LARGE SCALE OPTIMIZATION METHODS FOR FLEET MANAGEMENT IN LONG-HAUL TRANSPORTATION NETWORKS

A Thesis
Presented to
The Academic Faculty

by

İlke Bakır

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Industrial and Systems Engineering

Georgia Institute of Technology
December 2017

LARGE SCALE OPTIMIZATION METHODS FOR FLEET MANAGEMENT IN LONG-HAUL TRANSPORTATION NETWORKS

Approved by:

Professor Alan Erera, Advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Alejandro Toriello
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Natashia Boland
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor John-Paul Clarke
School of Aerospace Engineering
Georgia Institute of Technology

Professor Martin Savelsbergh
School of Industrial and Systems Engineering
Georgia Institute of Technology

Date Approved: 1 November 2017

To my family,

Sevil, Kemal, Alihan, and Bulut,

for being my inspiration and the source of my inner strength my entire life...

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my advisor, Dr. Alan Erera, for supporting me in so many ways throughout the ups and downs of my PhD. His invaluable guidance made me the researcher I am today. His quickness in identifying mistakes and missing pieces, and his skill in always asking the right questions have been a huge help in shaping and structuring this thesis.

I would like to extend my sincere thanks to Dr. Natashia Boland, who has been an extraordinary mentor, research collaborator, and committee member, for her endless support. Every time I sought her help or advice, she went above and beyond to provide the help/support I needed. For that, I will be forever grateful. I feel very fortunate to have the opportunity to work with her. I am extremely grateful to Dr. Martin Savelsbergh for his collaboration and mentorship, and for devoting his time to serve in my thesis committee. His guidance in research matters, as well as my professional decisions, has helped to shape my career path. It has been a privilege working with him. I thank Dr. Alejandro Toriello for providing valuable insights on how to enhance my thesis work and suggestions about future research directions, while serving in my thesis committee. I would also like to thank Dr. John-Paul Clarke for generously devoting his time to serve in my thesis committee, and contributing many interesting ideas on potential applications and extensions of my thesis work.

Being a PhD student at ISyE has been an extremely valuable learning experience from so many perspectives. It taught me the value of perseverance, will power, and patience; and also that of friendship, peer support, and kindness. Most importantly, it taught me how much we needed each other (as fellow PhD students) to pick ourselves up at times of difficulty, which was bound to happen at least once to all of

us over the course of the PhD. With that in mind, I would like to acknowledge my (hopefully lifelong) friends from ISyE: Ezgi, for being "my mother", my scheduler, anger buddy, and hug/tear factory; Tuğçe, for always being the voice of reason when I fail to hear the one inside my head; Bahar, for always being an inspiration and the perfect combination of fun and logic; Murat, for always managing to make me like and appreciate him despite his obvious annoyances; Işıl, for being so perceptive and empathetic; İdil, for understanding all of my quirks and tirelessly listening to my complaints; Mallory, for being the best and least politically correct bartender in Atlanta; Oğuzhan, for his encouragement at my times of low confidence; Burak, for his willingness to provide academic support; Beste, for checking up on me to make sure Im alive in most critical times; and Şeyma, for being the most fun person ISyE has ever seen. I have also been very fortunate to have met and spent time with James, Andres, Stefania, Daniel, Diego, Tim, Erin, Brian, Alvaro, Evren, Çağlar, Satya, and many more wonderful ISyE people that I have crossed paths with over the years.

Though I spent a considerable proportion of my time in Atlanta at ISyE, I managed to convince some really interesting and kind people outside of ISyE to be friends with me. Thanks so much to Fatma and Michael, for their moral and physical support, for feeding me and taking care of me when I needed it the most; Didem and Can, for always being there for me and Bulut, and for always having a unique perspective on life matters; and Gamze, Gökçin, Can Erdoğan, and Şerife for being a part of my Atlanta family. I would like to thank Melis, Ayşe Nur, Sezin, Sibel, Gizem, Pelin, and Yeliz for sharing happy or miserable, but nevertheless memorable moments either in Atlanta, at random locations in the world, or through Skype.

Words would fall short to express how thankful I am to my parents Sevil and Kemal, and my little brother Alihan; but I will try. If it wasn't from firsthand experience, it would have been impossible to imagine this much support can come from an ocean away. They always knew the right things to say (or not say), found

the most creative ways to share my burden, and constantly let me know (with or without words) that they are there for me with their unconditional love. I could not have done this without them. This thesis is theirs as much as it is mine.

TABLE OF CONTENTS

DEDICATION . . . . . **iii**

ACKNOWLEDGEMENTS . . . . . **iv**

LIST OF TABLES . . . . . **x**

LIST OF FIGURES . . . . . **xii**

SUMMARY . . . . . **xiii**

I INTRODUCTION **1**

1.1 Robust Empty Repositioning 2

1.2 Integrated Fleet Management 4

1.3 Partition Bounds for Multistage Stochastic Programs 5

II A ROBUST ROLLING HORIZON FRAMEWORK FOR EMPTY REPOSITIONING **7**

2.1 Introduction 7

2.2 Literature Review 11

2.3 Methodology 14

2.4 Computational Experiments 23

2.4.1 Problem Instances 23

2.4.2 Testing Framework 24

2.4.3 Computational Results 26

2.5 Conclusion 36

III AN INTEGRATED FLEET MANAGEMENT MODEL INTRODUCING ALTERNATIVE FUEL TRUCKS **38**

3.1 Introduction 38

3.2 Literature Review 41

3.3 Integrated Fleet Management Model 44

3.3.1 Integrated Fleet Management Model with Truck-mile Targets 45

3.3.2 Integrated Fleet Management Model with Loaded Truck-mile Targets . . . 50
3.3.3 Optimal Solutions of the Integrated Fleet Management Model with Loaded Truck-mile Targets ($\mathbf{P}'$) . . . 52
3.4 Solution Approaches . . . 63
3.4.1 Benders' Decomposition . . . 63
3.4.2 Variable Neighborhood Search . . . 69
3.5 Computational Experiments . . . 70
3.5.1 Benders' Decomposition . . . 72
3.5.2 Variable Neighborhood Search . . . 74
3.6 Conclusion . . . 75

IV SCENARIO SET PARTITION DUAL BOUNDS FOR MULTI-STAGE STOCHASTIC PROGRAMMING . . . 77

4.1 Literature Review . . . 80
4.2 Preliminaries and *EGSO* Bounds for Multistage Problems . . . 84
4.3 Expected Partition Bounds: A Hierarchy of Dual Bounds . . . 87
4.4 Scenario Set Partition Sampling . . . 92
4.4.1 Partition Sampling Based on Scenario Tree Structure . . . 94
4.4.2 Optimally Recombined Partitions . . . 97
4.4.3 Truncated Partition Bound Calculation . . . 98
4.5 Computational Results . . . 99
4.5.1 Test Instances . . . 99
4.5.2 Computation Time . . . 100
4.5.3 Comparing Partition Sampling with SAA . . . 103
4.5.4 Partition Bound Sampling Enhancements . . . 108
4.6 Conclusion . . . 115

V CONCLUSION . . . 117

APPENDIX A — SUPPLEMENTARY MATERIAL FOR CHAPTER II . . . 121

APPENDIX B — SUPPLEMENTARY MATERIAL FOR CHAPTER III **126**

APPENDIX C — SUPPLEMENTARY MATERIAL FOR CHAPTER IV **132**

REFERENCES **150**

LIST OF TABLES

3.1 Constant demand, constant target 54

3.2 Constant demand, increasing target 55

3.3 Constant target, region swapping demand 58

3.4 Constant target, volatile demand 59

3.5 Changing truck costs, constant demand 62

3.6 The effect of critical period cuts on Benders' convergence 75

4.1 Solution times (in seconds) for CAP 101 101

4.2 Best partition bounds (among 10 randomly sampled partitions) for CAP 101 102

4.3 Expected fraction of series/parallel computation time needed to achieve the partition bounds shown in Table 4.2 102

4.4 Best partition bound vs. lower limit of the 95% SAA confidence interval on SSLP instances. 107

4.5 Best partition bound vs. lower limit of the 95% SAA confidence interval: summary for CAP instances. 108

4.6 Difference of tree-aligned and misaligned partition bounds in a small DCAP 332 instance 111

A.1 Data associated with Figure 2.1a 123

A.2 Data associated with Figure 2.1b 123

A.3 Data associated with Figure 2.1c 123

A.4 Data associated with Figure 2.2 124

A.5 Data associated with Figure 2.3a 124

A.6 Data associated with Figure 2.3b 124

A.7 Data associated with Figure 2.5 125

A.8 Data associated with Figure 2.6 125

B.1 Problem instances used in computational experiments 126

B.2 Solution times in 10-terminal instances: Direct solution (OPT) vs. Benders' variants 127

B.3 Number of Benders' iterations in 10 terminal instances 128

B.4 Optimality gaps after 1 hour in 30-terminal instances for Gurobi (OPT) and Benders' variants 129

B.5 VNS Optimality gaps and solution time (seconds) performance in 10-terminal instances 130

B.6 VNS Optimality gaps and solution time (seconds) performance in 30-terminal instances 131

C.1 Partition bound vs. lower end of the 95% SAA confidence interval, CAP instances, $q = 5$. 139

C.2 Partition bound vs. lower end of the 95% SAA confidence interval, CAP instances, $q = 10$. 140

C.3 Partition bound vs. lower end of the 95% SAA confidence interval, CAP instances, $q = 15$. 141

C.4 Partition bound vs. lower end of the 95% SAA confidence interval, CAP instances, $q = 20$. 142

C.5 Best partition bound vs. lower limit of the 95% SAA confidence interval on SSLP instances. 143

C.6 Best partition bound vs. lower limit of the 95% SAA confidence interval on DCAP 332 instances. 144

C.7 Best partition bound vs. lower limit of the 95% SAA confidence interval on DCAP 342 instances. 145

C.8 Tree-aligned vs. misaligned partition bounds in DCAP 332 1x4x2 with $A = 8a + 4\Lambda + 12\lambda = 2.7$ 146

C.9 Tree-aligned vs. misaligned partition bounds in DCAP 332 1x4x2 with $A = 8a + 4\Lambda + 12\lambda = 4.6$ 146

C.10 Tree-aligned vs. misaligned partition bounds in DCAP 332 1x4x2 with $A = 8a + 4\Lambda + 12\lambda = 9.0$ 146

C.11 Upper and lower bounds for ρ when $L = 240$ 149

LIST OF FIGURES

2.1 Effect of R on outsourcing and transportation cost ($k = 1/3$, averaged over 30 instances) 28

2.2 Effect of N in a 30-terminal geography (averaged over 30 instances) (Marker size is proportional to the natural logarithm of total computation time.) 30

2.3 Effect of k (averaged over 30 instances) (Marker size is proportional to total computation time.) 31

2.4 Sharing group configurations for 30-terminal geography 33

2.5 Effect of inbound/outbound loaded demand balance (30 terminals, $k = 1/3$, averaged over 30 instances) 34

2.6 Effect of sharing group size (30 terminals, $k = 1/3$, averaged over 30 instances) 35

3.1 Greenhouse gas emissions by sector 39

3.2 Sample network with 6 terminals and 3 routes 48

3.3 Truck replacement costs when the truck costs change over time 61

3.4 Progression of Benders' iterations in BEN-0, BEN-3, and BEN-6 . . . 72

4.1 Best partition vs. expected partition 93

4.2 Example tree-aligned partitions for a tree with $|\Xi^1 \times \Xi^2 \times \Xi^3| = 1 \times 4 \times 3$ and $q = 2$ 95

4.3 Three alternative methods to construct partitions based on scenario tree structure 97

4.4 CAP 124 Partition sampling vs. SAA ($K^{max} = 30$) 104

4.5 DCAP 342 1x15x20 Partition sampling (with tree-aligned partitioning strategy and optimal recombination) vs. SAA ($K^{max} = 50$) 105

4.6 Example scenario tree with $|\Xi^1 \times \Xi^2 \times \Xi^3| = 1 \times 2 \times 4$ for DCAP instances 110

4.7 Partition bounds with different truncation strategies. DCAP 332 1x15x20, $q = 5$ 113

4.8 Computational savings vs. loss in bound quality for different truncation strategies 114

4.9 Bound progression with conservative truncation vs. no truncation (same computational effort) 115

SUMMARY

In this thesis, we explore topics in large-scale deterministic and stochastic optimization methods in transportation networks, with particular emphasis on methods of addressing uncertainty in fleet management problems. It consists of studies concentrating on emerging problems in fleet management and optimization under uncertainty.

The first study, *"A Robust Rolling Horizon Framework for Empty Repositioning"*, presents a robust optimization framework for an empty repositioning problem with demand uncertainty and provides insights on how to benefit from robustness while avoiding overconservatism. In this study, we demonstrate that by using the robust framework, significant improvements in service level can be attained, when compared to solution approaches with no robustness consideration. We also analyze how the level of conservatism affects solution quality, and identify cases in which the robust approach provides the best repositioning plans. Additionally, we experiment on ways to divide large transportation networks into "sharing groups" and limiting the repositioning moves to be mostly within sharing groups. By observing the outcome of the robust framework when different sharing group policies are used, we substantiate that significant savings can be obtained by only redefining the sharing groups, without making any structural changes in the network.

The second study, *"An Integrated Fleet Management Model Introducing Alternative Fuel Trucks"*, addresses the challenge of smoothly introducing alternative fuel trucks (AFTs) into an existing fleet while making necessary structural changes and maintaining feasible operations during the transition. In this study, we develop an integrated fleet management model, which incorporates the decisions for *(i)* opening

new maintenance/fueling facilities, and *(ii)* assigning trucks to travel routes for ensuring demand satisfaction, into a fleet replacement framework. We demonstrate that the integrated model finds non-obvious solutions by making use of information that is often overlooked by most fleet replacement strategies. For finding optimal or good heuristic solutions to the integrated fleet management model efficiently, we propose a Benders decomposition framework and a Variable Neighborhood Search (VNS) algorithm. Finally, in order to demonstrate the performance of these solution methods on realistic problem instances, we conduct a comprehensive computational study.

The third study, *"Scenario Set Partition Dual Bounds for Multistage Stochastic Programming"*, focuses on finding dual bounds, called "partition bounds", in multistage stochastic programming problems with a finite number of scenarios. In this work, we propose a sampling approach that randomly samples partitions of the scenario set, and returns the best one of the associated partition bounds. We also present ways to further improve this sampling method in terms of bound quality and computational burden, such as *(i)* taking advantage of the scenario tree structure in multi-stage instances, and *(ii)* seeking "good" partitions by solving a set partitioning problem over the scenario subsets that were previously observed in the sampling procedure. Furthermore, a heuristic approach for truncating the bound calculations in order to focus computational efforts on the most promising partitions is presented. With a comprehensive computational study, we demonstrate that by using the partition sampling approach, better dual bounds can be obtained with less computational effort, when compared to a Sample Average Approximation approach.

CHAPTER I

INTRODUCTION

Freight transportation plays a key role in not only moving goods but also creating jobs, generating revenue, and consuming materials and services produced by other sectors of the economy. According to the Bureau of Transportation Statistics (BTS), the freight transportation industry employed 4.6 million people in the United States in 2014 and comprised 9.5% of the nation's economic activity as measured by gross domestic product (GDP). [24] It has been a growing sector, due to population growth and expansion in economic activity. From 1998 to 2008, world merchandise freight exports nearly tripled in value from $5.4 trillion to $16 trillion. During this period, U.S. freight exports doubled from $682 billion to $1.3 trillion. [23] The number of trucks used in commercial transportation rose 37% between 1980 and 2002, and the total vehicle miles of traveled by trucks grew from 40 billion miles to 76 billion miles in 2002. [22]

Truck transport, among other modes of transport, constitutes a big portion of the freight transportation sector, both in domestic and international arenas. In 2004, 68.8%, of all domestic freight in the U.S. was moved by trucks. Furthermore, trucks transported 56.6% and 66.2% of the trade (in monetary value) with Canada and Mexico, respectively, in 2010. [23] Trucking industry, which has been growing consistently in the recent years, has important contributions to the U.S. economy. In 2013, for-hire transportation generated $481 billion for U.S. GDP, the largest contributor of which was for-hire trucking (with 27%). Additionally, in-house transportation sector was almost as large as the for-hire sector. [24]

As trucking industry grew rapidly, managing transportation efficiently gained importance for both for-hire and in-house trucking sectors. Among the costs incurred in trucking, fuel cost and truck/trailer lease or purchase payments make up the first and third largest categories (38% and 10%, respectively, in 2013 [8]). To address the potential inefficiencies in these categories, we focus on the fleet management aspect of truck transportation in this thesis. We explore various emerging problems in fleet repositioning and fleet replacement, especially in long-haul trucking, and develop solution methods. The thesis consists of three studies concentrating on three different challenges in the area. First, we emphasize potential inefficiencies demand uncertainty may cause in a transportation network while making optimal repositioning decisions, and propose a robust optimization framework to prevent such inefficiencies. Then, we examine transportation networks at a higher level, where the major focus is on fleet replacement decisions. We present a mathematical model for making these decisions, which is designed to be particularly useful in introducing alternative fuel vehicles into existing diesel fleets. Finally, we develop a scenario sampling method to find dual bounds in multistage stochastic programming problems, which is highly applicable to long-haul fleet management problems with various uncertain parameters such as demand, costs, and capacities.

1.1 Robust Empty Repositioning

The first study of this thesis concentrates on a robust optimization model for empty repositioning in large-scale transportation networks with demand uncertainty. It includes *(i)* a robust rolling horizon framework for empty repositioning, and *(ii)* a method of redesigning the transport moves in the network to support a hub-spoke structure, for computational and operational benefits. In this study, we present a network flow type robust optimization model for empty repositioning, based on the work of Erera et al. [46], which ensures protection against demand uncertainty by

enforcing an extra set of constraints augmenting the flow quantities. We then suggest a rolling horizon framework for using this robust model in a realistic transportation network setting. The rolling horizon framework updates the network parameters and "rolls the horizon" every time truck moves are executed or some demand values previously assumed to be uncertain are observed with certainty. With this computational framework, we perform comprehensive computational experiments using the robust model on test instances with various network sizes, demand scenarios, and levels of conservatism. We examine how the robust approach performs in terms of service level, transportation cost, and computational burden, as opposed to an empty repositioning model with no robustness consideration. Additionally, we observe how level of conservatism affects solution quality, and identify cases in which the robust approach provides the best repositioning plans. Later, we suggest that great savings in transportation cost and computation time can be achieved when the network is redesigned by dividing the set of terminals into smaller sets, called sharing groups, and limiting the repositioning moves to be mostly within sharing groups. To test our claim, we study repositioning plans found by using different sharing group configurations and provide insights as to how various characteristics of these configurations affect the outcome.

The major contributions of this study are threefold. First, a novel method of limiting the level of conservatism for the robust empty repositioning model is introduced. Extensive computational experiments reveal that robustness can provide great improvements in service level when enforced with realistic levels of conservatism, as opposed to trying to guarantee recovery against the worst case. Secondly, a rolling horizon framework for the robust repositioning model is developed to simulate real-life operations. Clearly, under demand uncertainty, a rolling horizon approach represents the operations more accurately compared to solving the repositioning model once in the beginning and operating with that repositioning plan during the entire planning

horizon. Finally, the sharing group design strategies we propose, which can provide great operational savings to trucking companies without any structural changes in the network, grant interesting insights for managing large-scale trucking networks.

1.2 Integrated Fleet Management

The second theme explored in this thesis focuses on a higher-level fleet management problem incurred in long-haul trucking operations, where we generalize a fleet replacement problem by also taking into consideration some infrastructural and operational decisions. We propose an integrated fleet management model designed to be particularly helpful in situations where new truck types (alternative fuel trucks, as the main focus) are being introduced into existing fleet. Though the main motivation of this study is smoothly introducing alternative fuel trucks (AFTs) into existing diesel fleet, the proposed methodology is applicable to managing any fleet mix at any stage of life. The model simultaneously makes *(i)* truck purchase/retirement, *(ii)* maintenance facility opening, and *(iii)* truck deployment decisions. To demonstrate the benefits of utilizing this computationally challenging model instead of simpler methods, we investigate the optimal solutions of this model on realistically designed problem instances, each representing important characteristic of the transportation network, such as constant vs. changing demand, constant vs. increasing targets on proportion of truck-miles traveled with AFTs, constant vs. time-dependent truck purchase and operating costs. On these problem instances, we demonstrate that the optimal solutions to the integrated model are non-trivial and that the consideration of infrastructural and operational decisions in the context of fleet management results in effective fleet replacement strategies by making use of information that a plain fleet replacement model would not have.

Various strategies for finding optimal and good heuristic solutions for the integrated fleet management model are explored. A Benders' decomposition framework

is introduced. The effectiveness of this framework in finding optimal solutions in relatively small problems, and finding solutions with very small optimality gaps in larger problems (when reasonable time limits are enforced) is demonstrated with a comprehensive computational study. Additionally, a variable neighborhood search (VNS) algorithm is designed for finding good solutions to the integrated model. The effectiveness of VNS in finding good solutions quickly is also demonstrated in the computational study.

There are existing works in the fleet replacement literature, which provide methods for making fleet replacement decisions jointly with infrastructural or truck routing/utilization decisions. However, to the best of our knowledge, the integrated fleet management model we present in this thesis is the first to integrate both strategic and operational decisions into the fleet replacement framework. The introduction of this new model is one of the main contributions of this study, along with identification of patterns in which the resulting fleet replacement plans respond to changing cost parameters and insights on how making fleet replacement decisions simultaneously with maintenance facility opening and truck deployment decisions affect fleet replacement strategies.

1.3 Partition Bounds for Multistage Stochastic Programs

In the last study of this thesis, we develop an approach for obtaining dual bounds in multistage stochastic programming problems with a finite number of scenarios. A method for obtaining a hierarchy of dual bounds, called expected group subproblem objective ($EGSO$) bounds, is presented in Sandıkçı et al. [108], where the bounds are calculated by using the solutions to "scenario group subproblems" (problems including variables and constraints for only a subset of scenarios) for every scenario subset of a certain group size. The bounds we propose, called the expected partition (EP) bounds, use partitions of the scenario set comprised of subsets of (nearly) equal

cardinality and obtain the same values as the $EGSO$ bounds. Furthermore, every partition bound, calculated by solving group subproblems of a single partition, yields a dual bound for the original problem. Therefore, even when it is not practical to calculate the EP bound exactly by calculating partition bounds for every possible partition of a certain group size, we can still obtain dual bounds from as few as a single partition of the scenario set. Motivated by the observation that the best of even a very small sample of partition bounds is very likely, in practice, to be better than the corresponding EP bound, we develop a sampling approach where we randomly sample scenario partitions and calculate partition bounds. We then suggest two enhancements to the approach: sampling partitions that align with the multistage scenario tree structure, and use of an auxiliary optimization problem to discover new best bounds based on the values of group subproblems already computed. The practical value of these ideas is illustrated on benchmark problems, and the approach compared to Sample Average Approximation. Finally, we give a heuristic to save computational effort by ceasing computation of a partition part-way through, if it appears unpromising.

Even though the idea of finding bounds for stochastic programming problems using solutions to scenario group subproblems is studied in the literature [86, 108, 109, 134], to the best of our knowledge, our solution approach is the first to suggest random sampling of scenarios to find such bounds. Another contribution of this study is the idea of distinguishing promising scenarios from the unpromising ones in order to decrease computation time wasted in calculating non-improving bounds.

CHAPTER II

A ROBUST ROLLING HORIZON FRAMEWORK FOR EMPTY REPOSITIONING

2.1 Introduction

One of the major challenges faced by a transportation service provider is the largely imbalanced nature of freight flows and the resulting need to reposition resources empty. The imbalanced nature of load requests causes containers to accumulate at certain terminals, while other terminals, which fulfill mostly outbound load requests, are in constant need of resources. When resources are needed at a terminal for outbound moves, it may not be possible to direct enough inbound loaded moves to that terminal on time, which, if no action is taken, results in lost demand or outsourcing the demand to another carrier at a much higher cost. To prevent such situations, as much as possible, carriers have to move resources empty. In empty repositioning decisions, balancing between the transportation cost incurred by moving resources empty and the benefit gained by not having to outsource future load requests is key.

The main challenge when developing effective empty repositioning strategies is future load uncertainty. As a result of the uncertainty in future load requests, a carrier may, at times, be unable to satisfy load requests or may, at times, be moving resources empty unnecessarily. Satisfying customer demand without employing a needlessly large fleet and without excessively repositioning empty resources requires careful planning in environments where demand is uncertain. Incorporating demand uncertainty into the empty repositioning strategies can be done in different ways. In this chapter, we focus on empty repositioning strategies based on robust optimization

methods.

A secondary challenge arises when a transportation service provider serves a large region and operates a large number of terminals, because, ideally, empty repositioning moves should be carried out between terminals that are in close proximity to each other, which not only controls cost (since long empty repositioning moves are obviously more expensive), but also helps control the complexity of the day-to-day operations. A natural strategy that accomplishes this, and is often seen in practice, is to divide the region served into subregions for empty repositioning purposes and to manage each subregion somewhat independently. These so-called *sharing group policies* partition the terminals in the transportation network into sharing groups, each with a single designated empty hub. In a sharing group policy, empty repositioning within a sharing group is between non-hub terminals and the hub, and empty repositioning between sharing groups is between the empty hubs. Such policies naturally introduce pooling benefits.

Early optimization approaches for solving empty repositioning problems are deterministic and model the problem on a time-expanded (or time-space) network with nodes representing terminals at specific time periods and arcs representing loaded and possible empty repositioning moves. Resource requirements at terminals are typically point estimates, because future load requests are uncertain in most settings. An important and recognized shortcoming of these approaches is that they do not capture future demand uncertainty, which may result in suboptimal empty repositioning plans or even infeasibility after the load requests have materialized.

To model the empty repositioning problem with demand uncertainty more accurately, we propose a variant of the robust optimization approach of [46] and demonstrate its effectiveness with a comprehensive computational study. Empty repositioning plans generated by the robust approach guarantee the existence of short empty

repositioning moves to recover feasibility after demand has materialized for every possible demand realization, or for a subset of possible demand realizations, assuming demand realizations come from a known uncertainty set. Since repositioning decisions are made prior to the materialization of demand, the proposed robust optimization model is similar to a two-stage stochastic (integer) programming model, in which first and second stage decisions represent the repositioning plan and the recourse opportunities given the demand realization, respectively.

In our robust approach, instead of explicitly solving a second stage problem, the existence of recovery actions is ensured by an additional set of constraints in the first stage problem. Furthermore, the level of uncertainty the recovery actions are to hedge against is controlled by limiting deviations from nominal values in a way that is similar to uncertainty budgets, an idea introduced in [17]. The core of the optimization problem that has to be solved is a minimum cost flow problem with flow balance and flow bound constraints to ensure that nominal demand is met and empty resources are repositioned at minimum cost. The problem is augmented with constraints that ensure recoverability, *i.e.*, that additional empty resources can be repositioned, if necessary, to accommodate deviations from nominal demand. Since only low-cost repositioning moves are considered for recovery actions, the costs associated with potential recovery actions are ignored.

In reality, the empty repositioning problem, of course, has a multi-stage nature, because demand information becomes available over time. To accurately represent this multi-stage nature, the proposed robust empty repositioning model is embedded in a rolling horizon framework. Consequently, each time the model is solved, it is assumed that only the current day's load requests are known with certainty and that the only information available about future load requests are nominal demand quantities and uncertainty intervals. Furthermore, only the current day's decisions are implemented before the horizon is rolled forward.

By taking advantage of relatively minor data reliance of robust optimization models compared to other optimization methods that address uncertainty, the proposed robust approach provides high-quality solutions while being simple in modeling and implementation, and computationally tractable even for large-scale systems. Computational experiments demonstrate that this approach generates empty repositioning plans that are significantly more immune to demand uncertainty (with only a slight increase in transportation cost) compared to plans generated with a deterministic approach using only nominal demand quantities. For example, in a 100-terminal network, averaged over 30 instances, the robust approach increases the service level (*i.e.*, the fraction of load requests that can be satisfied with the carrier's own fleet) by 2.6% when compared to a non-robust approach, while incurring only a 5.4% increase in transportation cost (the cost of outsourcing for the unsatisfied load requests is much more than 5.4%). By adjusting the conservatism of the robust approach, for the same set of instances, it is possible to achieve a 0.7% increase in service rate, while only incurring a 0.6% increase in transportation cost.

As briefly mentioned before, sharing group policies in empty repositioning are commonly used in practice when operating large-scale transportation service networks. In addition to being sensible from an operational perspective, such policies are also useful from a computational viewpoint. Clearly, a policy allowing empty repositioning between too many pairs of terminals will generate plans that are sensitive to demand uncertainty (due to a lack of risk pooling), as well as greatly increasing the computational burden (due to an excessive number of empty repositioning options). On the other hand, a sharing group policy with too few sharing groups or sharing groups with high flow imbalance can result in excessive empty repositioning costs (either because of long distances between empty hubs or frequent repositioning between empty hubs). Therefore, it is important for freight transportation service providers to carefully design the sharing group configuration if they plan to employ a sharing

group based empty repositioning strategy. As part of our computational study, we conduct experiments to assess the effects of the sharing group configuration on empty repositioning costs (using both the nominal model and the robust model). We find that it is beneficial to define the sharing groups in such a way that they have roughly equal inbound and outbound loaded flows (if possible) and to carefully trade-off the distance between non-hub terminals and the hub in a sharing group and the distance between hubs of different sharing groups.

