Usually, the aggregation is performed by combining
“small” nodes in a region into a “big” node. In this way, a large size city network is reduced into a small sketch-planning network. For example, Bovy and Jansten (1983) report that a fine level network (including all streets) with 1,286 zones and 12,871 links can be consolidated into a course level network (including only arterials) with only 47 zones and 544 links. Various research (e.g., Bronzini [1981], Walker [1992], Bovy and Jansten, and Hazelton and Polak) has been conducted to evaluate the effects of network aggregation by comparing the solution with actual flows. In particular, Bovy and Jansten show that an increase in the level of detail yields better results. In the experiments of evaluating assignment models on the networks with different levels of details, they found that the outcomes on the medium level network (covering arterials and collectors) are more improved compared to the outcomes on the coarse level network (covering only arterials). However, the assignment outcomes improve only slightly on the fine level network (covering all streets) even though the computational effort increases dramatically, indicating that only marginal improvement could be obtained beyond a certain level of network details.

The aggregation approaches in urban transportation can be applied to blocking problems to reduce network size. Additionally, due to specific feature that blocking problems involve freight and not passengers, additional aggregation approaches can be applied. Since freight can be assigned to a certain path, in blocking problems, commodity and route consolidation is possible. We describe these consolidation techniques in the following sections.

This chapter focuses on evaluating the trade-off between problem size and solution quality for various aggregation schemes in order to pursue appropriate ways of aggregation under run time and storage restrictions. Besides investigating the trade-off between problem size and solution quality in problem aggregation, this chapter also:

1. discusses blocking plan generation in real time;
2. evaluates the efficiency of model enhancements; and
3. shows the effects of volume constraints.
3.2 Problem Data

To evaluate our new model, we tested it on blocking problems from a major railroad. The first test problem represents a high fidelity network with 8,944 commodities, 1,050 nodes, and 1,547 undirected links. The second network is an aggregate strategic network with 7,170 commodities, 116 nodes, and 170 undirected links. Except for the car volume evaluation case study where the aggregate strategic network is used, all case studies use the data of the high fidelity network. We use CPLEX version 4.0 to implement our algorithm on a SGI workstation with 256 Megabytes of RAM, running IRIX version 6.2.

There are five types of yards in the railroad network—system, region, interchange, field and local yards. They have different roles in traffic classification. Most system and region yards and some local yards can classify both originating and through traffic (traffic that neither originates nor terminates at that location). Most interchange and field yards and some local yards can only classify the originating traffic. A small portion of local and field yards cannot classify any traffic. With respect to the above differences, we divide the yards into three groups—regular classification, end classification and non-classification. A regular classification yard can classify both through traffic and originating traffic, an end classification yard can only classify originating traffic, and a non-classification yard can only receive terminating traffic. The differences in the classification functions of the yards have a big impact on the formulation and associated problem size. For instance, the exclusion of all blocking paths with either end classification yards or non-classification yards as intermediate nodes greatly reduces the number of blocking paths in the flow subproblem. The number of variables in the block subproblem is also reduced since no (intermediate) block arcs originating at an end classification or a non-classification yard need be included.

Besides the number of classifications, railroads are also concerned with the lengths of physical routings in the network. To avoid circuitous routings, the railroad considers for each commodity only the paths with distance not exceeding 150% of the
shortest one. Such criteria in route selection allows us to focus on only a few routes for each commodity.

To preprocess the input data, we first generate legitimate physical routes that do not exceed 150% of the shortest path distance for that shipment. We consider only the blocks that appear in a subsequence of one of these routings for each commodity. This preprocessing allows us to reduce the total number of potential blocks significantly.

3.3 Problem Aggregation Schemes

The blocking problem is a special case of network design, a NP-hard problem. To exacerbate this, blocking networks of major railroads are very large. Hence, it is difficult to apply optimization-based approaches to the entire blocking network since storage requirements and necessary computational efforts are very large. Instead, we apply different schemes of problem aggregation to consolidate the problem into a set of smaller problems which can be loaded and solved on a workstation-class computer. The schemes of problem aggregation are node-based, route-based and commodity-based, respectively.

Problem aggregation involves certain assumptions that can result in an optimal solution to the aggregate problem that is not optimal for the original network. Problem aggregation then, involves a trade-off between solution time and optimality of the solution. The optimal solution to the detailed, or disaggregate, problem gives the best blocking plan. However, solving this problem to optimality requires massive amounts of computer memory and computing time. Thus in practice, working with the disaggregate problem may result in a poor quality solution if computational resources are limited since we never reach the optimal solution in the time available. In contrast, more aggregate blocking problems can be solved relatively easily but optimality of the original problem may be dramatically compromised. In this section, we attempt to evaluate the trade-off between solution time and optimality of solution under different aggregation schemes. The comparisons in this section will help us to determine an appropriate aggregation scheme for large-scale railroad blocking problems.
3.3.1 Node-Based Aggregation

We follow a three-step heuristic procedure to generate blocking plans for large networks. In the first step, we aggregate the network nodes and arcs, creating a smaller network using certain rules and assumptions. Second, we use our solution approach to generate a blocking plan for the aggregate blocking problem. Finally, we heuristically adjust the plan locally and expand it to the original network.

We perform node and network consolidation based on the roles and blocking capacity (the number of blocks that can be built) of yards in the system. Among the five types of classification yards, system and region yards are major classification yards in the system or in certain regions, and these yards usually have large blocking capacities. Local and field yards serve corresponding local areas, and the blocking capacities at these yards may vary widely depending on the range and activities in the local areas. In general, most field yards and the majority of local yards possess limited blocking capacities. Interchange yards are on the boundary of a railroad’s network, and these yards gather originating traffic at the boundary to be forwarded into the railroad’s network.

Given the functionality of different types of classification yards in the system, we evaluate a hub-and-spoke network structure, where the major classification yards (referred to as terminals) are hubs performing many classifications, and the small local yards are satellite nodes collecting and forwarding local traffic to the major terminals.

An aggregate network consists of only the major terminals, with small classification yards consolidated into other terminals based on the local movements in the current operation. According to their roles and blocking capacity, the terminals include all system and region yards and some high blocking capacity (greater than or equal to a threshold) local and field yards. The remaining classification yards are consolidated into terminals based on the current blocking plan. Commodities are consolidated by changing the origin (destination) of each commodity to the first (last) major terminal it visits in the current blocking plan. This aggregation essentially
fixes the local moves in the current blocking plan. Following network aggregation, the blocking problem is solved on the aggregate network with only the consolidated terminals and traffic.

### 3.3.2 Route-Based Aggregation

In railroad blocking problems, another approach that can be used to control problem size is to limit the number of routes through the physical network that will be allowed for each commodity. This approach limits both the number of flow variables \( f_{ij}^k \) and the number of block variables \( y_a \). Each physical route gives rise to a number of potential blocking paths since any subsequence of the terminals visited on a physical route is a potential blocking path. To some extent, the total number of potential blocks in the problem depends on the number of routes considered for each commodity. For example, the total number of blocks will be small if we limit block selection based only on the current routes, and this number will increase significantly as more routes are considered. In our work, the routes considered include the routes currently adopted by the railroad and some routes with distances less than 150% of the shortest path distance. In this way, we limit the number of potential blocks, which, otherwise, could lead to intractability for the network in our problem.

In the most aggregate case we allow only what we refer to as current block sequences. This aggregation is more restrictive than allowing only the current routes. In cases where current block sequences are used we allow commodities to use only their current routes and we allow the commodity to be classified only at terminals where it is classified in the current blocking plan. That is, in our final solution, a commodity may be classified at all the same yards where it is handled in current practice, or at some subset of those yards. The merits of using current block sequences as the basis for block selection are as follows:

1. The potential block set is reduced dramatically. Unlike physical routes which include dozens of yards for each OD pair, a block sequence typically contains only a few yards (in the range of 2 to 6) where the shipment is classified. Since
potential blocks are generated from possible pairs of yards, the potential block set is much smaller when block sequences are used.

2. Our model solution demonstrates the opportunities for improvement over current operations, even under this restrictive setting. By assembling blocks differently, the model solution may outperform current operations and suggest opportunities for improving the current blocking plan.

However, allowing classification of each commodity only at yards in its current block sequence might overlook opportunities for making efficient blocks at other yards, resulting in low quality solutions. In the following case study, we show that such negative effects are moderate.

3.3.3 Commodity-Based Aggregation

In the full-size problem, we have 8,944 commodity OD pairs, among which over 40% have car volume less than 10 cars per day, accounting for less than 7% of total traffic on the network. Due to their low volume, these commodities are unlikely to have dedicated blocks in an optimal solution. Therefore, combining these low volume commodities with other commodities might not affect solution quality significantly. On the other hand, limiting the size of the commodity set can greatly reduce problem size and have a significant effect on the computational effort required to find a good solution. Applying this approach to the full-size problem, which is too large to load into memory on our computer, we reduce the number of commodities to 5,588, thus obtaining a problem that can be loaded and solved.

To perform commodity-based aggregation, we represent certain low volume commodities as combinations of other commodities. For example, consider one commodity with origin O and destination A; another commodity with origin A and destination D; and finally, a low volume commodity with origin O, destination D, and volume v(O,D). We represent this O-D commodity by increasing the volume of both the O-A and A-D commodities by v(O,D) if the length of the O-A-D route does not exceed
150% of the shortest O-D route. If there are multiple commodity pairs whose combined routes connect O-D, e.g. O-B and B-D or O-C and C-D, we pick the one with the shortest distance. If, however, there is no pair of commodities that can be used to combine the small volume commodity, or, if the combined routes exceed 150% of the shortest O-D route, we keep the small volume commodity.

3.3.4 Comparison of Aggregation Schemes

Among the above aggregation schemes, node-based aggregation produces the most dramatic reduction in problem size since the number of commodities and blocks are both reduced when nodes are consolidated. Route-based aggregation affects the block domain only. Commodity-based aggregation reduces problem size by condensing the commodity set.

Each aggregation has certain negative impacts on solution quality due to the assumptions made about local movements (in node-based aggregation), route selection (route-based aggregation) and small volume traffic consolidation (commodity-based aggregation).

3.4 Problem Aggregation Case Studies

3.4.1 Implementation Details

In the first case study, we investigate the solution quality, under different run time limits, of different levels of node-based aggregation. We consider only the first four shortest paths as possible routes for each commodity. All the routes considered have distances less than 150% of the shortest path distance for the commodity. In some cases, there are fewer than four such routes.

In the second case study, we investigate the effects of route-based aggregation. To limit the computational resources needed to perform the study, we use one of the medium-level problems from the first case study. We vary the number of routes considered, with the most restrictive case allowing only current block sequences.
In the third case study, we compare the solution quality of a problem using the original network and commodity-based aggregation with the best solution obtained using node and route-based aggregation.

Finally, we compare the best solution from our model with current operations in the railroad to illustrate potential improvements.

In order to evaluate effects of various levels and types of aggregation, we consider several characteristics of the resulting problems. These include the number of potential blocks, the total computational time and the time necessary to solve the sub-problems in each iteration, and the solution quality. The quality of the solution under each level of aggregation can be evaluated based on two gaps; namely, the gap (Gap One) between the upper and lower bounds on the optimal solution to the aggregated problem and the gap (Gap Two) between the best known solution to the aggregated problem mapped back to the original network and the best known feasible solution to the original problem. To evaluate the trade-off between computational effort and solution quality, we compare the solutions obtained using models with varying levels of aggregation under a number of CPU time limits.

### 3.4.2 Node-Based Aggregation Case Study

The most aggregate network we consider consists only of system and region yards. Adding local, field and interchange yards with large blocking capacity (greater than or equal to a prespecified threshold) as terminals, we may obtain less aggregate blocking networks. Table 3.1 summarizes the different levels of aggregation we investigate and the resulting problem sizes. Case I is the most aggregate network, consisting only of system and region yards. Cases II to V are less aggregate networks where nodes include not only system and region yards but also some local, field and interchange yards with blocking capacity greater than or equal to the threshold specified in the parentheses next to each case. For comparison, we also list the size of the original network in Table 3.1.

Table 3.1 shows that most yards are non-terminals with limited blocking capacities. For example, more than 70% of the yards have blocking capacity lower than 3,
Table 3.1: Levels of Node-Based Aggregation and Problem Sizes

<table>
<thead>
<tr>
<th>Problem (threshold)</th>
<th>Node</th>
<th>Commodity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case I</td>
<td>24</td>
<td>458</td>
</tr>
<tr>
<td>Case II (13)</td>
<td>53</td>
<td>1,232</td>
</tr>
<tr>
<td>Case III (8)</td>
<td>109</td>
<td>2,841</td>
</tr>
<tr>
<td>Case IV (5)</td>
<td>179</td>
<td>4,250</td>
</tr>
<tr>
<td>Case V (3)</td>
<td>296</td>
<td>5,677</td>
</tr>
</tbody>
</table>

indicating that their function is to serve local traffic. Hence, assigning these small yards to the major terminals greatly reduces the problem size. However, problem simplification has its price; specifically, we might not achieve the optimal solution to the original problem because the local movements (from non-terminal yards to the hub terminals and vice versa) resulting from node consolidation might be different from the optimal local movements on the original network. Consequently, we test different levels of aggregation to evaluate the trade-off between fidelity and solution quality.

Table 3.2 presents the performance under different levels of aggregation when the total run time is fixed at 105 seconds, when Case I is solved to optimality, and 0.5, 1, 2, 3, 5 and 10 hours respectively. In the table, $T_{Block}$ and $T_{Flow}$ represent the average run time (in seconds) in block and flow subproblems, respectively; Number of Blocks stands for the number of potential blocks in each problem; Storage Space describes the computer memory needed to load and solve the problem; Solution is the best feasible solution after mapping back to the high fidelity network.

From the computational results, we can observe that network aggregation results in blocking problems that are easy to solve for low fidelity networks (e.g., Cases I and II), with Case I solved to optimality in less than 2 minutes. In contrast, no feasible solution is found with the same computational effort for the medium and high fidelity problems (Cases III to V). In particular, it takes more than an hour to generate the first feasible solution for the highest fidelity problem (Case V).