The remainder of the chapter is organized as follows. In Section 2.2, we give a brief overview of the relevant literature. In Section 2.3, we introduce the proposed robust optimization approach. Finally, we present and discuss the results of a comprehensive computational study in Section 2.4, and summarize our findings in Section 2.5.

2.2 Literature Review

Empty repositioning problems have been a research focus for a long time. Earliest dynamic models in the area utilize deterministic optimization, where uncertain demand is modeled via point estimates. Some examples of such studies are [81] and [89], which study fleet management in rail operations, and [132], which focuses on container management. Deterministic models are still being used by freight transportation companies to manage their operations, since these models are easy to implement and solve.

Even though deterministic models that utilize point estimates for demand have been proven useful in real-life applications, their feasibility is susceptible to deviations of demand realizations from point estimates. To address this issue, expected cost minimization approaches, which require solving dynamic or stochastic programming models, are widely used. Early examples of this line of research, focusing on truckload trucking operations are [100], [101], [56], and [30]. Later, adaptive estimation techniques are introduced for tractability of the dynamic programming approaches. [102]

introduces linear functional approximations to capture the future effects of decisions made today; and nonlinear functional approximations are introduced and effectively used in [58] and [59]. From a similar perspective, [43] and [62] suggest using principles from inventory theory to obtain repositioning policies based on reorder points. They both propose these policies as a simple alternative to dynamic programming approaches, which are computationally burdensome.

Another way of modeling uncertainty in dynamic resource management problems is stochastic programming. [34] proposes a two-stage restricted recourse model, where all the empty container allocation decisions are assumed to be made in the first stage, never to be re-optimized in the future, and second-stage decisions only act as corrective actions to the pre-specified repositioning plan for the entire horizon. This type of approaches are tractable, while restrictive in its assumptions. Other examples of two-stage stochastic models with various uncertain parameters are studied in [29], and [110], where a Sample Average Approximation (SAA) scheme with a Benders' decomposition variant is shown to compute high-quality solutions in large-scale supply chain design problems. [36] considers an empty container repositioning problem with multiple uncertain parameters in maritime operations that has very low-quality data. It proposes a multi-scenario optimization model and stresses that the formulation exhibits a certain algebraic structure and therefore can be solved via decomposition and parallel computing. More recently, [136] proposes using a two-stage stochastic programming model iteratively to approximately solve a complex multi-stage stochastic optimization problem arising in drayage operations.

Robust optimization is another approach for modeling uncertainty, and is often preferred because of its simplicity, computational tractability, and ability to generate good quality solutions (in terms of recoverability) with insufficient or unreliable data. The first work in robust optimization with coefficient uncertainty is [124]. It shows that such uncertainties can be incorporated into a linear programming form and

solved so that feasibility is guaranteed for all possible realizations of the uncertain coefficients. The foundations of robust convex optimization are established in [13]. Later, detailed analyses of robust linear programs are presented in [15] and [16]. The latter introduces the idea of uncertainty sets with budgets, where the budget limits the number of uncertain coefficients that can simultaneously assume their worst-case value.

Robust discrete optimization is studied in [79] and [16]. The former develops robust versions of many traditional discrete optimization problems, and the latter concentrates on network flow problems - particularly the theoretical and practical tractability of robust counterparts of polynomially solvable discrete optimization problems.

[12] extends the robust linear optimization approach to cases where a subset of decisions are made before the realization of uncertain parameters and the remaining decisions can be made after the realization; calling the model Adjustable Robust Counterpart (ARC). Similar to the ARC approach, [7] develops a two-stage robust optimization approach for network flow and design problems with uncertain right-hand side, and underlines the importance of the ability of two-stage robust models to control the level of conservatism, unlike single-stage robust models where feasibility is guaranteed for all possible uncertainty realizations.

There is a wide range of recent studies that effectively implement robust optimization methods in dynamic resource management problems. [80] and [32] concentrate on network design, where the latter proposes a tractable linear programming formulation for the robust optimization model. [14] proposes adjustable robust formulations for the retailer-supplier flexible commitment problem, and [18] develops a robust formulation for inventory problems where demand uncertainty is modeled deterministically when certain network characteristics are present. [112] suggests robust formulations to the multi-period inventory problem, [27] proposes a method for the

lot sizing model with uncertain and non-stationary demand structure, and [121] and [122] develop robust mixed-integer programs for the inventory routing problem with polyhedral demand uncertainty. [46], which constitutes a basis for this study, presents a robust optimization framework for dynamic empty repositioning and develops feasibility conditions for various sets of allowed recovery actions; and [57] provides a convex robust formulation for the location transportation problem.

2.3 Methodology

Consider a transportation service provider centrally managing a homogeneous fleet of reusable resources such as containers, railroad cars or trucks to serve a vast geographic area with a large number of terminals over a planning horizon of multiple time periods. Note that a time period can correspond to any measure of time, but it typically corresponds to a fraction of a day in this context. Load requests materialize over time and specify the number of resources to be transported from one terminal to another in a specific time period. Based on the known load requests as well as anticipated future load requests, central management determines a cost-effective empty repositioning plan to ensure resource availability to satisfy demand.

Before introducing the robust optimization model that addresses demand uncertainty, we first introduce the empty repositioning problem with deterministic demand. Assuming demand quantities are known with certainty in the beginning of the planning horizon, the traditional approach for finding a minimum-cost repositioning plan is to solve a deterministic minimum-cost flow problem with flow balance and flow bound constraints over the entire planning horizon, where flow lower bounds guarantee demand satisfaction. This deterministic problem will be referred to as the nominal repositioning problem (**NP**) from this point on.

To formally define the nominal problem, let D denote the set of terminals, and the planning horizon consist of ρ time periods $\{0, 1, 2, \ldots, \rho\}$. Let the set of nodes for each

terminal $d \in D$ be $V^d = \{v_0^d, v_1^d, \ldots, v_\rho^d\}$, and $\mathcal{V} = \cup_{d \in D} V^d$. There are *inventory arcs* (v_t^d, v_{t+1}^d) for each $d \in D$ and $t = 0, 1, \ldots, \rho - 1$, and *repositioning arcs* $(v_t^i, v_{t+h_{ij}}^j)$ from terminal i to terminal j whenever repositioning is allowed between those terminals and the travel time is h_{ij} periods. Let $\mathcal{A}_I = \{(v_t^d, v_{t+1}^d) : d \in D, 0 \leq t < \rho\}$ denote the complete set of inventory arcs, and $\mathcal{A}_R$ denote the set of repositioning arcs, which represent the allowed repositioning moves in the network.

We distinguish between two types of repositioning arcs: *loaded arcs*, which represent load requests from specific origin terminals to specific destination terminals in specific time periods and *empty arcs*, which represent the option to move resources empty between two terminals in specified time periods and are determined based on the carrier's empty repositioning strategy. Let $\mathcal{A}_R^L$ denote the set of loaded arcs, $\mathcal{A}_R^E$ denote the set of empty arcs. Then, $\mathcal{A}_R = \mathcal{A}_R^L \cup \mathcal{A}_R^E$, where

$\mathcal{A}_R^L = \{(v_t^i, v_{t+h_{ij}}^j) :$ a load request exists from i to j in period $t\}$

$\mathcal{A}_R^E = \{(v_t^i, v_{t+h_{ij}}^j) :$ there is the option of moving resources empty from i to j in period t, $0 \leq t \leq \rho - h_{ij}\}$

Finally, let each node $v \in \mathcal{V}$ have a net supply $b(v)$, which represents the number of empty resources available at the corresponding terminal in the corresponding time period. Let s be an auxiliary sink node with net supply $b(s) = -\sum_{v \in \mathcal{V}} b(v)$ and $\mathcal{A}_s$ denote the auxiliary arcs entering sink node s. Then, the time-expanded network representing this transportation system is $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N} = \mathcal{V} \cup \{s\}$ and $\mathcal{A} = \mathcal{A}_I \cup \mathcal{A}_R \cup \mathcal{A}_s$.

Using the above notation and terminology, the nominal problem can be formulated as

$$\mathbf{NP} \qquad \min_{\mathbf{x} \in \mathbb{Z}_+^{|\mathcal{A}|}} \{\mathbf{c}^T \mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{b},\ x(a) \geq l(a)\ \forall a \in \mathcal{A}_R^L\} \tag{2.1}$$

where the decision vector $\mathbf{x}$ quantifies the resource flow on each arc, $\mathbf{c} \in \mathbb{R}_+^{|\mathcal{A}|}$ denotes the transportation cost vector, $\mathbf{A} \in \{0, 1, -1\}^{|\mathcal{V}| \times |\mathcal{A}|}$ denotes the node-arc incidence

matrix associated with $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, and $\mathbf{b} \in \mathbb{Z}^{|\mathcal{N}|}$ and $\mathbf{l} \in \mathbb{Z}_+^{|\mathcal{A}_R^L|}$ denote node net supplies and flow lower bounds (dictated by load requests from customers), respectively. Note that **NP** is polynomially solvable with standard minimum-cost flow algorithms or linear programming.

Now suppose that the number of customer load requests on loaded arcs are uncertain. To address this uncertainty, we formulate a mathematical model based on the Adjustable Robust Counterpart (ARC) approach of [12], to be referenced as the robust repositioning problem (**RP**), which supports an uncertain demand structure where only nominal quantities (historical averages, forecasts, etc.) are known for future load requests.

Let the nominal demand on loaded arc $a \in \mathcal{A}_R^L$ be $l(a)$. We assume that every potential demand realization $\tilde{l}(a) \in \mathbb{Z}_+$ falls in the interval $[l(a) - \hat{l}^-(a), l(a) + \hat{l}^+(a)]$, where $\hat{l}^-(a) \in [0, l(a)]$ and $\hat{l}^+(a) \geq 0$ denote the maximum negative and positive deviations from the nominal demand quantity, respectively.

The purpose of the robust model is to ensure that the repositioning plan is *recoverable*, *i.e.*, there exists a set of empty moves (referred to as *recovery actions*) that can alter the repositioning plan obtained prior to the materialization of uncertain demand, so that the altered plan will be feasible with respect to realized loaded demand quantities. It is possible to ensure recoverability against all potential demand realizations, including a scenario where all loaded demand quantities simultaneously assume their worst-case values. It is also possible, and more realistic for practical purposes, to pursue a less conservative approach and ensure recoverability against only a subset of potential demand realizations. To control the level of conservatism, we follow a method similar to the *budget of uncertainty* idea introduced in [17].

To control how much of the demand uncertainty **RP** will protect against, we define the following limited perturbation set φ_k, where we guarantee recoverability

only in cases where a certain proportion of the worst-case deviations are realized:

$$\varphi_k = \{\boldsymbol{\delta} \in \mathbb{Z}^{|\mathcal{A}_R^L|} : \delta(a) \in [-k\hat{l}^-(a), k\hat{l}^+(a)] \quad \forall a \in \mathcal{A}_R^L\}.$$

Here, $k \in [0,1]$ denotes the proportion of worst-case deviation from nominal demand for which **RP** will guarantee recoverability. Note that when $k = 0$, every realization is assumed to conform to nominal, and when $k = 1$, the robust model protects against all potential realizations $\tilde{\mathbf{l}} \in [\mathbf{l} - \hat{\mathbf{l}}^-, \mathbf{l} + \hat{\mathbf{l}}^+]$.

To assess the recoverability of repositioning plans found by **RP**, we first need to define the set of allowed recovery actions. As mentioned in Section 2.1, the cost of recovery actions will be ignored so that the originally two-stage problem can be reduced to a single integer program. Because recovery will not be modeled explicitly in the robust problem, it is crucial that the set of recovery actions consists only of low-cost moves. Given the set of recovery arcs $\mathcal{R} \subseteq \mathcal{A}_R$, the set of recovery actions is defined as:

$$W = \{\mathbf{w} \in \mathbb{Z}^{|\mathcal{A}|} : w(a) \geq 0 \quad \forall a \in \mathcal{R}, \quad w(a) = 0 \quad \forall a \in \mathcal{R}_0\}$$

where $\mathcal{R}_0 = \{(v_0^d, j) \in \mathcal{R} : d \in D, j \in \mathcal{V}\}$ denotes the set of recovery arcs associated with the initial time period. We assume that the decisions associated with the initial period are fixed and therefore cannot be altered with recovery actions, *i.e.*, we assume the resources will have already left their original locations en-route to their intended destinations by the time recovery actions are to be taken, and hence cannot be rerouted. Note that similar recovery sets can be defined if the decisions associated with multiple periods are assumed to be fixed.

Using the definitions of limited perturbation set φ_k and set of recovery actions W, a formulation for **RP** is

$$\mathbf{RP}(W,k) \qquad \min_{\mathbf{x} \in \mathbb{Z}_+^{|\mathcal{A}|}} \{\mathbf{c}^T\mathbf{x} : \quad \mathbf{x} \in X(\mathbf{l}),$$

$$\forall\, \mathbf{l} + \boldsymbol{\delta} \in \mathcal{Z}_k \quad \exists \mathbf{w} \in W : \mathbf{x} + \mathbf{w} \in X(\mathbf{l} + \boldsymbol{\delta})\} \qquad (2.2)$$

where $\mathcal{Z}_k = \{\tilde{\mathbf{l}} \in \mathbb{Z}_+^{|\mathcal{A}_R^L|} : l(a) - k\hat{l}^-(a) \leq \tilde{l}(a) \leq l(a) + k\hat{l}^+(a) \quad \forall a \in \mathcal{A}_R^L\}$ denotes the set of possible joint demand realizations corresponding to the limited perturbation set φ_k, $X(\mathbf{l}) = \{\mathbf{x} \in \mathbb{Z}^{|\mathcal{A}|} : \mathbf{Ax} = \mathbf{b}, x(a) \geq l(a) \quad \forall a \in \mathcal{A}_R^L\}$ denotes the set of feasible flows with respect to flow balance and nominal demand satisfaction constraints, and $X(\mathbf{l}+\boldsymbol{\delta})$ is the set of feasible flows with respect to flow balance and perturbed demand satisfaction constraints.

We can write **RP** in extended form as:

$$\begin{aligned} \mathbf{RP}(W,k) \qquad \min_{\mathbf{x}\in\mathbb{Z}_+^{|\mathcal{A}|},\mathbf{w}\in\mathbb{Z}_+^{|\varphi_k|\times|\mathcal{A}|}} \{\mathbf{c}^T\mathbf{x} : \quad & \mathbf{x} \in X(\mathbf{l}), \\ & \mathbf{x} + \mathbf{w}_{\boldsymbol{\delta}} \in X(\mathbf{l}+\boldsymbol{\delta}) \quad \forall\, \boldsymbol{\delta} \in \varphi_k, \\ & \mathbf{w}_{\boldsymbol{\delta}} \in W \quad \forall\, \boldsymbol{\delta} \in \varphi_k\} \end{aligned} \tag{2.3}$$

Since **RP** is primarily a problem of feasibility, we choose not to model costs associated with the set of recovery decisions, $\mathbf{w}$. This feature constitutes a basis for the more tractable alternative robust formulation, which guarantees recoverability not by explicitly deciding on the recovery actions but by enforcing a set of *recoverability constraints* in addition to the constraints of the nominal problem (**NP**). To further describe the recoverability constraints, it is necessary to define *vulnerability of node sets* and *inbound-closed node sets*.

Definition 2.1 (Node Set Vulnerability)**.** *For a set of nodes $U \subseteq \mathcal{N}$, the vulnerability $\vartheta(U,k)$ is defined as*

$$\vartheta(U,k) = \sum_{\substack{a=(i,j)\in\mathcal{A}_R^L \\ i\in U, j\in\Delta^{out}(i)\setminus U}} k\hat{l}^+(a) + \sum_{\substack{a=(i,j)\in\mathcal{A}_R^L \\ i\in\Delta^{in}(i)\setminus U, j\in U}} k\hat{l}^-(a)$$

Note that when the uncertainty set is φ_k, the vulnerability of node set U quantifies the sum of maximum excess of loaded outflow from node set U and maximum shortage of loaded inflow into node set U, multiplied by k.

Definition 2.2 (Inbound-closed Node Set)**.** *Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ denote a network. A set of nodes $C \subseteq \mathcal{N}$ is inbound-closed if there exists no directed path in $\mathcal{G}$ from any node $i \in \mathcal{N} \setminus C$ to any node $j \in C$.*

Let $\mathcal{G}_W = (\mathcal{N}, \mathcal{A}_W)$ represent the recovery network induced by the set of recovery actions W, where $\mathcal{A}_W$ denotes the set of recovery arcs. It is important to observe that in $\mathcal{G}_W$, an inbound-closed node set has no possible incoming recovery flow. Therefore, such sets need to contain enough pooled inventory so that recovery is guaranteed in worst-case demand realizations. In what follows, we provide the necessary and sufficient conditions for guaranteed recovery flow. The proofs are relatively straightforward adaptations of the original proofs in [46], but they are provided in Appendix A.1 for the sake of completeness.

Let $\mathcal{G}_W(\mathbf{x}, \boldsymbol{\delta})$ represent the recovery network given a repositioning plan $\mathbf{x}$ and a specific perturbation realization $\boldsymbol{\delta} \in \varphi_k$, where the net supply vector $\mathbf{I} \in \mathbb{Z}^{|\mathcal{N}|}$ is adjusted based on $\mathbf{x}$ and $\boldsymbol{\delta}$ to reflect the marginal net inventory of resources available (or needed) in the recovery problem.

$$I(v_0^d) = 0, \quad d \in D$$

$$I(v_1^d) = x(v_1^d, v_2^d) + \sum_{i \in \Delta^{in}(v_1^d)} \delta(i, v_1^d) - \sum_{j \in \Delta^{out}(v_1^d)} \delta(v_1^d, j), \quad d \in D$$

$$I(v_t^d) = x(v_t^d, v_{t+1}^d) - x(v_{t-1}^d, v_t^d) + \sum_{i \in \Delta^{in}(v_t^d)} \delta(i, v_t^d) - \sum_{j \in \Delta^{out}(v_t^d)} \delta(v_t^d, j), \quad d \in D, 1 < t \leq \rho$$

$$I(s) = -\sum_{v \in \mathcal{V}} I(v)$$

Proposition 2.1. *A feasible solution $\mathbf{x}$ of* **NP** *is recoverable with recovery action set W after a perturbation realization $\boldsymbol{\delta}$ if and only if there exists a feasible flow in $\mathcal{G}_W(\mathbf{x}, \boldsymbol{\delta})$.*

Proposition 2.2. *Let $\mathcal{U}_W$ denote the collection of inbound-closed node sets in $\mathcal{G}_W$.*

There exists a feasible flow in $\mathcal{G}_W(\mathbf{x}, \boldsymbol{\delta})$ *if and only if for every set of nodes* $U \in \mathcal{U}_W$

$$\sum_{v \in U} I(v) \geq 0$$

Theorem 2.1. *A feasible solution* $\mathbf{x}$ *of* ***NP*** *is also feasible for the robust problem* ***RP*** *if and only if for every node set* $U \in \mathcal{U}_{\mathcal{W}}$

$$\sum_{a \in \Delta^{out}(U) \cap \mathcal{A}_I} x(a) \geq \vartheta(U, k). \tag{2.4}$$

The theorem provides the following alternative formulation for the robust model **RP**:

$$\mathbf{ARP}(W, k) \qquad \min\{\mathbf{c}^T\mathbf{x} : \quad \mathbf{x} \in X(\mathbf{l}), \sum_{a \in \Delta^{out}(U) \cap \mathcal{A}_I} x(a) \geq \vartheta(U, k) \quad \forall\, U \in \mathcal{U}_W\} \tag{2.5}$$

Observe that the number of variables in **ARP** is equal to the number of variables in **NP**, whereas **RP** models all recovery actions explicitly and therefore introduces a very large number of variables. Additionally, **ARP** has a significantly smaller number of constraints compared to **RP** when the recovery network is designed pragmatically, *i.e.*, when only a small number of low-cost moves are allowed for recovery flow. It is particularly important to note that the number of recoverability constraints in **ARP** is only dependent on the structure of the recovery network and not on the possible realizations in uncertainty set φ_k. Even though the number of such constraints is directly related to the number of inbound-closed node sets in the recovery network, which may be exponential, it is manageable when the recovery network is relatively small.

In transportation applications, empty repositioning decisions are typically made and implemented in a rolling horizon framework, where the optimization model is solved for a given planning horizon but only today's decisions are implemented. Subsequently, model parameters are updated with the implemented decisions and tomorrow's realized demand quantities, and the optimization model can be solved once more to obtain decisions for tomorrow; and so on.

We propose to use our two-stage robust repositioning model in the rolling horizon framework. In the two-stage robust model, the first stage represents the repositioning decisions made prior to the materialization of demand, and the second stage represents the recovery decisions made after the realized demand quantities are observed. Using this model in a rolling horizon framework provides opportunities to guarantee recoverability every time the horizon is rolled forward and the next period's demand is materialized, much like seeking recourse options in a multi-stage stochastic programming model.

When implementing **ARP** with rolling horizon, it may suffice for practical purposes to guarantee recoverability only for the near future rather than the entire planning horizon, *(i)* as a measure to prevent overconservatism, and *(ii)* since we will have other opportunities to ensure future recoverability after recent decisions are implemented and the horizon is rolled. To demonstrate this idea, consider a time-space network where the travel time between terminals is no more than one period and repositioning is allowed between all pairs of terminals. Then, ensuring recoverability of only the immediate next period's repositioning actions would suffice, since recovery actions for the periods that are further into the future will still be available after the horizon is rolled. It is also important to note that in this case, even though feasibility is not a concern, looking further into the future for recoverability can provide more cost-effective repositioning plans by taking advantage of cheaper repositioning moves that are available in the future and therefore would be missed by the myopic approach of ensuring recoverability only for the immediate next period. Therefore, it would be the most practical to consider recoverability for a reasonable number of periods into the near future, without being overly myopic or conservative. With this idea, instead of enforcing the recoverability constraints (2.4) for all node sets that are inbound-closed in $\mathcal{G}_W$, we enforce such constraints for a limited sub-collection of inbound-closed node sets. For this, we define a *robustness horizon*, R, and limit

the collection of inbound-closed sets $\mathcal{U}_W$ to include only the sets that contain nodes associated with time periods that are no more than R periods into the future. This approach helps in controlling the level of conservatism, as well as limiting the number of recoverability constraints (2.4) in **ARP** for computational tractability.

To further control the level of conservatism in **ARP**, we define another robustness parameter, N, which represents the maximum number of terminals contributing to an inbound-closed node set. Introducing such a parameter to control conservatism is motivated by the fact that it is not likely in real-life instances for worst-case empty resource deficits to occur in a large number of terminals simultaneously. Therefore, imposing recoverability constraints for the inbound-closed node sets with a large number of terminals is likely to be too conservative for practical purposes, as well as being computationally burdensome. With this idea, we limit $\mathcal{U}_W$ to include only the inbound-closed sets that contain nodes associated with a maximum of N terminals.

To summarize, the level of conservatism associated with **ARP** can be controlled by controlling the following:

(i) $\vartheta(U, k)$, right-hand side of the recoverability constraints (2.4)

(ii) $\mathcal{U}_W$, collection of inbound-closed node sets in $\mathcal{G}_W$ (the number and structure of the node sets)

Controlling the right-hand side is possible by adjusting the value of the *vulnerability multiplier*, k, which has no effect on the computational burden of building and solving the model. On the contrary, controlling the number and structure of inbound-closed node sets, which is possible by changing the values of robustness parameters R and N, directly affects the computational burden of **ARP**. When not controlled via the robustness parameters, **ARP** can become intractable due to potentially exponential number of recoverability constraints. However, when the recovery network $\mathcal{G}_W = (\mathcal{N}, \mathcal{A}_W)$ has certain characteristics, limiting $\mathcal{U}_W$ by robustness parameters R and N

can provide tractable **ARP** models that exhibit a linear increase in the number of recoverability constraints with increasing number of terminals.

2.4 Computational Experiments

In this section, we first describe the testing framework that is devised to analyze the proposed robust optimization methodology, and then present and discuss results of our computational study. To assess the performance and efficiency of the proposed methodology, we use a simulation framework. Furthermore, to evaluate the value of robustness, we conduct computational experiments using the robust model **ARP** in comparison to the nominal model **NP**. In the simulation framework; each day, a set of load requests materializes and an optimization model is solved to determine an empty repositioning plan. Based on this plan, the current day's repositioning moves are executed, the net supply quantities are updated based on the executed moves, and the horizon is rolled forward to start from the next day.

2.4.1 Problem Instances

The basis for our computational study is a set of instances that are representative of real-life empty repositioning settings. Multiple instances (representing multiple demand realizations) are randomly generated for the same underlying network geography and demand distribution. The inputs to the instance generation process are the geographical locations of a set of terminals and the probabilities of each terminal being the origin or destination of a load request. Using that information, nominal demand quantities and deviations from the nominal values are obtained via Monte Carlo simulation, and then the fleet size and initial resource locations are determined.

In generating the problem instances, we assume that a constant number of load requests materialize each day of the planning horizon. Each load request is the result of a multinomial trial with the origin and destination terminals determined using the

origin and destination probabilities associated with each terminal. To generate realistic instances, it is assumed that the transportation carrier operates a hub-and-spoke network, where regional hubs act as consolidation centers for their spokes (non-hub terminals). Load requests are routed so that loads first go from the origin terminal to the origin regional hub, then, if necessary, from the origin regional hub to the destination regional hub, and, finally, from the destination regional hub to the destination terminal. Thus, even though a load request can originate at any terminal and be destined for any terminal, it is routed along the appropriate path in the hub-and-spoke network.

Once load requests are generated and routed in the described manner for a large number of days, average loaded demand is calculated for every arc $a \in \mathcal{A}_R^L$ (in the hub-and-spoke network) and that quantity is recorded as the nominal demand quantity, denoted as $l(a)$. The maximum positive and negative deviations from nominal values, denoted as $\hat{l}^+(a)$ and $\hat{l}^-(a)$, are also recorded.

Using the nominal demand quantities, we determine a fleet size that can feasibly satisfy the nominal demand and realistically accommodate some demand uncertainty. To achieve this, we solve a minimum cost network flow problem on a wrapped time-expanded network, which represents the transportation system, for a planning horizon of one week. The minimum fleet size required to satisfy the nominal loaded demand is then increased by a factor to account for demand uncertainty. In order to obtain meaningful results, 30 instances are generated for any given network geography and demand structure, and computational results are presented as averages of these 30 instances.

2.4.2 Testing Framework

In order to test the performance of the described robust optimization approach for empty repositioning problems, we develop a simulation framework that mimics the

daily generation and execution of repositioning plans. For each day of the planning horizon, a time-expanded network is generated, a variant of **ARP** is solved, the repositioning decisions of the current day are executed, and the horizon is rolled to start from the next day. The time-expanded network uses homogeneous time periods of six hours and covers a workweek, *i.e.*, five working days. Consequently, the time-expanded network has $5 \times 4 \times |D|$ nodes (plus one auxiliary sink node).

The variant of **ARP** that we use in our experiments, **ARP'**, differs from the original **ARP** in allowing outsourcing in case the nominal demand satisfaction and recoverability cannot be feasibly attained with the current fleet size. **ARP'** explicitly models the option of outsourcing load requests to ensure feasibility. For this, we introduce outsourcing variables to the original **ARP** formulation and assign a sufficiently large penalty cost to outsourcing so that it occurs only when it is absolutely necessary for feasibility. Note that positive values of outsourcing variables can also be interpreted as unsatisfied demand, depending on whether the carrier operates on an outsourcing or a lost demand policy.

ARP' is formulated as follows:

$$\textbf{ARP'}(W,k) \quad \min_{\substack{\mathbf{x}\in\mathbb{R}_+^{|\mathcal{A}|} \\ \mathbf{y}\in\mathbb{R}_+^{|\mathcal{A}_R^L|}}} \{\mathbf{c}^T\mathbf{x} + M\mathbf{1}^T\mathbf{y} : \sum_{a\in\Delta^{out}(i)} x(a) - \sum_{a\in\Delta^{in}(i)} x(a) = b_i \quad \forall i \in \mathcal{N} \tag{2.6}$$

$$x(a) + y(a) \geq l(a) \quad \forall a \in \mathcal{A}_R^L \tag{2.7}$$

$$\sum_{a\in\Delta^{out}(U)\cap\mathcal{A}_I} x(a) \geq \vartheta(U,k) \quad \forall U \in \mathcal{U}_W\}, \tag{2.8}$$

where $M > 0$ is a sufficiently large positive number that represents the outsourcing (or lost demand) cost, and $\mathbf{y}$ represents the outsourcing (or lost demand) vector.

When **ARP'** is solved for a week of five workdays starting from today, the outcome is an empty repositioning plan and a set of load requests to be outsourced. In the simulation framework, the next step is implementing the moves (*i.e.*, served load

requests, outsourced load requests, and empty repositioning) of the first day, which translates into adjusting the net supply values at the nodes impacted by these moves.

After today's moves are executed, the horizon is rolled forward by one day, and tomorrow's demand is observed. In order to roll the horizon, the nodes associated with today's periods are deleted from the time-expanded network, along with the relevant inventory and repositioning arcs. Furthermore, nodes and arcs for one more day are appended to the end of the time-expanded network. Once the network is updated with the rolling horizon logic, we assume that one day has passed and that load requests for the (new) first day are observed. To reflect the load requests that have materialized, the flow lower bounds, $l(a)$, of the relevant arcs are changed from nominal to realized demand quantities. An algorithmic summary of the testing procedure is presented in Algorihtm 2.1.

Algorithm 2.1 Testing Framework

for $t \in \{1, 2, 3, \ldots, 20\}$ **do**
  Observe real demand of day t
  Generate $\mathcal{G}_t$, the time-expanded network starting with day t
  Solve **ARP'** on $\mathcal{G}_t$
  Implement decisions associated with day t
end for

To assess the value of robustness, we compare the performance of the empty repositioning plans generated by the robust approach to those generated without robustness consideration. The approach without robustness consideration simply replaces **ARP'** with **NP'** in Algorithm 2.1, where **NP'** denotes a variant of **NP** that allows outsourcing for feasibility.

2.4.3 Computational Results

We test the performance of various robust and non-robust approaches on geographies with 30, 80, and 100 terminals. Section 2.4.3.1 focuses on analyzing the impact of the choice of robustness parameters, *i.e.*, the robustness horizon (R), the number of

terminals in an inbound closed set (N), and the vulnerability multiplier (k). The 30, 80, and 100-terminal geographies are divided into 5, 7, and 10 sharing groups, respectively, and the sharing groups are designed so that the terminals in sharing groups are within close proximity to each other and the number of terminals in each sharing group is the same (or as close to each other as possible). Section 2.4.3.2 focuses on the impact of the sharing group configuration, *i.e.*, the number and make up of the sharing groups, and therefore various sharing group configurations are explored in detail for a 30-terminal geography.

The performance measures that we focus on are the outsourcing ratio (the number of outsourced load requests divided by the total number of load requests), the transportation cost incurred by the carrier, and the solution time.

In the body of the text, we mainly rely on graphs and figures to present the results of computational experiments, but, for completeness, tables with detailed results are provided in Appendix A.2.

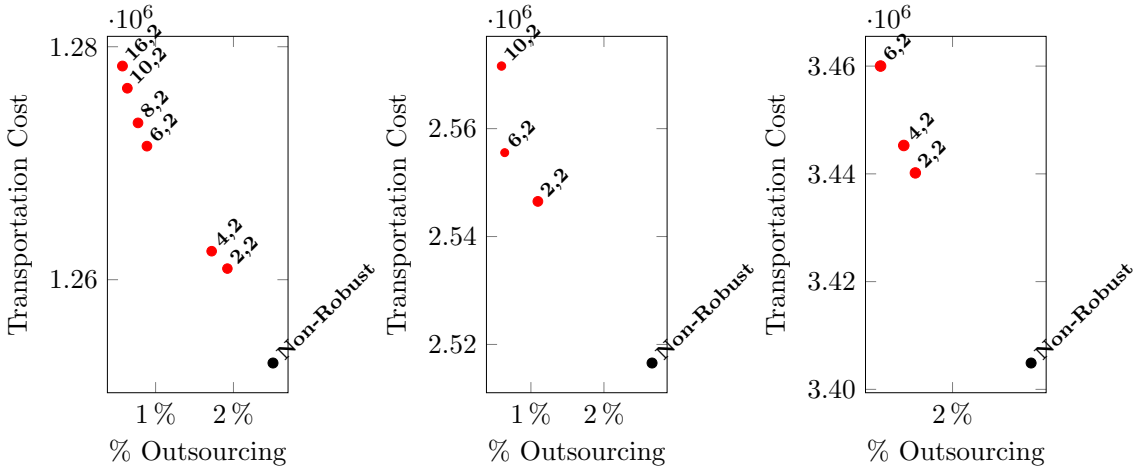
2.4.3.1 Value of Robustness

In this section, we concentrate on the benefits of robust, as opposed to nominal, empty repositioning; as well as the impact of level of conservatism on solution quality. All computational experiments reported in this section are conducted with uncertainty sets $[\max\{0, l - k\hat{l}^-\}, l + k\hat{l}^+]$, where the demand realizations come from a uniform distribution with support $[\max\{0, l - \hat{l}^-\}, l + \hat{l}^+]$. Different values for the parameter k, which specifies the fraction of the worst-case deviation from nominal demand that the recoverability constraints are to protect against, are considered in the test setting. These values are 1, 1/2, 1/3, 1/4. Note that higher values of k correspond to higher levels of protection (*i.e.*, are more conservative).

The robusntness horizon (R) and the maximum number of terminals contributing to an inbound-closed node set (N) affect solution performance both in terms of

solution quality and computational tractability. Therefore, different values for those parameters are considered in our computational experiments, namely 2,4,6,8,10,16, and 20 for R and 2,3,4 and 5 for N. In the visual representation of results, a marker label (R, N) is representative of a test case where at most R time periods are accounted for in the recoverability constraints, and there are at most N terminals in each inbound-closed node set.

The robust approach, as opposed to a non-robust (*i.e.*, nominal) approach, moves the empty resources to "the right places at the right times" so that the terminals that are more vulnerable to demand uncertainty can be feasibly served with recovery moves. Naturally, while ensuring that the resources are at the right places at the right times, the robust approach moves the resources around more than a nominal approach. It is important to note that in our experimental setting, we place emphasis on improving the service level (and not so much reducing the total transportation cost), so we set the unit outsourcing cost to be very high. However, it is also possible to set the unit outsourcing cost closer to the unit transportation cost, in order to represent environments where incurring a low total transportation cost is a comparably important goal to improving the service level.



(a) 30 Terminals (b) 80 Terminals (c) 100 Terminals

Figure 2.1: Effect of R on outsourcing and transportation cost ($k = 1/3$, averaged over 30 instances)

Figure 2.1 presents total transportation cost and outsourcing ratio for different levels of conservatism on 30, 80, and 100-terminal networks. It illustrates the effect of robustness in improving service level while causing a slight increase in total transportation cost. Tables A.1, A.2, and A.3 of Appendix A.2 provide further details and demonstrate that a considerable increase in service level can be achieved with an acceptable increase in computation time when robustness parameters are selected pragmatically. In the discussion to follow, we study the effects of each of the three robustness parameters (R, N, and k) independently.

First, we study the effects of limiting the robustness horizon, R. As seen in Figure 2.1a, when the values of k and N are kept constant, increasing R results in substantial increase in service level up to a certain point, around $R = 10$. However, as R increases from 10 to 16, the increase in service level is less significant and it comes at (relatively speaking) greater increase in transportation cost. Furthermore, as we increase R from 16 to 20 (not included in the figure because of difficulty in viewing), outsourcing actually increases, from 0.57% to 0.60%, contrary to the decreasing trend observed for lower R values. This change in the trend can intuitively be explained with overconservatism, since a robust approach looking too far into the future can accumulate resources at terminals that have high demand in the distant future at the expense of outsourcing demand in the present. Therefore, the value of R must be chosen so that the resulting repositioning plan reflects the service level advantage of robustness and not the adverse effects of overconservatism.

Second, we focus on the effects of limiting N, the maximum number of terminals in an inbound-closed node set. Since increasing N increases the number of inbound-closed node sets (and hence the number of recoverability constraints) exponentially, it was not computationally feasible to test for very large values of N, especially in larger geographies. But it can be clearly seen in Figure 2.2 that increasing N causes a considerable increase in computation time, while not substantially improving the service

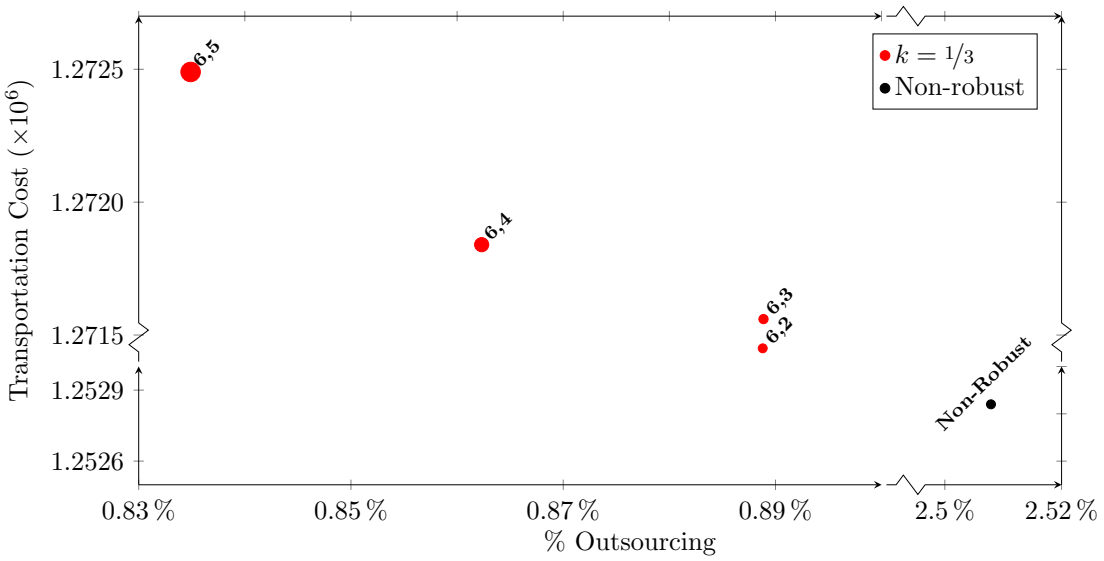
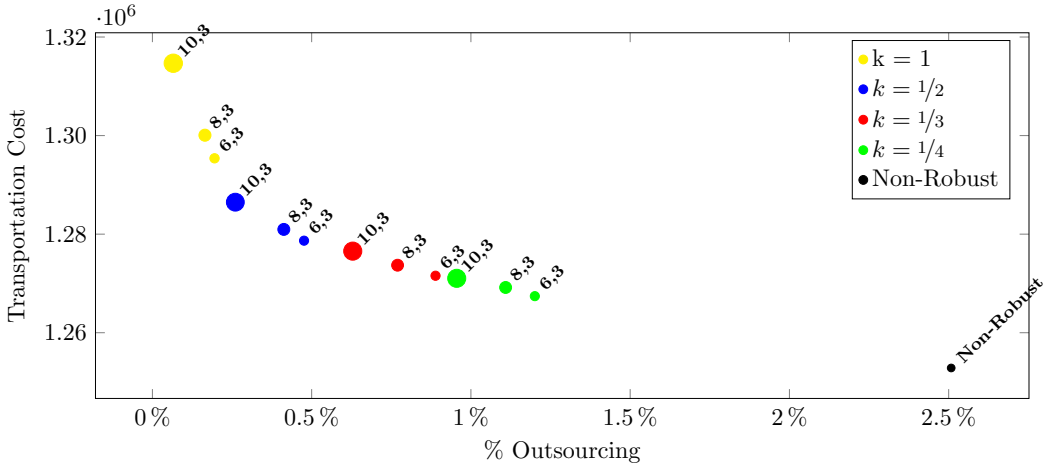


Figure 2.2: Effect of N in a 30-terminal geography (averaged over 30 instances) (Marker size is proportional to the natural logarithm of total computation time.)

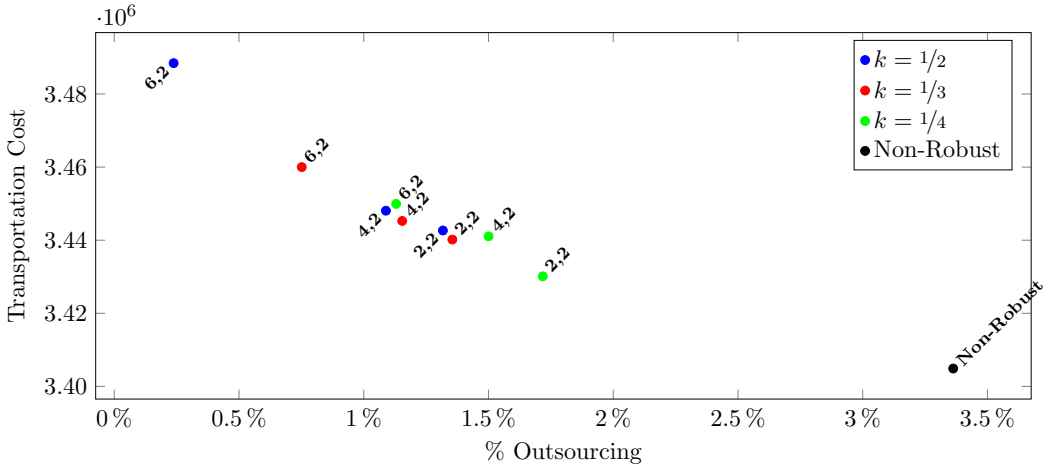
level. We observed a similar trend in the outcomes of our experiments with $N = 2$ and $N = 3$ in larger geographies. Generally, considering more than two terminals in a recoverability constraint does not improve overall solution performance significantly. This outcome was expected, since we only allow empty repositioning between empty hubs and their spokes and not between spokes. Setting $N = 2$ corresponds to having every spoke with its designated hub in inbound-closed node set, and allowing more terminals in such sets by increasing N only slightly improves the service level.

Finally, the impact of the vulnerability multiplier, which dictates how wide the robust uncertainty interval is compared to the underlying demand distribution, on solution quality is investigated. Examining Figure 2.3, it is clear that on average, as k increases from $1/4$ to $1/2$, the service level increases without causing the total transportation cost to increase by much and without a noticeable increase in computation time. However, as can be seen especially in the 30-terminal geometry, increasing k to 1, *i.e.*, protecting against the worst-case demand realizations, can actually results in only slightly better service levels, but at significantly higher total transportation costs. Furthermore, in some individual test instances, we observe lower service levels

than the non-robust approach when protecting against worst-case demand realizations by setting $k = 1$. Similar to increasing R excessively, this is due to overconservatism. When $k = 1$, the repositioning plans are protected against all uncertainty, which means that the plan can be recovered even when all future loaded demand is realized at the worst-case values. Generally, this much conservatism is unnecessary for practical purposes, and it generates solutions that are even more vulnerable to uncertainty than those produced by a non-robust approach, by accumulating resources at certain terminals for potential future need at the cost of outsourcing demand at other terminals today.



(a) 30-terminal geography



(b) 100-terminal geography

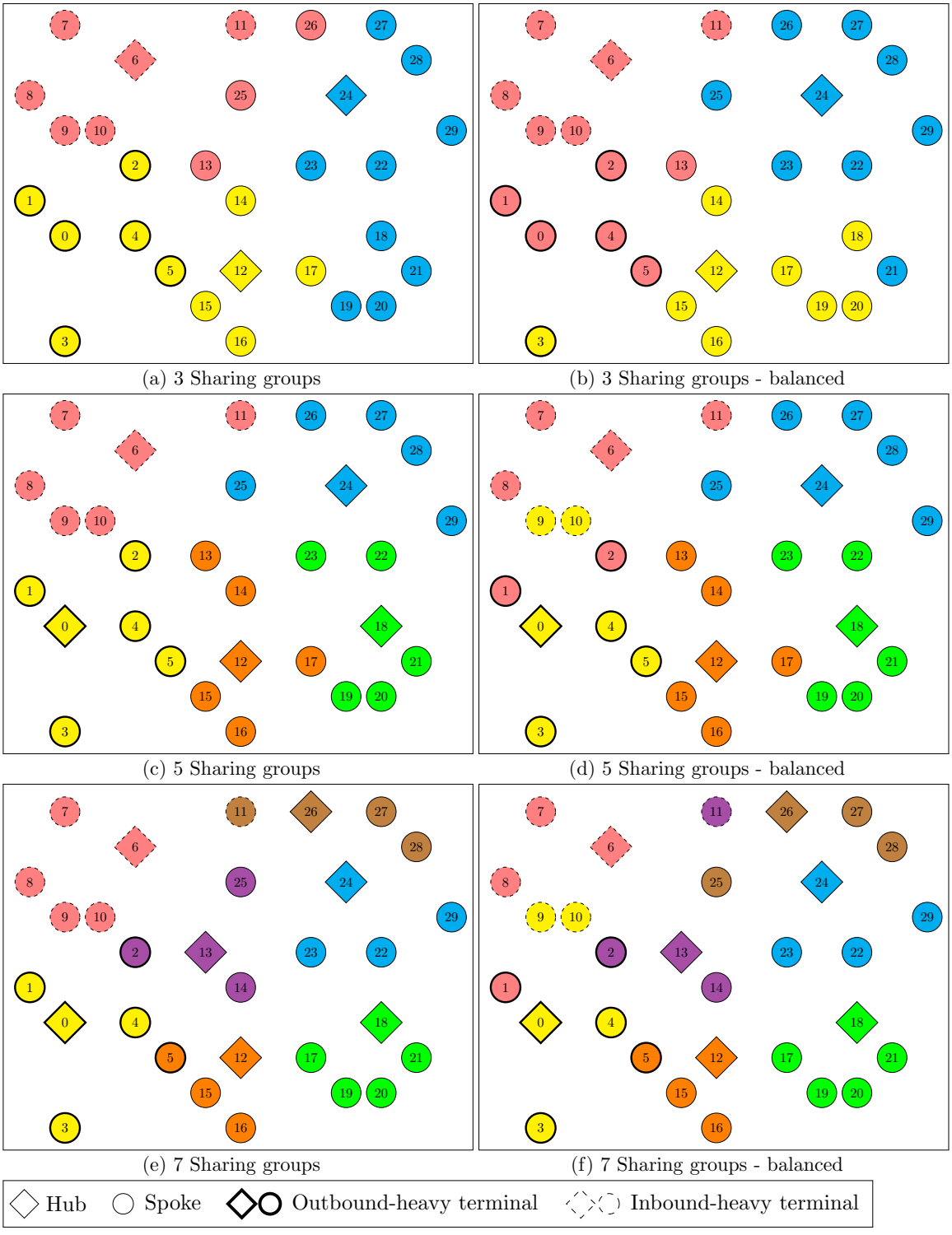
Figure 2.3: Effect of k (averaged over 30 instances) (Marker size is proportional to total computation time.)

In our computational experiments where *(i)* about 10% of the terminals act as empty hubs, *(ii)* the maximum travel time between an empty hub and a designated spoke and between two empty hubs are 3 and 10 periods, respectively, and *(iii)* uncertain demand quantities follow a uniform distribution; we observe that the proposed robust methodology with robustness parameters $(R, N) = (8, 2)$ and $k = 1/3$ produces high-quality repositioning plans in acceptable computation times, *i.e.*, about 250 seconds for instances with 30 terminals, about 4,350 seconds for instances with 80 terminals, and about 10,400 seconds for instances with 100 terminals. In a similar manner, robustness parameters that would provide the most desirable outcome in terms of service level, total transportation cost, and computational burden can be determined for different geographies and network parameters.

2.4.3.2 Sharing Group Configuration

In this section, we concentrate on the impact of sharing group configuration on service level and total transportation cost. More specifically, we investigate the effects of *(i)* the number and size of the sharing groups; and *(ii)* the volume of nominal inbound and outbound loaded demand of sharing groups. All computational experiments reported in this section are conducted on a 30-terminal geography with either 3, 5, or 7 sharing groups, where the number of terminals in each sharing group is the same (or as close to each other as possible), and each sharing group is either balanced, in terms of inbound and outbound loaded demand, or imbalanced. The different sharing group configurations are shown in Figure 2.4.

First, we examine the effects of sharing group balance in terms of inbound and outbound loaded demand. An imbalanced sharing group configuration has some sharing groups that dominantly act as the origin or the destination of load requests, whereas in a balanced configuration all sharing groups have roughly equal inbound and outbound load requests. It can be clearly seen in Figure 2.5 that the adverse effects of



(a) 3 Sharing groups

(b) 3 Sharing groups - balanced

(c) 5 Sharing groups

(d) 5 Sharing groups - balanced

(e) 7 Sharing groups

(f) 7 Sharing groups - balanced

Figure 2.4: Sharing group configurations for 30-terminal geography

uncertainty (outsourced load requests and excessive empty repositioning) are observed more strongly in a system with an imbalanced sharing group configuration (for both robust and non-robust empty repositioning plans). A balanced sharing group configuration provides an extra layer of risk pooling, so that within each sharing group, terminals with lower-than-expected net loaded demand can provide empty resources to those with higher-than-expected net loaded demand. This way, *(i)* outsourcing is reduced, as resources are more likely to be available within the sharing group when needed, and *(ii)* less empty repositioning occurs between sharing groups, as sharing groups are more self-sufficient in terms of demand and supply of empty resources.

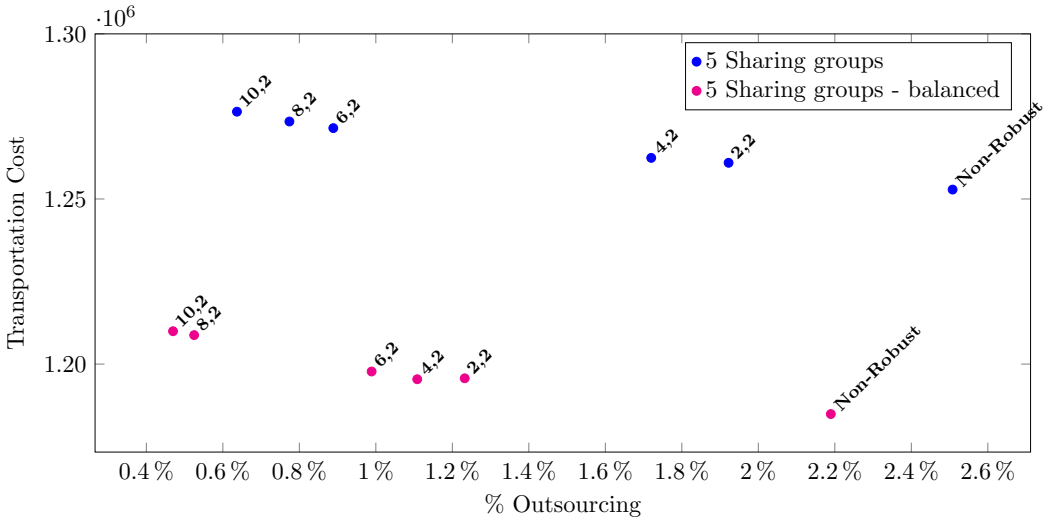


Figure 2.5: Effect of inbound/outbound loaded demand balance (30 terminals, $k = 1/3$, averaged over 30 instances)

Next, we focus on how the number and size of sharing groups affect outsourcing and total transportation cost. Figure 2.6 reveals patterns that can be explained with risk pooling and average travel distance within and between sharing groups. When the number of sharing groups increases from 3 to 5, the larger number of empty hubs eases the demand intensity on each hub, hence lessening the adverse effects of demand uncertainty (in both robust and non-robust empty repositioning plans). It is also worth noting that a sharing group configuration with 3 sharing groups naturally has sharing groups that cover a larger geographic area, and thus longer

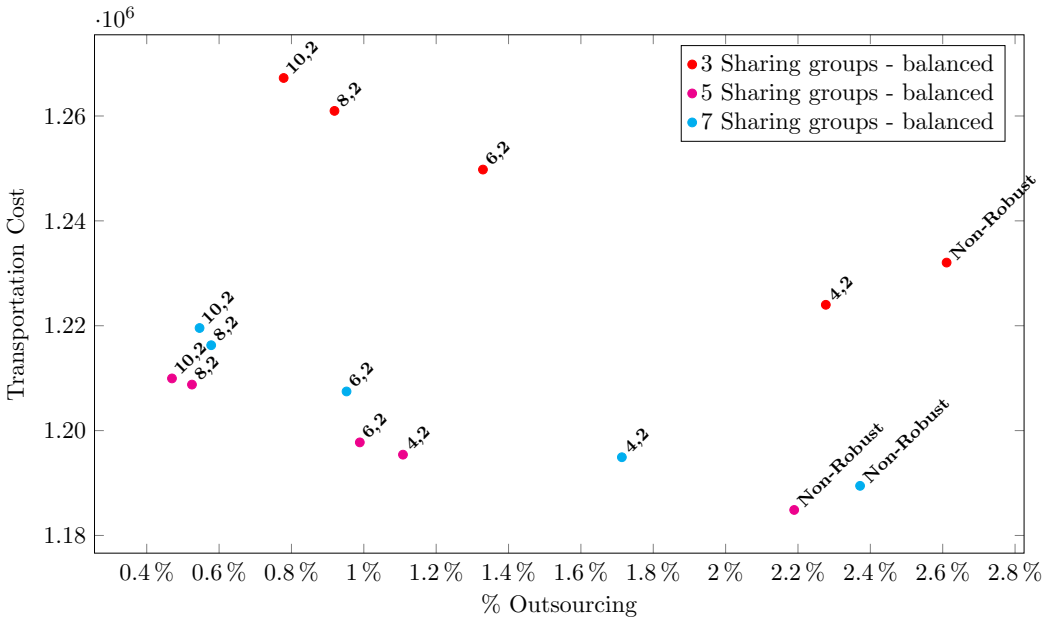


Figure 2.6: Effect of sharing group size (30 terminals, $k = 1/3$, averaged over 30 instances)

spoke-hub distances and therefore higher empty repositioning costs within a sharing group. When the number of sharing groups increases from 5 to 7, the decrease in sharing group size causes a decrease in the risk pooling within the sharing groups, and, as a consequence, an increase in empty repositioning between sharing groups. This leads to *(i)* greater total transportation cost, as the travel distances between sharing groups are relatively long compared to those within sharing groups, and *(ii)* more outsourcing, as the longer intergroup travel times can cause empty resources to arrive too late.

For every geography and demand intensity, sharing group configurations with appropriately sized and balanced (in terms of inbound and outbound loaded demand) groups can be designed in order to take advantage of pooling benefits.

2.5 Conclusion

In this chapter, we introduced a rolling horizon framework featuring a robust optimization model for an empty repositioning problem subject to arc demand uncertainty. The proposed robust model finds repositioning plans that are feasible for nominal demand quantities and recoverable (by using a predefined set of recovery actions) for every possible demand realization, or a subset of demand realizations. By using this two-stage robust model in a rolling horizon framework, we accurately represent the multi-stage nature of a typical transportation carrier's day-to-day operations. In order to pragmatically control the level of conservatism of the robust model, we defined model-specific robustness parameters.

We demonstrated the value of this framework with a comprehensive computational study, which includes problem instances with various network and fleet sizes, robustness parameters, and recovery action sets. Even though our computational study mainly concentrates on instances with *(i)* hub-and-spoke networks (and hub-to-spoke recovery actions), *(ii)* relatively small fleets, and *(iii)* demand imbalance, and heavily prioritizes service level over other performance measures (such as incurred transportation costs); the robust rolling horizon framework can be used with appropriate network and robustness parameters to accurately represent numerous network structures, demand distributions, and objectives. The computational results illustrate that when the robustness parameters are selected pragmatically, this framework can improve service level significantly compared to a nominal approach, while causing only a slight increase in transportation cost. Robustness parameters also help in keeping the formulation tractable, as the robust model tends to have exponentially many recoverability constraints when it aims for recoverability against all possible demand realizations.

Finally, we investigated how various sharing group configurations can affect the

quality of the repositioning plans in terms of operational and computational simplicity. By designing the sharing group configuration in a way that balances expected transportation volumes within and between sharing groups, we were able to improve service level and reduce transportation cost in our test instances.

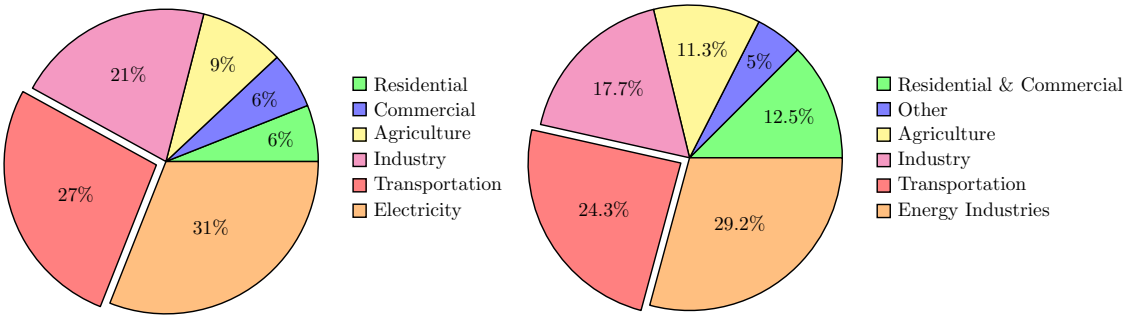
CHAPTER III

AN INTEGRATED FLEET MANAGEMENT MODEL INTRODUCING ALTERNATIVE FUEL TRUCKS

3.1 Introduction

Today, climate change has become a very important concern for the sustainability of the planet. Therefore, analyzing the sources of greenhouse gas (GHG) emissions and working on emission reduction strategies are now more important than ever. As can be seen on Figure 3.1a, the transportation sector is one of the largest contributors to U.S. GHG emissions. Furthermore, between 1990 and 2013, GHG emissions in this sector increased more in absolute terms than any other sector. Medium- and heavy-duty trucks, which constitute the focus of this research, are responsible for 23% of emissions in the transportation sector. [45] The situation is similar in the European Union (EU), where transportation generates 24.3% of total GHG emissions (Figure 3.1b). Heavy-duty vehicles (HDVs) produce about a quarter of CO_2 emissions from the road transport sector. More importantly for this study, despite efforts for improvements in fuel efficiency, CO_2 emissions from HDVs rose by 36% between 1990 and 2010, mainly due to increasing road freight traffic. The European Commission (EC) recognizes that changes need to be made in order to meet their goal of reducing GHG emissions from transport to 60% below 1990 levels by 2050. [44]

Due to environmental concerns, potential fuel savings, and encouraging government funding, there have been widespread efforts to switch to greener fleets in heavy-duty trucking. In 2012, Coca-Cola added more than 750 hybrid electric delivery trucks (with 35 ft trailers) to its U.S. and Canada fleet, which in turn amounted to 10% of their overall North American fleet. [37] In October 2015, they added 1000



(a) In the U.S. (2013)

(b) In the EU (2012)

Figure 3.1: Greenhouse gas emissions by sector

more Alternative Fuel Vehicles (AFVs) to their fleet, making their fleet the largest heavy-duty hybrid fleet in North America. [40] In 2008, the largest port complex in the U.S., located in Southern California, started a truck replacement initiative for adding 219 liquified natural gas (LNG) trucks to be used in their operations, which resulted in 182 LNG trucks actively on the road in June 2012 and more trucks whose purchases were under way. [38] Heavy-duty AFVs also attracted interest from companies with smaller fleets. For example, M&M Cartage, a freight hauling company based in Kentucky, recently purchased 30 compressed natural gas (CNG) trucks that reduce CO_2 emissions by 11 grams per mile compared to diesel trucks, and plan to purchase 100 more by 2018. [39] Another example is Monarch Beverage Company, located in Indiana. Since 2009, they purchased 80 CNG haulers for their fleet and built a CNG fueling station, which resulted in fuel cost savings of over $2 million a year. [41]

This research is motivated by the practice of introducing alternative fuel trucks (AFTs) into an existing diesel fleet in long-haul trucking. Though the method proposed in this study is not limited to facilitating the introduction of AFTs but also can be used for managing any stage of any fleet mix, it is particularly helpful for smoothly integrating new fleet types that also require some infrastructural changes in the trucking network, such as additional or modified maintenance facilities or fueling

stations, into existing fleet.