Figures 3-1a to 3-1g show the Gap Two values for the aggregation problems under different run time limits. We can observe that the curves in these figures typically
<table>
<thead>
<tr>
<th></th>
<th>Case I</th>
<th>Case II</th>
<th>Case III</th>
<th>Case IV</th>
<th>Case V</th>
</tr>
</thead>
<tbody>
<tr>
<td>T_Block (sec.)</td>
<td>0.8</td>
<td>7</td>
<td>83</td>
<td>477</td>
<td>1,925</td>
</tr>
<tr>
<td>T_Flow (sec.)</td>
<td>0.7</td>
<td>8</td>
<td>88</td>
<td>461</td>
<td>1,860</td>
</tr>
<tr>
<td>Number of Blocks</td>
<td>522</td>
<td>1,947</td>
<td>6,343</td>
<td>15,971</td>
<td>28,076</td>
</tr>
<tr>
<td>Storage Space (MB)</td>
<td>4</td>
<td>7</td>
<td>24</td>
<td>46</td>
<td>85</td>
</tr>
<tr>
<td>105 Sec. Solution</td>
<td>713,742</td>
<td>688,179</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Gap One</td>
<td>0%</td>
<td>10.97%</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Gap Two</td>
<td>41.41%</td>
<td>36.34%</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>1/2 Hour Solution</td>
<td>713,742</td>
<td>669,363</td>
<td>617,913</td>
<td>775,404</td>
<td>N/A</td>
</tr>
<tr>
<td>Gap One</td>
<td>0%</td>
<td>0.540%</td>
<td>7.25%</td>
<td>46.9%</td>
<td>N/A</td>
</tr>
<tr>
<td>Gap Two</td>
<td>41.41%</td>
<td>32.62%</td>
<td>22.42%</td>
<td>53.62%</td>
<td>N/A</td>
</tr>
<tr>
<td>1 Hour Solution</td>
<td>713,742</td>
<td>669,290</td>
<td>613,104</td>
<td>703,777</td>
<td>N/A</td>
</tr>
<tr>
<td>Gap One</td>
<td>0%</td>
<td>0.36%</td>
<td>5.27%</td>
<td>30.04%</td>
<td>N/A</td>
</tr>
<tr>
<td>Gap Two</td>
<td>41.42%</td>
<td>32.60%</td>
<td>21.47%</td>
<td>39.43%</td>
<td>N/A</td>
</tr>
<tr>
<td>2 Hour Solution</td>
<td>713,742</td>
<td>669,255</td>
<td>610,251</td>
<td>642,650</td>
<td>756,903</td>
</tr>
<tr>
<td>Gap One</td>
<td>0%</td>
<td>0.037%</td>
<td>4.04%</td>
<td>29.10%</td>
<td>57.75%</td>
</tr>
<tr>
<td>Gap Two</td>
<td>41.41%</td>
<td>32.59%</td>
<td>20.90%</td>
<td>27.32%</td>
<td>49.96%</td>
</tr>
<tr>
<td>3 Hour Solution</td>
<td>713,742</td>
<td>669,249</td>
<td>608,799</td>
<td>581,516</td>
<td>756,903</td>
</tr>
<tr>
<td>Gap One</td>
<td>0%</td>
<td>0.029%</td>
<td>3.02%</td>
<td>10.67%</td>
<td>57.75%</td>
</tr>
<tr>
<td>Gap Two</td>
<td>41.41%</td>
<td>32.59%</td>
<td>20.62%</td>
<td>15.21%</td>
<td>49.96%</td>
</tr>
<tr>
<td>5 Hour Solution</td>
<td>713,742</td>
<td>669,240</td>
<td>607,605</td>
<td>573,333</td>
<td>756,903</td>
</tr>
<tr>
<td>Gap One</td>
<td>0%</td>
<td>0.019%</td>
<td>2.20%</td>
<td>8.57%</td>
<td>57.38%</td>
</tr>
<tr>
<td>Gap Two</td>
<td>41.41%</td>
<td>32.59%</td>
<td>20.38%</td>
<td>13.59%</td>
<td>49.96%</td>
</tr>
<tr>
<td>10 Hour Solution</td>
<td>713,742</td>
<td>669,231</td>
<td>606,366</td>
<td>565,425</td>
<td>658,296</td>
</tr>
<tr>
<td>Gap One</td>
<td>0%</td>
<td>0.010%</td>
<td>1.43%</td>
<td>4.88%</td>
<td>38.93%</td>
</tr>
<tr>
<td>Gap Two</td>
<td>41.41%</td>
<td>32.59%</td>
<td>20.13%</td>
<td>12.02%</td>
<td>30.42%</td>
</tr>
</tbody>
</table>

Table 3.2: Computational Results for Node-Based Aggregation
Figure 3-1a: Best Feasible Solutions at 105 seconds

Figure 3-1b: Best Feasible Solutions at 1/2 Hour

Figure 3-1c: Best Feasible Solutions at 1

Figure 3-1d: Best Feasible Solutions at 2 Hours

Figure 3-1e: Best Feasible Solutions at 3 Hours

Figure 3-1f: Best Feasible Solutions at 5 Hours

Figure 3-1g: Best Feasible Solutions at 10 Hours

Figure 3-1: Best Feasible Solutions under Different Run Time Limits
exhibit a U-shape, with the lowest cost (or lowest Gap Two value) blocking plan generated by the problem at the bottom of each curve. This suggests the best solution is usually generated by some intermediate level of aggregation under each time limit. For the problems on the left of the bottom of the curve, the level of aggregation is excessive because the fast convergence (low Gap One value) due to problem simplicity is outweighed by the solution quality compromised through aggregation. For the problems on the right of the bottom of the curve, however, the level of aggregation is not sufficient to reduce the problem to a tractable size that converges to a satisfactory solution under the specific run time limits. We can see that the size of the problem that gives the best solution changes as computing time restrictions vary. When the run time is 105 seconds, the best solution is generated by Case II and no feasible solution can be identified on higher fidelity networks in such a short time. As run times are increased to between 0.5 and 2 hours, the best solutions are generated by the medium fidelity (Case III) network. As solution time increases to 3 and 10 hours, the Case IV network provides the best results. Case V, on the other hand, will take more than 10 hours to converge to a better solution.

Figures 3-2a to 3-2d show the Gap Two values for Cases II to V under different run time restrictions. (Gap Two for Case I does not change after the aggregate problem reaches optimality in 105 seconds.) From the figures, we can see that the gaps for Cases II to IV decrease as run time increases, and the rate of decrease decelerates as run time restrictions are higher. For example, Gap Two values decrease 0.03% and 2.29% only from 0.5 hour to 10 hours for Cases I and II, respectively. The pattern shown in these figures indicates that the Gap Two values reach a limit and cease to decrease when the run time is long enough. The limit measures the solution quality compromised in node aggregation, compared to the best solution obtained on the disaggregate network. This suggests that our assumption of hub-and-spoke structure from which our node aggregation was performed does not fully capture actual railroad blocking processes. Under the hub-and-spoke assumption, every local yard only gathers and distributes traffic for the major terminals, and the classifications and blocking activities are performed only at the major terminals. Due
to capacity limits and congestion considerations, however, it is not efficient to perform all classifications at the major terminals. Besides their main function of gathering and distributing local traffic, some local yards may classify some traffic and build some blocks to other terminals. Since the hub-and-spoke system does not recognize the classification functions of the local yards, some opportunities of designing better (lower cost) blocking plans are ignored. For example, 997 yards are considered as small local yards in Case II. Among these yards, over 50 yards have blocking capacity over 8 blocks per day. Excluding the blocks built to the major terminals, these small local yards can still build some blocks to other terminals based on their residual capacities. Ignoring this possibility of building blocks from the local yards causes the gap between the solution to the aggregate and disaggregate problems.

From these figures, we notice that the gap decreases as the level of network fidelity increases. This occurs because the small local yards to be consolidated in the more disaggregate network have lower blocking capacities. As a result, the opportunities to
build blocks using the residual capacities at the local yards diminish as the network becomes more disaggregate, and the solutions to the more disaggregate problems become closer to the best possible solutions to the original problem. Unlike the other cases whose solution values improve as the run time limit increases, the solution values for Case V do not change even with significant increases in the run time limit. Specifically, the solution value remains the same throughout the run time limit range of 2 to 5 hours. This occurs because the run time for one iteration is very long in Case V, since the network is more detailed— it takes over one hour to run one iteration. Therefore, no solution is available when the run time restriction is under 1 hour. Further, since the feasible solution is not updated, the Gap Two value stays the same as run time increases from 2 to 5 hours.

From the above comparisons and analyses, we find that the level of node-based aggregation has a major impact on solution quality. The gaps (Gap Two) between the model solutions for aggregate problems and the best known feasible solution reflect the solution quality compromised in pursuing node-based aggregation. Given enough computing time, this gap, which is introduced by fixing local movements during problem aggregation, decreases significantly as the level of network fidelity increases.

### 3.4.3 Route-Based Aggregation Case Study

In the above node aggregation case study, we include the first four shortest paths with distance within 150% of the shortest path for each commodity. In order to evaluate the effects of adding or removing potential routes from consideration, we select different numbers of routes on a medium size node-aggregated network (Case III). Table 3.3 summarizes the effects of route-based aggregation on solution quality for the various cases. In the first case we allow only current block sequences to be used. The others allow only the first one, two, four, six, eight and ten shortest paths for each commodity.

We report $Init.LB$, representing the lower bound generated in the first iteration; $Init.FS$, the initial feasible solution that gives the objective value obtained in the first iteration; $T.Block$, the average running time in the block subproblem; $T.Flow$, the
Table 3.3: Computational Results for Route-Based Aggregation

<table>
<thead>
<tr>
<th>Number or Type of Routes Considered</th>
<th>Current</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage Space (MB)</td>
<td>16</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>28</td>
</tr>
<tr>
<td>Init.LB</td>
<td>185,574</td>
<td>212,313</td>
<td>212,163</td>
<td>212,085</td>
<td>212,073</td>
<td>212,073</td>
<td>212,073</td>
</tr>
<tr>
<td>Init_FS</td>
<td>244,284</td>
<td>330,868</td>
<td>336,717</td>
<td>348,574</td>
<td>354,237</td>
<td>336,717</td>
<td>326,254</td>
</tr>
<tr>
<td>T.Block(sec)</td>
<td>8</td>
<td>45</td>
<td>62</td>
<td>83</td>
<td>104</td>
<td>111</td>
<td>114</td>
</tr>
<tr>
<td>T.Flow(sec)</td>
<td>11</td>
<td>50</td>
<td>69</td>
<td>88</td>
<td>109</td>
<td>112</td>
<td>116</td>
</tr>
<tr>
<td>T.total1 (sec)</td>
<td>266</td>
<td>1,520</td>
<td>2,489</td>
<td>4,260</td>
<td>3,560</td>
<td>3,568</td>
<td>3,680</td>
</tr>
<tr>
<td>N.Iter1</td>
<td>14</td>
<td>16</td>
<td>19</td>
<td>22</td>
<td>20</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>LB1</td>
<td>213,556</td>
<td>213,145</td>
<td>212,684</td>
<td>212,328</td>
<td>212,362</td>
<td>212,205</td>
<td>212,177</td>
</tr>
<tr>
<td>FS1</td>
<td>224,214</td>
<td>222,666</td>
<td>223,844</td>
<td>223,149</td>
<td>221,739</td>
<td>221,928</td>
<td>222,192</td>
</tr>
<tr>
<td>Gap One1</td>
<td>4.75%</td>
<td>4.3%</td>
<td>5.0%</td>
<td>4.8%</td>
<td>4.2%</td>
<td>4.4%</td>
<td>4.5%</td>
</tr>
<tr>
<td>Gap Two1</td>
<td>21.54%</td>
<td>21.23%</td>
<td>21.46%</td>
<td>21.33%</td>
<td>21.05%</td>
<td>21.08%</td>
<td>21.14%</td>
</tr>
<tr>
<td>T.total2 (sec)</td>
<td>36,000</td>
<td>36,000</td>
<td>36,000</td>
<td>36,000</td>
<td>36,000</td>
<td>36,000</td>
<td>36,000</td>
</tr>
<tr>
<td>N.Iter2</td>
<td>1,894</td>
<td>378</td>
<td>274</td>
<td>210</td>
<td>169</td>
<td>161</td>
<td>156</td>
</tr>
<tr>
<td>LB2</td>
<td>217,369</td>
<td>215,559</td>
<td>214,927</td>
<td>214,199</td>
<td>213,752</td>
<td>213,873</td>
<td>214,095</td>
</tr>
<tr>
<td>FS2</td>
<td>219,462</td>
<td>217,719</td>
<td>217,282</td>
<td>216,980</td>
<td>216,642</td>
<td>216,651</td>
<td>216,946</td>
</tr>
<tr>
<td>Gap One2</td>
<td>1.0%</td>
<td>1.0%</td>
<td>1.08%</td>
<td>1.28%</td>
<td>1.33%</td>
<td>1.28%</td>
<td>1.31%</td>
</tr>
<tr>
<td>Gap Two2</td>
<td>20.60%</td>
<td>20.25%</td>
<td>20.16%</td>
<td>20.10%</td>
<td>20.04%</td>
<td>20.04%</td>
<td>20.10%</td>
</tr>
</tbody>
</table>

average running time in the flow subproblem; and two sets of performance indicators under two different time limits. The first time limit is selected when the gap between the lower bound and feasible solution closes to within 5% for each case. Thus $T_{Total1}$, the total running time, is different for each case. The second time limit, $T_{Total2}$, is fixed at 10 hours for all cases. At each time limit, we observe $N_{iter}$, the total number of iterations; $LB$, the best lower bound; $FS$, the best feasible solution; $Gap One$, the percentage gap between $FS$ and $LB$; and $Gap Two$, the percentage gap between the best known solution to the aggregated problem mapped back to the original network and the best known feasible solution to the original problem.

From Table 3.3, we can see that including fewer routes (one or two) increases the lower bound on the optimal solution value. However, it is easier to generate a feasible solution for these smaller problems. Note that the quality of solution obtained when Gap One was closed to within 5% is very similar for all cases, but these solutions were found much more quickly for the more aggregate problems. Comparing the solutions after a fairly long run time, say 10 hours, we see that the large problems generate better quality solutions, reducing the number of classifications by over 2,000 in the
best feasible solutions compared to the problem with only current block sequences. However, the relative improvement is moderate, less than 1.5%. This suggests that the effects of route-based aggregation on solution quality is moderate. The reason is that the railroad network is very sparse, so that there are lots of common terminals among the first 10 shortest paths. Meanwhile, most of these common terminals are also included in current block sequences. Compared to the first few shortest paths, the current block sequences are usually much shorter, consisting only of certain subsequences in the first few shortest paths. In our experiment, the average number of terminals on a shortest path is 12, but it is less than 4 on a current block sequence. Since the current block sequences are considered in the existing blocking plan, that reflects the railroad's long operating experience, the terminals on the sequences are likely the efficient locations to build blocks compared to other terminals not in the sequences but on the shortest paths. Investigating the blocking plans based on the shortest paths, we find most yards where classifications are performed are considered in the current block sequences even though significantly more terminals are included in the shortest paths. Comparing the blocking plans with different route-aggregation levels, we find that the common blocks are over 80%. However, due to the differences in the number of terminals considered, the potential block set is dramatically smaller in the case with current block sequences only. Therefore, the run time and storage requirements are greatly reduced in generating a blocking plan based on the current block sequences, as evidenced by the comparisons in Table 3.3.

Table 3.4 shows how appropriate problem aggregation and the corresponding solutions change as the run time limit varies. We ran computational trials for each possible node-based aggregation/route-based aggregation combination. For each run time limit, we report the best aggregation scheme. The aggregation approach is represented by a pair of numbers, the first one referring to the case in the node-based aggregation case study, and the second referring to the number or type of route(s) considered in the route-based aggregation case study.