We present an integrated fleet management model, which we will refer to as $\mathbf{P}$, that takes into consideration multiple levels of interdependent decisions that may be faced by a company with a long-haul trucking fleet while facilitating fleet replacement. The model makes fleet purchase and retirement decisions, while ensuring feasible operations and also exploring possible infrastructural changes. As the main focus of this study is introducing new types of trucks into the fleet, the infrastructural change taken into account in $\mathbf{P}$ is modifying some existing terminals to make them compatible for maintaining and/or fueling the new truck types. The planning horizon is divided into time periods, and the model is designed to ensure economical fleet replacement decisions, feasible maintenance and fueling for every truck type, and customer demand satisfaction at every time period.

Integrated fleet management model is examined in detail from multiple perspectives. Two variants of the model, which dictate AFT purchases by enforcing a certain proportion of the total truck-miles and total loaded truck-miles, respectively, to be traveled by AFTs, are introduced and some characteristics of these variants are investigated. Then, in order to demonstrate the benefits of including infrastructural and operational considerations in the model, a computational study examining the optimal solutions of the model is conducted for various demand, target AFT proportion, and cost scenarios. Finally, a Benders' decomposition approach and a variable neighborhood search (VNS) algorithm are proposed, and their performances are assessed with extensive computational experiments.

The remainder of this chapter is organized as follows: In Section 3.2, existing literature on fleet replacement problems is presented in detail. Section 3.3 presents the two variants of the integrated fleet management problem ($\mathbf{P}$, $\mathbf{P}'$) and a computational study analyzing the effects of integrating infrastructural and operational decisions into the model on optimal solutions of $\mathbf{P}'$. Section 3.4 describes a Benders' decomposition

framework and a variable neighborhood search algorithm to efficiently solve $\mathbf{P}'$, and Section 3.5 provides an extensive computational study demonstrating the performance of these solution methodologies. Finally, Section 3.6 summarizes our findings.

3.2 Literature Review

Equipment replacement decisions have been important in production and service industries for a long time. Various decision-making strategies, especially optimization methods are suggested for this problem in the literature. The earlier examples of these optimization methods utilize dynamic programming (DP), such as [126] and [11]. Following those, [42] extends their work to take into account technological improvements. Later, a new dynamic programming formulation for the problem is presented in [130]. [35] extend the decision set by introducing some new decisions, and demonstrate their proposed method in real-life applications. [116] suggests using rolling horizon logic in a dynamic programming setting for equipment replacement decisions, and shows important properties of regeneration points.

As the dynamic programming models improved more and more to represent real-life situations, various problem parameters are modeled to be uncertain. [116] mention incorporating probabilistic machine breakdowns into their model. [90] introduce a Markov decision process (MDP) that operates in an equipment replacement setting where the timing of technological improvement, as well as the additional revenue it may generate, are uncertain. Later, [63] presents a dynamic program where asset utilizations are probabilistic, and [66] considers a problem where the demand is stochastic and machine utilizations vary by the age of the machine. A model where technological breakthroughs are assumed to arrive stochastically and result in new cost structures that are also stochastic is developed in [68].

In relatively recent examples in dynamic programming literature, various aspects of the equipment replacement problem are still being explored. For example, [67]

investigates the cases where the optimal infinite-horizon solution for equipment replacement is also a good solution to the finite-horizon problem. Other studies, such as [52] and [97], present successful applications of the deterministic DP formulation to real-life problems. [52] uses an equipment replacement DP with a backward recursion on Texas Department of Transportation vehicle fleet data, while [97] study 4×4 vehicle replacement within the International Committee of the Red Cross.

Another line of research relies on linear (LP), integer (IP), or mixed-integer programming (MIP) models to represent and solve equipment replacement problems. A basic LP for multiple period equipment replacement problem that models multiple equipment types is presented in [54], and it is later extended to support airline operations in [91]. [72] presents a fleet replacement IP model that looks into keep or replace decisions of each individual asset at every time period, and suggests a Lagrangian relaxation for solving the large IP. [103] adapts a basic equipment replacement MIP to support demand fluctuations by making capacity expansion decisions together with equipment replacement decisions. [65] presents an IP to solve a parallel replacement problem with capital budgeting constraints, and explores some polyhedral characteristics. Later, [73] demonstrates the effectiveness of that IP in a real-life case study on the operations of a city transit bus operator in Europe. [25] define a MIP for parallel replacement of equipment under economies of scale, and proposes some effective cutting planes for the problem.

When equipment replacement decisions arise, there are usually other interdependent decisions that require attention. This is a crucial fact that motivated this research, as we propose making infrastructure, fleet replacement, and fleet deployment decisions jointly. There are examples in the equipment replacement literature that stem from the same motivation. [119] suggests an IP that simultaneously makes fleet replacement and fleet operating decisions in an urban transit setting. [64] and [133] present integer and linear programming models, respectively, that determine tactical

(replacement) and operational (utilization) decisions at the same time. [133] also provides a decomposition-based solution methodology for the integrated replacement and utilization problem. [70] introduce an optimal control model that highlights the benefits of replacement and utilization decisions being made jointly, rather than sequentially in a fleet of vessels. In similar spirit, [125] describe an integer programming approach to combine fleet replacement decisions with aggregated task assignment decisions, and [69] study the fleet replacement and composition problem while explicitly accounting for vehicle routing costs.

Quite a few recent studies incorporate environmental considerations into equipment replacement decisions. Particularly in the field of transportation, where reduction in carbon emissions is crucial, such considerations are very important. [125] considers optimal fleet replacement and assignment for a CNG bus fleet. Another model for making replacement decisions in a fleet with CNG and diesel busses is developed by [60], which finds cost-minimizing strategies that meet pollutant reduction goals. Similarly, to reduce carbon emissions, [55] and [53] propose IPs minimizing purchase, operating, maintenance, and emissions costs for electric passenger cars and electric commercial vehicles, respectively. [96] develops an IP that considers purchasing and retirement of trucks with multiple fuel options, while also ensuring demand satisfaction in an aggregate sense.

[9] develops a two-stage game theoretic model that helps with the decision of how much to rely on alternative fuel vehicles and evaluates the implications of greening of the transportation fleet. Also using a game theoretical approach, [31] presents a cooperative game considering various stakeholders for the electric vehicle adoption decision in commercial fleets. They give the fleet of La Poste, the national postal operator of France, as an example setting where their methodology can be useful. Later, [77] proposes a discrete-time, stochastic dynamic programming formulation for the electric vehicle adoption decision of La Poste, which supports uncertainty fuel

and battery acquisition prices. A Markov decision process for fleet replacement is proposed in [120], which decides on whether to keep the existing technology, upgrade to a newer technology which produces a smaller environmental burden, or wait for an even newer, cleaner technology which may be introduced soon. Another modeling and solution approach that embraces uncertainty in fleet replacement is presented in [6], where a stochastic program is formulated using conditional value at risk (CVaR) to account for uncertainty in the decision process.

3.3 Integrated Fleet Management Model

In this section, two variants of a MIP formulation for making fleet replacement decisions in a setting where alternative fuel trucks (AFTs) are being introduced into the existing fleet of diesel trucks (DTs) is presented. Then, the benefits of integrating maintenance facility opening and routing considerations into this model is emphasized by presenting optimal solutions to the model under numerous demand, target, and cost scenarios, and inspecting the effects of integrated decision making.

The MIP formulation, referenced as the "integrated fleet management model" simultaneously takes into account strategic, tactical, and operational decisions. Since AFTs require very different maintenance and fueling methods than diesel trucks, the strategic decision of when and where to open new maintenance facilities to accommodate AFTs is modeled explicitly. Tactical decisions concern when and how many of each truck type to purchase and retire, and operational decisions manage truck deployment to predefined service routes. The model resembles a multi-commodity network flow model, while also having network design characteristics due to the maintenance facility opening decisions. The two variants of the model are distinguished by how they induce AFT purchases. The first variant ($\mathbf{P}$) and the second variant ($\mathbf{P}'$) enforce certain proportions of total truck-miles and total loaded truck-miles, respectively, to be traveled by AFTs at every time period.

3.3.1 Integrated Fleet Management Model with Truck-mile Targets

The model is defined over a network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where nodes represent terminals and arcs represent connections between terminals. Trucks are assumed to operate on a set of predefined routes, $\mathcal{R}$. These routes are designed to be short enough for AFTs to run without requiring intermediate fuel or maintenance stops. Customer demand, which requires truck loads to be transported between specific origins and destinations at specific time periods, must be satisfied by the fleet at hand. Furthermore, targets specifying the minimum proportion of AFT truck-miles to total truck-miles are enforced, as an incentive for the trucking company to switch to an AFT fleet.

Three types of decisions are modeled in **P**. These decisions, which are defined over a planning horizon of T time periods are listed below.

- Opening maintenance facilities to service alternative fuel trucks, denoted by binary decision variables y_{jt} for every terminal $j \in \mathcal{N}$ and time period $t = 1, \ldots, T$
- Purchase and retirement of diesel (D) and alternative fuel (A) trucks, where decision variables
  - p_t^k represent the number of trucks of type $k \in \{A, D\}$ to be purchased in the beginning of period $t \in \{1, \ldots, T\}$, and
  - $r_{\tau t}^k$ represent the number of trucks of type $k \in \{A, D\}$ purchased in the beginning of period $\tau \in \{1, \ldots, t\}$ to be salvaged in the beginning of period $t \in \{1, \ldots, T\}$.
- Assigning diesel and alternative fuel trucks to routes in the network, denoted by decision variables $x_{i\tau t}^k$ to represent the number of trucks of type $k \in \{A, D\}$ purchased in the beginning of period $\tau \in \{1, \ldots, t\}$ to be assigned to route $i \in \mathcal{R}$ in period $t \in \{1, \ldots, T\}$

Decision variables other than the ones representing the maintenance facilities are modeled as continuous variables. This is a reasonable approximation for reasonably-sized fleets, where the number of trucks assigned to routes are relatively large. The parameters used in the model are as follows:

- C_{jt} Cost of opening a maintenance facility for AFTs at terminal j in period t, and operating it until the end of the planning horizon
- P_t^k Cost of purchasing a truck of type k at the beginning of period t (discounted to present time)
- $R_{\tau t}^k$ Salvage value of retiring a truck of type k purchased at the beginning of period τ at the beginning of period t (discounted to present time)
- $c_{\tau t}^k$ Cost of operating a truck of type k purchased at the beginning of period τ in period t
- $\bar{\bar{p}}_t^D$ Number of diesel trucks existing in initial fleet, which were purchased in period t ($t \leq 0$)
- n_i Number of times route i can be traversed by a truck in one period
- τ_a Travel time on arc a
- D_{at} Demand on arc a in period t, in number of full truck loads
- M_{it} Big-M value denoting the maximum number of alternative fuel trucks that will benefit from opening an alternative fuel truck maintenance facility along route i (Calculated as $M_{it} = \frac{\max_{a \in \mathcal{A}(i)} D_{at}}{n_i}$, where $\mathcal{A}(i)$ denotes the arcs of route i)
- b_t Target proportion of the AFT-miles to total truck-miles traveled in period t

The model formulation of **P** is presented below. (3.1) is a cost-minimizing objective function, where maintenance facility opening/operating, truck purchasing and operating costs are considered, as well as salvage values of retiring trucks. It is important to note that an end-of-horizon cost is incorporated into the truck operating cost

at the end of the planning horizon, $c^k_{\tau T}$, for each truck type $k \in \{A, D\}$ and purchase period $\tau \leq T$, in order to accurately represent end-of-horizon effects. Constraint set (3.2) represents inventory balance equations for existing fleet that was purchased at or before time period 0. (3.3) and (3.4) are typical inventory balance and demand satisfaction constraints, respectively. (3.5) represents big-M constraints that prohibit assignment of alternative fuel trucks to routes with no maintenance facility for those trucks. M_{it} values are calculated as the maximum number of trucks necessary to satisfy the demand of every arc along route i in period t, in order to prevent the utilization of more AFTs than necessary for demand satisfaction. Finally, target constraints that enforce at least a certain proportion of the truck-miles must be traveled by alternative fuel trucks are presented in (3.6). (Note that satisfying truck-mile targets is the same as satisfying targets in terms of number of trucks, since the trucks are assumed to be continuously traversing the routes.)

$$(\mathbf{P}) \quad \min \sum_{j\in\mathcal{N},t} C_{jt}y_{jt} + \sum_{k,t} P^k_t p^k_t - \sum_{k,t,\tau\leq t} R^k_{\tau t} r^k_{\tau t} + \sum_{i\in\mathcal{R},k,t,\tau\leq t} c^k_{\tau t} x^k_{i\tau t} \tag{3.1}$$

$$\text{s.t.} \sum_{i\in\mathcal{R}} x^D_{i\tau t} = \bar{\bar{p}}^D_\tau - \sum_{s=\tau}^{t} r^D_{\tau s} \qquad t = 1,\ldots,T, \;\; \tau \leq 0 \tag{3.2}$$

$$\sum_{i\in\mathcal{R}} x^k_{i\tau t} = p^k_\tau - \sum_{s=\tau}^{t} r^k_{\tau s} \qquad k \in \{\text{A, D}\}, \;\; t = 1,\ldots,T, \;\; \tau \leq t \tag{3.3}$$

$$\sum_{k,i\in\mathcal{R}(a),\tau\leq t} n_i x^k_{i\tau t} \geq D_{at} \qquad a \in \mathcal{A}, \;\; t = 1,\ldots,T \tag{3.4}$$

$$x^A_{i\tau t} \leq M_{it} \sum_{j\in\mathcal{N}(i),s\leq t} y_{js} \qquad i \in \mathcal{R}, \;\; x, \;\; \tau \leq t \tag{3.5}$$

$$\sum_{i\in\mathcal{R},\tau\leq t} x^A_{i\tau t} \geq b_t \sum_{k,i\in\mathcal{R},\tau\leq t} x^k_{i\tau t} \qquad t = 1,\ldots,T \tag{3.6}$$

$$y_{jt} \in \{0,1\} \qquad j \in \mathcal{N}, \;\; t = 1,\ldots,T \tag{3.7}$$

$$p^A_t, p^D_t \geq 0 \qquad t = 1,\ldots,T \tag{3.8}$$

$$r^A_{\tau t}, r^D_{\tau t} \geq 0 \qquad t = 1, \ldots, T, \;\; \tau \leq t \tag{3.9}$$

$$x^A_{i\tau t}, x^D_{i\tau t} \geq 0 \qquad i \in \mathcal{R}, \;\; t = 1, \ldots, T, \;\; \tau \leq t \tag{3.10}$$

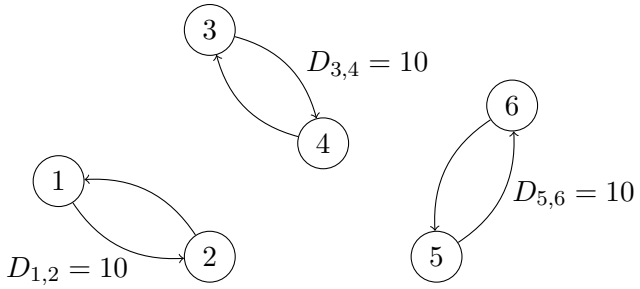


Figure 3.2: Sample network with 6 terminals and 3 routes

An important shortcoming of **P** in realistically representing fleet management operations is that it allows purchasing and operating AFTs only to satisfy AFT target requirements, even when they are not needed for demand satisfaction. To exemplify a scenario where this may happen, consider a single-period problem on the 6-terminal geography presented in Figure 3.2 with three demand arcs (1, 2), (3, 4), and (5, 6), each having a demand of 10 truck loads; and three routes $\{(1,2),(2,1)\}$, $\{(3,4),(4,3)\}$, and $\{(5,6),(6,5)\}$, each having a travel time of 1 period. Suppose that the target requirements dictate that at least half of the total miles traveled must be traveled by AFTs. In this setting, let solution (1) be opening 1 maintenance facility and purchasing/running 20 AFTs and 20 DTs, and solution (2) be opening 2 maintenance facilities and purchasing/running 20 AFTs and 10 diesel trucks. If solution (1) is cheaper than solution (2), **P** would choose solution (1), which requires purchasing 10 more AFTs than required by the demand and running them empty so that the target requirement is satisfied. Clearly, a solution where trucks are running empty without satisfying any demand is undesirable from an operational viewpoint, even when it is cheaper. In what follows, we discuss the circumstances under which this never happens when the following simplifying assumptions hold:

(1) Demand is constant, *i.e.*, $D_{at} = D_a, \quad \forall t = 1, \ldots, T.$

(1) There is no initial fleet, *i.e.*, $\bar{\bar{p}}^D_\tau = 0, \quad \forall \tau \leq 0.$

(1) There is only one target constraint (requiring at least b proportion of the fleet to be AFTs) at the beginning of the planning period, *i.e.*, $b_t = b, \quad \forall t = 1, \ldots, T.$

(1) Maintenance facility opening and operating cost is the same for all terminals, *i.e.*, $C_{jt} = C_t, \quad \forall j \in \mathcal{N}.$

Proposition 3.1. *Let $F \subseteq \mathcal{N}$ represent the set of terminals where maintenance facilities are opened at the beginning of the planning horizon and $\mathcal{R}(F) \subseteq \mathcal{R}$ represent the set of routes passing through the terminals in F. Furthermore, let $\mathcal{A}(F) = \{a \in \mathcal{A} : \exists R \in \mathcal{R}(F) : a \in R\}$ denote the set of arcs on all routes passing through the terminals in F. Under assumptions* **(1) - (4)***, running empty AFTs only to satisfy targets will never be optimal under* $\mathbf{P}$ *unless the demand on all arcs $a \in \mathcal{A}(F)$ is satisfied with AFTs.*

Let $TC^k = P^k_1 + \sum_{t=1}^{T} c_{1t} + Q^k_{1,T+1}$ for $k \in \{A, D\}$ denote the total lifetime cost of both truck types. ($Q^k_{t_1,t_2}$ represents the present worth of all costs incurred after period $t_2 - 1$ if a truck of type k that is purchased at period t_1 is kept and optimally replaced indefinitely.) Note that this definition of lifetime cost assumes that the payback period of the trucks is longer than the planning horizon, but it can be easily extended to the case where the payback period is shorter than T. Let X^A_F and X^D_F denote the minimum number of trucks required to satisfy all the demand in $\mathcal{A}(F)$ and $\mathcal{A} \setminus \mathcal{A}(F)$, respectively,

$$X^A_F = \min_{\mathbf{x} \in \mathbb{R}^{|\mathcal{R}(F)|}_+} \{\mathbf{1}^T \mathbf{x} : \sum_{i \in \mathcal{R}(F)} x_i \geq D_a, \forall a \in \mathcal{A}(F)\}$$

$$X^D_F = \min_{\mathbf{x} \in \mathbb{R}^{|\mathcal{R}|}_+} \{\mathbf{1}^T \mathbf{x} : \sum_{i \in \mathcal{R}} x_i \geq D_a, \forall a \in \mathcal{A} \setminus \mathcal{A}(F)\}$$

and $e_F = \frac{b}{1-b} X_F^D - X_F^A$ quantify the number of AFTs that must run empty to satisfy the target constraints with a given set of maintenance facilities, F. Furthermore, let X^{min} denote the minimum number of trucks required to serve the demand, *i.e.*,

$$X^{min} = \min_{\mathbf{x} \in \mathbb{R}_+^{|\mathcal{R}|}} \{\mathbf{1}^T \mathbf{x} : \sum_{i \in \mathcal{R}} x_i \geq D_a, \forall a \in \mathcal{A}\}$$

and $F^{min} = \arg\min_{F \subseteq \mathcal{N}} \{|F| : X_F^A + X_F^D = X^{min}, \frac{X_F^A}{X_F^A + X_F^D} \geq b\}$ represent the smallest set of terminals, opening maintenance facilities at which will allow the demand to be satisfied with X^{min} trucks and the target to be satisfied.

Theorem 3.1. *Under assumptions* **(1) - (4)**, *an optimal solution of* $\mathbf{P}$ *will never require running empty AFTs only to satisfy targets if*

(i) $TC^A > TC^D$ *and* $(X_F^A + e_F - bX^{min})TC^A + (X_F^D - (1-b)X^{min})TC^D > (|F^{min}| - |F|)C_1$, *or*

(i) $TC^A \leq TC^D$ *and* $(X_F^A + e_F - X_{F^{min}}^A)TC^A + (X_F^D - X_{F^{min}}^D) > (|F^{min}| - |F|)C_1$

for every $F \subseteq \mathcal{N}$ *with* $e_F > 0$.

3.3.2 Integrated Fleet Management Model with Loaded Truck-mile Targets

For cases where the conditions in Theorem 3.1 do not hold, we propose a modified model $\mathbf{P}'$ that enforces target constraints based on loaded travel. Clearly, when AFT targets are enforced on loaded truck-miles, the AFTs will never be purchased and operated empty only to satisfy targets. $\mathbf{P}'$ is formulated below, where the additional set of variables l_{at}^k denotes the number of truck loads of type k passing through arc a in period t for all $k \in \{A, D\}$, $a \in \mathcal{A}$, and $t = 1, \ldots, T$. The target constraints in this formulation requires at least a certain proportion of loaded truck-miles to be traveled by AFTs.

$$\mathbf{P}' \quad \min \sum_{j \in \mathcal{N}, t} C_{jt} y_{jt} + \sum_{k,t} P_t^k p_t^k - \sum_{k,t,\tau \leq t} R_{\tau t}^k r_{\tau t}^k + \sum_{i \in \mathcal{R}, k, t, \tau \leq t} c_{\tau t}^k x_{i\tau t}^k \tag{3.11}$$

s.t. (3.2), (3.3), (3.5), (3.7), (3.8), (3.9), (3.10)

$$\sum_{i\in\mathcal{R}(a),\tau\le t} n_i x^k_{i\tau t} \ge l^k_{at} \qquad k\in\{\text{A, D}\},\ \ a\in\mathcal{A},\ \ t=1,\dots,T \tag{3.12}$$

$$l^A_{at} + l^D_{at} = D_{at} \qquad a\in\mathcal{A},\ \ t=1,\dots,T \tag{3.13}$$

$$\sum_{a\in\mathcal{A}} \tau_a l^A_{at} \ge b_t \sum_{k,a\in\mathcal{A}} \tau_a l^k_{at} \qquad t=1,\dots,T \tag{3.14}$$

$$l^A_{at}, l^D_{at} \ge 0 \qquad a\in\mathcal{A},\ \ t=1,\dots,T, \tag{3.15}$$

Constraint set (3.12) couples the $x^k_{i\tau t}$ and l^k_{at} variables, and (3.13) ensures demand satisfaction. Finally, constraint set (3.14) enforce loaded truck-mile targets for AFTs. From this point on in this chapter, the computational experiments are going to be conducted on this variant of the integrated fleet management model, rather than **P**.

In order to simplify the model, sufficient conditions for never having AFT purchases or maintenance facility openings at time periods other than "critical periods" are developed in Theorem 3.2. (Here, critical periods are period 1 and the periods when there is an increase in AFT targets.) This idea is implemented in the Benders' decomposition framework we introduce in Section 3.4.1 to improve convergence behavior.

Theorem 3.2. *Let* $C(t_1,t_2,a,d) = d\times(P^D_{t_1}+\sum_{t=t_1}^{t_2-1} c^D_{t_1,t} - R^D_{t_1,t_2}) + a\times(P^A_{t_2}+\sum_{t=t_2}^{T} c^A_{t_2,t} + Q^A_{t_2,T+1})$ *denote the total cost of purchasing* d *diesel trucks at period* t_1*, operating them until* t_2*, retiring them and purchasing* a *AFTs at period* t_2*, and operating those indefinitely. Under assumptions* **(1)**, **(2)** *and* **(4)**, *an optimal solution of* $\mathbf{P}'$ *will never require opening maintenance facilities or purchasing AFTs at periods other than* $\{1\}\cup\mathcal{T}$ *if*

(i) $C(t_1,t_2,1,1) < C(t_1,t_2-1,1,1)$*, or*

(i) $C(t_1,t_2,1,d^*) > C(t_1,t_2-1,1,d^*)$ *and* $M_{t_2-1} - M_{t_2} \le C(t_1,t_2,1,d^*) - C(t_1,t_2-$

$1, 1, d^*)$

for all $t_1 = 1, \ldots, T$ *and* $t_2 \geq t_1$, *where* $d^* = \frac{\min_{i \in \mathcal{R}} n_i}{\max_{i \in \mathcal{R}} n_i}$, *and* $\mathcal{T} = \{1 \leq t \leq T : b_t \geq b_{t-1}\}$.

3.3.3 Optimal Solutions of the Integrated Fleet Management Model with Loaded Truck-mile Targets ($\mathbf{P}'$)

In this section, we present optimal fleet management strategies found by the integrated fleet management model with loaded truck-mile targets ($\mathbf{P}'$) on small benchmark problem instances with various demand, target, and cost characteristics. In each class of problem instances, we investigate the optimal fleet management plans and emphasize the impact of integrating maintenance facility and routing decisions into the fleet replacement model.

The problem instances are defined over a small network with 10 terminals and 24 time periods, where there is an initial fleet of DTs. 2 truck types (DTs and AFTs) are considered for being purchased and dispatched in the planning horizon. Truck operating costs in the mathematical model represent both fuel costs and planned/unplanned maintenance costs. In order to include aging effects in the model, these costs increase linearly with the age of the truck. Salvage values of trucks are calculated based on a straight line depreciation method, where the salvage value drops to zero at the end of the truck's useful life. In order to minimize end-of-horizon effects, we assume that the fleet at the end of the planning horizon is kept and operated indefinitely, while replacing the trucks at their optimal replacement periods. Finally, all cost components are designed to reflect the time value of money. In our computational experiments, one period corresponds to a month, and the interest rate is chosen accordingly.

Customer demand is randomly generated based on given origin/destination probabilities for each terminal. To represent the inherent demand imbalance in transportation systems, the origin/destination probabilities are set up so that some regions are

mostly origins of loaded moves (outbound-heavy), some regions are mostly destinations of loaded moves (inbound-heavy), and some regions have a balanced demand pattern.

The simplest problem instances have truck costs, demand, and AFT targets that do not change over time. In more complicated cases, we consider problem instances with increasing targets, changing demand, and changing truck costs. In every problem instance with constant truck costs, we examine the cases in which AFTs are cheaper/more expensive than diesel trucks to own and operate, and maintenance facilities are expensive/cheap to open and operate.

3.3.3.1 Constant Demand, Constant Target

The simplest problem instances are ones with constant truck costs, demand, and AFT targets. Here, customer demand is the same every period, and there is a constant target proportion for AFT loaded truck-miles, which is imposed at the beginning of the planning horizon and remains in effect until the end. Additionally, truck costs do not change over time, *i.e.*, the costs associated with purchasing and operating an AFT (or a DT) is independent from the purchase (or operating) period. In these problem instances, one truck type always has a lower total lifetime cost than the other, so the model always favors one truck type. With truck preference being clear, the only drivers of fleet replacement decisions are maintenance facility costs and AFT targets.

When truck costs, demand, and AFT targets are constant over time, the integrated fleet management problem resembles a single period model, where the objective is to decide on the optimal fleet mix. If there was no initial fleet, the optimal solution of $\mathbf{P}'$ would be to open necessary maintenance facilities and purchase as many trucks of each type as required by the single period problem at the beginning of the planning horizon, and keep replacing the trucks as they reach their optimal replacement age. When there is an initial fleet of DTs, the main question becomes when and with which

Table 3.1: Constant demand, constant target

(a) Expensive AFT, expensive maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	80.84	62.52	130.00	0.00	228.84	62.52
5	64.00	0.00	64.00	0.00	228.84	62.52
15	84.00	0.00	84.00	0.00	228.84	62.52
21	80.84	0.00	80.84	0.00	228.84	62.52

Maintenance facilities: 1 opened at period 1

(b) Expensive AFT, cheap maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	84.97	58.40	130.00	0.00	232.97	58.40
5	64.00	0.00	64.00	0.00	232.97	58.40
15	84.00	0.00	84.00	0.00	232.97	58.40
21	84.97	0.00	84.97	0.00	232.97	58.40

(c) Cheap AFT, expensive maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	0.00	143.37	130.00	0.00	148.00	143.37
5	44.03	19.97	64.00	0.00	128.03	163.33
15	84.00	0.00	84.00	0.00	128.03	163.33

Maintenance facilities: 1 opened at period 1

(d) Cheap AFT, cheap maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	0.00	143.37	130.00	0.00	148.00	143.37
5	0.00	64.00	64.00	0.00	84.00	207.37
15	0.00	84.00	84.00	0.00	0.00	291.37

truck type to replace the initial fleet. Depending on cost savings, both replacing the entire initial fleet immediately or waiting until the optimal replacement periods of the trucks can be optimal. In the examples presented in Table 3.1, the latter option is optimal, so the trucks of the initial fleet are replaced at their optimal replacement periods. The initial fleet consists of *(i)* 130 trucks, which are already past their optimal replacement age, *(ii)* 64 trucks of age 15, whose optimal replacement period is period 5, and *(iii)* 84 trucks of age 5, whose optimal replacement period is period 15. Furthermore, the optimal replacement age of DTs and AFTs are 20 and 27, respectively.