From the table we can see that

- When current block sequences are available, the best solution is generated using
<table>
<thead>
<tr>
<th>Run Time Limit</th>
<th>With Current Block Sequences Available</th>
<th>Without Current Block Sequences Available</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Aggregation</td>
<td>Solution</td>
</tr>
<tr>
<td>1/2 Hour</td>
<td>V, Current</td>
<td>568,077</td>
</tr>
<tr>
<td>1 Hour</td>
<td>V, Current</td>
<td>533,652</td>
</tr>
<tr>
<td>2 Hour</td>
<td>V, Current</td>
<td>532,116</td>
</tr>
<tr>
<td>3 Hour</td>
<td>V, Current</td>
<td>529,608</td>
</tr>
<tr>
<td>5 Hour</td>
<td>V, Current</td>
<td>528,294</td>
</tr>
<tr>
<td>10 Hour</td>
<td>V, Current</td>
<td>526,017</td>
</tr>
</tbody>
</table>

Table 3.4: Appropriate Level of Aggregation and Solution

the most disaggregate network (Case V) with current block sequences in every instance. This suggests that maintaining a high level of network fidelity can offset the solution quality compromised by limiting the routes considered.

- When current block sequences are not available, the best solutions are generated by the problem using a medium fidelity network (Case III) and the shortest path when the run time limit is under 2 hours, and by using a higher fidelity network (Case IV) with the first four shortest paths when the run time limit is over 3 hours. This suggests that the appropriate level of aggregation needs to be adjusted as the run time limit varies. As the run time limit increases, a higher level of network fidelity and lower level of route-based aggregation is better for generating high quality solutions.

- For each time limit, the best solutions without knowing current block sequences are not as good as the solutions when they are known. This occurs because when we restrict our analysis to include only current block sequences, problem size is sufficiently reduced to allow us to generate satisfactory solutions on the less aggregated (Case V) network in reasonable time.

3.4.4 Commodity-Based Aggregation Case Study

From the above case studies and Table 3.4, we see that the best solutions are generated by networks with minimal node-based aggregation using current block sequences. We also observe that the solution quality improves as the level of fidelity (in terms of node-based aggregation) increases, indicating that the solution quality may be further
<table>
<thead>
<tr>
<th></th>
<th>Case V</th>
<th>High Fidelity with Commodity Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>T_Block(sec.)</td>
<td>48</td>
<td>120</td>
</tr>
<tr>
<td>T_Flow(sec.)</td>
<td>70</td>
<td>780</td>
</tr>
<tr>
<td>Number of Blocks</td>
<td>7,323</td>
<td>11,383</td>
</tr>
<tr>
<td>1/2 Hour Solution</td>
<td>568,077</td>
<td>541,533</td>
</tr>
<tr>
<td>Gap One</td>
<td>25.89%</td>
<td>20.31%</td>
</tr>
<tr>
<td>Gap Two</td>
<td>12.55%</td>
<td>6.79%</td>
</tr>
<tr>
<td>1 Hour Solution</td>
<td>533,652</td>
<td>526,320</td>
</tr>
<tr>
<td>Gap One</td>
<td>12.32%</td>
<td>14.60%</td>
</tr>
<tr>
<td>Gap Two</td>
<td>5.73%</td>
<td>4.28%</td>
</tr>
<tr>
<td>2 Hour Solution</td>
<td>532,116</td>
<td>517,320</td>
</tr>
<tr>
<td>Gap One</td>
<td>9.15%</td>
<td>9.36%</td>
</tr>
<tr>
<td>Gap Two</td>
<td>5.42%</td>
<td>2.49%</td>
</tr>
<tr>
<td>3 Hour Solution</td>
<td>529,608</td>
<td>512,184</td>
</tr>
<tr>
<td>Gap One</td>
<td>7.44%</td>
<td>7.10%</td>
</tr>
<tr>
<td>Gap Two</td>
<td>4.93%</td>
<td>1.47%</td>
</tr>
<tr>
<td>5 Hour Solution</td>
<td>528,294</td>
<td>507,315</td>
</tr>
<tr>
<td>Gap One</td>
<td>6.22%</td>
<td>4.86%</td>
</tr>
<tr>
<td>Gap Two</td>
<td>4.67%</td>
<td>0.51%</td>
</tr>
<tr>
<td>10 Hour Solution</td>
<td>526,017</td>
<td>504,741</td>
</tr>
<tr>
<td>Gap One</td>
<td>4.77%</td>
<td>3.68%</td>
</tr>
<tr>
<td>Gap Two</td>
<td>4.22%</td>
<td>0%</td>
</tr>
</tbody>
</table>

Table 3.5: Computational Results for Commodity-Based Aggregation

improved by solving the problem on the high fidelity (original) network. However, due to memory limits on our computer, we are unable to load and solve the full-sized problem even when we restrict ourselves to current block sequences. In the following, therefore, we apply commodity-based aggregation to consolidate small volume commodities with volumes less than 10 cars/day. It is worth noting that the commodity-based aggregation is effective when there are a large number of small volume commodities that can be consolidated to large volume shipments. Hence, this aggregation scheme might not be effective on an aggregate network because the commodities are consolidated commodities gathered from local yards. The car volumes of these commodities usually are very large.

In Table 3.5, we compare the solution quality of the high fidelity network with commodity aggregation and the best solution from the previous case studies, i.e.,
Case V with current block sequences. From Table 3.5, we can see that the solution quality, under each time limit, can be improved by solving the commodity aggregated problem on the high fidelity network. Indeed, this scheme resulted in the best known solution used to evaluate the solution quality in the above case studies.

3.4.5 Computer Storage Requirements

*Computer Storage Space* is an important factor in determining the size of problem that can be loaded and solved. There are two types of storage space on each computer, *primary storage space* and *secondary storage space*. Primary storage space on a computer refers to *random access memory* (RAM), where programs and data are stored. Secondary storage space on a computer takes the form of a removable floppy disk, fixed hard disk, tape drive, or other device in which information is stored on a series of magnetic charges on some medium. Information cannot be moved directly from secondary storage devices to the *Central Processing Unit* (CPU), it must first be moved to the computer's primary storage, increasing total processing time dramatically. Even though secondary storage might be used to dynamically store and retrieve information to load a large problem, excessive time in storing and retrieving information from secondary storage makes the solution time extremely long. Hence, it is not practical to use secondary space to store the information that has to be retrieved frequently, and thereby the size of problem that each computer can solve in reasonable time is ultimately determined by available primary storage on the computer.

In our case studies, we refer to *computer storage requirements* as the primary storage space needed to load programs and data in each iteration of subgradient optimization. We use secondary storage for pre- and post-processing, but we do not use secondary storage for storing information and programs in subgradient optimization iterations because of its slowness.

In our experiments, computational storage space is assumed to be fixed. In reality, the storage space depends on the specific computer to be used, which in turn determines the appropriate aggregation schemes to be performed on the problem. Summarizing computational results in our experiments, Table 3.6 shows how the ap-
Table 3.6: Appropriate Level of Aggregation and Solution Value with Storage Constraints

<table>
<thead>
<tr>
<th>Run Time Limit</th>
<th>32 MB</th>
<th>64 MB</th>
<th>128 MB</th>
<th>256 MB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Aggregation</td>
<td>Solution</td>
<td>Aggregation</td>
<td>Solution</td>
</tr>
<tr>
<td>1/2 Hour</td>
<td>IV, C, 30%</td>
<td>592,731</td>
<td>V, C, 0</td>
<td>568,077</td>
</tr>
<tr>
<td>1 Hour</td>
<td>IV, C, 30%</td>
<td>588,437</td>
<td>V, C, 0</td>
<td>533,652</td>
</tr>
<tr>
<td>2 Hour</td>
<td>IV, C, 30%</td>
<td>586,278</td>
<td>V, C, 0</td>
<td>532,116</td>
</tr>
<tr>
<td>3 Hour</td>
<td>IV, C, 30%</td>
<td>584,739</td>
<td>V, C, 0</td>
<td>529,608</td>
</tr>
<tr>
<td>5 Hour</td>
<td>IV, C, 30%</td>
<td>583,422</td>
<td>V, C, 0</td>
<td>528,294</td>
</tr>
<tr>
<td>10 Hour</td>
<td>IV, C, 30%</td>
<td>582,460</td>
<td>V, C, 0</td>
<td>528,017</td>
</tr>
</tbody>
</table>

Figure 3-3: Solution Values under Storage Space Constraints

Appropriate problem aggregation and the corresponding solution value changes as the available computer storage space varies under different run time limits. For each run time limit, we report the best aggregation scheme under different storage spaces of 32 MB, 64 MB, 128 MB and 256 MB. The aggregation approach is represented by a set of numbers or letters, the first one referring to the case in the node-based aggregation case study, the second referring to the number or type of route(s) considered in the route-based aggregation case study, and the third one referring to the level of commodity-based aggregation, i.e., the percentage of small volume commodities consolidated. C represents current block sequences and F represents the high fidelity network.
Figure 3-3 shows how the solution values change as run time and storage space increases. From the figure, we can see that higher fidelity problems can be loaded and solved to generate better (lower cost) solutions as available computer storage space increases and that the solution quality can be further improved as run time limits increase. This suggests that, besides run time restrictions, the solution quality also depends on the available hardware used to generate the blocking plan. The solution quality can be much improved by using more powerful computers. From the comparisons, we notice that the solution quality increases significantly as storage space and run time increase from 32 MB to 64 MB and 0.5 hour to 2 hours. However, the improvement is moderate as the storage space and run time are further increased, as evidenced by the minor solution change as the storage space and run time double from 128 MB and 5 hours to 256 MB and 10 hours. This suggests that solution quality approaches a limit and the marginal contributions of additional space and time diminishes as storage space and run time are large enough to load and solve problems on the high fidelity network. Given the profiles of solution values in figure 3-3, it is reasonable to project that the solution quality will not be increased much even if the storage space and run time are further increased (e.g. to 1 gig bytes and 100 hours). Additional storage space and run time enables more disaggregate problems to be solved, and the most disaggregate problem is the original problem without any aggregation. Compared to the most disaggregate problem, the problem yielding the lowest cost solutions with 256 MB storage and 10 hour run time considers only current block sequences and aggregates small volume commodities on the original network (i.e., no node-based aggregation). In earlier experiments, we have shown that consolidating small volume commodities and considering only current blocking sequences have limited effects on solution quality. Therefore, the solution quality is not expected to improve significantly even if large storage space and a long solution time are available.
<table>
<thead>
<tr>
<th></th>
<th>Railroad’s Plan</th>
<th>Our Plan</th>
<th>Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Classifications/car</td>
<td>1.52</td>
<td>1.38</td>
<td>0.14 (9.4%)</td>
</tr>
<tr>
<td>Estimated Shipment Time (days)</td>
<td>3.5</td>
<td>3.35</td>
<td>0.15 (4.5%)</td>
</tr>
<tr>
<td>Time to Generate Plan(hours)</td>
<td>72</td>
<td>10</td>
<td>62 (88%)</td>
</tr>
</tbody>
</table>

Table 3.7: Model Solution vs. Current Practice

3.4.6 Comparison to Current Operations

In order to illustrate the potential improvements of applying our proposed blocking model to the railroad, we compare the best solution we obtained with the one used by the railroad. The results are summarized in Table 3.7. We observe that our solution reduces the number of planned classifications by 9.4%, which can translate into millions of dollars in savings annually. Also, our improved blocking plan can reduce average shipment time by 4.5%, which enhances the railroad’s level of service. Lastly, our blocking plan can be generated relatively rapidly using a workstation-class computer.

3.5 Real Time Blocking Plan Generation

In the following experiment, we consider generating real time blocking plans in light of disruptions. On certain occasions, railroads might confront unexpected situations that cause the existing blocking plan to be inexecutable. Examples include:

- Some anticipated shipments are withdrawn at certain locations, or unanticipated traffic is on deck at other locations.

- Bad weather conditions might reduce blocking capacity at some yards such that certain traffic has to be rerouted and classified at other locations.

- Direct blocks have to be built for some high priority shipments to avoid further delay.

Under these situations, the existing plan might not be feasible or effective, and a new blocking plan has to be generated in a short time. To generate a real time
blocking plan, we have to determine the appropriate problem size to solve based on available computer storage space and run time. The results in Table 3.6 suggest that the largest disaggregate problem that the computer storage space allows, should be adopted to generate high quality solutions even when the available run time is as short as 30 minutes. The remaining issue is how to generate a good plan in a very short time. Before a new plan is generated, we have an existing plan and new demand and/or operating environment. There are two ways of generating a real time blocking plan, differentiated by the way the existing plan is utilized. The *cold-start* approach solves the new blocking problem from scratch, without utilizing the useful information in the existing plan. The *warm-start* approach, however, generates a new blocking plan by modifying the existing plan. We notice that impacts of the unexpected situations on the blocking plan are limited to certain yards and blocks, and the rest of the blocking plan requires few changes. Therefore, instead of solving the blocking model from scratch, the alternative approach inserts the existing plan into the blocking model and modifies it based on current situations to generate a new feasible plan. Such an approach can generate a feasible plan in one iteration, if feasible solutions exist. Table 3.8 summarizes the computational time (*Time*) needed, solution value (*Cost*) and optimality gap (*Gap* between the solution and the best feasible solution generated in 10 hours) for the following three cases:

- **Case A**: 5% of the shipments have different demand values,
- **Case B**: 10 blocks (none in the existing plan) have to be built in the new plan,
- **Case C**: 10 blocks in the existing plan cannot be built in the new plan.

The combination of the above three cases captures the direct effects of possible unexpected events on the blocking plan. To illustrate the efficiency of the alternative approach, we compare the solutions from the cold and warm start methods on the most disaggregate network (the high fidelity network with current block sequences only and 40% of the commodities consolidated) in Table 3.8. We only investigate one iteration of the algorithm in each experiment.
Table 3.8: “One-Shot” Feasible Plan Generation Comparisons

Table 3.8 shows that the blocking plan generated based on an existing plan yields much lower costs compared to the plan generated using the cold start method for each case. The solution time for one iteration is almost the same for the two approaches, around 15 minutes for the most disaggregate problem. When the problem size is smaller due to limited available storage space, the time to generate a real time blocking plan could be even shorter. The run time of 15 minutes or lower is usually acceptable in real time operations.

Despite its effectiveness in generating a plan quickly, the warm start approach has its own limitations as follows:

- It requires that there exists a high quality blocking plan and that the changes to the existing blocking plan cannot be dramatic in order to generate a new low-cost feasible blocking plan. The “one-shot” feasible plan is generated based on the existing plan with limited adjustments to reach feasibility. Hence, the solution quality of the new plan depends on (1) the quality of the existing plan, and (2) the closeness of the new and prior operating environments. If the quality of the existing plan is low or the operating environment is dramatically different, the above approach might lead to a new blocking plan with high costs.