It is also important to note that even when there is no initial fleet (and the problem can be reduced to a single period model), routing consideration in the integrated model provides information that the model does not have otherwise. For example, the locations of the maintenance facilities to be opened depend on characteristics of the routes passing through each terminal. Furthermore, the number of trucks of each type to be purchased/retired depend on the truck routes, because truck routes determine *(i)* the number of trucks that can be served by the opened maintenance facilities, and *(ii)* the number of trucks that would suffice to satisfy AFT targets.

In addition to the impact of including routing decisions in the integrated model, that of including maintenance facility opening decisions can also be observed in Table 3.1. The benefits of having zero maintenance facility costs is very clear when AFTs are cheaper than DTs to own an operate, since the model opens maintenance facilities at every terminal and purchases as many AFTs as possible at optimal replacement periods. Even in the cases where AFTs are more expensive than DTs, opening many maintenance facilities (as opposed to only one) allows using AFTs on the most efficient routes in terms of target satisfaction. This way, the AFT target can be satisfied with fewer AFTs, which provides cost savings in terms of truck owning and operating costs.

3.3.3.2 Constant Demand, Increasing Target

When truck costs and demand does not change over time, the main reasons to replace existing trucks or purchase new ones become aging of the trucks and AFT targets. In this class of problem instances, we consider AFT targets enforcing an AFT loaded truck-mile proportion of 0.3 at the beginning of the planning horizon, which increases every 3 periods to reach 0.8 at the end.

Table 3.2: Constant demand, increasing target

(a) Expensive AFT, expensive maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	0.00	143.37	130.00	0.00	148.00	143.37
5	61.42	2.58	64.00	0.00	145.42	145.94
15	4.64	79.36	84.00	0.00	66.06	225.31

Maintenance facilities: 1 opened at period 1, 4 opened at 13

(b) Expensive AFT, cheap maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	0.00	143.37	130.00	0.00	148.00	143.37
5	64.00	0.00	64.00	0.00	148.00	143.37
15	13.86	70.14	84.00	0.00	77.86	213.51

(c) Cheap AFT, expensive maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	6.05	137.32	130.00	0.00	154.05	137.32
5	37.98	26.02	64.00	0.00	128.03	163.33
15	0.00	84.00	84.00	0.00	44.03	247.33
21	0.00	6.05	6.05	0.00	37.98	253.38

Maintenance facilities: 1 opened at period 1, 4 opened at 13

(d) Cheap AFT, cheap maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	0.00	143.37	130.00	0.00	148.00	143.37
5	0.00	64.00	64.00	0.00	84.00	207.37
15	0.00	84.00	84.00	0.00	0.00	291.37

With increasing targets, the optimal solution can require immediately purchasing/replacing as many trucks as necessary to satisfy the final target, purchasing/replacing as it becomes necessary by the targets, or replacing at optimal replacement periods with enough AFTs to satisfy the targets until the next replacement period, depending on the cost savings associated with each option. The best strategy among these three, along with the number of trucks of each type to be purchased/replaced is not obvious without solving the integrated fleet management problem, since these decisions are impacted by the available truck routes.

Table 3.2 presents optimal solutions to the problem instance as in Section 3.3.3.1, with increasing targets and under various truck and maintenance facility cost assumptions. It can be observed that when AFTs are more expensive, the need to purchase AFTs originates from the targets, so the optimal fleet management strategy requires replacing at optimal replacement periods (replacing the 130 trucks beyond their optimal replacement periods immediately, the 64 trucks of age 15 at period 5, and the 84 trucks of age 5 at period 15) and purchasing as many AFTs as necessary to satisfy the targets until the next replacement period. When AFTs are cheaper and maintenance facilities are cheap, replacing the entire initial fleet with AFTs at the optimal replacement periods is optimal. When AFTs are cheaper and maintenance facilities expensive, the optimal solution is to open one maintenance facility at the beginning, then another one when it is necessary for target satisfaction (at period 13); and purchase as many AFTs as allowed by the limited number of maintenance facilities at optimal replacement periods. In this particular cost setting, it is interesting to note that the second maintenance facility is opened at period 13, but no AFT purchases are made. Opening a maintenance facility allows using existing AFTs on routes that were previously unavailable due to the lack of maintenance facilities. This way, more demand-intensive routes can be used, and the proportion of loaded AFT-miles can increase without adding new AFTs to the fleet.

3.3.3.3 Constant Target, Changing Demand

When the AFT target is set at the beginning of the planning horizon and does not increase, and there is no anticipated change in truck costs, the main motivations for truck replacements are cost savings by switching truck types and changing customer demand. In this class of problem instances, we consider customer demand patterns where *(i)* origin/destination heavy regions change in time, and *(ii)* total demand is volatile (increasing/decreasing/increasing).

The problem instances in this computational study have non-uniform demand throughout the geography: some regions have more outbound demand while others have more inbound or balanced demand. The first problem instance considered in this class represents the case when inbound-heavy, outbound-heavy, and balanced regions are periodically swapped. While the total demand does not change significantly during these swaps, the localization of major origins and destinations change. This demand pattern poses a challenge particularly for locating the maintenance facilities to be opened. Since the demand intensity of routes changes with changing demand, the most advantageous terminals for opening maintenance facilities are not obvious. Due to the demand region swaps, a central terminal may be selected for opening a maintenance facility, even if it is not the most preferable terminal for any of the regions at any point in time. It may even be necessary (or profitable) to open more maintenance facilities than would have been necessary if the regions remained the same throughout the planning horizon.

Table 3.3 presents optimal fleet management strategies for various truck and maintenance facility cost scenarios when demand regions change every 8 periods ("region swap"). When AFTs are more expensive than DTs, the optimal truck replacement plans are the same regardless of the maintenance facility cost: purchasing just enough AFTs to satisfy targets at period 1, then replacing the rest of the initial fleet at optimal replacement periods, and purchasing or retiring as necessary if total demand

Table 3.3: Constant target, region swapping demand

(a) Expensive AFT, expensive maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	58.60	62.00	130.00	0.00	206.63	62.03
5	63.70	0.00	63.70	0.00	206.63	62.03
9	0.00	0.00	0.30	0.00	206.36	62.03
15	84.00	0.00	84.00	0.00	206.36	62.03
17	1.50	0.00	0.00	0.00	207.90	62.03
21	58.60	0.00	58.60	0.00	207.90	62.03

Maintenance facilities: 4 opened at period 1

(b) Expensive AFT, cheap maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	58.60	62.00	130.00	0.00	206.63	62.03
5	63.70	0.00	63.70	0.00	206.63	62.03
9	0.00	0.00	0.30	0.00	206.36	62.03
15	84.00	0.00	84.00	0.00	206.36	62.03
17	1.50	0.00	0.00	0.00	207.90	62.03
21	58.60	0.00	58.60	0.00	207.90	62.03

(c) Cheap AFT, expensive maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	57.00	63.60	130.00	0.00	205.04	63.62
5	21.50	42.30	63.70	0.00	162.78	105.88
9	0.00	0.00	0.30	0.00	162.51	105.88
15	84.00	0.00	84.00	0.00	162.51	105.88
17	0.00	1.50	0.00	0.00	162.51	107.43
21	0.00	57.00	57.00	0.00	105.47	164.46
24	0.00	7.20	7.20	0.00	98.28	171.65

Maintenance facilities: 6 opened at period 1

(d) Cheap AFT, cheap maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	0.00	120.70	130.00	0.00	148.00	120.66
5	0.00	63.70	63.70	0.00	84.27	184.39
9	0.00	0.00	0.30	0.00	84.00	184.39
15	0.00	84.00	84.00	0.00	0.00	268.39
17	0.00	1.50	0.00	0.00	0.00	269.93

increases or decreases at region swap periods. It is possible for the optimal solution to require some trucks to be kept beyond their optimal ages if they will have to be retired a few periods later due to a decrease in total demand. The retirement of DTs (of age 24) at period 9, even though the optimal replacement age of DTs is 20, is an example of this phenomenon.

When demand regions get swapped, the number of trucks operating on the routes passing through the maintenance facilities change, so a maintenance facility cannot serve the same number of AFTs throughout the entire planning horizon. So, even though AFTs are cheaper, purchasing as many AFTs as the maintenance facilities can possibly serve could create infeasibility or inefficiency. As it can be observed in Table 3.3, when AFTs are cheaper and maintenance facilities are costly, the optimal plan requires opening a maintenance facility at terminal 6. The fleet mix with maximum number of AFTs that can be served by this maintenance facility is reached at the last period. In order to reach that fleet mix at the end, the model delays some AFT purchases until later periods. This way, replacing some of the DTs purchased earlier

in the planning horizon with AFTs later in the horizon can provide cost savings. To take advantage of these cost savings, the optimal solution dictates purchasing just enough AFTs to satisfy the AFT target in the beginning, and replace the DTs purchased early in the planning horizon with AFTs whenever possible (as many as can be served by the opened maintenance facilities).

Table 3.4: Constant target, volatile demand

(a) Expensive AFT, expensive maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	134.11	107.87	130.00	0.00	282.11	107.87
5	87.07	10.25	0.00	0.00	369.18	118.12
9	0.00	0.00	194.98	0.00	174.20	118.12
13	0.00	0.00	87.12	10.34	87.07	107.78
17	195.12	0.00	0.00	0.00	282.20	107.78
21	86.98	10.34	0.00	0.00	369.18	118.12

Maintenance facilities: 6 opened at period 1

(b) Expensive AFT, cheap maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	144.44	97.54	130.00	0.00	292.44	97.54
5	79.38	17.94	0.00	0.00	371.82	115.48
9	0.00	0.00	194.98	0.00	176.84	115.48
13	0.00	0.00	97.46	0.00	79.38	115.48
17	195.12	0.00	0.00	0.00	274.50	115.48
21	97.32	0.00	0.00	0.00	371.82	115.48

(c) Cheap AFT, expensive maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	128.99	112.99	130.00	0.00	276.99	112.99
5	80.67	16.65	0.00	0.00	357.66	129.64
9	0.00	0.00	194.98	0.00	162.68	129.64
13	0.00	0.00	82.01	15.45	80.67	114.19
17	80.77	114.35	0.00	0.00	161.44	228.54
21	40.29	57.03	0.00	0.00	201.72	285.58

Maintenance facilities: 4 opened at period 1

(d) Cheap AFT, cheap maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	144.44	97.54	130.00	0.00	292.44	97.54
5	0.00	97.32	0.00	0.00	292.44	194.86
9	0.00	0.00	194.98	0.00	97.46	194.86
13	0.00	0.00	97.46	0.00	0.00	194.86
17	0.00	195.12	0.00	0.00	0.00	389.98
21	0.00	97.32	0.00	0.00	0.00	487.30

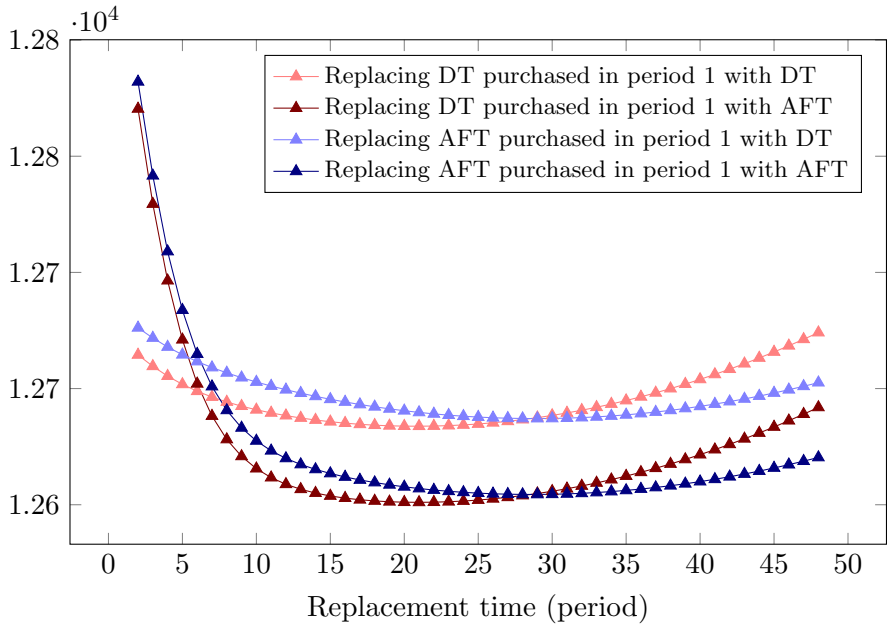
The problem instances with volatile demand are generated so that there will be overall increases/ decreases in the demand every 4 periods. (Periods 5, 17, and 21 are demand increase periods, while periods 9 and 13 are demand decrease periods.) When total demand is volatile, purchasing as many AFTs in the beginning as will be necessary (or profitable) throughout the entire horizon, as well as purchasing/retiring trucks as they become necessary or profitable can be optimal. Depending on the purchase costs and salvage values of the two truck types, the steepness of the increase in operating costs due to aging of trucks, and the timing of demand changes, either option can contribute in the feasibility or optimality of the fleet management plan. Table 3.4 provides an example problem instance with volatile demand, where

total demand changes every 4 periods and it follows an increasing, decreasing, and increasing pattern. In this example, it is not optimal to purchase more trucks than necessary and keeping them for future use. Rather, following the demand trend and purchasing when there is a demand increase, and retiring when there is a demand decrease is optimal. It is notable that in the case when AFTs are cheaper than DTs and maintenance facilities are expensive, some AFTs (purchased at period 1) are retired at period 13 due to a demand decrease, even though AFTs are cheaper and retiring them is not desirable. This is due to the fact that one maintenance facility would not be enough to serve all of the AFTs after the demand decrease. So, rather than opening another maintenance facility, which is costly, it is cheaper to retire some AFTs when demand falls, and purchasing AFTs again when demand increases.

3.3.3.4 Changing Truck Costs

All of the previous problem instances presented in this section were generated under the assumption that truck purchase and operating costs do not change over time. Here, we experiment with a class of problem instances where the operating cost of AFTs decreases over time, while that of DTs increases. These instances represent scenarios where the operating cost mostly consists of fuel cost, and unit diesel cost increases while unit natural gas cost decreases. As Figure 3.3 shows, AFTs get cheaper to own and operate after period 7, even though they were more expensive than DTs at the beginning of the planning horizon. Table 3.5 presents optimal solutions of problem instances with *(i)* constant demand, constant target (same as in Section 3.3.3.1), and *(ii)* constant demand, increasing target (same as in Section 3.3.3.2), when the truck costs change in the described manner. In all problem instances, the model favors DTs until period 7, and starts favoring AFT purchases after that.

In the instances with constant demand and target, the most important driver of truck replacements is changing truck costs and maintenance facility costs (when



(This plot shows total cost of purchasing a truck of specified type at period 1, and replacing it with the specified truck type at the specified replacement period, and then using that truck indefinitely while replacing optimally with the same truck type.)

Figure 3.3: Truck replacement costs when the truck costs change over time

they are nonzero). When the maintenance facility cost is zero, the optimal plan suggests replacing the trucks at their optimal replacement periods with just enough AFTs to satisfy the targets if the replacement period is before period 7, and with as many AFTs as possible if the replacement period is at or after period 7. The only exception is the DT replacements in period 4 with new DTs. The optimal solution replaces in period 4, even though the optimal replacement period is 5, since it is desirable to replace these DTs with AFTs before the planning horizon ends (at period 24) and the optimal replacement age for DTs is 20 periods. In this instance class, when maintenance facilities are costly, the optimal solution opens one maintenance facility in the beginning, and replaces the trucks at their optimal replacement periods throughout the planning horizon. In early periods, it purchases just enough AFTs to satisfy targets, and after period 7, it purchases the maximum number of AFTs that can be served with the opened maintenance facility.

Table 3.5: Changing truck costs, constant demand

(a) Constant target, expensive maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	80.84	62.52	130.00	0.00	228.84	62.52
5	64.00	0.00	64.00	0.00	228.84	62.52
15	0.00	84.00	84.00	0.00	144.84	146.52
21	64.03	16.81	80.84	0.00	128.03	163.33

Maintenance facilities: 1 opened at period 1

(b) Constant target, cheap maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	84.97	58.40	130.00	0.00	232.97	58.40
4	64.00	0.00	64.00	0.00	232.97	58.40
15	0.00	84.00	84.00	0.00	148.97	142.40
21	0.00	84.97	84.97	0.00	64.00	227.37
24	0.00	64.00	64.00	0.00	0.00	291.37

(c) Increasing target, expensive maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	80.84	62.52	130.00	0.00	228.84	62.52
4	0.00	27.81	27.81	0.00	201.04	90.33
7	0.00	36.19	36.19	0.00	164.84	126.52
10	0.00	19.42	19.42	0.00	145.42	145.94
15	0.00	64.58	64.58	0.00	80.84	210.52
21	37.98	42.86	80.84	0.00	37.98	253.38

Maintenance facilities: 1 opened at period 1, 4 opened at 13

(d) Increasing target, cheap maint. fac.

	Purchases		Retirements		Fleet (# Trucks)	
Period	Diesel	AFT	Diesel	AFT	Diesel	AFT
0	-	-	-	-	278.00	0.00
1	84.97	58.40	130.00	0.00	232.97	58.40
4	20.12	13.90	34.02	0.00	219.07	72.30
7	0.00	29.98	29.98	0.00	189.09	102.28
13	0.00	27.81	27.81	0.00	161.28	130.09
15	0.00	56.19	56.19	0.00	105.09	186.28
21	0.00	84.97	84.97	0.00	20.12	271.25
24	0.00	20.12	20.12	0.00	0.00	291.37

When demand is constant and the targets increase in time, it could be optimal to purchase enough AFTs in early periods to satisfy future targets. But it could also be optimal to purchase just enough AFTs to satisfy the current target, and keep replacing DTs with just enough AFTs to satisfy targets when the target increases. As can be seen in Table 3.5, the latter is optimal in these particular problem instances, even though the former policy was optimal when the trucks costs were constant in instances with the same demand and target pattern (see Table 3.2). Since targets increase every 3 periods, periods 4, 7, and 10 are target increase periods. Both when maintenance facilities are expensive and cheap, the optimal solution requires purchasing just enough AFTs to satisfy targets at periods 4 and 7. The instance with zero maintenance facility costs does not require any replacements in period 10, because even though the fleet mix contains less AFTs then the instance with high maintenance facilities, it is able to satisfy the targets without any AFT purchases due to the numerous maintenance facilities and the availability of many demand-intensive routes. In the case of expensive maintenance facilities, the optimal solution keeps

increasing the number of AFTs at optimal replacement periods or target increase periods until it reaches the maximum number of AFTs that can be served with the opened maintenance facilities at the end of the planning horizon.

3.4 Solution Approaches

3.4.1 Benders' Decomposition

The integrated fleet management problem with loaded truck-mile targets ($\mathbf{P}'$) is decomposed into Benders' master and subproblems based on the levels of considered decisions. The higher level (maintenance facility opening) decisions are considered in the master problem and the lower level (fleet purchase/retirement and fleet assignment) decisions are considered in the subproblem. With this decomposition, the master problem and the subproblem become a pure binary programming problem and a linear programming problem, respectively.

The master problem (**MP**) is presented below. It has a very simple structure, since it is only responsible for the maintenance facility decisions. In the formulation, θ represents the lower bound for the subproblem objective.There are no constraints other than Benders' cuts in **MP**. Optimality cuts (3.17) are generated by the extreme points of the dual sub problem (**DSP**), $\mathcal{P}$, and feasibility cuts are generated by the extreme rays of **DSP**, $\mathcal{Q}$.

$$\mathbf{MP}(\mathcal{P},\mathcal{Q}) \quad \min\ \theta + \sum_{j\in\mathcal{N},t} C_{jt}y_{jt} \tag{3.16}$$

$$\text{s.t. } \theta \geq \sum_{t,\tau\leq 0} \bar{\bar{p}}^D_\tau \tilde{\alpha}^D_{\tau t} + \sum_{a,t} D_{at}\tilde{\pi}_{at} - \sum_{i,t,\tau\leq t} M_{it} \sum_{j\in\mathcal{N}(i),s\leq t} \tilde{\gamma}_{i\tau t}y_{js} \ \ (\tilde{\boldsymbol{\alpha}},\tilde{\boldsymbol{\pi}},\tilde{\boldsymbol{\gamma}}) \in \mathcal{P} \tag{3.17}$$

$$0 \geq \sum_{t,\tau\leq 0} \bar{\bar{p}}^D_\tau \tilde{\alpha}^D_{\tau t} + \sum_{a,t} D_{at}\tilde{\pi}_{at} - \sum_{i,t,\tau\leq t} M_{it} \sum_{j\in\mathcal{N}(i),s\leq t} \tilde{\gamma}_{i\tau t}y_{js} \ \ (\tilde{\boldsymbol{\alpha}},\tilde{\boldsymbol{\pi}},\tilde{\boldsymbol{\gamma}}) \in \mathcal{Q} \tag{3.18}$$

$$\theta \text{ free} \tag{3.19}$$

$$y_{jt} \in \{0,1\} \qquad j \in \mathcal{N}, \;\; t = 1, \ldots, T \tag{3.20}$$

To make the master problem formulation stronger, the following four cuts are added at the beginning of the Benders' procedure.

$$\sum_{j \in \mathcal{N}} \sum_{t=1}^{\min\{t:b_t>0\}} y_{jt} \geq 1 \tag{3.21}$$

$$\sum_{j \in \mathcal{N}} \sum_{t=1}^{T} y_{jt} \leq N^{min} \tag{3.22}$$

$$y_{jt} = 0 \qquad \forall j \in \mathcal{M}, \quad \forall t = 1, \ldots, T \tag{3.23}$$

$$y_{jt} = 0 \qquad \forall j \in \mathcal{N}, \quad \forall t \notin \{1\} \cup \mathcal{T} \tag{3.24}$$

1. Constraint (3.21) ensures that there will be at least one maintenance facility by the earliest target requirement.

2. Constraint (3.22) ensures that there will be at most N^{min} maintenance facilities, where N^{min} denotes the minimum number of maintenance facilities required to serve all routes in the network. The value of N^{min} can be computed by solving a simple set covering problem.

3. Constraints (3.23) provide that there will never be maintenance facilities at dominated terminals. (A "dominated terminal" is defined as a terminal, routes passing through which are a subset of the routes passing through another terminal. The set of dominated terminals is defined as $\mathcal{M} = \{j \in \mathcal{N} : \exists j' \in \mathcal{N} \setminus \{j\}$ such that $\mathcal{R}(j') \supseteq \mathcal{R}(j)\}$.)

4. Constraints (3.24) are valid only when the assumptions of Theorem 3.2 hold. They ensure that maintenance facilities can be opened only at period 1 or "critical periods", $\mathcal{T}$ (periods that have an increase in the AFT target requirements).

The subproblem, **SP** is modeled as shown below, given the solution to the master problem, $\tilde{\boldsymbol{y}}$. It has the same structure as $\mathbf{P}'$, the only difference being the fixed maintenance facility decisions, $\tilde{y}_{jt}$. **DSP**, the dual subproblem, is the dual of **SP**.

$$\mathbf{SP}(\tilde{\boldsymbol{y}}) \quad \min \sum_{k,t} P_t^k p_t^k - \sum_{k,t,\tau\le t} R_{\tau t}^k r_{\tau t}^k + \sum_{i\in\mathcal{R},k,t,\tau\le t} c_{\tau t}^k x_{i\tau t}^k \tag{3.25}$$

$$\text{s.t. } (3.2), (3.3)$$

$$(3.12), (3.13), (3.14)$$

$$x_{i\tau t}^A \le M_{it} \sum_{j\in\mathcal{N}(i),s\le t} \tilde{y}_{js} \qquad i\in\mathcal{R},\ \ t=1,\dots,T,\ \ \tau\le t \tag{3.26}$$

$$(3.8), (3.9), (3.10), (3.15)$$

$$\mathbf{DSP}(\tilde{\boldsymbol{y}}) \quad \max \sum_{t,\tau\le 0} \bar{\bar{p}}_\tau^D \alpha_{\tau t}^D + \sum_{a,t} D_{at}\pi_{at} - \sum_{i,t,\tau\le t} M_{it} \sum_{j\in\mathcal{N}(i),s\le t} \tilde{y}_{js}\gamma_{i\tau t} \tag{3.27}$$

$$\text{s.t. } \alpha_{\tau t}^A + \sum_{a\in\mathcal{A}(i)} n_i \beta_{at}^A - \gamma_{i\tau t} \le c_{\tau t}^A \qquad i\in\mathcal{R},\ \ t=1,\dots,T,\ \ \tau\le t \tag{3.28}$$

$$\alpha_{\tau t}^D + \sum_{a\in\mathcal{A}(i)} n_i \beta_{at}^D \le c_{\tau t}^D \qquad i\in\mathcal{R},\ \ t=1,\dots,T,\ \ \tau\le t \tag{3.29}$$

$$\sum_{s\ge t} -\alpha_{ts}^k \le P_t^k \qquad k\in\{\text{A, D}\},\ \ t=1,\dots,T \tag{3.30}$$

$$\sum_{s\ge t} \alpha_{\tau s}^k \le -R_{\tau t}^k \qquad k\in\{\text{A, D}\},\ \ t=1,\dots,T,\ \ \tau\le t \tag{3.31}$$

$$-\beta_{at}^A + \pi_{at} + (1-b_t)\tau_a\nu_t \le 0 \qquad a\in\mathcal{A},\ \ t=1,\dots T \tag{3.32}$$

$$-\beta_{at}^D + \pi_{at} + b_t\tau_a\nu_t \le 0 \qquad a\in\mathcal{A},\ \ t=1,\dots T \tag{3.33}$$

$$\alpha^A_{\tau t}, \alpha^D_{\tau t} \text{ free} \qquad t = 1, \ldots, T, \;\; \tau \leq t \tag{3.34}$$

$$\beta^A_{at}, \beta^D_{at} \geq 0 \qquad a \in \mathcal{A}, \;\; t = 1, \ldots, T \tag{3.35}$$

$$\gamma_{i\tau t} \geq 0 \qquad i \in \mathcal{R}, \;\; t = 1, \ldots, T, \;\; \tau \leq t \tag{3.36}$$

$$\nu_s \geq 0 \qquad s \in \mathcal{T} \tag{3.37}$$

$$\pi_{at} \text{ free} \qquad a \in \mathcal{A}, \;\; t = 1, \ldots, T \tag{3.38}$$

The reader should note that an infeasibility in $\mathbf{SP}(\tilde{\boldsymbol{y}})$ can only stem from not having enough AFT maintenance facilities when necessary. With this observation in mind, a constraint enforcing at least one maintenance facility to be opened earlier than the master solution $\tilde{\boldsymbol{y}}$ suggests, as shown in Equation (3.39), is added to the master problem whenever a Benders' feasibility cut is being added. (In Equation (3.39), $\tilde{t}_j$ denotes the opening time of a maintenance facility in terminal j suggested by master solution $\tilde{\boldsymbol{y}}$. It is $T+1$ if $\tilde{\boldsymbol{y}}$ does not open a maintenance facility in terminal j during the planning horizon.)

$$\sum_{j \in \mathcal{N}} \sum_{t < \tilde{t}_j} y_{jt} \geq 1 \tag{3.39}$$

$\mathbf{MP}(\mathcal{P}, \mathcal{Q})$ and $\mathbf{DSP}(\tilde{\boldsymbol{y}})$ formulations are used in a Benders' decomposition framework as shown in Algorithm 3.1. First, the master problem **MP** is generated using sets $\mathcal{P}$ and $\mathcal{Q}$ for optimality and feasibility cuts, respectively. Using the solution of the master problem, $\tilde{\boldsymbol{y}}$, the dual subproblem **DSP** is generated. If it can be solved optimally, the optimal solution is added to set $\mathcal{P}$, and the upper bound is updated if necessary. If the dual subproblem is unbounded, an unbounded ray is found and added to set $\mathcal{Q}$. The process continues until the value of θ converges to the upper bound.