- It is only useful at generating a new plan quickly, and it has only limited ability to generate an optimal or near-optimal plan. In subgradient optimization, the optimal blocks are selected based on the associated Lagrangian multiplier values. Since introducing an initial plan does not provide any improved information about the Lagrangian multipliers, its impacts on solution convergence are limited.
Therefore, the warm start approach is useful for generating real time blocking plans quickly. To generate higher quality blocking plans, however, we should follow the formal solution approach discussed in Chapter 2.

3.6 Efficiency Enhancements

To understand the advantage of using connectivity cut constraints and advanced dual solutions, we report and compare the implementation results with and without these enhancements. We evaluate model performance on a medium aggregate blocking problem (Case Three) by comparing the quality of initial and final lower bounds and feasible solutions, the running times and total number of iterations performed.

To evaluate the effects of our initial dual solution on the quality of the solution generated and on run time, we consider three scenarios. In the first, called Zero Dual, we initiate subgradient optimization with Lagrangian multipliers at zero. In the second, called Init Dual, we initiate subgradient optimization with the initial dual feasible solution discussed in Section 5. And in the third, called Dual Ascent, we initiate subgradient optimization with a dual feasible solution generated using our dual-ascent method.

To illustrate the improvement from adding connectivity cut constraints, we compare the performances with and without cut constraints, for each of these three scenarios.

As in the case study of route selection, we report Init LB, Init FS, T Block and T Flow for initial lower bound, upper bound and run times in the subproblems. Also, we present and compare T Total (total run time), N iter (number of iterations), LB (final lower bound), FS (final upper bound) and Gap (upper-lower bound gap) under two different time limits. The first one is set for the gaps in the problems with cut constraints are within 5%, and the second one is fixed at 24 hours.

The results for the medium aggregate network blocking problem are summarized in Table 3.9. Unlike Table 3.2, the solution values do not include local movement so that we can observe the effects of the efficient enhancements on the blocking problems...
Table 3.9: Computational Results Comparisons on Medium Level Aggregate Network

of similar size. The addition of cutset inequalities to the block subproblem improves significantly the solution quality and run time, and allows a dramatic reduction in the gap between the lower and upper bounds. In the case when cuts are included, the initial dual solution has little impact on the final solution quality. However, the number of iterations and total solution time are cut by more than half when an initial dual solution is provided by our dual ascent procedure.

3.7 Volume Constraints in the Blocking Problem

As discussed in Chapter 3, the flow subproblem of network design with volume constraints is a multicommodity flow problem and column generation is applied to solve the subproblem. However, if the volume constraints are not present, then, the flow subproblem is a set of simple shortest path problems. To compare the computational effort and solution quality, we contrast the results using the low fidelity (the second test problem) network with and without car volume constraints. (The test problems in previous experiments do not have volume constraints because the railroad cannot provide reliable capacity data.) The results are presented in Table 3.10, where both cut connectivity constraints and dual-ascent advanced start are used.
### Table 3.10: Comparisons With and Without Car Volume Constraints

<table>
<thead>
<tr>
<th></th>
<th>Init.LB</th>
<th>Init.FS</th>
<th>T.Block</th>
<th>T.Flow</th>
<th>T.Total</th>
<th>N.Iter</th>
<th>Final.LB</th>
<th>Final.FS</th>
<th>F.Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>With</td>
<td>409,773</td>
<td>499,458</td>
<td>313.8</td>
<td>869.2</td>
<td>61,530</td>
<td>39</td>
<td>420,622</td>
<td>442,059</td>
<td>4.8%</td>
</tr>
<tr>
<td>Without</td>
<td>424,029</td>
<td>447,729</td>
<td>288.9</td>
<td>260.0</td>
<td>4,778</td>
<td>4</td>
<td>424,029</td>
<td>442,755</td>
<td>4.2%</td>
</tr>
</tbody>
</table>

Notice that the computational effort when car volume constraints are eliminated is greatly reduced, since the flow subproblem is simplified to a set of shortest path problems. Interestingly, we find that the car volume constraints have very little effect on the final solution, with only 6% binding. Consequently, in the rest of case studies, we feel the omission of the car volume constraints is justifiable. The advantage of eliminating them is twofold: we reduce run time by simplifying the flow subproblem; and we reduce storage requirements, enabling us to solve larger (more disaggregate) network blocking problem.

### 3.8 Summary

This chapter illustrates the applications of the blocking model and algorithms through various case studies based on problems from a major railroad.

- We evaluate the trade-off between problem size and solution quality, suggesting appropriate aggregation schemes under different run time limits and available storage space. Experiments show that the best blocking plan can be generated on the most disaggregate network with commodity-based aggregation and current block sequences.

- We propose an efficient way of generating blocking plans in real time by modifying an existing plan. Utilizing the information in the existing plan, a high quality blocking plan can be generated in one iteration.

- We investigate the effects on solution quality and run time of the enhancements proposed in Chapter 2. We find that both adding connectivity constraints and
introducing dual-ascent solutions improve solution quality and reduce solution time significantly.

- We show the effects of volume constraints in the blocking model by comparing the run time and solution quality for the problems with and without volume constraints. Even though the volume constraints are not very tight, the solution time increases dramatically with the constraints included.

In the above experiments, all data are assumed to be known and deterministic. In the next two chapters, we will provide the models and case studies for the blocking problems with uncertain data.
We have discussed modeling and solution approaches for deterministic railroad blocking problems, where relevant supply, demand, cost and capacity information is assumed to be known with certainty. In real world operations, however, some of above information, especially customer demand is not available at the time the blocking plan must be developed. This chapter discusses the models and solution approaches for railroad blocking under uncertainty.

4.1 Uncertainties in the Railroad Blocking Problem

The blocking plan is a medium-term tactical plan, with the planning horizon ranging from one quarter to one year. During this period, the operating environment might be very different from time to time, due to variations in demand, weather conditions, equipment availability, etc.. Despite variability in the operating environment, most railroads adopt medium-term blocking plans because it takes a long time and a great deal of labor input to develop a blocking plan and coordinate with other tactical plans. In practice, most railroads develop blocking plans based on average demand and ca-
pacity and the resulting plan does not capture variations in demand or capacity. This planning method is effective when variations in uncertainties are moderate, however, variations in demand are often large, and the blocking plans based on average demand perform badly on peak days. This creates a major challenge to execute the blocking plans when large variations in demand are realized. To address this challenge, we propose to design robust blocking plans that are capable of incorporating variations in demand and supply.

In railroad operations, a number of factors, including demand, supply and costs, could be uncertain when a blocking plan has to be developed. Among all uncertainties, demand is particularly volatile. In addition to dramatic seasonal fluctuations, there are significant variations in daily demands. Network supply, in terms of blocking capacities at yards, can be affected by weather conditions and special events. For example, bad weather conditions might make some classification tracks unusable at certain yards and the blocking capacities could thus be reduced. Compared to the uncertainties in demand and supply, costs are relatively stable, especially when the objective of the blocking problem is to minimize the total number of classifications.

Changes in demand and capacity have large impacts on the optimality and even feasibility of the blocking plans. A blocking plan based on certain demands might not be able to accommodate all traffic as demand increases. Similarly, a blocking plan based on existing capacity at yards might not be implementable when some yards have reduced or no capacity due to bad weather conditions.

In railroad planning, train schedules, crew schedules and yard operating plans are designed based on the blocking plan. Any unpredictable changes to the blocking plan have several downstream effects, making plan adjustments and execution in real time operations difficult and costly. Therefore, railroads have multiple objectives in designing blocking plans. First, railroads wish to achieve the lowest operating costs of delivering traffic from the origins to the destinations. Second, railroads wish to maintain plan consistency and minimize the requirements for plan adjustments. Unfortunately, these objectives are at odds. The lowest operating costs are usually achieved at the price of frequent plan adjustments, making the blocking plans un-
stable. Whereas a consistent plan may be achieved by ignoring the opportunities of plan adjustments that potentially reduce the operating costs. Deterministic solution approaches cannot deal with these competing objectives and uncertainties. There is a need for an approach considers uncertainty and balances these two objectives.

In the next sections, we discuss the first issues of defining and modeling robustness and in the following sections, we describe stochastic programming models for robust planning. In the end, we describe a general form robust blocking model and corresponding solution approaches.

4.2 Robustness and Robust Planning

In this section, we first discuss the definition of robustness with specific robustness criteria, and we present general-form mathematical formulations for modeling robustness under different robustness criteria. Then, we introduce two types of robust plans, i.e., static plans and dynamic plans, differentiated by the flexibility of changing the plans. We conclude this section with the literature review on robustness and robust optimization.

4.2.1 Robustness Definition

Robustness is achieved when a plan or design performs well under all or most potential realizations of uncertainties. Depending on the specific planning environments, uncertainties could result from randomness in demand, supply, costs or other model parameters. Performance is usually evaluated according to specific criteria, which could be cost or service oriented. The cost oriented criteria attempt to minimize certain costs using robust plans, examples of these criteria include:

- **Minmax Regret Criterion**, which minimizes the cost for the worst case among all possible realizations of uncertainties;

- **Mean Value Minimization Criterion**, which minimizes the expected costs under all realizations of uncertainties;
Closeness to Optimality Criterion, which minimizes the deviation from the optimal solutions under all realizations of uncertainties. (The closeness to optimality is usually measured by the percentage deviation of certain solution from the optimal solution to the specific realizations of the uncertainties, as discussed in Kouvelis and Yu [1996].)

A service oriented criterion attempts to maintain certain levels of service through robust planning. Level of service could be defined as requirements to move commodities from their origins to their destinations within certain amounts of time, or requirements to maintain a certain level of network connectivity (for example, a minimum number of disjoint paths might be required for certain commodities).

The significance of robustness arises from the difficulty of quickly changing a plan or design. If a plan or design can be changed freely and instantly, robustness is not an issue since we can always adopt the optimal plan or design for the specific realization of uncertainties. The difficulty in changing a plan or design might involve certain costs or time for migrating an existing plan to a new one. When the time required for plan changes is long, a fixed plan or design, which satisfies the specific robustness criteria, has to be developed in advance. When the plan changes can be made quickly, a robust plan or design is determined based on the associated plan change costs. If the costs required for plan changes are extremely high compared to the operating costs, robustness can be achieved by executing a static (unchanging) plan or design that satisfies the specific robustness criteria; If plan change costs are trivial compared to the operating costs, robustness can be achieved by adopting flexible dynamic plans or designs that are adjusted frequently and freely according to specific realizations of uncertainties. For most occasions, plan change costs are not extreme, and, therefore, robustness will be achieved by balancing the trade-offs between operating costs and plan change costs.
4.2.2 Mathematical Formulations for Robustness

With respect to specific robustness criteria, we can model robustness through mathematical programming. Before presenting the specific formulations, we introduce the following notations:

- $s$: a specific realization of uncertainties;
- $S$: the set of all possible realizations of uncertainties;
- $x$: decision variables;
- $X$: the set of decision variables;
- $f()$: objective function;
- $f^*(x)$: the optimal value for objective function $f$;
- $g()$: service level function;
- $a$: a specific service level;
- $c$: a threshold of service level satisfaction;
- $Pr()$: probability function; and
- $E_s$: mathematical expectation over uncertainty $s$.

Using the above notations, we can formulate various robustness criteria. Even though the specific problems might be different, the corresponding objectives or constraints reflecting robustness can be generalized. In the following, we provide the general formulations for specific robustness criteria without considering the details for the specific applications.

The minmax regret criterion can be formulated as:

$$\min_{x \in X} \max_{s \in S} f(x, s).$$  (4.1)
The mean value minimization criteria can be formulated as:

$$min_{x \in X} E_s[f(x, s)].$$

(4.2)

The closeness to optimality criteria can be formulated based on different rules. For example, the absolute value deviation criteria is:

$$min_{x \in X} \sum_{s \in S} |f(x, s) - f^*(x, s)|,$$

(4.3)

and the mean square deviation criteria is:

$$min_{x \in X} \sum_{s \in S} [f(x, s) - f^*(x, s)]^2.$$

(4.4)

The service oriented criteria can be formulated as:

$$Pr[g(x, s) \geq \text{or } \leq a] \geq \text{or } \leq c.$$  

(4.5)

Notice that the objective functions ($f$) in the above formulations are in general forms. Besides the operating costs, these objectives might also capture the plan change costs when the plan adjustments are allowed.

### 4.2.3 Static vs. Dynamic Plans

Depending upon the specific operating and planning environment, a robust plan could be static or dynamic. A static plan requires a fixed plan to be executed in the planning period, for all potential realizations of uncertainties. A dynamic plan allows the plan to be modified dynamically as more and more information becomes certain. The selection of the specific type of plan in railroad blocking depends on how difficult it is to change the plan. A static plan is relevant when changes to the plan cannot be made or are too costly once information about the uncertainties becomes known. A dynamic plan, on the other hand, applies when the blocking plan can be altered relatively easily.
Since a static plan is always feasible for dynamic planning, a dynamic plan usually yields a lower cost than a static plan when the same amount of information about uncertainty is considered. On the other hand, plan modifications in dynamic planning involve certain adjustments to operations and sequential planning activities. Therefore, we consider plan change costs to reflect the difficulty of plan adjustments in dynamic models, which will be discussed later.

4.2.4 Literature Review

Robustness and robust planning are addressed in various applications. Watanabe and Ellis (1993) survey the conceptual and modeling developments for robustness in the areas involving economics, ecology, water and environment management. Stiegler (1939) defines robustness as economic flexibility. A plan or policy is robust if it is relatively insensitive to a range of possible economic conditions. In an ecological application, Holling (1973) uses the concept of resilience to describe the ability of a dynamic multispecies ecological system to persist with the same basic structures when subjected to stress. In a research on water resources planning, Hashimoto, et al. (1982) evaluate system performance from the following viewpoints:

1. **Reliability**: how often the system fails;

2. **Resiliency**: how quickly the system returns to a satisfactory state once a failure has occurred; and

3. **Vulnerability**: how significant the likely consequences of failure might be.

Haimes and Hall (1977) introduce sensitivity measures of the objective function as secondary objectives and reformulate the original problem as a multi-objective program.

Recently, Stoer (1992) describes robustness in telecommunications, where robustness is captured by network survivability against facility failures. Stoer presents mathematical formulations for survivable network design by introducing *redundant* designs that pursue disjoint connected paths for routing messages. Mulvey, et al.
(1995) adopt a scenario based approach to model robustness under two dimensions. The first dimension, *solution robustness*, checks if the design remains “close” to optimality for all scenarios of input data. The second, *model robustness*, checks if the design remains “almost” feasible for all variable data inputs. Keouvelis and Yu (1997) and Gutierrez, et al. (1996) describe and formulate robustness for problems where associated probabilities of realizations of uncertainties are known. They adopt min-max regret rules to minimize the worst-case costs among all possible realizations of uncertainties.

In this research, we represent uncertainty by *scenarios*, which are realizations of the stochastic elements, (namely, demand and supply in terms of block and volume capacities) in the blocking problem. For problems with multiple uncertainty factors, the scenarios represent possible combinations of realizations of these factors. To illustrate, if the uncertainty can fully be captured by *daily* variations in demand, then, these variations can be represented by seven scenarios, one for each day’s demands. In the following sections, we present various robust models for railroad blocking problems.

### 4.3 Static Blocking Models

In this section, we discuss *static* blocking models that assume a fixed blocking plan is executed regardless of the specific realizations of uncertainties. These models apply to problems where plan adjustments are too costly or impossible. In the following section, we present *dynamic* blocking models that allow blocking plan adjustments when realizations of uncertainties become available. The models discussed in these two sections are differentiated by their specific robustness definition. The *worst-case models* define robustness as achieving the lowest cost for the worst-case, and *expected performance models* define robustness as attaining the lowest expected cost over all realizations of uncertainty.

We adopt the same notations as in Chapter 3, and introduce scenario \( s \) as a certain
realization of uncertainties, with \( S \) representing the set of all scenarios.

### 4.3.1 Static Worst-Case Model

A static worst-case model is to find a blocking plan that is feasible to every possible scenario and cost achieved minimizes the maximum among all possible scenarios. The model formulation is:

\[
\begin{align*}
(\text{ROB}_\text{wcs}) \quad \min_{\{f_k^s, y_a\}} & \max_{s \in S} \sum_{k \in \mathcal{K}(s)} \sum_{q \in \mathcal{Q}(k)} PC_q^k v^k(s) f_q^k \\
\sum_{q \in \mathcal{Q}(k)} f_q^k(s) \delta_a^q & \leq y_a, \quad \forall k \in \mathcal{K}(s), \forall a \in \mathcal{A}, \forall s \in S \\
\sum_{q \in \mathcal{Q}(k)} f_q^k(s) & = 1, \quad \forall k \in \mathcal{K}(s), \forall s \in S \\
\sum_{a \in \mathcal{A}} y_a \xi_a^s & \leq B(i)(s), \quad \forall i \in \mathcal{N}, \forall s \in S \\
\sum_{k \in \mathcal{K}(s)} \sum_{q \in \mathcal{Q}(k)} \sum_{a \in \mathcal{A}} v^k(s) f_q^k(s) \delta_a^q \delta_a^s & \leq V(i)(s), \quad \forall i \in \mathcal{N}, \forall s \in S \\
f_q^k(s) & \geq 0, \quad \forall q \in \mathcal{Q}(k), \forall k \in \mathcal{K}(s), \forall s \in S \\
y_a & \in \{0, 1\}, \quad \forall a \in \mathcal{A}.
\end{align*}
\]  

A general discussion of the worst-case robust optimization problem can be found in Kouvelis and Yu [1997]. The solution approach they present applies Benders decomposition (Benders [1962]) to a two-stage stochastic programming model, referred as the \textit{L-shaped method} (Birge and Louveaux [1997]). In Benders decomposition, the above mixed-integer-programming (MIP) problem is reformulated as a \textit{Restricted Benders Master Problem (RMP)} and a \textit{flow subproblem}. Solving the RMP provides a tentative blocking plan and a lower bound on the original problem. Based on the tentative blocking plan, the flow subproblem is solved, either proving optimality of the current plan or generating a \textit{Benders cut} to add to the RMP. The original problem is solved, then, by alternately solving the RMP and the flow subproblem.

The RMP can be formulated as:
Restricted Master Problem

\[(RMP) \min z \quad (4.13)\]

\[
s.t. \sum_{a \in A} y_a \xi_t^a \leq B(i), \ \forall i \in \mathcal{N} \quad (4.14)\]

\[
\sum_{a \in \text{cut}(k)} y_a \delta_a^q \geq 1, \quad \forall q \in \mathcal{Q}(k), \forall \text{cut}(k) \in [\mathcal{O}(k), \mathcal{D}(k)], \forall k \in \mathcal{K}(s), \forall s \in \mathcal{S} \quad (4.15)\]

\[
y_a \in \{0, 1\} \quad \forall a \in A \quad (4.16)\]

Benders cuts to be added in each iteration. \quad (4.17)

We will define the Benders cuts and their generation later. The objective of the RMP is to find the tightest Benders cut with block capacity (4.14) and connectivity (4.15) constraints satisfied.

Flow Subproblem

For a given tentative blocking plan \((y_a)\), we have for each scenario \(s \in \mathcal{S}\):

\[
BP(s, y_a) : \min W_s(y_a) = \sum_{k \in \mathcal{K}(s)} \sum_{q \in \mathcal{Q}(k)} PC_q^k v^k(s) f_q^k(s) \quad (4.18)\]

\[
s.t. \sum_{q \in \mathcal{Q}(k)} f_q^k(s) \delta_q^a \leq y_a, \quad \forall k \in \mathcal{K}(s), \forall a \in A \quad (4.19)\]

\[
\sum_{q \in \mathcal{Q}(k)} f_q^k(s) = 1, \quad \forall k \in \mathcal{K}(s) \quad (4.20)\]

\[
\sum_{k \in \mathcal{K}(s)} \sum_{q \in \mathcal{Q}(k)} \sum_{a \in A} v^k(s) f_q^k(s) \delta_q^a \xi_t^a \leq V(i), \quad \forall i \in \mathcal{N} \quad (4.21)\]

\[
f_q^k(s) \geq 0, \quad \forall q \in \mathcal{Q}(k), \forall k \in \mathcal{K}(s) \quad (4.22)\]

The above flow subproblem is a multicommodity flow problem and can be solved using column generation as discussed in Chapter 3.

Let \(\lambda^k_q(s), \sigma_k(s), \) and \(\pi_i(s)\) be the duals associated with constraints 4.19, 4.20 and 4.21, respectively. For any set of feasible duals to the above flow subproblem, and any
tentative blocking plan \((y_a)\), a *Benders cut* for RMP or lower bound on the optimal value of \(\text{ROB\_WC}\), is:

\[
z \geq \sum_{a \in A} \left( \sum_{k \in K(s)} \lambda^k_y(s) \right) y_a + \sum_{k \in K(s)} \sigma^k(s) + \sum_{i \in N} V(i) \pi_i(s). \tag{4.23}
\]

Cuts of the form of 4.23 are added to approximate more accurately the original problem, \(\text{ROB\_WC}\).

The detailed procedure is summarized as follows:

- **Step 0:** *Initialization.* Set iteration \(i := 0\), and upper bound \(UB = +\infty\), lower bound \(LB = 0\).

- **Step 1:** *Tentative Blocking Plan Generation.* Solve the RMP at iteration \(i\) to obtain \(z^i = z^*\), the optimal solution of RMP at iteration \(i\). If \(LB < z^i\), set \(LB = z^i\).

- **Step 2:** *Upper and Lower Bounds Update.* Solve the flow subproblems for each scenario \(s\) at iteration \(i\), and obtain \(w^i_s(y_a)\). Set \(z_{max}^i = \max_{s \in S} w^i_s(y_a)\). If \(UB > z_{max}^i\), set \(UB = z_{max}^i\). Given the dual solution to the flow subproblem at iteration \(i\), generate a Benders cut for each scenario and add them to the current RMP.

- **Step 3:** *Optimality Check.* If \(\frac{UB-LB}{UB} < \epsilon\), a specified tolerance between the upper and lower bounds, STOP; otherwise, go to Step 1.

Even though different stochastic factors are incorporated differently, supply (capacities at yards), in the right hand variables and demand in the objective function and constraint coefficients, the resulting MIP is a deterministic math program. The result is that the solution approach need not be altered. Tractability, however, is an issue because problem size expands rapidly with increases in the number of scenarios.

It is worth noting that this formulation requires the solution to be feasible for all cases, including the extreme cases with low probabilities distribution of occurrence. This type of robustness criterion might be appropriate in the case where failure of
service and inability to alter plans are unacceptable; however, for railroad blocking, this model will likely result in plans that are overly restrictive and expensive.

4.3.2 Static Expected Performance Model

Unlike the static worst-case model, the static expected performance model seeks the static solution with the minimum expected objective function value, where each possible scenario \(s\) has an associated probability, \(p(s)\). Like the worst-case model, the expected performance model generates a single, static solution that must be feasible for all scenarios. The model is formulated as:

\[
(\text{ROB-EP}) \quad \min E \left[ \sum_{k \in K(s)} \sum_{q \in Q(k)} PC_q^k \nu^k(s) f_q^k(s) \right] 
\]

\[
\text{s.t.} \quad \sum_{q \in Q(k)} f_q^k(s) \delta_a^q \leq y_a \quad \text{a.s., } \forall s \in S \quad (4.25)
\]

\[
\sum_{q \in Q(k)} f_q^k(s) = 1 \quad \text{a.s., } \forall s \in S \quad (4.26)
\]

\[
\sum_{a \in A} y_a \xi_a^s \leq B(i)(s) \quad \text{a.s., } \forall s \in S \quad (4.27)
\]

\[
\sum_{k \in K(s)} \sum_{q \in Q(k)} \sum_{a \in A} \nu^k(s) f_q^k(s) \delta_a^q \xi_a^s \leq V(i)(s) \quad \text{a.s., } \forall s \in S \quad (4.28)
\]

\[
f_q^k(s) \geq 0 \quad \text{a.s., } \forall q \in Q(k), \forall s \in S \quad (4.29)
\]

\[
y_a \in \{0, 1\} \quad \text{a.s., } \forall a \in A
\]

Similar to the worst-case model, this model can be solved using the L-shaped method. We summarize the solution procedure as follows:

- Step 0: Initialization
  
  Set \(i := 0\), and \(UB = +\infty\), \(LB = 0\)

- Step 1: Tentative Blocking Plan Generation
  
  Solve Benders RMP and obtain \(Z_{MP}^i\)
  
  If \(LB < Z_{MP}^i\), set \(LB = Z_{MP}^i\)
• Step 2: Upper and Lower Bounds Update

Solve the flow subproblems for each scenario, and obtain $Z_{FLOW}^i(s)$

Set $Z_{mean}^i = \sum_{s=1}^{[S]} P(s) Z_{FLOW}^i(s)$, where $Z_{mean}^i$ represents the expected objective function value at iteration $i$.

If $UB > Z_{mean}^i$, set $UB = Z_{mean}^i$

Generate Benders cuts and add to RMP.

• Step 3: Optimality Check

If $\frac{UB-LB}{UB} < \epsilon$, STOP. Otherwise, go to Step 1.

where $Z_{mean}$ represents the expected cost.

The commonality of static worst-case and static expected performance models is the pursuit of a static blocking plan that is feasible to all potential scenarios and yields the minimum objective value. Static plans, however, are not effective as variability increases. Birge (1982) establishes that large error bounds arise when one solves static expected performance problems, and larger errors are expected for static worst-case models.

4.4 Dynamic Expected Performance Blocking Models

Unlike static blocking models that search for a fixed plan for all possible scenarios, dynamic blocking models allow the plans to change given the realizations of uncertainties. The solution to a dynamic blocking model is a “core” plan, and a set of plan changes. The “core” plan is used to determine the resources needed to execute the plan. Such a “core” plan is needed because personnel and equipment must be scheduled in advance before realizations of uncertainties.

The common objective in dynamic models is to search for a dynamic plan that balances flow costs and plan change costs such that total costs are minimized. There are several ways to model this, depending on the assumptions on plan change costs and the specific planning objectives.
As with the static models, dynamic models can be formulated using the worst-case and expected performance criteria. Since the worst-case models usually are not appropriate for the railroad blocking problem, we concentrate on expected performance models. Dynamic expected performance models search for dynamic plans that achieve the lowest expected total costs, including flow costs and plan change costs. The models we present differ in the manner that the change costs are modeled and their representation in the objective function. A general form of the objective function applied to the railroad blocking model is

$$\min E\left[ \sum_{k \in C(s)} \sum_{q \in Q(k)} PC_{q}(s) + F(\bar{y}_a, y_a(s)) \right]$$

where \( F \) is the cost function for plan changes. Plan changes represent the deviation of individual plans \( (y_a(s)) \) from the core plan \( (\bar{y}_a) \).

A special case is a flexible plan model, assuming that blocking plans can be altered freely to accommodate various realizations of uncertain demand. The flexible plan model decomposes by scenario into a set of \( |S| \) deterministic blocking problems, one for each scenario. Their solutions can be attained using the algorithms discussed in Chapter 2. Clearly, this naive model is relevant only for problems where the costs of changing the plan are negligible. In railroad blocking operations, the costs of change are not negligible, in fact, they can be significant when the change involves significant changes in the resources necessary to execute the plan. The solution to this problem provides a lower bound on the least cost, feasible solution.

### 4.4.1 Quadratic Dynamic Model

Jauffred (1997) presents a dynamic expected cost model in which the plan change cost function \( (F) \) is quadratic and symmetric. For our application, this implies that the costs of adding or deleting a block are the same. His modeling approach applied to the blocking problem is:
\[
\min E\left[ \sum_{k \in K(s)} \sum_{q \in Q(k)} PC^k q^v(s) f^k_q(s) + \frac{1}{2} (y_a(s) - \bar{y}_a)'Q(y_a(s) - \bar{y}_a) \right] \quad (4.32)
\]

s.t. \[
\sum_{q \in Q(k)} f^k_q(s) \delta^q_a \leq y_a(s), \quad \forall k \in K(s), \forall a \in A, \forall s \in S \quad (4.33)
\]
\[
\sum_{q \in Q(k)} f^k_q(s) = 1, \quad \forall k \in K(s), \forall s \in S \quad (4.34)
\]
\[
\sum_{a \in A} y_a(s) \xi^a_i \leq B(i)(s), \quad \forall i \in N, \forall f \in S \quad (4.35)
\]
\[
\sum_{k \in K(f)} \sum_{q \in Q(k)} u^k(s) f^k_q(s) \delta^q_a \xi^a_i \leq V(i)(s), \quad \forall i \in N, \forall f \in S \quad (4.36)
\]
\[
f^k_q(s) \geq 0, \quad \forall q \in Q(k), \forall k \in K(s), \forall s \in S \quad (4.37)
\]
\[
\bar{y}_a, \quad y_a(s) \in \{0, 1\}, \quad \forall a \in A, \forall f \in S \quad (4.38)
\]

where \( Q \) is a positive semi-definite matrix.

Unlike the above static models, this model allows the blocking plan to change to accommodate variations in daily demand and capacity. That is, the individual plans \((y_a(s))\) may differ by scenario. We denote \( \bar{y}_a \) as the core plan and imposes costs, denoted by the matrix \( Q \), on modifying the core plan. (In Jauffred’s original work, \( \bar{y}_a \) is referred to as the average plan for the problems where \( y_a \) is a continuous variable. Since block variable \( y_a \) is a binary variable, \( \bar{y}_a \) is not an average plan in the blocking problems.)