Algorithm 3.1 Benders' decomposition for $\mathbf{P}'$

$\mathcal{P} \leftarrow \emptyset$, $\mathcal{Q} \leftarrow \emptyset$, $UB \leftarrow \infty$, $done \leftarrow False$
while $done = False$ **do**
  Solve $\mathbf{MP}(\mathcal{P}, \mathcal{Q})$, obtain $\tilde{\boldsymbol{y}}$ and $\tilde{\theta}$
  if $\tilde{\theta} = UB$ **then**
    $done = True$
  else
    Solve $\mathbf{DSP}(\tilde{\boldsymbol{y}})$
    if Optimal solution $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*, \boldsymbol{\gamma}^*, \boldsymbol{\nu}^*, \boldsymbol{\pi}^*)$ to $\mathbf{DSP}(\tilde{\boldsymbol{y}})$ exists **then**
      if $f(\tilde{\boldsymbol{y}}) < UB$ **then** $[f(\tilde{\boldsymbol{y}})$:= *Optimal objective of* $\mathbf{DSP}(\tilde{\boldsymbol{y}})]$
        $UB \leftarrow f(\tilde{\boldsymbol{y}})$
      end if
      $\mathcal{P} \leftarrow \mathcal{P} \cup (\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*, \boldsymbol{\gamma}^*, \boldsymbol{\nu}^*, \boldsymbol{\pi}^*)$
    else if $\mathbf{DSP}(\tilde{\boldsymbol{y}})$ unbounded **then**
      Find unbounded ray $(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\nu}, \boldsymbol{\pi})$
      $\mathcal{Q} \leftarrow \mathcal{Q} \cup (\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\nu}, \boldsymbol{\pi})$
      Add constraint $\sum_{j \in \mathcal{N}} \sum_{t \leq \tilde{t}_j} y_{jt} \geq 1$ to $\mathbf{MP}(\mathcal{P}, \mathcal{Q})$
    end if
  end if
end while

3.4.1.1 Pareto-optimal Cuts

To accelerate the basic Benders' decomposition approach described in the previous section, we enhance the optimality cuts using the notion of Pareto-optimal cuts, introduced by [87]. A Pareto-optimal cut is defined as a cut that is not dominated by any other cut in terms of strength. Let $\tilde{\boldsymbol{y}}$ be a solution to the master problem **MP**, and $f(\tilde{\boldsymbol{y}})$ be the subproblem objective value associated with $\tilde{\boldsymbol{y}}$. Furthermore, let $\boldsymbol{y^0}$ be a point in the relative interior of the convex hull (a "core point") of the feasible region associated with $\boldsymbol{y}$ variables. (Note that in **MP**, the feasible region of $\boldsymbol{y}$ variables is $\{0,1\}^{|\mathcal{N}| \times T}$.) Then, a Pareto-optimal cut for the Benders' master problem can be found by first solving $\mathbf{DSP}(\tilde{\boldsymbol{y}})$ to obtain the objective value $f(\tilde{\boldsymbol{y}})$, and then solving the $\mathbf{PODSP}(\tilde{\boldsymbol{y}}, \boldsymbol{y^0})$ for finding a Pareto-optimal cut.

Let the solution found by $\mathbf{DSP}(\tilde{\boldsymbol{y}})$ be $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*, \boldsymbol{\gamma}^*, \boldsymbol{\nu}^*, \boldsymbol{\pi}^*)$, with the subproblem objective value $f(\tilde{\boldsymbol{y}})$. $\mathbf{PODSP}(\tilde{\boldsymbol{y}}, \boldsymbol{y^0})$ finds a dual solution $(\boldsymbol{\alpha^0}, \boldsymbol{\beta^0}, \boldsymbol{\gamma^0}, \boldsymbol{\nu^0}, \boldsymbol{\pi^0})$ that

would result in the same subproblem objective value, while establishing a cut that is at least as strong as $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*, \boldsymbol{\gamma}^*, \boldsymbol{\nu}^*, \boldsymbol{\pi}^*)$. The new solution having the same objective value $f(\tilde{\boldsymbol{y}})$ is ensured by (3.41), while the strength of the new cut is attained by (3.40).

$$\textbf{PODSP}(\tilde{\boldsymbol{y}}, \boldsymbol{y^0}) \quad \max \sum_{t,\tau \leq 0} \bar{\bar{p}}^D_\tau \alpha^D_{\tau t} + \sum_{a,t} D_{at}\pi_{at} - \sum_{i,t,\tau \leq t} M_{it} \sum_{j \in \mathcal{N}(i), s \leq t} y^0_{js}\gamma_{i\tau t} \tag{3.40}$$

$$\text{s.t.} \sum_{t,\tau \leq 0} \bar{\bar{p}}^D_\tau \alpha^D_{\tau t} + \sum_{a,t} D_{at}\pi_{at} - \sum_{i,t,\tau \leq t} M_{it} \sum_{j \in \mathcal{N}(i), s \leq t} \tilde{y}_{js}\gamma_{i\tau t} = f(\tilde{\boldsymbol{y}}) \tag{3.41}$$

$$(3.28), (3.29), (3.30), (3.31), (3.32), (3.33)$$

$$(3.34), (3.35), (3.36), (3.37), (3.38)$$

When implementing Benders' decomposition with Pareto-optimal cuts to solve $\mathbf{P}'$, we consider different core points to investigate the effect of core point selection. The following core points are considered in our computational experiments:

1. A core point where all y_{jt} values are $\frac{1}{2}$

$$\boldsymbol{y^0_1} = \{\boldsymbol{y} \in \mathbb{R}^{|\mathcal{N}| \times T} : y_{jt} = \frac{1}{2} \ \ \forall j \in \mathcal{N}, t = 1, \ldots T\}$$

2. A core point where all y_{jt} values are 0.01, except for the variable associated with opening a maintenance facility at the most demand intensive terminal, j^*, at period 1

$$\boldsymbol{y^0_2} = \{\boldsymbol{y} \in \mathbb{R}^{|\mathcal{N}| \times T} : y_{j^*1} = 0.99, y_{j^*t} = 0.01 \ \ \forall t = 2, \ldots T,$$
$$y_{jt} = 0.01 \ \ \forall j \in \mathcal{N} \setminus \{j^*\}, t = 1, \ldots T\}$$

3. A core point where all y_{jt} values are 0.01, except for the two variables associated with opening a maintenance facility at the most and the second most demand

intensive terminals, j^* and j^{**}, at period 1

$$\boldsymbol{y_3^0} = \{\boldsymbol{y} \in \mathbb{R}^{|\mathcal{N}|\times T} : y_{j^*1} = 0.99, y_{j^{**}1} = 0.99, y_{jt} = 0.01 \;\; j \in \{j^*, j^{**}\} \; \forall t = 2, \ldots T,$$
$$y_{jt} = 0.01 \;\; \forall j \in \mathcal{N} \setminus \{j^*, j^{**}\}, t = 1, \ldots T\}$$

4. A core point $\boldsymbol{y_4^{0,k}}$ that starts as $\boldsymbol{y_2^0}$ and is updated at every iteration k using the current solution to the master problem, $\tilde{\boldsymbol{y}}^k$ with the following update mechanism (inspired by the core point update mechanism of [95]):

$$\boldsymbol{y_4^{0,k}} = \frac{1}{2}\boldsymbol{y_4^{0,k-1}} + \frac{1}{2}\tilde{\boldsymbol{y}}^k$$

3.4.2 Variable Neighborhood Search

A variable neighborhood search (VNS) heuristic is proposed for finding good solutions to the integrated fleet management model ($\mathbf{P}'$) quickly. For any given solution, neighborhoods are determined based on the maintenance facilities that are opened. Let M denote the set of (terminal, time period) pairs at which maintenance facilities are opened. For example, $M = \{(j_1, t_1), (j_2, t_2), \ldots\}$ specifies the case where maintenance facilities are opened at terminal j_1 at the beginning of period t_1, at terminal j_2 at the beginning of t_2, etc. The neighborhood of a given M is generated with the following five operations:

1. *Add*(j): $M = M \cup \{(j, 1)\}$
2. *Remove*(S): $M = M \setminus S$
3. *Swap*(S): $M = M \setminus S \cup S'$, where $S' = \{(f(j), t) : (j, t) \in S\}$ and $f(j)$ denotes the terminal with the largest number of routes in common with j that is not in M
4. *Delay*$((j, t), \tau)$: $M = M \setminus \{(j, t)\} \cup \{(j, t + \tau)\}$
5. *Advance*$((j, t), \tau)$: $M = M \setminus \{(j, t)\} \cup \{(j, t - \tau)\}$

Let k denote the maximum number of elements in M that can be changed when generating a neighborhood. A complete k-neighborhood of a solution with a maintenance facility set M, denoted as $\mathcal{N}(M,k)$, consists of the following maintenance facility sets:

1. $Add(j)$ for the k most demand intensive terminals, j
2. $Remove(S)$ for every $S \subseteq M$ with $|S| = k$
3. $Swap(S)$ for every $S \subseteq M$ with $|S| = k$
4. $Delay((j,t),\tau)$ for every $(j,t) \in M$ and $1 \leq \tau \leq k$
5. $Advance((j,t),\tau)$ for every $(j,t) \in M$ and $1 \leq \tau \leq k$

VNS uses an adaptive k, which is progressively increased when the algorithm cannot find an improving solution in the k-neighborhood, $\mathcal{N}(M,k)$. Algorithm 3.2 summarizes VNS for the integrated fleet management model, $\mathbf{P}'$. Here, $z(M)$ represents the optimal objective value when the maintenance facility set is M. It can be found by solving the model $\mathbf{P}'$ with the maintenance facility variables (y_{jt}) fixed according to M, denoted as $\mathbf{P}'(M)$.

3.5 Computational Experiments

In this section, we present a computational study for demonstrating the performance of the Benders' decomposition and the variable neighborhood search (VNS) frameworks presented in Section 3.4. Numerous problem instances with varying characteristics are generated and the solution approaches are examined in terms of solution time and quality in comparison with a direct solution approach using a commercial solver.

Computational experiments are run on a heterogeneous cluster of machines with Xeon E5520, Xeon E5-2670, E5-2603, E5-2650v3 and Xeon E5430 processors using Gurobi 5.6.3 Python interface (with Python 2.7).

Algorithm 3.2 Variable neighborhood search for $\mathbf{P}'$

$M \leftarrow \emptyset$
while $\mathbf{P}'(M)$ not feasible **do** *[Initial Solution]*
    $M \leftarrow M \cup \{(j^*, 1)\}$, where j^* is the most demand-intensive terminal that has not yet been included in M
end while
while $k \leq k^{max}$ **do**
    Randomly choose M^s from the k^{th} neighborhood of M, $M^s \in \mathcal{N}(M, k)$ *[Shaking]*
    $\mathcal{N}' \leftarrow \mathcal{N}(M^s, k)$
    while $\mathcal{N}' \neq \emptyset$ **do** *[Local Search]*
        Pick $M' \in \mathcal{N}'$
        if $z(M') < z(M)$ **then**
            $M \leftarrow M'$
        end if
        $\mathcal{N}' \leftarrow \mathcal{N}' \setminus \{M'\}$
    end while
    $k \leftarrow k + 1$
end while

Problem instances are generated in a similar manner to those described in Section 3.3.3. Instances with 10 terminals and 24 periods are solved to optimality and used for demonstrating aspects of the solution methods that require optimal solutions; and instances with 30 terminals and 24 periods are solved by enforcing time limits, as solving those to optimality in reasonable time is not always possible by either direct solution or Benders' decomposition.

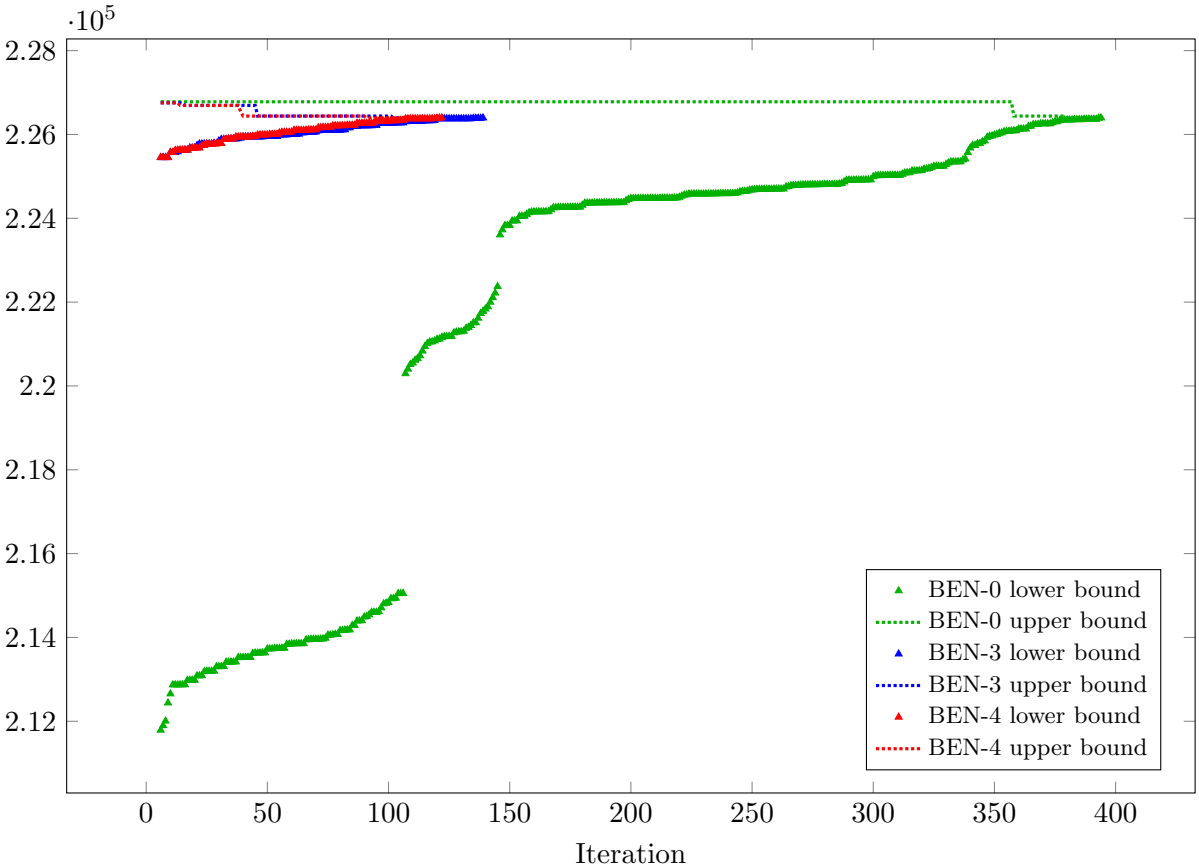
A wide range of cost, demand, and target patterns are considered in our computational experiments. The cases of *(i)* constant and increasing targets; *(ii)* constant, region swapping, increasing, and volatile demand; *(iii)* constant (both when DTs are cheaper and AFTs are cheaper) and changing truck costs; and *(iv)* high and low maintenance facility costs are represented in the computational experiments. Table B.1 shows all of the instance classes used in the experiments, which are generated based on these changing parameters.

Each problem is directly solved with Gurobi, Benders' decomposition with and without Pareto-optimal cuts, and VNS; and solutions, along with solution times, are

reported.

3.5.1 Benders' Decomposition

In investigating the solution performance of the Benders' decomposition framework described in Section 3.4.1, Benders' decomposition algorithm with *(i)* no Pareto-optimal cuts, and Pareto-optimal cuts with *(ii)* a core point $\boldsymbol{y_1^0}$, *(iii)* a core point $\boldsymbol{y_2^0}$, *(iv)* a core point $\boldsymbol{y_3^0}$, and *(v)* a core point updated at every iteration, $\boldsymbol{y_4^{0,k}}$, are run on all problem instances. The solution times are then compared to the time it takes to directly solve the problems. Additionally, the Benders' variants are studied comparatively in order to draw conclusions about the performance of Pareto-optimal cuts.



(Test instance: 10 terminals, cheap maintenance facilities, cheap AFTs, constant targets, nonzero initial fleet, increasing demand)

Figure 3.4: Progression of Benders' iterations in BEN-0, BEN-3, and BEN-6

Small problem instances with 10 terminals and 24 periods are solved to optimality using both a direct solution approach and Benders' decomposition. The results of this computational experiment can be observed in Table B.2, where the solution times (in seconds) of each problem instance is presented, along with the average solution time for every solution method. In these instances, Benders' decomposition does not, on average, perform better than the direct solution approach in terms of solution time. However, it should be noted that in more than half of the instances, there is at least one Benders' decomposition variant that solves the problem in a shorter time than the direct solution approach. Basic Benders' decomposition with no Pareto-optimal cuts seem to perform the best in terms of solution time, as the other variants (with Pareto-optimal cuts) tend to spend a significant amount of time for finding Pareto-optimal cuts. Even though Pareto-optimal cuts do not improve solution time in the majority of our problem instances, it is noteworthy that they significantly reduce the number of Benders' iterations. It can be observed in Table B.3 that on average, Benders' variants with Pareto-optimal cuts terminate in fewer iterations than basic Benders' decomposition. To demonstrate how Pareto-optimal cuts lead to fewer iterations by closing the optimality gap quicker, Figure 3.4 illustrates the progression of the upper and lower bounds in a single problem instance with *(i)* basic Benders' decomposition, *(ii)* Benders' decomposition with core point $\boldsymbol{y_3^0}$, and *(iii)* Benders' decomposition with continuously updated core points, $\boldsymbol{y_4^{0,k}}$.

As it was not practical to try to solve the larger problem instances with 30 terminals and 24 periods to optimality with reasonable computing power, all of the solution methods were run up to a certain time limit in these larger instances. A 1-hour time limit is enforced in this set of computational experiments, and the same time limit is enforced on both Benders' decomposition and direct solution using Gurobi. Table B.4 presents optimality gaps associated with each solution method after one hour of

solution time. With the exception of three instances, both direct solution and Benders' decomposition were able to find a feasible solution. Despite the fact that one approach does not uniformly perform better than the other in terms of solution quality after an hour, the majority of instances have at least one Benders' decomposition variant, which finds a solution that is at least as good as that found by Gurobi. Additionally, on average, about three of the five Benders' decomposition variants finds solutions with smaller optimality gaps than those of Gurobi after one hour of solution time. Even though the computational experiments on the small instances revealed that Benders' variants with Pareto-optimal cuts take a longer time, on average, to find the optimal solution than basic Benders' decomposition; the time spent on finding Pareto-optimal cuts, especially in the early iterations, pays off in terms of improved early optimality gaps.

Finally, the effects of critical period cuts, (3.24), on the convergence behavior of Benders' decomposition are investigated. When solving problem instances with constant demand, targets, and truck / maintenance facility costs, critical period cuts, (3.24), can be added to the master problem at the beginning of the solution procedure, thanks to Theorem 3.2. These cuts aim at reducing the number of Benders' iterations and the total solution time. Table 3.6 presents solution time and number of iterations with and without critical period cuts on problem instances where the assumptions of Theorem 3.2 hold. Clearly, introducing these cuts improves the convergence of Benders' decomposition for the integrated fleet management model in practice.

3.5.2 Variable Neighborhood Search

The variable neighborhood search (VNS) algorithm described in Section 3.4.2 is implemented to solve the 10-terminal and 30-terminal problem instances used in Section 3.5.1. A k^{max} value of 3 is used in all experiments. The resulting optimality gaps and solution times are presented in Tables B.5 and B.6 for 10-terminal and 30-terminal

Table 3.6: The effect of critical period cuts on Benders' convergence

	Expensive maintenance facilities			Cheap maintenance facilities		
	Expensive AFTs	Cheap AFTs	Changing truck costs	Expensive AFTs	Cheap AFTs	Changing truck costs
Solution time (sec)						
Critical period cuts	102.41	135.02	155.09	197.47	920.26	9581.16
No critical period cuts	108.19	157.97	382.36	750.55	2331.80	14217.80
Number of iterations						
Critical period cuts	14.10	13.10	21.40	20.40	84.20	231.78
No critical period cuts	14.90	15.80	55.80	83.00	221.50	474.29

Note: The figures in this table are averages of 10 problem instances with 10 terminals, constant AFT targets, and given cost characteristics.

instances, respectively. The optimality gaps presented for 10-terminal instances are the true optimality gaps, since we were able to solve these instances to optimality and get the optimal objective values. However, since we were not able to obtain the optimal objective values for the 30-terminal instances, the optimality gaps presented in Table B.6 are calculated based on the best lower bound found either by Gurobi or Benders' decomposition variants.

As Table B.5 illustrates, VNS finds solutions that are optimal or very close to optimal in reasonable solution times in 10-terminal instances. Despite the small size of these instances and therefore the fact that optimal solutions can be found relatively quickly, VNS is able to save about 18% solution time on average while being about 0.2% away from the optimal solution. Similarly in 30-terminal instances, VNS finds solutions with about 0.3% optimality gap, on average, in about 30 minutes, on average. Furthermore, when the optimality gaps of VNS and Gurobi after an hour are compared, as presented in Table B.6, it can be observed that VNS finds better solutions in more than half of the instances considered in this experimental setting.

3.6 Conclusion

In this chapter, a novel modeling approach for introducing AFTs into an existing diesel fleet is introduced. Two variants of the proposed integrated fleet management model, which differ in how they enforce AFT targets, are presented. It is observed

that the model with target AFT truck-mile proportions can require purchasing more AFTs than necessary to satisfy demand and running them empty just to satisfy target proportions; and some characteristics of the problem instances guaranteeing that this type of policy will never be optimal are determined. In order to prevent purchasing extra AFTs and running them empty in the general case, we introduce a model with target loaded truck-mile proportions for AFTs.

Optimal fleet management strategies found by the integrated model are thoroughly investigated, and it is demonstrated that integrating maintenance facility opening and truck deployment decisions into a fleet replacement framework results in nontrivial solutions that depend on the route configuration of the network, which is an aspect of the transportation system often overlooked by basic fleet replacement strategies.

In order to efficiently solve the integrated fleet management model, a Benders' decomposition framework and a VNS algorithm are proposed. These methodologies are tested against a direct solution approach. While none of the Benders' variants perform better than direct solution approach on average in small instances that can be solved to optimality; there is at least one Benders' variant that finds the optimal solution quicker than the direct solution approach in majority of the problem instances. Furthermore, in larger instances, Benders' variants find better solutions (with lower optimality gaps) than Gurobi within a 1 hour time limit. Finally, VNS is tested against the direct solution approach and it is demonstrated to find very good solutions (0.2 - 0.3 % optimality gap) in a fraction of time required solve the model to optimality, in addition to quickly achieving smaller optimality gaps than those can be achieved with Gurobi within a 1 hour time limit in numerous problem instances.

CHAPTER IV

SCENARIO SET PARTITION DUAL BOUNDS FOR MULTISTAGE STOCHASTIC PROGRAMMING

Stochastic programming provides an approach to optimized decision-making that models uncertain parameters with probability distributions. The values of these parameters are typically revealed over time, in a multistage setting, and decisions made at each stage hedge against possible realizations of parameters revealed in future stages. Such multistage stochastic programming models are of enormous practical value, and have been studied for many years now, with the literature offering a wealth of theoretical and algorithmic tools for solving them: see, for example, Birge and Louveaux [20], Powell [98, 99], Ruszczynski and Shapiro [105], Sahinidis [107], Schultz [111], Sen and Zhou [115], Shapiro [117], Shapiro et al. [118] and the many references therein.

Many important practical applications are effectively addressed with multistage stochastic programming models that include integer variables, leading to their formulation as multistage stochastic mixed integer programs (SMIPs). Solving multistage SMIPs is especially challenging, and decomposition approaches are typically required as the size of the scenario tree grows. Here we address the case that the number of scenarios is small enough to enumerate explicitly, while still too large to permit direct solution of the deterministic equivalent formulation. This case arises often in applications reported in the literature, and has been the focus of much research effort. Computing dual bounds for measuring the quality of primal solutions remains a key challenge in this case.

A recent stream of research work has investigated the computation of dual bounds

for SMIPs derived from the solution of *scenario group subproblems*, which each include variables and constraints for only a subset of scenarios. Sandıkçı et al. [108], in a two-stage setting, explore dual bounds provided by the expected value over all groups of a fixed cardinality, q, of the group subproblem (including a reference scenario). For a fixed q, they name this the $EGSO(q)$ bound. They prove that the $EGSO(q)$ bound is non-decreasing in q, and give computational results showing that the bound is strong relative to that given by linear relaxations, even for small group sizes, and that as q increases, the bound quite rapidly approaches the optimal value. However, computing the $EGSO(q)$ bound can be challenging, as it requires solution of a SMIP with q scenarios, for every possible cardinality q subset of the set of all scenarios. This work has since been extended in several directions. Maggioni et al. [85] generalizes the $EGSO$ bound idea to multistage stochastic programming while also allowing multiple reference scenarios. We discuss the extensions described in Maggioni et al. [86], Sandıkçı and Özaltin [109] and the working paper of Zenarosa et al. [134], which are closest to our work, in Section 4.1.

In this paper, we consider dual bounds that we refer to as *partition bounds*. A single partition bound is obtained by combining the group subproblem values for each group that is a subset in a single fixed partition of the scenario set. Our first contribution is to prove that for partitions into subsets of nearly equal size, the expected value over all such partitions yields a hierarchy of bounds. By "nearly equal", we mean that all groups have size either q, or $q-1$; we call such a partition a q-*partition*. We denote the expected value of the partition bound over all q-partitions by $EP(q)$. Observing that the $EGSO$ bound of [108] readily extends from the two-stage to the multi-stage setting, we prove that the $EP(q)$ bound is equal to the $EGSO(q)$ bound in the case that q divides the number of scenarios, L, and interpolates between the $EGSO(q-1)$ and $EGSO(q)$ values otherwise. We thus obtain the result that the $EP(q)$ bound increases monotonically in q.

While this hierarchical property of the expected q-partition bounds is primarily of theoretical interest, q-partitions can provide bounds in practice. By solving $\lceil L/q \rceil$ SMIPs, each with q or $q-1$ scenarios, a single partition bound results. In empirical studies, we found that the distribution of such partition bounds for typical benchmark instances is not very far from symmetric, so the probability that a partition has above-average value is not small. Similar observations can be made from distributions provided in Sandıkçı and Özaltin [109]. Thus, calculating a partition bound for only a few, randomly sampled partitions, and taking the best of these bounds, is highly likely to result in a dual bound greater than the corresponding EP bound, and hence, greater than the $EGSO$ bound. We are thus motivated to seek partition bounds by random sampling of partitions.

Random sampling has been a valuable tool for tackling stochastic programs. In particular, the Sample Average Approximation (SAA) method (see, for example, Kleywegt et al. [78], Ruszczynski and Shapiro [105]) has proven of great utility. In SAA, a set of scenarios is sampled, the sampled problem is solved, and the resulting solution evaluated using a, usually, larger, sample of the scenarios. This process is repeated a number of times, allowing statistical bounds, including confidence intervals, to be computed [88, 92]. In SAA, the expected value of the group subproblem is known to provide a dual bound [78]. Indeed, modulo replacement[1], the average SAA group subproblem value can be interpreted as an estimate of the $EGSO$ bound for groups of the same size.

Here, we propose to randomly *sample partitions* of the set of scenarios, with replacement, and to compute the best partition bound over the partitions sampled. The practical value of this idea is illustrated on benchmark instances, showing that

[1]SAA samples scenarios with replacement, allowing the same scenario to be sampled more than once, while the $EGSO$ bound assumes that all scenarios in a group are distinct.

randomly sampled q-partition bounds are better than those determined to be statistically likely using SAA with the same computational effort. The sampling approach is enhanced by leveraging the scenario tree structure and by constructing optimally recombined partitions from scenario subsets that are previously used in the algorithm. Empirical tests show that both these ideas can significantly improve the quality of the final bound. On benchmark problems, sampling as few as 30 partitions closes 94% of the wait-and-see gap, on average, for moderate group sizes (size 10); which is 79% more than that can be closed with the SAA estimates. By using the observation that the bound from a given partition can be heuristically estimated part-way through its computation, we suggest a method to improve the efficiency of the approach. Strategies that compare the heuristically estimated partition bound with the best such bound found so far, in order to terminate the partition bound computation part-way through, are tested empirically. In many cases, such strategies substantially reduce the computational effort with very little impact on the quality of the final bound.

The remainder of this paper is organized as follows. In Section 4.1, we provide an overview of related literature. In Section 4.2, we introduce our notation, and review the $EGSO$ bounds and their extension to the multistage case. In Section 4.3, we introduce the EP bounds, and prove that they yield a hierarchy of bounds. In Section 4.4, we present the random partition sampling approach, and enhancements to it. In Section 4.5, we provide the results of computational experiments on two- and three-stage benchmark problems. We give conclusions and discuss promising extensions to this work in Section 4.6.

4.1 Literature Review

Challenges in solving multistage stochastic programs with integer variables are well documented; see, for example, Ahmed [2], Klein Haneveld and van der Vlerk [76],

Schultz [111], Sen and Zhou [115] (A comprehensive list of relevant scholarly work published between 1996 and 2007 is provided by van der Vlerk [127].) These challenges have spurred many recent algorithmic developments, with substantial effort focused on decomposition approaches.

One line of work employs stage-wise decomposition and convexification of the value function at each scenario tree node; see, for example, the work of Klein Haneveld et al. [75], Sen and Higle [113], Sen and Sherali [114], van der Vlerk [128], and for a computational comparison of some alternative approaches, see Ntaimo and Sen [94].

Another line of research, based on scenario-wise decomposition [26], has been vigorously pursued in recent years, not least for its strong potential for parallel implementation. For example, Ahmed [1] provides a scenario decomposition approach for 0-1 two-stage problems. The computational efficacy of a synchronous parallel implementation of this approach is demonstrated in Ahmed [1], with algorithmic improvements, and an asynchronous parallel implementation is provided in Ryan et al. [106]. For two-stage stochastic mixed integer programming, Kim and Zavala [74] develop software based on parallel implementation of a scenario decomposition method that uses Lagrangian relaxation to improve the dual bounds. Also solving a (stabilized) Lagrangian dual master problem, but exploiting its special structure to do so in parallel, is the method of Lubin et al. [84]. A related approach is progressive hedging [104], that has been used as an effective primal heuristic for SMIPs [28, 33, 82, 129, 131], including parallel implementations. Its connections with dual decomposition have been exploited recently to derive hybrid approaches [61].