Note that this dynamic model reduces to the static expected performance model when the eigenvalues of \( Q \) are sufficiently high, and reduces to the flexible model when \( Q \) is a null matrix.

Jauffred’s solution algorithm is motivated by the following optimality condition for problems with continuous variables (Jauffred [1997]):

\[
\bar{y}_a = E[y_a(s)]. \quad (4.39)
\]

This optimality condition implies that the optimal plan \( \bar{y}_a \) (referred to as the
average plan) is the average of the individual plans \(y_a(s)\). Jauffred details an iterative approach to solve 4.32. When applied to the blocking problem, the steps are summarized as:

- **Step 0: Initialization.** Set iteration index \(t = 0\). Select an initial average plan \(\bar{y}_a^t\).

- **Step 1: Scenario Solution.** Given \(\bar{y}_a^t\), for every scenario \(s \in \mathcal{S}\), solve:

  \[
  \min \sum_{k \in \mathcal{K}(s)} \sum_{q \in \mathcal{Q}(k)} PC_q v^k(s)[f_q^k(s)]^k + \frac{1}{2}([y_a(s)]^t - [\bar{y}_a]^t)'Q([y_a(s)]^t - [\bar{y}_a]^t) \tag{4.40}
  \]

  \[
  s.t. \quad \sum_{q \in \mathcal{Q}(k)} f_q^k(s) = 1, \forall k \in \mathcal{K}(s) \tag{4.41}
  \]

  \[
  \sum_{q \in \mathcal{Q}(k)} f_q^k(s) = 1, \forall k \in \mathcal{K}(s) \tag{4.42}
  \]

  \[
  \sum_{a \in \mathcal{A}} y_a(s)^t \delta_a^q \leq B(i)(s), \forall i \in \mathcal{N} \tag{4.43}
  \]

  \[
  \sum_{k \in \mathcal{K}(s)} \sum_{q \in \mathcal{Q}(k)} f_q^k(s) \delta_a^q \leq V(i)(s), \forall i \in \mathcal{N} \tag{4.44}
  \]

  \[
  f_q^k(s) \geq 0, \forall q \in \mathcal{Q}(k), \forall k \in \mathcal{K}(s) \tag{4.45}
  \]

  \[
  y_a(s)^t \in [0,1], \forall a \in \mathcal{A}. \tag{4.46}
  \]

- **Step 2: Solution Update.** Let

  \[
  \bar{y}_a^{t+1} = \bar{y}_a^t + \frac{Q}{k} (E[y_a(s)^t] - \bar{y}_a^t). \tag{4.47}
  \]

- **Step 3: Candidate Plan Generation.** In each iteration \(t\), solve for each scenario, \(s \in \mathcal{S}\):

  \[
  z^t = \min \sum_{k \in \mathcal{K}(s)} \sum_{q \in \mathcal{Q}(k)} PC_q v^k(s)[f_q^k(s)]^k + \frac{1}{2}([y_a(s)]^t - [\bar{y}_a]^t+1)'Q([y_a(s)]^t - [\bar{y}_a]^t+1) \tag{4.48}
  \]

  \[
  s.t. \quad \sum_{q \in \mathcal{Q}(k)} f_q^k(s) \delta_a^q \leq y_a(s)^t, \forall k \in \mathcal{K}(s), \forall a \in \mathcal{A} \tag{4.49}
  \]
\[ \sum_{q \in Q(k)} f_q^k(s) = 1, \forall k \in \mathcal{K}(s) \quad (4.50) \]
\[ \sum_{a \in A} y_a(s) t^c_a \leq B(i)(s), \forall i \in \mathcal{N} \quad (4.51) \]
\[ \sum_{k \in \mathcal{K}(f)} \sum_{q \in Q(k)} \sum_{a \in A} \eta(s)^k q^k(s) s^a q^c_a \leq V(i)(s), \forall i \in \mathcal{N} \quad (4.52) \]
\[ f_q^k(s) \geq 0, \forall q \in Q(k), \forall k \in \mathcal{K}(s) \quad (4.53) \]
\[ y_a(s)^t \in \{0, 1\}, \forall a \in A. \quad (4.54) \]

- **Step 4: Convergence Check.** If \(|\tilde{y}^{t+1}_a - \tilde{y}^t_a| \leq \epsilon\), stop; otherwise, set \(t := t+1\) and go to Step 1.

- **Step 5: Plan Selection.** Select:

\[ z^* = \min_t z^t. \quad (4.55) \]

The shortcomings of this model when applied to the blocking problem are:

1. The quadratic formulation of the average plan model assumes that the plan change cost is the same for adding or removing a block from the core plan. This assumption is violated in railroad blocking whenever it is cheaper to remove a block than to add one, as is often the case.

2. The solution algorithm for the average plan model requires the repeated solution of deterministic blocking problems. Since the blocking problem is a special case of network design, an NP-hard problem, this solution approach might be impractical for large-scale blocking problems, like those encountered at Class I railroads.

### 4.4.2 Linear Dynamic Model

The linear dynamic model is a mixed integer program with three sets of binary variables representing the core plan and changes (additions or deletions) to it. Specifically, for the blocking problem:
• $\bar{y}_a$: equals 1 if block $a$ is in the core plan, 0 otherwise;

• $w_a(s)$: equals 1 if block $a$ is added to the core plan for scenario $s$, 0 otherwise;

• $z_a(s)$: equals 1 if block $a$ is deleted from the core plan for scenario $s$, 0 otherwise.

In this model, we differentiate the cost per block for adding to or deleting from the core plan. We let $f^w_a$ denote the cost to add a block and $f^*_a$ be the cost to delete one. The objective is to minimize the expected total costs, including both the core plan costs and the change costs.

To eliminate some illogical cases, such as adding a block that is already present, simultaneously adding and deleting a block, or deleting a block that does not exist in the core plan, we include the following constraints in our dynamic model:

$$w_a(s) + z_a(s) \leq 1, \quad \forall s \in S, \forall a \in A \quad (4.56)$$

$$\bar{y}_a \geq z_a(s), \quad \forall a \in A, \quad s \in S \quad (4.57)$$

$$\bar{y}_a + w_a(s) \leq 1, \quad \forall a \in A, \quad s \in S. \quad (4.58)$$

Constraints (4.56) require that only one change activity is allowed for a block in each scenario. Constraints (4.57) allow a block to be deleted only if it is present in the core plan. Constraints (4.58) allow no block to be added if the block is already in the core plan.

If costs are positive, then constraints 4.56 and 4.58 do not need to be included since no optimal solution would violate them. In particular, $w_a(s) + z_a(s) \geq 1$ is not possible in an optimal solution if change costs are both positive, and $\bar{y}_a + w_a(s) \geq 1$ is also not possible if the cost of adding a block to the core plan is positive.

Using the same notations as previously presented, our binary dynamic blocking model is:

$$\min \left\{ \sum_{s \in S} \sum_{q \in Q(s)} PC^q_k v^k(s) f^k_q(s) + \sum_{s \in S} \sum_{a \in A} [f^w_a w_a(s) + f^*_a z_a(s)] \right\} \quad (4.59)$$
\[ \text{s.t.} \quad \sum_{q \in Q(k)} f^k_q(s) \delta_a^q \leq \bar{y}_a + w_a(s) - z_a(s) \quad \text{a.s.,} \quad \forall k \in K(s), \forall a \in A, \forall s \in S \]  
(4.60)

\[ \sum_{q \in Q(k)} f^k_q(s) = 1 \quad \text{a.s.,} \quad \forall k \in K(s), \forall s \in S \]  
(4.61)

\[ \sum_{a \in A} [\bar{y}_a + w_a(s) - z_a(s)] \xi^a_i \leq B(i)(s) \quad \text{a.s.,} \quad \forall i \in N, \forall s \in S \]  
(4.62)

\[ \sum_{k \in K(s)} \sum_{q \in Q(k)} \sum_{a \in A} u^k(s) f^k_q(s) \xi^a_i \leq V(i)(s) \quad \text{a.s.,} \quad \forall i \in N, \forall s \in S \]  
(4.63)

\[ \bar{y}_a - z_a(s) \geq 0 \quad \text{a.s.,} \quad \forall a \in A, \forall s \in S \]  
(4.64)

\[ f^k_q(s) \geq 0 \quad \text{a.s.,} \quad \forall q \in Q(k), \forall k \in K(s), \forall s \in S \]  
(4.65)

\[ \bar{y}_a, w_a(s), z_a(s) \in \{0,1\} \quad \text{a.s.,} \quad \forall a \in A, \forall s \in S. \]  
(4.66)

This formulation is an extension of 2.19 that includes the added dimension of scenarios. Optimization over multiple scenarios results in plans that optimally balance core plan and change costs.

This model is flexible in that it can model very different situations: those in which plan change is impossible to those in which plan change is inexpensive. When change costs are sufficiently high, a static core plan is the optimal solution, that is, each scenario specific plan is exactly the core plan. In contrast, when change costs are zero, each scenario specific plan is optimized without regard to the core plan, and in fact, the core plan is meaningless.

We adapt our approach of Chapter 2 to this linear dynamic model. By relaxing the forcing constraints, we can decompose 4.59 - 4.66 into two separate subproblems, one in the binary design variables and the other in the continuous flow variables. Due to the introduction of additional variables and constraints, however, the subproblems become more complex and more difficult to solve.

Flow Subproblem

Now instead of one flow subproblem, there are |S| flow subproblems, one for each scenario. The flow subproblem for robust network design is:

\[ (\text{ROB-FLOW}) \quad \min_{E} \left[ \sum_{k \in K(s)} \sum_{q \in Q(k)} PC^k_q u^k(s) f^k_q(s) + \sum_{s \in S} \sum_{k \in K} \sum_{a \in A} \lambda^k_a(s) \sum_{q \in Q(k)} f^k_q \delta^q_a \right] \]  
(4.67)
\[
\begin{align*}
\text{s.t.} \quad & \sum_{q \in Q(k)} f^k_q(s) = 1 \quad \forall k \in \mathcal{K}(s), \forall s \in S \quad (4.68) \\
& \sum_{k \in \mathcal{K}(s)} \sum_{q \in Q(k)} \sum_{a \in A} v^k(s) f^k_q(s) \delta^k_a \xi^a_i \leq V(i)(s) \quad \forall i \in \mathcal{N}, \forall s \in S \quad (4.69) \\
& f^k_q(s) \geq 0 \quad \forall q \in Q(k), \forall k \in \mathcal{K}(s), \forall s \in S. \quad (4.70)
\end{align*}
\]

Note that the objective function is separable in the flow variables and the flow subproblem has block angular structure, with one block representing a scenario specific flow subproblem. Therefore, we can decompose the flow subproblem into \(|S|\) separate flow subproblems, one for each scenario. Hence, the computational effort to solve flow subproblems increases proportionally to the number of scenarios. As in Chapter 2, we solve these large scenario specific flow subproblems using column generation.

**Block Subproblem**

The block subproblem for the linear dynamic model is:

\[
\begin{align*}
\text{(ROB.BLK)} \quad & \min \sum_{s \in S} \sum_{a \in A} [f^a_w w_a(s) + f^a_z z_a(s)] - \sum_{s \in S} \sum_{k \in \mathcal{K}} \sum_{a \in A} \lambda^k_a(s) [\overline{y}_a + w_a(s) - z_a(s)] \\
\text{s.t.} \quad & \sum_{a \in A} [\overline{y}_a + w_a(s) - z_a(s)] \xi^a_i \leq B(i)(s) \quad a.s., \quad \forall i \in \mathcal{N}, \forall s \in S \quad (4.72) \\
& \overline{y}_a - z_a(s) \geq 0 \quad a.s., \quad \forall a \in \mathcal{A}, \forall s \in S \quad (4.73) \\
& \overline{y}_a \in \{0, 1\} \quad a.s., \quad \forall a \in \mathcal{A} \quad (4.74) \\
& w_a(s), z_a(s) \in \{0, 1\} \quad a.s., \quad \forall a \in \mathcal{A}, \forall s \in S. \quad (4.75)
\end{align*}
\]

Compared to the blocking subproblem \textbf{B.PATH} in Chapter 2, this model has additional sets of integer variables in the block subproblem. Moreover, unlike the flow subproblems, the block subproblem does not decompose into separate subproblems for each scenario since they are tied together by the core plan variables, \(\overline{y}_a\). As a result, the computational effort to solve the block subproblem is much more difficult than when only a single scenario is considered.
We have to modify the solution approach of Chapter 2 to account for the multiple scenarios. The only change necessary is to modify the connectivity constraints, by requiring at least one connected path for each commodity in every scenario. The modified constraints, reflecting the added variables, are:

\[
\sum_{a \in \text{cut}(k)} (y_a + w_a(s) - z_a(s)) \geq 1, \forall \text{cut}(k) \in [\mathcal{O}(k), \mathcal{D}(k)], \forall k \in \mathcal{K}(s), \forall s \in S. \tag{4.76}
\]

The rest of the solution procedure remains the same as discussed in Chapter 2.

### 4.5 Summary

In this chapter, we review different ways of defining and formulating robustness for the problems with uncertainties. We discuss different formulations and solution algorithms for railroad blocking problem with variable inputs. Then, we propose a mixed integer program formulation and solution algorithm for multiple scenario railroad blocking problems. The solution approach is an extension to the approach for the deterministic problems discussed in Chapter 2, which again decomposes the problem into two separate problems using Lagrangian relaxation. In the next chapter, we test our model and solution algorithm on problems from a major railroad.
Chapter 5

Robust Railroad Blocking: A Case Study

In this chapter, we provide a case study for incorporating uncertainties in railroad blocking. Due to the available information in the data provided by a major railroad, we focus on the problem of capturing variations in demand, with deterministic supply.

5.1 Alternative Blocking Plans

In the following sections, we define and classify our blocking plans as Static or Dynamic.

5.1.1 Static Blocking Plans

A static blocking plan requires that the same blocking plan be executed every day, regardless of realized demands. Although a static plan achieves the highest possible level of plan consistency, the price is that the plan might be far away from the optimal plans for individual scenarios.

Depending on the number of scenarios considered, there are several ways to generate a static blocking plan, namely:

1. If the blocking plan is generated based on a single scenario, we can solve a single
scenario railroad blocking problem for a particular realization of demand, such as average demand, maximum demand or the demand on a certain day (e.g. demand on a peak day);

2. If the blocking plan is based on multiple scenarios, we can solve the multiple scenario blocking model discussed in Chapter 4, with sufficiently high plan change costs.

5.1.2 Dynamic Blocking Plans

A dynamic blocking plan balances flow costs and plan change costs to achieve a plan with the lowest expected total costs. The resulting dynamic plan includes two parts: the core plan that is used to determine necessary resources to execute the plan; and the scenario specific plans that modify the core plan to optimize for individual scenarios.