For an interesting study that compares stage-wise and scenario-wise decomposition for a class of problems with special structure, see Beier et al. [10].

Scenario-wise decomposition can be generalized to decomposition into sets, or groups, of scenarios, with the subproblem for each group retaining all non-anticipativity

constraints between scenarios in the set, but the non-anticipativity constraints between scenarios in different groups relaxed. This idea was exploited by Aldasoro et al. [5], Escudero et al. [48, 49] in the context of branch-and-fix coordination algorithms, and by Escudero et al. [47, 50] with non-anticipativity constraints between groups (called "clusters" in this work) relaxed in a Lagrangian fashion. The groups form a partition of the set of all scenarios, the *scenario set*, that is induced by the subtrees corresponding to the nodes in one stage of the scenario tree. A hierarchy of bounds is observed by Escudero et al. [50]: for any Lagrangian multiplier vector, (and hence for the Lagrangian dual value), the Lagrangian relaxation dual bound is non-increasing in the stage of the scenario tree used to induce the partition (the earlier the stage, the better the bound).

The work of Sandıkçı et al. [108] developing $EGSO(q)$ bounds describes approaches for computing dual bounds for two-stage SMIPs via the solution of many scenario group subproblems. Papers of Sandıkçı and Özaltin [109] and Maggioni et al. [86], and the working paper of Zenarosa et al. [134] describe extensions of these ideas. Sandıkçı and Özaltin [109] study bounds from collections of group subproblems (without reference scenario) for groups that cover the scenario set. Such a collection is called a "blockset". They prove that an appropriately weighted sum of the group subproblem values over all groups in a blockset gives a lower bound. They also show that if all groups in a blockset contain no more than b scenarios, each scenario appears in exactly m groups, and the blockset that gives the best possible lower bound with these requirements is used, then the bound from the $m = 1$ case is better than the others. This suggests that restricting attention to blocksets that form a partition of the set of all scenarios, rather than a cover, is beneficial. When the set of all scenarios is of size L, Sandıkçı and Özaltin [109] consider partitions in which all groups have cardinality q, in the case that q divides L, and in which all groups but one have cardinality q, and one group has cardinality $L \mod q$, otherwise. They provide

computational results showing the distribution of the resulting dual bound over all partitions of a set of 16 scenarios, for each of $q = 1,\ 2,\ 4,\ 8$, and 16, showing how the dual bound improves with group problem size and computation time increases. The distribution of primal bounds, derived from solutions to the group subproblems, is also given. The suggestion to stop MIP solution of each group subproblem prior to proven optimality is explored, and shown to have the potential to greatly decrease run times with relatively less impact on the quality of the bounds. Finally, a parallel implementation for a given partition is shown to be able to compute primal solutions and gaps for instances with an enormous number of scenarios in reasonable wall clock time.

Zenarosa et al. [134] generalize the Sandıkçı et al. [108] *EGSO* bound to include multiple reference scenarios in each group subproblem, and the resulting dual bound is shown to be monotonically increasing in the subset size. Like Sandıkçı and Özaltin [109], Zenarosa et al. [134] consider collections of group subproblems, however their collections are constructed from scenario-node cuts in the scenario tree. A given scenario-node cut in the scenario tree consists of a set of scenario tree nodes that induce a partition of the scenarios, with a subset of the partition for each node in the cut. For each node in the cut, a group subproblem is constructed, with a group of scenarios and a set of reference scenarios, both of which are subsets of the scenarios corresponding to that node. The group problem values are combined to compute what is called the value of the "cut-group subproblem". For a fixed cut, and a fixed set of reference scenarios for each node in the cut, taking the expected value of the cut-group subproblem over all possible group subproblems yields a dual bound. This bound is shown to increase monotonically in the group size. By using solutions of cut-group subproblems, Zenarosa et al. [134] also derive primal bounds, and prove monotonicity properties of their expected values. Computational results, including with a parallel implementation, show the utility of these ideas.

Maggioni et al. [86] show how to generate dual bounds based on a set of reference scenarios, and taking the expected value of the group subproblems over all groups of a fixed cardinality. They show that the resulting bound increases monotonically in both the cardinality of the groups, and as the set of reference scenarios expands. Maggioni et al. [86] also suggest ideas for upper bounds, and provide inequalities on their expected values, in a similar vein to some of those given in Zenarosa et al. [134]. Maggioni et al. [86] provide numerical tests based on a real logistics SMIP application.

Finally, we mention that there is another stream of research focusing on solving stochastic programs by using partitions of the scenario set [19, 51, 123]. Though very effective in practice, these methods are quite different to the approach we propose in this paper. For example, *partition-based approaches* are defined by Song and Luedtke [123] as aggregation-based methods where scenario-specific constraints are aggregated according to partitions of the scenario set in order to obtain dual bounds; and the partitions are iteratively refined later.

4.2 Preliminaries and *EGSO* Bounds for Multistage Problems

We consider the multistage stochastic programming problem, with T stages, and with random data $\tilde{\xi}$ having finite support. In particular, the random data is defined on a probability space with discrete realization set $\Xi \subseteq \Xi^1 \times \cdots \times \Xi^T$, arranged according to a scenario tree; each realization $\xi \in \Xi$ corresponds to a path in the scenario tree from its root node to a leaf node, and every such path uses T nodes. We use the notation $\xi_{[t]}$ for $(\xi_{t'})_{t'=1}^{t} = (\xi_1, \ldots, \xi_t)$, the realization, ξ, for stages $1, \ldots, t$. Since all scenarios share a single tree node in the first stage, it is assumed that $\tilde{\xi}_1$ is deterministic. We take the multistage stochastic program (MSP) to have the following form:

$$z_{MSP} := \min\{f_1(x_1) + \mathbb{E}[g_2(x_1, \xi_{[2]})] \ : \ x_1 \in \mathcal{X}^1\}$$

where for each $t = 2, \ldots, T$, $g_t(x_{t-1}, \xi_{[t]})$ is defined as

$$g_t(x_{t-1}, \xi_{[t]}) = \min\{f_t(x_t, \xi_{[t]}) + \mathbb{E}[g_{t+1}(x_t, \xi_{[t+1]})|\xi_{[t]}] \ : \ x_t \in \mathcal{X}^t(x_{t-1}, \xi_{[t]})\},$$

where $g_{T+1} \equiv 0$; and stage t decision variables, x_t, are assumed to be of dimension n_t, so $x_t \in \mathbb{R}^{n_t}$ for each $t = 1, \ldots, T$. We allow any finite-valued functions f_t and any set-valued functions $\mathcal{X}^t$, provided (MSP) has an optimal solution, and provided the restriction to any proper subset of Ξ has an optimal solution. For practical implementation, we also require a solver that can handle problems in the form of (MSP).

Since $\tilde{\xi}$ has finite support, we may write $\Xi = \{\xi^1, \ldots, \xi^L\}$ for some positive integer L, and index the scenario set Ξ with $\mathcal{S} = \{1, \ldots, L\}$. Define $\mathcal{H}(t, s)$ to be the scenario tree node for the scenario with index s at stage t, for each $t = 1, \ldots, T$ and $s \in \mathcal{S}$. This permits us to write the (MSP) in its *extensive form* as

$$\begin{aligned} z_{MSP} = \ \min \ & \sum_{s \in \mathcal{S}} p_s \sum_{t=1}^{T} F_t^s(x_t^s) & \\ \text{s.t.} \ & (x_t^s)_{t=1}^T \in X^s, & \forall s \in \mathcal{S} \\ & y_{\mathcal{H}(t,s)} = x_t^s, & \forall t = 1, \ldots, T, \ s \in \mathcal{S}, \end{aligned}$$

where p_s is the probability that ξ^s is realized, and it is assumed that $\sum_{s \in \mathcal{S}} p_s = 1$. We define $F_t^s(x_t^s) = f_t(x_t^s, \xi_{[t]}^s))$, and

$$X^s = \{(x_t)_{t=1}^T \ : \ x_1 \in \mathcal{X}^1, \ x_t \in \mathcal{X}^t(x_{t-1}, \xi_{[t]}^s)), \ \forall t = 2, \ldots, T\}.$$

$y_h \in \mathbb{R}^{n_t}$ for each scenario tree node h at stage t is an auxiliary variable introduced to model the *non-anticipativity constraints (NACs)*, ensuring that decisions made at stage t do not depend on knowledge of realizations at later stages.

We now define the *group subproblem* for a subset $S \subseteq \mathcal{S}$ of the scenarios, denoted by $GR(S)$ to be

$$v_{GR}(S) = \frac{1}{\rho(S)} z_{GR}(S),$$

where

$$\begin{aligned} z_{GR}(S) = \ \min \ & \sum_{s\in S} p_s \sum_{t=1}^{T} F_t^s(x_t^s) & \\ \text{s.t.} \ & (x_t^s)_{t=1}^T \in X^s, & \forall s \in S \\ & y_{\mathcal{H}(t,s)} = x_t^s, & \forall t = 1,\ldots,T,\ s \in S, \end{aligned}$$

and we define

$$\rho(S) = \sum_{s\in S} p_s.$$

Note that, as in Sandıkçı and Özaltin [109], we use a simpler form of the group subproblem to that given by Sandıkçı et al. [108], without a reference scenario. Note also that our assumption that (MSP) restricted to any proper subset of Ξ has an optimal solution ensures that all group subproblems have an optimal solution.

Observe that $v_{GR}(\mathcal{S}) = z_{GR}(\mathcal{S}) = z_{MSP}$ and that for each $s \in \mathcal{S}$, the value of the group subproblem for the set consisting of the single scenario, s, is simply

$$v_{GR}(\{s\}) = \min\{\sum_{t=1}^{T} F_t^s(x_t^s) \ : \ (x_t^s)_{t=1}^T \in X^s\};$$

we call this the *single-scenario subproblem*. The so-called "wait-and-see" solution has value

$$z_{WS} = \sum_{s\in\mathcal{S}} p_s v_{GR}(\{s\}) = \sum_{s\in\mathcal{S}} z_{GR}(\{s\}).$$

It is well known that this provides a dual (*i.e.*, lower) bound on z_{MSP}, since it is precisely equivalent to solving (MSP) with all NACs omitted. Thus $z_{WS} \leq z_{MSP}$.

We now replicate the key result of Sandıkçı et al. [108], adjusted and extended to our multistage context (see also Maggioni et al. [85]). The *expected group subproblem objective* for an integer q with $1 \leq q \leq L = |\mathcal{S}|$ is denoted by $EGSO(q)$ and defined by

$$EGSO(q) = \frac{1}{\binom{L-1}{q-1}} \sum_{S\in\Omega_q} \rho(S) v_{GR}(S) = \frac{1}{\binom{L-1}{q-1}} \sum_{S\in\Omega_q} z_{GR}(S),$$

where Ω_q denotes the set of all subsets of $\mathcal{S}$ of cardinality q. The result is as follows.

Theorem 4.1 ([108], Theorem 1)**.**

$$z_{WS} = EGSO(1) \leq EGSO(2) \leq \cdots \leq EGSO(L-1) \leq EGSO(L) = z_{MSP}.$$

The proof of Theorem 4.1, as well as some necessary preliminary results, are given in Appendix C.1. The slight difference to the result in [108], where $z_{WS} \leq EGSO(1)$ rather than $z_{WS} = EGSO(1)$, is due to the omission of the reference scenario.

4.3 Expected Partition Bounds: A Hierarchy of Dual Bounds

We use the notation Π to denote a partition of the scenario set, $\mathcal{S}$, so $\Pi = \{S_1, \ldots, S_m\}$, for some m, with $S_i \subseteq \mathcal{S}$ for all $i = 1, \ldots, m$ and $S_i \cap S_{i'} = \emptyset$ for $i' = 1, \ldots, m$ with $i' \neq i$. It is well known that solving the group subproblem over all subsets in a partition yields a lower bound on (MSP).

Proposition 4.1. *Let Π be a partition of $\mathcal{S}$, with $\Pi = \{S_1, \ldots, S_m\}$, for some positive integer m. Then, defining $z^P(\Pi)$ to be the total value of the problems over all subsets of the partition Π, we have*

$$z^P(\Pi) = \sum_{i=1}^{m} z_{GR}(S_i) \leq z_{MSP}.$$

Thus a dual bound on z_{MSP} is obtained from a single partition. In practice, provided a solver is available for problems of the form of (MSP), and is computationally effective for problems with a modest number of scenarios, it can be applied in parallel to yield a single-partition bound for any partition in which all subsets are of modest size.

To obtain a hierarchy of bounds from partitions, we propose using the following particular type of partition, which, for a given integer q partitions the scenario set into sets of size q or "almost" q, specifically into sets of size q or $q-1$.

Definition 4.1. *Given a positive integer q, the q-partition set of $\mathcal{S}$, denoted by Λ_q, is the set of all partitions of $\mathcal{S}$ into m subsets of size q and m' subsets of size $q-1$, where $m' = q\lceil \frac{L}{q} \rceil - L$, $m = \lceil \frac{L}{q} \rceil - m'$ and $L = |\mathcal{S}|$.*

It is straightforward to verify that the q-partition set justifies its name, *i.e.*, that

$$mq + m'(q-1) = L.$$

Note that if q is "too large" relative to $L = |\mathcal{S}|$, then it is possible that m could be negative, and hence Λ_q could be ill-defined. Note also that, to be of practical interest, partitions should consist of sets having sizes small relative to the entire set of scenarios, otherwise the benefits of being able to solve many smaller problems in parallel to construct a lower bound from the partition is lost. Fortunately, as the following observation indicates, q can be quite large relative to L without m dropping to zero. For example, $q \leq \sqrt{L}$ suffices.

Lemma 4.1. *The q-partition set of $\mathcal{S}$, Λ_q, is well defined, i.e., $m \geq 1$, if and only if*

$$q|L \quad or \quad L \geq (q-r)q + r, \tag{4.1}$$

where $r = L - q\lfloor \frac{L}{q} \rfloor$.

The expected partition bound that we propose to consider is obtained by taking the average of all q-partition single-partition bounds.

Definition 4.2. *The* expected partition (EP) bound *for subsets of (almost) size q is denoted by $EP(q)$ and given by*

$$EP(q) = \frac{1}{|\Lambda_q|} \sum_{\Pi \in \Lambda_q} z^P(\Pi)$$

In what follows, we repeatedly use the observation that q-partitions of a set S can be enumerated by enumerating all permutations of its $\ell = |S|$ elements, and then taking each consecutive sequence of q (and then $q-1$) elements to form one of

the subsets in the partition. However, many permutations of $1, \ldots, \ell$ give the same partition, since each sequence of length q (or $q-1$) can be permuted without changing the partition, and the m sequences themselves can be permuted without changing the partition. In general, the number of distinct partitions of a set of size $\ell = mq$ into m sets of size q is

$$\frac{\ell!}{(q!)^m m!}.$$

By similar arguments, each set in Ω_q appears in

$$\frac{(\ell - q)!}{(q!)^{m-1}(m-1)!}$$

distinct partitions of S. Both formulae are easily extended to q-partitions of $\mathcal{S}$, as given in the proof of the proposition below.

Proposition 4.2. *Let q be a positive integer satisfying (4.1), and define m and m' as in Definition 4.1, where $L = |\mathcal{S}|$. Then*

$$EP(q) = \frac{qm}{L} EGSO(q) + \frac{(q-1)m'}{L} EGSO(q-1)).$$

Proof. Let $\ell = mq$ and $\ell' = m'(q-1)$, so $\ell + \ell' = L$. The number of distinct partitions of a set of size L into m sets of size q and m' sets of size $q-1$ is given by

$$|\Lambda_q| = \left(\begin{array}{c} L \\ \ell \end{array} \right) \frac{\ell!}{(q!)^m m!} \frac{\ell'!}{((q-1)!)^{m'} m'!},$$

which can easily be simplified to

$$|\Lambda_q| = \frac{L!}{(q!)^m m! ((q-1)!)^{m'} m'!}.$$

Now, for P a given subset of $\mathcal{S}$ of size q, any q-partition containing P must induce a partition of $\mathcal{S} \setminus P$ into two sets: S^1, the set of scenarios contained in subsets of size q of the partition, (but not in P), and S^2, the set of scenarios contained in subsets of size $q-1$ of the partition. Obviously $|S^1| = (m-1)q = \ell - q$ and $|S^2| = m'(q-1) = \ell'$.

Thus to construct all q-partitions that contain P, one may consider (i) all ways of choosing S^1 from $\mathcal{S} \setminus P$, (ii) all ways of partitioning S^1 into sets of size q, and (iii) all ways of partitioning S^2 into sets of size $q-1$, in combination. For case (i), there are $\begin{pmatrix} L-q \\ \ell-q \end{pmatrix}$ such ways. For case (ii) there are $(\ell-q)!/((q!)^{(m-1)}(m-1)!)$ ways. Similarly, for case (iii) there are $\ell'!/(((q-1)!)^{m'}m'!)$ ways. Putting these in combination, we see that each distinct subset of $\mathcal{S}$ of size q appears in

$$\eta_q := \begin{pmatrix} L-q \\ \ell-q \end{pmatrix} \frac{(\ell-q)!}{(q!)^{(m-1)}(m-1)!} \frac{\ell'!}{((q-1)!)^{m'}m'!}$$

partitions. The above expression can easily be simplified to

$$\eta_q = \frac{(L-q)!}{(q!)^{(m-1)}(m-1)!((q-1)!)^{m'}m'!}.$$

Similarly, each distinct subset of $\mathcal{S}$ of size $q-1$ appears in

$$\eta'_q := \begin{pmatrix} L-q+1 \\ \ell \end{pmatrix} \frac{\ell!}{(q!)^{m}m!} \frac{(\ell'-q+1)!}{((q-1)!)^{(m'-1)}(m'-1)!}$$

partitions; the above expression can easily be simplified to

$$\eta'_q = \frac{(L-q+1)!}{(q!)^{m}m!((q-1)!)^{(m'-1)}(m'-1)!}.$$

Now

$$\begin{aligned} \frac{1}{|\Lambda_q|} \sum_{\Pi \in \Lambda_q} z^P(\Pi) &= \frac{1}{|\Lambda_q|} \sum_{\Pi \in \Lambda_q} \sum_{S \in \Pi} z_{GR}(S) \\ &= \frac{1}{|\Lambda_q|} \left(\eta_q \sum_{S \in \Omega_q} z_{GR}(S) + \eta'_q \sum_{S \in \Omega_{q-1}} z_{GR}(S) \right). \end{aligned} \tag{4.2}$$

But

$$\begin{aligned} \frac{\eta_q}{|\Lambda_q|} &= \frac{(L-q)!}{(q!)^{(m-1)}(m-1)!((q-1)!)^{m'}m'!} \frac{(q!)^m m!((q-1)!)^{m'}m'!}{L!} \\ &= \frac{(L-q)!q!m}{L!} = \frac{qm}{L} \frac{(L-q)!(q-1)!}{(L-1)!} = \frac{qm}{L} \frac{1}{\binom{L-1}{q-1}}. \end{aligned}$$

Thus

$$\frac{\eta_q}{|\Lambda_q|}\sum_{S\in\Omega_q} z_{GR}(S) = \frac{qm}{L}\frac{1}{\binom{L-1}{q-1}}\sum_{S\in\Omega_q} z_{GR}(S) = \frac{qm}{L}EGSO(q). \tag{4.3}$$

Similarly,

$$\begin{aligned}\frac{\eta'_q}{|\Lambda_q|} &= \frac{(L-q+1)!}{(q!)^m m!((q-1)!)^{(m'-1)}(m'-1)!}\frac{(q!)^m m!((q-1)!)^{m'}m'!}{L!}\\ &= \frac{(L-q+1)!(q-1)!m'}{L!} = \frac{(q-1)m'}{L}\frac{(L-q+1)!(q-2)!}{(L-1)!}\\ &= \frac{(q-1)m'}{L}\frac{1}{\binom{L-1}{q-2}}.\end{aligned}$$

Thus

$$\frac{\eta'_q}{|\Lambda_q|}\sum_{S\in\Omega_{q-1}} z_{GR}(S) = \frac{(q-1)m'}{L}\frac{1}{\binom{L-1}{q-2}}\sum_{S\in\Omega_{q-1}} z_{GR}(S) = \frac{(q-1)m'}{L}EGSO(q-1). \tag{4.4}$$

The result follows by substituting from (4.3) and (4.4) into (4.2).

□

We now obtain our main result as a corollary.

Theorem 4.2. *If q divides evenly into $L = |\mathcal{S}|$ then*

$$EP(q) = EGSO(q)$$

and otherwise, provided that q satisfies (4.1),

$$EGSO(q-1) \leq EP(q) \leq EGSO(q),$$

with the first inequality strict unless $EGSO(q-1) = EGSO(q)$.

Proof. Let m and m' be as defined in Definition 4.1. Now if q divides evenly into L, then $m' = 0$, so $qm = L$, and the result follows by Proposition 4.2. Otherwise, suppose that q satisfies (4.1). Then, by Lemma 4.1, $m \geq 1$, obviously $\frac{qm}{L} > 0$ and $\frac{(q-1)m'}{L} \geq 0$, and, furthermore, $\frac{qm}{L} + \frac{(q-1)m'}{L} = 1$. Thus, by Proposition 4.2, it must be that $EP(q)$ is a convex combination of $EGSO(q-1)$ and $EGSO(q)$, with a strictly positive coefficient on the latter in the combination. Since $EGSO(q-1) \leq EGSO(q)$, the result follows. □

This result shows that the $EP(q)$ bounds provide a sequence of dual bounds for z_{MSP} that are monotonically nondecreasing in q, and that coincide with the $EGSO$ bounds in the case that the subset cardinality divides evenly into the cardinality of the scenario set, and otherwise interpolates between the bounds for two consecutive cardinalities.

4.4 Scenario Set Partition Sampling

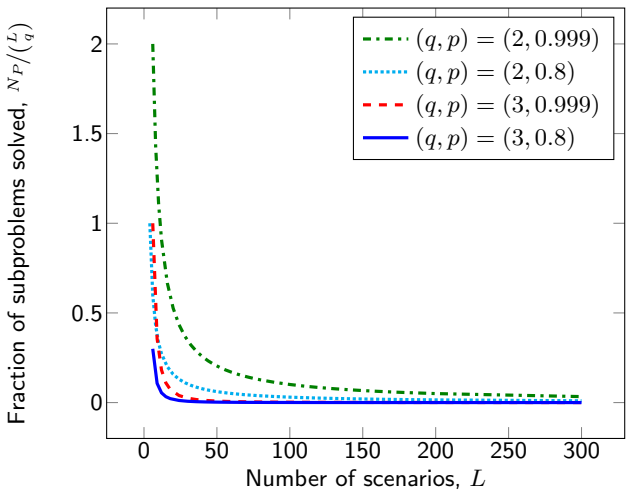
The practical value of the results in the previous section comes from the observation that there is a high likelihood that computing only a very small number of q-partition single-partition bounds will yield a better bound than $EP(q)$, provided the distribution of $z^P(\Pi)$ over $\Pi \in \Lambda_q$ is not highly right-skewed. Specifically, given $\tilde{\Lambda}_q \subset \Lambda_q$, an independently sampled subset of the q-partitions, we have from Proposition 4.1 that

$$\max_{\Pi \in \tilde{\Lambda}_q} z^P(\Pi) \leq z_{MSP}.$$

If, for example, the distribution of the values of $z^P(\Pi)$ over $\Pi \in \Lambda_q$ is symmetric about its average, then the probability that a single, randomly chosen $\Pi \in \Lambda_q$ will have value no greater than its average is 0.5. Since its average is exactly $EP(q)$, we have that

$$Pr(\max_{\Pi \in \tilde{\Lambda}_q} z^P(\Pi) \geq EP(q)) \quad = \quad 1 - Pr(z^P(\Pi) < EP(q),\ \forall \Pi \in \tilde{\Lambda}_q) \quad \approx \quad 1 - (0.5)^{|\tilde{\Lambda}_q|}.$$

To illustrate the utility of this, consider a problem with $L = 24$ scenarios, and take $q = 3$. To compute $EP(3) = EGSO(3)$, we must compute the solution to $\binom{24}{3} = 2024$ group subproblems, each with 3 scenarios. However, if we randomly sample 10 partitions, we need to solve only $\frac{L}{q} \times |\tilde{\Lambda}_q| = \frac{24}{3} \times 10 = 80$ such group subproblems, and will have found a bound at least as good with probability $1-(0.5)^{10} > 0.999$. This idea is demonstrated in Figure 4.1 for a symmetric distribution of partition bounds. As the number of scenarios increases, the fraction of subproblems that need to be solved in order to find a better partition bound than the expected bound (EP) with a given



Note. q: Group size, p: Target probability of finding a better partition bound than $EP(q)$ after solving N_P subproblems, $N_P = \frac{L}{q} \times \lceil \log_2 \frac{1}{1-p} \rceil$.

Figure 4.1: Best partition vs. expected partition

probability gets smaller. Even in the case that the distribution of the values of $z^P(\Pi)$ over $\Pi \in \Lambda_q$ is somewhat right-skewed, for example, say $Pr(z^P(\Pi) \geq EP(q)) = 0.2$ only, then an independent random sample of 31 partitions, requiring solution of only 248 group subproblems, is sufficient to ensure that the best bound generated by one of them is at least $EP(q)$ with probability at least 0.999. On benchmark instances, our observation echoes that made by Sandıkçı and Özaltin [109]: these distributions, in the case of q-partitions, are not highly skewed.

This motivates Algorithm 4.1, which computes exact lower bounds from randomly sampled q-partitions. The stopping criterion enforces a maximum on the number of partitions sampled, K^{max}, which is a given parameter of the algorithm. Although the primary purpose of Algorithm 4.1 is to generate a good dual bound, there may also be an opportunity to generate upper bounds during the course of the algorithm, making use of the solution to each group subproblem. For example, Sandıkçı and Özaltin [109] describe such a heuristic. Even though randomly sampled partitions provide effective results, as we discuss in Section 4.5, we explore several ways to select partitions more intelligently, so as to yield better partition bounds. These are described in the remainder of this section.

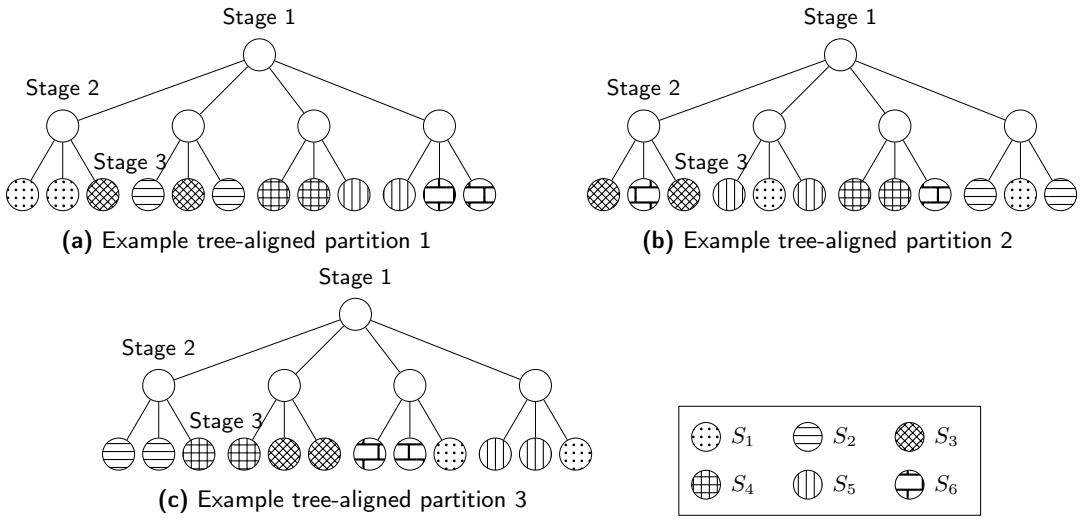
Algorithm 4.1 *Scenario Set Partition Sampling* (S, L, q, K^{max})

Inputs: Set of scenarios S, $L = |S|$, group size q, number of partitions K^{max}
$LB := -\infty$, $k := 1$
while $k \leq K^{max}$ **do**
    $S := \mathcal{S}$
    for all $i = 1, \ldots, \lceil \frac{L}{q} \rceil$ **do**
        / Sample, without replacement, the next subset in the partition */*
        if $i \leq L - (q-1)\lceil \frac{L}{q} \rceil$ **then**
            S_{ki} := a random sample of q scenarios from S
        else
            S_{ki} := a random sample of $q - 1$ scenarios from S
        end if
        / Solve the multistage SP over scenarios in the current subset */*
        Solve $GR(S_{ki})$ to get optimal value $z_{GR}(S_{ki})$
        $S := S \setminus S_{ki}$
    end for
    $z_k := \sum_{i=1}^{\lceil L/q \rceil} z_{GR}(S_{ki})$
    / Update exact lower bound */*
    $LB := \max\{LB, z_k\}$
    [Optional]: seek an optimally recombined partition (Section 4.4.2) to improve LB
    $k := k + 1$
end while

4.4.1 Partition Sampling Based on Scenario Tree Structure

When using partition sampling on multistage problem instances, taking advantage of the scenario tree structure can potentially result in improved partition bounds. Selecting partitions that group together scenarios with as many common nodes in the scenario tree as possible can provide better partition bounds compared to random partition sampling, due to fewer NACs being violated. On the other hand, partitions that keep "similar" scenarios (scenarios with common nodes in the scenario tree) in different groups also have the potential to provide better bounds than randomly sampled partitions, due to each group subproblem being a more accurate representation of the original problem. Here, we explore both strategies.