We refer to a specific dynamic blocking plan as a flexible plan if the blocks can be added or removed freely in order to adapt daily blocking plans to optimize for each specific day’s demands without incurring any costs. In fact, a flexible plan is a set of optimal blocking plans, one for each day. Therefore, flexible plans have the lowest possible costs. Nonetheless, execution of a flexible plan in the complex railroad operating environment is very costly and even impossible because:

1. The flexible plan requires frequent block changes in the yards, causing confusion in day to day operations; and

2. Frequent blocking plan changes affect other tasks, such as humping and line operations.

Consequently, it is unrealistic to expect that flexible plans can be executed in today’s railroads.

In general, the generation of a dynamic blocking plan depends on specific costs. If plan change costs are relatively high, indicating that plan consistency is the main objective in plan development, then adjustments to the core plan become moderate.
and the resulting blocking plan is close to a static plan. Alternatively, if plan change
costs are relatively low, indicating that routing efficiency becomes more important,
then individual plans are adjusted frequently to minimize routing costs for specific
demand scenarios, leading to a blocking plan close to the flexible plan. In the following
case studies, we illustrate how dynamic plans change as change costs are altered.

5.2 Implementation Issues

In this chapter, we evaluate multiple scenario optimization techniques using railroad
blocking data from a Class I U.S. railroad. We consider deterministic yard capacities
and link distances, and demand that varies daily. We use CPLEX version 4.0 to
implement our algorithm on a SGI workstation with 256 Megabytes of RAM, running
IRIX version 6.2.

5.2.1 Daily Variations in Demand

As discussed in Chapter 4, multiple scenario models increase problem size and solution
complexity dramatically, and are limited in the problem size that can be handled
by given computer hardware. In the following experiments, we apply our multiple
scenario models to blocking problems on a small network with 51 terminals.

Commodity information collected in the data set include average demand and
daily variation factors in a week. That is, each day’s demand in a typical week can be
calculated by multiplying average demand by the appropriate daily variation factor.
In Table 5.1, we list the daily demands over the network in the week, where, we can
see, the demand is low on Monday, and reaches the peak on Friday. Observing the
daily demands in the week, we notice that the total number of commodities with
non-zero flow on some day is 1,089, 28% more than the daily average number of
commodities with non-zero flow. This occurs because not all commodities have posi-
tive demand every day and each day, the set of commodities with no demand might
differ. In practice, railroads develop a blocking plan based on the average demand.
Average demand is calculated by summing the daily demands and dividing by 7 (i.e.,
average demand = $\sum_{t=1}^{7} demand_{day_t}/7$. Since the number of commodities considered in generating the blocking plan is greater than the number of commodities on any single day, some terminals have to save space and blocks for some nonexistent commodities on certain days. This causes inefficiency and under-utilization of resources in terminals.

### 5.2.2 Plan Change Costs

One set of important parameters in generating dynamic blocking plans are the plan change costs, as they are the factors that balance routing efficiency and plan consistency. Given our objective function, adding or removing a block is justified only if the savings in classifications exceeds the unit cost of plan change. Unfortunately, our Class I railroad could not provide reliable estimates of these costs. Consequently, we use a range of plan change costs in our experiments to assess the effects of various cost assumptions.

### 5.2.3 Convergence Criteria

Although we can solve blocking problems with a single scenario to optimality within hours, solving multiple scenario models for the same problem is so memory and run time intensive that it is often unrealistic to expect optimal solutions in many hours. In order to maintain consistency and comparability among approaches, we adopt a universal stopping criterion of terminating iterations if the gap between the upper and lower bounds (that is, $\frac{\text{upper bound} - \text{lower bound}}{\text{lower bound}}$) is within 1%.
<table>
<thead>
<tr>
<th>Problem</th>
<th># Blocks</th>
<th>Cost</th>
<th>B-Time</th>
<th>F-Time</th>
<th>Iter</th>
<th>Total Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUN</td>
<td>795</td>
<td>104,520</td>
<td>4</td>
<td>13</td>
<td>27</td>
<td>459</td>
</tr>
<tr>
<td>MON</td>
<td>760</td>
<td>79,944</td>
<td>3</td>
<td>11</td>
<td>22</td>
<td>308</td>
</tr>
<tr>
<td>TUE</td>
<td>786</td>
<td>99,786</td>
<td>3</td>
<td>12</td>
<td>25</td>
<td>375</td>
</tr>
<tr>
<td>WED</td>
<td>790</td>
<td>124,299</td>
<td>3.5</td>
<td>13</td>
<td>28</td>
<td>462</td>
</tr>
<tr>
<td>THU</td>
<td>827</td>
<td>133,497</td>
<td>5</td>
<td>15</td>
<td>29</td>
<td>580</td>
</tr>
<tr>
<td>FRI</td>
<td>820</td>
<td>137,172</td>
<td>5</td>
<td>14</td>
<td>28</td>
<td>532</td>
</tr>
<tr>
<td>SAT</td>
<td>807</td>
<td>129,381</td>
<td>4</td>
<td>13</td>
<td>26</td>
<td>442</td>
</tr>
<tr>
<td>Average Demand</td>
<td>866</td>
<td>120,909</td>
<td>5.5</td>
<td>30</td>
<td>35</td>
<td>1242.5</td>
</tr>
</tbody>
</table>

Table 5.2: Network Design Solutions Under a Single-Scenario

5.3 Computational Results

5.3.1 Single-Scenario Blocking Problems

In Table 5.2, we provide the optimal solutions to our deterministic blocking model using average demand and individual daily demands. # Blocks and Cost represent the number of blocks and flow costs in the best solution generated; B-Time and F-Time represent the average run times in the BLOCK and FLOW subproblems, respectively; Iter represents the number of iterations; and Total Time represents the total run time in seconds. Currently, a blocking plan based on average demand is adopted by the railroad as a static plan, and the associated cost is 120,909. As a comparison, the average daily cost for the flexible plan is 115,514, 4.5% less than that obtained by solving the single-scenario model with average demand. However, we will show in the following experiments that objective function value for the average demand single-scenario model might greatly underestimate realized costs.

5.3.2 Single-Scenario Blocking Plan Evaluation

In this experiment, we evaluate the expected realized costs and the planned costs estimated by solving single-scenario models for a set of static blocking plans. After solving a model based on a certain single-scenario demand, there is an objective function value associated with the plan, indicating the total number of classifications
necessary to deliver the traffic in the demand scenario. We refer to the objective value from these models as the planned cost based on the single scenario. Flowing a certain daily traffic using the blocks in the blocking plan generated by the single-scenario model leads to the actual cost (the number of classifications) to deliver the traffic on that day. Taking the average over realized costs in the week yields the expected daily costs for the demands in the week. We refer to the expected daily costs as the realized cost. Clearly, the planned cost based on single scenarios might be far from the costs that will be realized in delivering the traffic in the week. Since we know the realized costs are the actual costs to the railroads, the realized costs not the planned costs will be compared for each evaluated solution.

The single scenarios we consider include: average demand for each commodity, maximum demand (the maximum among the seven daily demands) for each commodity, peak traffic day (Friday) demand, medium traffic day (Sunday) demand and low traffic day (Monday) demand. We summarize our comparisons in Figure 5-1. Notice that not all blocking plans based on single scenarios are feasible for all demand scenarios. This occurs because the single scenarios considered might not cover all potential shipments in the week so that some shipments might not have a connected blocking path (a sequence of blocks in the blocking plans based on single scenarios). To assure fair comparisons, we add 50 classifications for any car left unserved.

From the comparisons shown in the figure, we can see that there exists a huge gap between the realized cost and the planned cost from the deterministic model for each case. Except for the case based on the maximum demand, the realized costs for all other cases are much higher than the planned costs. For example, the expected realized cost is 74,867 (61%) higher than the planned cost based on average demand. This suggests the following:

- The planned costs for solutions to deterministic models should not be used to estimate actual costs when uncertainty exists. A low cost deterministic plan could lead to high realized costs, as highlighted in the case based on low traffic day demand, where the gap is as high as 76,274, about 60% over the planned cost. Realized costs must always be the basis for evaluation of different plans;
If a static blocking plan is generated based on a single demand scenario, this scenario should have wide coverage of all potential traffic OD pairs. In this experiment, the lowest realized costs are generated by the two cases considering all commodity OD pairs, i.e., average demand and maximum demand cases, and these costs are significantly lower than those based on the partial commodity set, i.e., demands on peak, medium and low traffic days, respectively. This occurs because the blocking plan based on the partial commodity set uses all existing capacities for classifying the traffic in the partial commodity set. Even though the blocking plan might specify efficient blocking paths for some commodities, it might cause circuitous paths for the commodities not in the set of considered commodities.

5.3.3 Robust Planning Improvements

The above experiment shows that the lowest realized costs are obtained with the single scenarios based on the average demand calculation defined earlier. In the
following experiments, we show how the realized costs can be reduced further through robust planning. First, we generate a static blocking plan by incorporating all seven daily demand scenarios in our multiple scenario model. This problem can be solved using our robust blocking model with sufficiently high plan change costs. Then, we generate a dynamic blocking plan by allowing the plan to be changed. Change costs are equivalent to 20 classifications for each added block and 10 classifications for each removed block. This problem can be solved by applying the robust blocking model detailed in Chapter 4. We summarize cost comparisons in Figure 5-2, where RC_AVG and PC_AVG represent the realized and planned costs for blocking plans based on average demand, and RC_STA and RC_DYN represent the costs from the static and dynamic blocking plans described above. From the figure, we observe that:

- The expected realized cost (120,084) of the static blocking plan based on all daily scenarios is not only lower than the realized cost (197,183) based on the average demand for all commodities (as expected), it is also lower than the planned cost for the average demand scenario (120,909). This suggests that utilizing detailed demand information can improve upon even plan costs.

- Realized costs can be further reduced by adopting dynamic plans (from 120,084 to 117,925). The improvement over the static plan is due to plan changes which enable more efficient routings to be identified for each demand scenario. The cost reduction from the static plan to the dynamic plan for multiple scenarios is not significant compared to the cost reduction from the single scenario to multiple scenarios. Nonetheless, it is justified to consider dynamic plans since the computational efforts for multiple scenario models are similar for static and dynamic planning.

The above study shows that even though the static blocking plan based on the average demand scenario leads to the lowest realized cost among all single scenario models, the realized cost can be reduced through robust planning by incorporating multiple scenarios and allowing plans to change.
In addition to the extra computational effort, there are two difficulties in generating and evaluating robust blocking plans, namely:

1. Not all demand scenarios are available at the time when the blocking plan has to be generated; and

2. There are no reliable estimates for plan change costs from railroads.

Therefore, we establish two sets of cases to evaluate how individual scenario selection and plan change costs affect the generation of robust blocking plans.

**Impacts of Scenario Selection on Robust Planning**

In the following experiment, we show how the robust blocking plan changes as more and more demand information is included in generating the blocking plan. We sort daily demands by volume and add the daily demands gradually. In Case I, we only consider the demand on Thursday, the one with the largest daily volume. Then, we incrementally consider the demand for Friday, Saturday, Sunday, Wednesday, Tuesday and Monday to create Cases II through VII, respectively. Again, we assume that the individual plans are allowed to change with costs of 20 and 10 for block addition and
removal, respectively. We summarize the results in Figure 5-3, including both flow costs and plan change costs for each case.

From Figure 5-3, we observe:

- As the number of scenarios increases, the realized costs are decreased because the need for plan adjustments decreases dramatically;

- With additional demand information included, flow cost decreases gradually as more low cost routings are identified through individual plan adjustments;

- The marginal contributions of additional demand information, in terms of cost reduction, decreases as more and more demand scenarios are included. For example, the realized costs for the blocking plan based on 4 daily demands (Wednesday through Sunday) is only 4% higher than the costs for the plan based on all 7 daily scenarios even though the computational effort and storage requirements are significantly larger in the latter case. This suggests that low realized costs can be achieved by incorporating only a subset of all demand scenarios. In a separate case study, we will show how to select scenarios efficiently when not all scenarios are considered.
Table 5.3: Plan Change Cost Assumptions in Robust Blocking Plans

<table>
<thead>
<tr>
<th>Problem</th>
<th>Core Size</th>
<th>C-Cost</th>
<th>F-Cost</th>
<th>B-Time</th>
<th>F-Time</th>
<th>Iter</th>
<th>Total Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>One</td>
<td>813</td>
<td>0</td>
<td>120,084</td>
<td>202</td>
<td>161</td>
<td>137</td>
<td>49,731</td>
</tr>
<tr>
<td>Two</td>
<td>-</td>
<td>0</td>
<td>115,514</td>
<td>379</td>
<td>205</td>
<td>176</td>
<td>102,784</td>
</tr>
<tr>
<td>Three</td>
<td>1,027</td>
<td>0</td>
<td>115,514</td>
<td>228</td>
<td>156</td>
<td>169</td>
<td>64,896</td>
</tr>
<tr>
<td>Four</td>
<td>1,027</td>
<td>0</td>
<td>115,514</td>
<td>263</td>
<td>158</td>
<td>181</td>
<td>76,201</td>
</tr>
<tr>
<td>Five</td>
<td>812</td>
<td>240</td>
<td>116,795</td>
<td>232</td>
<td>156</td>
<td>131</td>
<td>50,828</td>
</tr>
<tr>
<td>Six</td>
<td>811</td>
<td>210</td>
<td>117,925</td>
<td>235</td>
<td>156</td>
<td>179</td>
<td>69,989</td>
</tr>
<tr>
<td>Seven</td>
<td>813</td>
<td>100</td>
<td>119,796</td>
<td>230</td>
<td>152</td>
<td>208</td>
<td>79,456</td>
</tr>
<tr>
<td>Eight</td>
<td>818</td>
<td>60</td>
<td>118,776</td>
<td>216</td>
<td>160</td>
<td>151</td>
<td>56,776</td>
</tr>
<tr>
<td>Nine</td>
<td>813</td>
<td>0</td>
<td>120,084</td>
<td>209</td>
<td>154</td>
<td>187</td>
<td>67,881</td>
</tr>
</tbody>
</table>

Table 5.4: Robust Plans under Different Change Cost Assumptions

Impacts of Plan Change Costs on Robust Planning

In the following experiment, we consider all demand scenarios and we investigate the impacts of change costs on robust planning. Table 5.3 summarizes the range of plan change costs in this experiment. In Case 1, the plan change costs are sufficiently large, and the resulting blocking plan is a static plan. In Case 2, the plan change costs are 0, and the resulting blocking plan is a flexible plan. In Cases 3 to 9, the plan change costs assume some intermediate values, and the resulting blocking plans are a set of dynamic plans.

In Table 5.4, we summarize the computational results for the above cases, terminating our algorithms when the gaps between upper and lower bounds are within 3%. 
Core Size denotes the number of blocks in the core plan, C-Cost denotes the total change costs, F-Cost denotes the total flow costs, B-Time denotes the average run time for block subproblem, F-Time denotes the average run time for flow subproblem, Iter denotes the total number of iterations, and Total Time denotes the total run time.
From Table 5.4, we observe:

1. When only block addition costs are imposed (Cases 3 and 4), the core plan includes all blocks that are used in any scenario, and individual plans delete all blocks except what they use. The costs of the resulting robust plans are identical to the one in the flexible plan (Case 2).

2. When both arc addition and deletion costs are positive, the size of the core plan drops significantly since it is costly to delete a block from the core plan. Only the most common blocks remain in the core plan with other blocks being added to or deleted from the individual scenario plans when a net reduction in costs is possible.

3. Compared to run times for single-scenario models, in the multiple scenario problems, the flow subproblems' run times are approximately equal to the sum of the run times for each scenario specific flow subproblem, indicating that run time increases linearly with the number of scenarios. This occurs because the flow subproblem can be decomposed by scenario. However, run times for multiple scenario block subproblems are considerably greater than the sum of run times for the single scenario block subproblems due to the increase in the number of integer variables and scenario specific constraints. As the block subproblem is not decomposable by scenario, the computational effort grows relatively rapidly as more variables and constraints are added in the multiple scenario models.

5.3.4 Robust Planning with Limited Scenarios

In the previous experiments, we have shown that the blocking plan based on multiple scenarios leads to much lower realized costs than the plan based on single scenarios. We also notice that the computational effort for generating the blocking plans (in terms of run time and storage space requirements) can be excessive for multiple scenarios. We have shown that the marginal contribution of additional demand information decreases as more and more demand scenarios are included and that low
realized cost blocking plans can be generated with a limited number of scenarios. In order to generate the blocking plan under limited scenarios, we first generate a core plan solving a blocking problem based on the selected scenario(s). Then, we reoptimize the multiple scenario model given the blocks in the core plan. The blocks in the core plan are not allowed to be changed in static planning, but they can be altered (with some costs incurred) in dynamic planning. The solution from reoptimization yields the realized costs for the problem with limited scenarios.

In the following experiments, we investigate how to select demand scenarios efficiently. We first consider the blocking plans based on single scenarios to search for the single scenario that leads to the lowest realized cost. Then, we investigate how to select additional scenarios to be combined with the lowest cost scenario to further reduce realized cost effectively.

Robust Planning with a Single Scenario

From the case study on single scenario blocking plans, we conclude that wide coverage of all potential traffic is essential in generating low cost blocking plans. Therefore, all scenarios considered in this experiment include all commodities, and the difference among the scenarios exists in the volume of each commodity. Among the single scenarios we consider, average daily demand and maximum daily demand have been discussed in the earlier case studies. We refer to the difference between the demand volumes in the average demand and maximum demand cases as the volume gap. The other three scenarios we consider have demand volume between the average daily demand (AVG) and the maximum daily demand (MAX) scenarios. The demand volumes for each commodity in these three scenarios (AVG+10, AVG+20 and AVG+30) are 10%, 20% and 30% of the volume gap over the average demand for each commodity.

Again, we generate the blocking plans based on the above five single scenarios, and flow the daily traffic based on the blocking plans to calculate the realized costs for each plan. Figure 5-4 shows the comparisons of realized costs for the static blocking plans based on the above single scenarios. From the comparisons, we observe that
realized costs of plans based on average demand and maximum demand are higher than those of the plans based on scenarios with intermediate demand volumes. This occurs because the lowest cost blocking path for each potential commodity is based on the specific volume for the commodity. Therefore, the lowest cost blocking paths based on the average demand scenario might not be feasible with higher volumes due to the volume capacity constraints. In these cases, blocking paths with more classifications have to be used to flow these commodities, thereby increasing realized costs. On the other hand, the lowest cost blocking paths tend to be more indirect when the blocking plan is generated based on the maximum demand since a high level of demand must be accommodated over a limited capacity network. So, the blocking plan based on maximum demand might ignore the lower cost paths for the low volume commodities on certain days. Hence, the lowest cost blocking plan is usually generated by a scenario with demand volume between average and maximum for each commodity. Comparing realized costs, we notice that the lowest realized cost is generated by the blocking plan based on the scenario with volumes set at 20% of the volume gap over the average demand.

Figure 5-5 shows the flow and change cost breakdowns when plan adjustments
Figure 5-5: Realized Cost Comparisons for Dynamic Plans Based on Single Scenarios are allowed. Change costs are set at 20 classifications for each added block and 10 classifications for each deleted block. As in the above experiment on static plans, the blocking plans based on average demand and maximum demand scenarios yield higher realized costs than the plans based on intermediate demands, and the lowest cost blocking plan is again generated based on the scenario with volumes set at 20% of the volume gap over the average demand. However, the differences in the total realized costs among different cases are moderate compared to those in the static plan case study. This suggests that inefficient routings arising due to incomplete demand information can be corrected through dynamic plan adjustments. Compared to static plans, the realized costs for dynamic plans are significantly lower. This occurs because plan adjustments produce individual plans that are more suitable for the realized demand scenarios. Even though plan changes incur costs, the savings in operating costs is so large that they justify the costs of plan changes.
Robust Planning with Added Scenarios

In the above case study on robust planning with single scenarios, we find that the lowest cost single-scenario blocking plan is generated by the scenario AVG-20. In this case study, we show how to add scenarios efficiently when more than one scenario can be considered in generating a blocking plan. The way we proceed in this study is to compare realized daily and planned costs for each day and select scenarios to add in decreasing order of their differences between realized and planned costs. Table 5.5 summarizes the realized costs and planned costs for the daily demand scenarios, one for each day in the week. We can see that the largest difference between realized and planned costs occurs on Day 6, the peak traffic day. Therefore, we begin by generating a blocking plan based on the two scenarios, AVG-20 and Day 6 demands. Figure 5-6 shows the realized costs for static and dynamic blocking plans based on the AVG-20 and Day 6 demand, the AVG+20 single scenario, and the 7 daily scenarios. From the figure, we can see that the realized costs when two scenarios are considered are reduced dramatically compared to the best single scenario case. The resulting realized cost is very close to the realized cost for the seven scenarios plan, the lowest cost plan for our problem. It suggests that efficient blocking plans can be generated with limited scenarios if the scenarios are selected intelligently. Compared to static plans, the realized costs for dynamic plans are relatively close. When allowing the plans to change, the realized costs based on the single scenario (AVG+20) is 129,579, less than 10% higher than the costs based on seven day scenarios. This suggests that the marginal contributions of additional scenarios are moderate for dynamic plans.

5.3.5 Simulation Case Study

In the above case studies, we assume that we have complete information on demand on each day (scenario) when generating the blocking plan. However, sometimes, the exact information on demand may not be available or cannot be fully utilized at the time when the plan has to be generated. Examples include:
<table>
<thead>
<tr>
<th>Day</th>
<th>Realized Cost</th>
<th>Planned Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>174,889</td>
<td>104,520</td>
</tr>
<tr>
<td>2</td>
<td>81,501</td>
<td>79,944</td>
</tr>
<tr>
<td>3</td>
<td>157,184</td>
<td>99,786</td>
</tr>
<tr>
<td>4</td>
<td>126,868</td>
<td>124,299</td>
</tr>
<tr>
<td>5</td>
<td>188,343</td>
<td>133,497</td>
</tr>
<tr>
<td>6</td>
<td>564,039</td>
<td>137,172</td>
</tr>
<tr>
<td>7</td>
<td>140,217</td>
<td>129,381</td>
</tr>
</tbody>
</table>

Table 5.5: Realized vs. Planned Costs for Daily Demand Scenarios

Figure 5-6: Realized Cost Comparisons for Plans with Different Scenario Selection
• The number of potential scenarios in one month are so many that it is not realistic to incorporate all scenarios in generating the blocking plan. Instead, we have to generate the blocking plan based on a reduced set of representative scenarios. The representative scenarios could be the demand in one week or the average demand on each day of the week.

• The demand in a certain period is so volatile that it is not possible to represent demand with a limited number of discrete scenarios. Instead, we know only the approximate distribution of demands.

Unlike the above case studies where we have perfect information that can be represented by discrete demand scenarios, we have only partial information on demand in this case study. The partial information includes the expected value and fluctuation range for each commodity on any single day. The specific demand is generated by simulation under the assumption that the distribution of demand follows the uniform distribution. In order to evaluate blocking plans under different perspectives, we generate random demands in a month (4 weeks) with the following features:

• Week 1 (High10) is a High Demand Week, where the average daily demand for each commodity in this week is 10% higher than the average demand for the commodity in the month;

• Week 2 (Low10) is a Low Demand Week, where the average daily demand for each commodity in the week is 10% lower than the average demand for the commodity in the month;

• Week 3 (Var10) is a Low Variance Demand Week, where the range of fluctuation is between 10% above and below the average demand; and

• Week 4 (Var20) is a High Variance Demand Week, where the range of fluctuation is between 20% above and below the average demand.

Given the above description, the average daily demand in the month is the same as the average daily demand in the previous case studies with perfect information.
Table 5.6: Comparisons of Realized Costs with Different Static Blocking Plans

Through the experiments in this case study, we want to evaluate the following:

1. The performance of blocking plans with partial demand information relative to those with perfect demand information;

2. The effects of demand variability on the performance of the blocking plans;

3. The performance of different blocking plans based on different scenarios.

As in previous case studies, we consider blocking plans based on single scenarios and multiple scenarios. The single-scenario case studies consider the full set of commodities on any given day in the month. *Case AVG* represents the scenario with volume equal to the average daily demand over the month, and *Case MAX* represents the scenario with the maximum daily demand for each commodity in the month. The multiple scenario case studies consider seven daily demand scenarios. *Case AVG-7* represents seven scenarios with demand equal to the average demand for each day in the week. *Case LOW* represents seven scenarios with daily demand equal to 10% lower than the average demand for each day in the week, and *Case HIGH* represents seven scenarios with daily demand equal to 10% over the average demand for each day in the week. Table 5.6 summarizes the average daily realized costs for the four weeks' demands in the month under the various static blocking plans based on single and multiple scenarios.

From the above results, we observe that

- Compared to the average realized costs in the case study with perfect demand information, the realized costs for the single scenarios (AVG and MAX) and
multiple scenarios (AVG-7) cases are much higher. The differences show the realized cost savings when perfect information is available;

- The realized costs for the low demand case are very similar for the different blocking plans. This suggests that low demand scenarios are not important in affecting the resulting realized costs, and a robust blocking plan generator should focus on the scenarios with high demand and fluctuation;

- As expected, realized costs are much higher in the weeks when demand variation is higher, even though the average daily demand is the same for each commodity. This indicates that large variations in demand leads to dramatic increases in realized costs. Therefore, railroads should effectively manage the sources of demand variability to reduce demand fluctuations;

- The blocking plan based on a high demand week yields the lowest average realized costs, 271,296, or 20% lower than the cost for the next best plan based on scenarios with average demand. This suggests that demand scenarios with higher demand volumes should be used in generating low cost blocking plans, rather than average demand scenarios.

Table 5.7 summarizes the average daily realized costs for the demands in the month when plans are allowed to change with costs of 20 classifications for each added block and 10 classifications for each deleted block. Results in the table show that the realized costs are greatly reduced under dynamic blocking plans and the differences in realized costs among different blocking plans become moderate. This occurs because dynamic plan adjustments identify more efficient routings when realized demand scenarios are known. Compared to the case study with perfect demand information, the reduction in realized costs relative to planned costs is more dramatic in this experiment. This suggests that allowing plan adjustments is more important when perfect information is not available at the time the plan is developed.
Table 5.7: Comparisons of Realized Costs with Different Dynamic Blocking Plans

5.4 Summary

In the chapter, we test the model and solution approach discussed in Chapter 4 on problems from a major railroad where the demand is stochastic. From the comparisons between planned costs and realized costs, we have shown that the realized costs should be used to evaluate different blocking plans. Through various experiments, we show how the realized costs can be reduced by introducing multi-scenario and dynamic planning. We provide case studies to show the effects of scenario selection and plan change costs on blocking plans. In a simulation case study, we illustrate impacts of demand fluctuations on the solution quality from multi-scenario models and show that a low cost blocking plan can be generated using scenarios with higher demand volumes.
Chapter 6

Contributions and Future Directions

6.1 Contributions

In the thesis, we study the railroad blocking problem. We provide models and algorithms to find near-optimal blocking plans. This requires a combination of state-of-art modeling and algorithmic approaches, including:

- **Lagrangian Relaxation and Decomposition:** By relaxing the difficult forcing constraints, Lagrangian relaxation enables us to decompose the complex mixed integer programming problem into two simple subproblems that can be solved separately, thereby greatly reducing the storage requirement and computational effort to solve the problem.

- **Dual Ascent:** The dual feasible solution generated by the dual-ascent method provides an advanced start for the subgradient optimization procedure. The algorithm converges much more quickly with this advanced start and generates better feasible solutions.

- **Valid Cut Generation:** The added cut connectivity constraints, combined with Lagrangian relaxation, yields tighter lower bounds than those generated
from the alternative LP relaxation. Also, these constraints require a connected path for each commodity, making it possible to generate feasible solutions efficiently when volume constraints do not exist.

- **Column Generation**: Column generation is applied to the flow subproblems, which are multicommodity flow problems with a huge number of potential paths. Using column generation, we can efficiently solve large-scale LPs in reasonable amounts of time.

In solving these huge mathematical programs, we evaluate the trade-off between problem size and computational difficulty. We show that the level of network fidelity has a big impact on solution quality, and a low cost blocking plan can be generated on high fidelity networks with commodities and routes consolidated. Compared to the current railroad operations, our blocking plan could potentially reduce the number of classifications and operating costs by 10%. Further, service levels could be improved by reducing the average trip time for shipments by 4.5%.

We also evaluate the effects of considering stochasticity in input data, specifically daily demands. We present multiple scenario models and describe how to solve them by modifying our solution algorithm for the single scenario model. We find that the blocking plan capturing variations in demands reduces expected realized costs over 60%, compared to the plan generated based on a single scenario (the current practice in railroad planning). The costs can be further reduced by adopting dynamic blocking plans that allow plan adjustments when realizations of uncertainties become available.

### 6.2 Future Directions

#### 6.2.1 Model Development

- Our multiple scenario model does not allow certain shipments to be serviced later. By adding probabilistic constraints, service reliability issues and impacts could be evaluated.
In expanding from a single to a multiple scenario model, many integer variables and constraints are added, resulting in a computationally burdensome blocking problem. Storage and run time requirements quickly become excessive. Therefore, a more condensed formulation and fast solution algorithms should be pursued for multiple scenario models.

6.2.2 Algorithmic Development

- Experiments show that the combination of Lagrangian relaxation and Benders decomposition, referred to as Cross decomposition (Van Roy [1986]), can lead to noticeable improvements in solving general network design problems. This solution approach as well as the L-shaped method should be applied and evaluated for the multiple scenario blocking models.

- It would be interesting to evaluate the performance of parallel algorithms in our solution approach. Lagrangian relaxation decomposes the problem into subproblems, which may be solved in parallel.

6.2.3 Application Context

Developing integrated models and algorithms for railroad blocking, train scheduling, and other operating problems would produce more globally optimal plans for the railroads.
Bibliography