Algorithm 4.2 defines our method of generating q-partitions that groups scenarios with common nodes together, called the "tree-aligned" partitioning strategy. Starting



(a) Example tree-aligned partition 1

(b) Example tree-aligned partition 2

(c) Example tree-aligned partition 3

Figure 4.2: Example tree-aligned partitions for a tree with $|\Xi^1 \times \Xi^2 \times \Xi^3| = 1 \times 4 \times 3$ and $q = 2$

with any scenario tree node j^* in the penultimate stage, $T - 1$, it randomly selects scenarios whose scenario tree path includes j^*, denoted by $\mathcal{N}_{j^*}$, to add to a first partition subset S_1. To keep track of which scenarios have been assigned to a partition subset, the selected scenario is removed from $\mathcal{N}_{j^*}$. This is repeated until either the current subset, S_i, is full, or there are no more scenarios that include node j^* ($\mathcal{N}_{j^*}$ is empty). In the former case, a new subset is started, (i is incremented), and the process continues. In the latter case, a new tree node, j^*, with $\mathcal{N}_{j^*}$ not empty, having as many tree nodes in common with the previous tree node used to select a scenario, is found, and again the process continues. This strategy ensures that the resulting partition solution satisfies as many NACs as possible. The concept of a tree aligned partition is illustrated in Figure 4.2, which shows three tree-aligned partitions on a small, 3-stage scenario tree, with $|\Xi^1 \times \Xi^2 \times \Xi^3| = 1 \times 4 \times 3$ and $q = 2$. Note that provided $q \neq |\Xi^t \times \Xi^{t+1} \times \cdots \times \Xi^T|$ for some t, in which case there is a unique tree-aligned partition, the number of tree-aligned partitions is still large relative to the total number of partitions. For example, a $1 \times 2 \times 3$ tree with $q = 2$ has 9 distinct tree-aligned partitions out of a total 90 distinct partitions and a $1 \times 3 \times 2$ tree with $q = 3$ has 6 distinct tree-aligned partitions out of a total 20.

Algorithm 4.2 *Partitioning Strategy, "Tree-Aligned"* $(S, L, q, \mathcal{H})$

Inputs: Set of scenarios S, $L = |S|$, group size q, scenario tree $\mathcal{H}$
$\mathcal{N}_j$: the set of scenarios whose scenario tree path includes node j
$j^* :=$ any scenario tree node in the penultimate stage, $T-1$
for all $i = 1, \ldots, \lceil \frac{L}{q} \rceil$ **do**
  $S_i := \emptyset$
  while $(i \leq L-(q-1)\lceil \frac{L}{q} \rceil$ and $|S_i| < q)$ or $(i > L-(q-1)\lceil \frac{L}{q} \rceil$ and $|S_i| < q-1)$ **do**
    if $\mathcal{N}_{j^*} \neq \emptyset$ **then**
      $k :=$ randomly picked scenario from $\mathcal{N}_{j^*}$
      $S_i := S_i \cup \{k\}$
      Remove scenario k (from $\mathcal{N}_j$, for all nodes j on its scenario tree path)
    end if
    while $\mathcal{N}_{j^*} = \emptyset$ **do**
      $j^* :=$ its parent node
    end while
    while a child node of j^* has $\mathcal{N}_{j^*} \neq \emptyset$ **do**
      $j^* :=$ randomly picked child node of j^* with $\mathcal{N}_{j^*} \neq \emptyset$
    end while
  end while
end for

To evaluate this method, we test the tree-aligned ("tree", in short) partitioning strategy against two others: a "random" strategy, which randomly partitions the scenario set without considering the tree structure; and a "misaligned" strategy, which keeps the scenarios with common nodes in separate groups as much as possible. These three different partitioning strategies are illustrated on a 3-stage example with 9 scenarios in Figure 4.3.

An extensive computational study on the quality of partition bounds attained by each of the three partitioning strategies is conducted, and the results are discussed in Section 4.5. Computational results reveal that the tree-aligned partitioning strategy provides considerable improvements in bound quality over the other two strategies on the 3-stage instance class used in our computational study. However, the performance of different partitioning strategies depends on specific characteristics of the problem instance under consideration, therefore it is not always obvious which strategy would

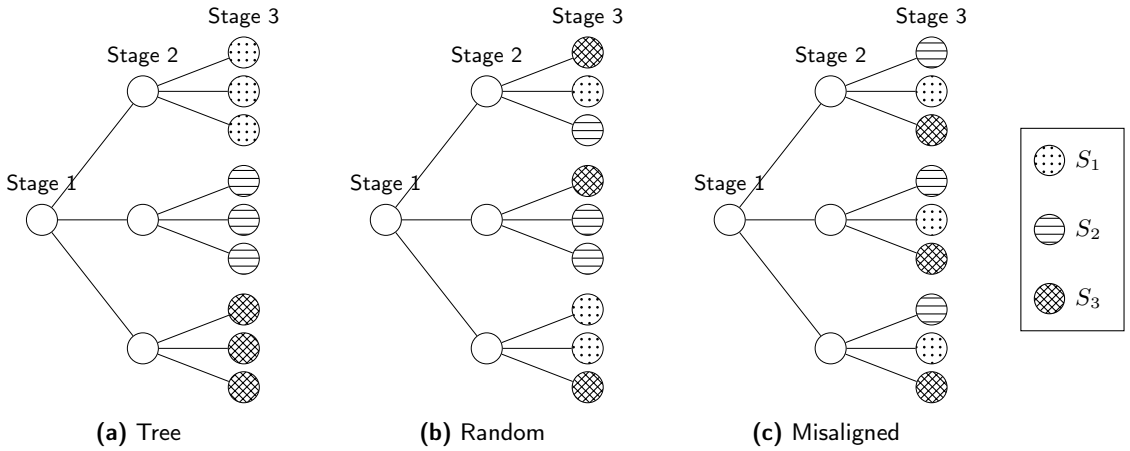


Figure 4.3: Three alternative methods to construct partitions based on scenario tree structure

result in better partition bounds. In Section 4.5, we present a discussion on the effects of some instance-specific characteristics that affect bound quality in tree-aligned and misaligned partitioning strategies.

4.4.2 Optimally Recombined Partitions

As the scenario set partition sampling procedure randomly samples partitions, it computes the group subproblem objectives, $z_{GR}(S)$, associated with scenario subsets, S, of these partitions. Once a sufficient number of partitions are sampled, it may be possible to recombine the previously sampled subsets into a new and potentially better partition. Thus, we seek optimally recombined partitions by solving a set partitioning problem over the previously sampled subsets, S, and the corresponding group subproblem objectives.

Recalling that $\mathcal{S}$ represents the set of scenario indices, let $\mathcal{C}$ represent the collection of previously sampled scenario subsets, and $\mathcal{C}(i)$ denote the collection of scenario subsets containing scenario i. An optimally recombined partition can be found by solving

$$\underset{x\in\{0,1\}^{|\mathcal{C}|}}{\arg\max}\{\sum_{S\in\mathcal{C}} z_{GR}(S)x_S : \sum_{S\in\mathcal{C}(i)} x_S = 1, \quad \forall i \in \mathcal{S}\}.$$

We incorporate optimal recombination into the partition sampling procedure by

solving the set partitioning problem at certain iterations. If a recombined partition that is different from the previously sampled partitions provides a better bound than the current best, then we update the current best bound accordingly. Otherwise, we continue to sample partitions.

4.4.3 Truncated Partition Bound Calculation

During the progression of the partition sampling algorithm, only a small fraction of partitions actually improve the best partition bound. (This is to be expected, since if k partitions have been solved so far, the probability that the next partition yields a better value is $1/(k+1)$.) This suggests the possibility of reducing wasted computational effort by using the group subproblem values calculated part-way through a partition to heuristically estimate the final partition bound, and to decide whether a partition is "promising", or not, before all the group subproblems of the partition are solved.

In order to determine how promising a partition is after solving only a subset of the group subproblems, we define a heuristic estimate for the partition bound $z^P(\Pi)$. When group subproblems associated with only the first r scenario subsets $\{S_1, \ldots, S_r\}$ of partition Π are solved, the heuristic estimate $\hat{z}_r^P(\Pi)$ is defined as

$$\hat{z}_r^P(\Pi) = \frac{1}{\sum_{i=1}^{r} \rho(S_i)} \sum_{i=1}^{r} z_{GR}(S_i).$$

We consider the following rule for eliminating unpromising partitions, parameterized by the triple (α, β, γ) with $\alpha, \beta \in [0, 1]$ and $\gamma > 0$.

> *Eliminate partition* Π *if, after at least* α *fraction of the group subproblems are solved, the heuristic estimate of* $z^P(\Pi)$ *remains lower than* γ *times the best bound, for the next* r' *group subproblems solved, where* r' *is at least* β *fraction of the remaining groups.*

In other words, when m and $z^P(\Pi^*)$ denote the number of groups in partition Π and the best partition bound found so far, respectively, under the truncation rule

parametrized by (α, β, γ), we characterize a partition Π as "unpromising" if

$$\hat{z}_i^P(\Pi) \leq \gamma z^P(\Pi^*), \qquad \forall i = r, \ldots, r + r',$$

provided $r \geq \lceil \alpha m \rceil$ and $r' = \lceil \beta(m - r) \rceil - 1$. Note that eliminating a partition, *i.e.*, ceasing computation of the value of the group subproblems in the partition, will save the computation time of at least $(1 - \beta)(m - r)$ group subproblems.However, as the estimate is only a heuristic, it may also eliminate a partition giving a new best bound. The higher the values of α and β, and the lower the value of γ, the more conservative the truncation rule. A more conservative rule will save less computing time, at less risk of eliminating a partition yielding a better bound.

In our computational experiments, we explore the extent of computational savings that can be obtained from truncated bound calculation. Furthermore, we present examples in which the computational effort saved by eliminating unpromising partitions is used towards promising partitions, resulting in an improved partition bound.

4.5 Computational Results

Computational experiments are performed for three different classes of problems: stochastic capacitated facility location problem (CAP), stochastic server location problem (SSLP), and dynamic capacity acquisition and allocation problem (DCAP). The former two classes are 2-stage, and the latter is 3-stage. Computations are run on a heterogeneous cluster of machines with Xeon E5520, Xeon E5-2670, E5-2603, E5-2650v3 and Xeon E5430 processors using Gurobi 5.6.3 Python interface (with Python 2.7). Different values for the group size, q, are considered, while ensuring that q divides the number of scenarios, L, evenly, for simplicity.

4.5.1 Test Instances

The stochastic capacitated facility location problem (CAP) considered here is described in [83]. The first stage decisions determine which facilities to open, and the

second stage decisions give the fraction of customer demand to be satisfied by each open facility. The instances we use come from [21]. Each instance has 5000 scenarios. Here, we use the first 240 of them, and experiment with group sizes $q = 5, 10, 15, 20$.

The stochastic server location problem (SSLP) considered in this paper is described in [93], and the instances used in experiments come from [4]. The first stage decisions concern installation of servers at possible locations, and the second stage decisions define assignment of clients to servers. Group sizes $q = 2, 5, 10, 25$ are used in experiments on SSLP instances with 50 and 100 scenarios.

The dynamic capacity acquisition and allocation problem (DCAP) is described in [3], and the 3-stage instances used in computational experiments of this paper are acquired from [135]. The capacity acquisition decisions are made at stages 1 and 2; and based on the acquired capacities of the resources, allocation decisions are made at stages 2 and 3. The test instances are named as "DCAP $n_i n_j$2 1xn_2xn_3", where n_i and n_j represent the number of resources and tasks, respectively; n_2 represents the number of second stage nodes in the scenario tree, and n_3 represents the number of third stage nodes originating from every second stage node. The instances used in this study have 200, 300, and 500 scenarios. Computational experiments are conducted using group sizes $q = 2, 5, 10, 20, 30, 40, 50$.

Details of these test problems are given in Appendix C.2.

4.5.2 Computation Time

As briefly mention in Section 4.3, partition bounds provide a computationally efficient way to obtain dual bounds for problems that have too many scenarios to allow direct solution of the deterministic equivalent, or even to permit the model to be loaded without encountering memory issues. In this section, we investigate the computational burden of partition bounds by providing solution times associated with

different numbers of scenarios (L) and group sizes (q) for a typical capacitated facility location problem instance (CAP 101) using a Xeon E5-2603 processor.

We experiment with $L = 240$, 480, and 960 scenarios and various group sizes. For each number of scenarios and group size, 10 partitions are randomly sampled and the solution times are reported. Table 4.1a presents the total solution time (averaged over 10 partitions) to represent the case when all group subproblems are solved sequentially, and Table 4.1b presents the maximum solution time among all group subproblems (averaged over 10 partitions) to represent the case when the group subproblems are solved in parallel. To better understand the tradeoff between solution time and bound quality, best partition bounds (among the 10 partitions) associated with each (L, q) pair are reported in Table 4.2 and the fractions of series/parallel computation time required to achieve the partition bounds in Table 4.2 are presented in Table 4.3.

Table 4.1: Solution times (in seconds) for CAP 101

(a) Total solve time

	q					
L	5	10	20	40	80	$q = L$
240	52.71	88.97	242.15	329.19	466.04	5,585.30
480	98.04	162.74	484.67	778.17	944.44	15,351.15
960	223.83	240.61	579.96	1,515.23	1,871.99	87,683.96

(b) Maximum solve time

	q					
L	5	10	20	40	80	$q = L$
240	2.08	10.07	49.47	121.75	168.80	5,585.30
480	2.70	9.99	58.79	161.05	234.98	15,351.15
960	5.01	8.70	38.09	162.28	184.90	87,683.96

Note. The values in both tables are averages of 10 solutions. For $q < L$, 10 partitions are randomly sampled and the average total (or maximum) solution time is reported. For $q = L$, the original problem with L scenarios is solved 10 times and the average solution time is reported.

Table 4.2: Best partition bounds (among 10 randomly sampled partitions) for CAP 101

	q					
L	5	10	20	40	80	$q = L$
240	732,005.22	733,311.23	733,967.62	734,158.09	734,302.79	734,354.25
480	729,194.92	730,530.73	731,161.12	731,434.18	731,604.76	731,631.30
960	729,929.05	731,283.14	731,888.98	732,184.13	732,345.42	732,490.10

Table 4.3: Expected fraction of series/parallel computation time needed to achieve the partition bounds shown in Table 4.2

	q				
L	5	10	20	40	80
240	0.094/0.0004	0.159/0.0018	0.434/0.0089	0.589/0.0218	0.834/0.0302
480	0.064/0.0002	0.106/0.0007	0.316/0.0038	0.507/0.0105	0.615/0.0153
960	0.026/0.0001	0.027/0.0001	0.066/0.0004	0.173/0.0019	0.213/0.0021

Note. The computation times under consideration are for computing 10 partition bounds, the best ones among which are presented in Table 4.2. The reported fractions represent the computation time required for solving all group subproblems in series and in parallel, respectively, divided by the computation time required for solving the original problem with $q = L$.

It is clear that in each one of the $L = 240$, 480, and 960 cases, using moderate group sizes such as $q = 5$, 10, or 20, allows computing partition bounds in very reasonable times even when parallel implementation is not possible, as opposed to using very large group sizes or directly solving the original problem. It is also important to note that when trying to directly solve the original problem, memory restrictions must be taken into consideration, in addition to computation time issues. For example, directly solving the CAP 101 instance with 960 scenarios is not possible for moderate memory sizes, as it uses over 50 GB of memory just to load the model.

Since larger group sizes provide better partition bounds, it is desirable to use

the largest group size allowed by the processing power, memory, and parallel implementation capability. But even when the computing platform at hand has moderate processing power and memory, finding partition bounds with small group sizes is a computationally efficient way of obtaining good dual bounds, especially if parallel computing is available.

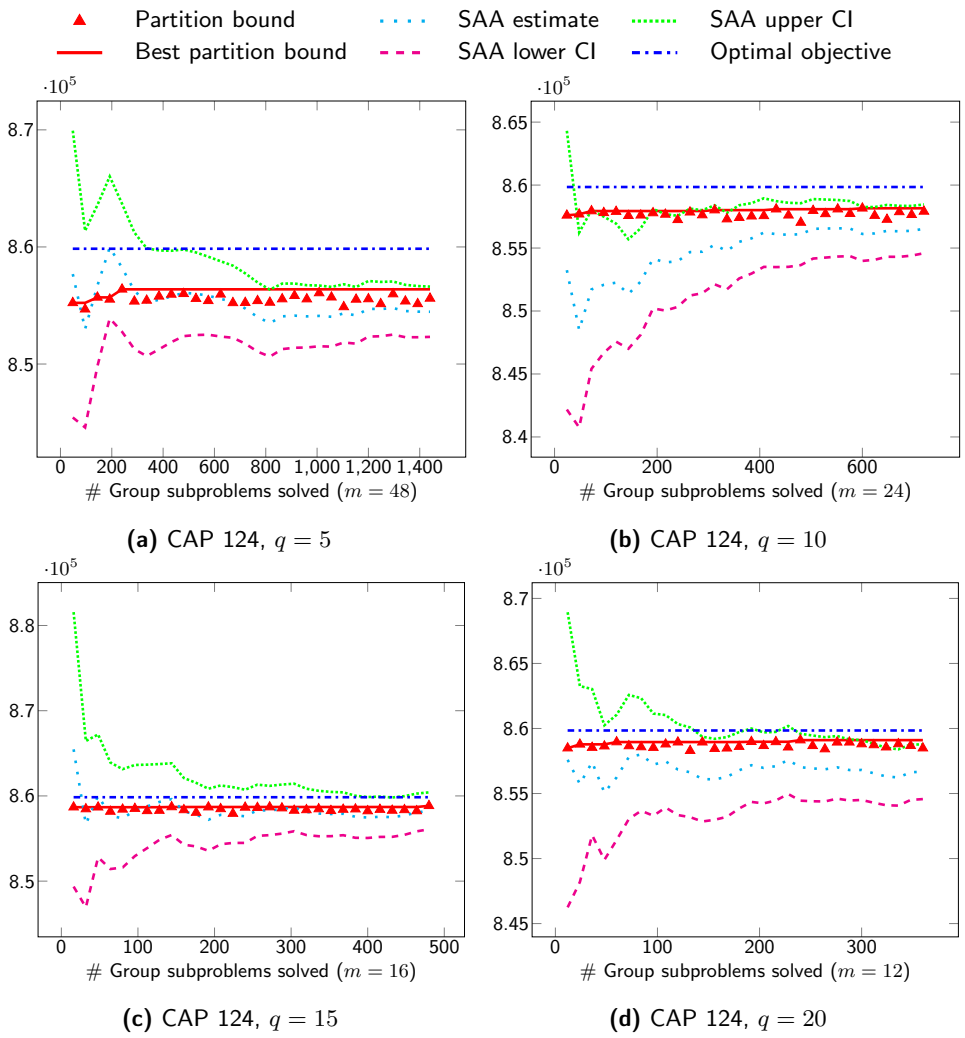
4.5.3 Comparing Partition Sampling with SAA

Partition bounds are compared against Sample Average Approximation (SAA) estimates found by solving the same size and number of group subproblems and then calculating confidence intervals around the resulting estimates. The lower end of the confidence interval is taken to be the SAA lower bound estimate. For each instance and each q value tested, we run Algorithm 4.1 with $K^{max} = 30$ or 50, requiring solution of $n' = K^{max}L/q$ group subproblems, each of size q. We then apply SAA to the instance, using similar computational effort: we take n' independent samples of size q, generated, (with replacement), based on the probability distribution associated with scenario occurrences. For each scenario sample, S, the sample average value, denoted as $z_{SA}(S)$ is found by solving the following problem (see Kleywegt et al. [78]):

$$z_{SA}(S) = \min\{\frac{1}{q}\sum_{s\in S}\sum_{t=1}^{T} F_t^s(x_t^s) \ : \ (x_t^s)_{t=1}^T \in X^s, \forall s \in S, \ y_{\mathcal{H}(t,s)} = x_t^s, \forall t = 1,\ldots,T, \ s \in S\}.$$

After n' subproblems are solved, for samples $S_1, S_2, \ldots, S_{n'}$, the corresponding SAA estimate is $\hat{z}_{n'}^{SAA} = \frac{1}{n'}\sum_{i=1}^{n'} z_{SA}(S_i)$. Then the sample standard deviation is $\hat{s}_{SAA}^2 = \frac{1}{n'-1}\sum_{i=1}^{n'}\left(z_{SA}(S_i) - \hat{z}_{n'}^{SAA}\right)^2$ and the confidence interval with a level of confidence α is $\hat{z}_{n'}^{SAA} \pm t_{\frac{\alpha}{2},n'-1}\sqrt{\frac{\hat{s}_{SAA}^2}{n'}}$.

Figure 4.4 illustrates, using a typical CAP instance, how partition bounds compare to SAA estimates and 95% confidence intervals around SAA estimates. Partition bounds associated with 30 independent partitions are presented, along with the SAA estimates calculated by solving the same size and number of group subproblems. For comparability of results in terms of computational burden, SAA estimates on the



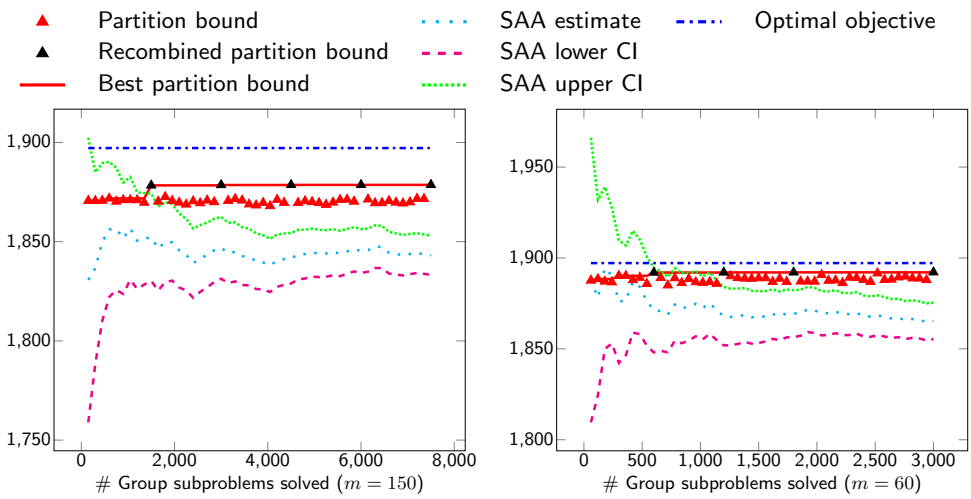
(a) CAP 124, $q = 5$

(b) CAP 124, $q = 10$

(c) CAP 124, $q = 15$

(d) CAP 124, $q = 20$

Figure 4.4: CAP 124 Partition sampling vs. SAA ($K^{max} = 30$)



(a) DCAP 342 1x15x20, $q = 2$

(b) DCAP 342 1x15x20, $q = 5$

(c) DCAP 342 1x15x20, $q = 10$

(d) DCAP 342 1x15x20, $q = 20$

Figure 4.5: DCAP 342 1x15x20 Partition sampling (with tree-aligned partitioning strategy and optimal recombination) vs. SAA ($K^{max} = 50$)

plots are updated only when a partition bound is fully computed. It can be observed that the best partition bound is significantly greater than the lower limit of the 95% confidence interval around the SAA estimate. Also, in most cases, the best partition bound exceeds the SAA estimate. Usefully, the best partition bound improves quite rapidly, so good bounds are available very early in the sampling process.

Figure 4.5 shows the progression of partition bounds and SAA estimates, on a 3-stage DCAP instance, when the tree-aligned partitioning strategy is used (as described in Section 4.4.1) and an optimally recombined partition is attempted (as described in Section 4.4.2) every 10 iterations. As in the previous example, the best partition bound exceeds the lower limit of the 95% confidence interval around the SAA estimate. Additionally, good partition bounds are obtained very early in the partition sampling process, while SAA estimates require a considerably long warm-up phase.

Similar behavior is observed in all CAP, SSLP, and DCAP instances. Tables 4.4 and 4.5 present

(i) the gap obtained from the best partition bound after $k = 30$ q-partitions , Δ_k^P (Equation 4.5),

(ii) the gap obtained from the lower limit of the SAA 95% confidence interval after a computational effort equivalent to $k = 30$ q-partitions, Δ_k^{SAA} (Equation 4.6), and

(iii) the proportion of the latter gap that is closed by the best partition bound, $\delta_k^{SAA,P}$ (Equation 4.7).

The gaps are calculated respective to the wait-and-see solution, z_{WS}. Since the test instances have a considerable amount of "sunk cost", which is the cost that has to be incurred regardless of solution quality, the objective values associated with different solutions do not seem very different from each other. In order to be able to objectively

compare different solutions in terms of quality, we subtract the lowerbound z_{WS} from the objective values in our reporting. Provided that OPT represents the the optimal value over all L scenarios, the gaps are calculated as follows:

$$\Delta_k^P = \frac{(OPT - z_{WS}) - (max_{i=1,\ldots,k}\{z^{P_i}\} - z_{WS})}{OPT - z_{WS}} \tag{4.5}$$

$$\Delta_k^{SAA} = \frac{(OPT - z_{WS}) - \left(\hat{z}_{km}^{SAA} - t_{0.025,km-1}\sqrt{\frac{\hat{s}_{SAA}^2}{km}} - z_{WS}\right)}{OPT - z_{WS}}, \text{ where } m = \frac{L}{q} \tag{4.6}$$

$$\delta_k^{SAA,P} = \frac{\Delta_k^{SAA} - \Delta_k^P}{\Delta_k^{SAA}} \tag{4.7}$$

Table 4.4 demonstrates the gaps Δ_{30}^P, Δ_{30}^{SAA}, $\delta_{30}^{SAA,P}$ for the SSLP instances with different group sizes q, and Table 4.5 gives the means and standard deviations of Δ_{30}^P, Δ_{30}^{SAA} and $\delta_{30}^{SAA,P}$, taken over all 16 CAP instances with different values of q. It can be seen in both tables that after 30 partitions, the gaps from the best partition bound are noticeably tighter than those from the SAA confidence interval lower limit for the same computational effort. Furthermore, it can be seen in Appendix C.3 that in the majority of instances, the best partition bound is attained within 10 or 20 partitions.

Table 4.4: Best partition bound vs. lower limit of the 95% SAA confidence interval on SSLP instances.

Instance	L	q	Δ_{30}^P	Δ_{30}^{SAA}	$\delta_{30}^{SAA,P}$
SSLP 5-25-50	50	2	41.60%	73.36%	43.29%
		5	11.62%	19.97%	41.83%
		10	0.00%	30.50%	100.00%
		25	0.00%	21.19%	100.00%
SSLP 10-50-100	100	2	45.29%	67.90%	33.30%
		5	9.01%	25.51%	64.67%

Table 4.5: Best partition bound vs. lower limit of the 95% SAA confidence interval: summary for CAP instances.

L	q	Δ_{30}^{P}		Δ_{30}^{SAA}		$\delta_{30}^{SAA,P}$	
		Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
240	5	14.87%	4.53%	32.35%	8.34%	52.23%	14.08%
240	10	6.45%	3.37%	25.71%	8.44%	71.07%	16.96%
240	15	3.53%	2.34%	19.17%	8.80%	80.23%	14.27%
240	20	2.40%	1.87%	17.14%	10.16%	83.54%	12.88%

4.5.4 Partition Bound Sampling Enhancements

For problem instances with more than 2 stages, the method of sampling partitions based on scenario tree structure is described in Section 4.4.1. We conduct a computational study for testing the bound quality for tree-aligned, random, and misaligned partitioning strategies in DCAP instances. It is observed that in all problem instances belonging to the DCAP instance class, when partitions are generated with the tree-aligned strategy, the resulting partition bounds are better than those for random and misaligned partitioning strategies. With 30 partitions, the tree-aligned strategy closes, on average, 85% of the wait-and-see gap with group size of only 5, and 96% with size 10. As reported in detail in Tables C.6 and C.7 of Appendix C.3, the tree-aligned strategy obtains significantly better bounds than those found using other partitioning strategies, even with group sizes many times larger. The tree-aligned partitions close up to 3.5 times (1.9 times, on average) the wait-and-see gap that can be closed by the "random" partitions, which are constructed without taking the scenario tree structure into account. Therefore, in the remainder of this paper, we only report the tree-aligned partition bounds associated with the DCAP instances (unless stated otherwise).

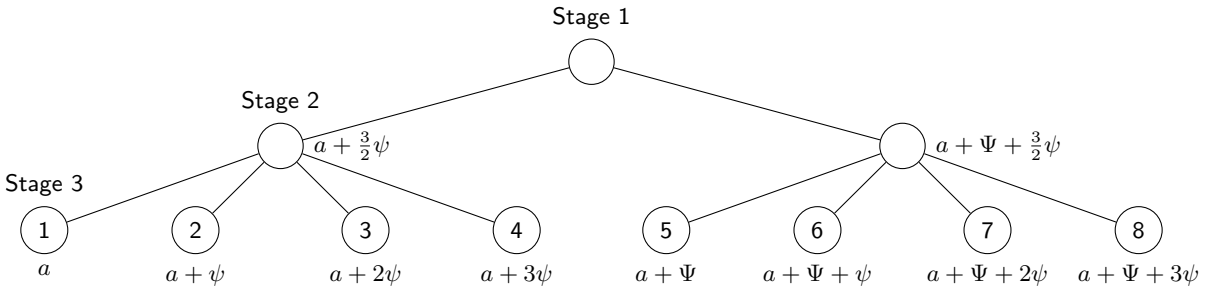
Clearly, there are potential advantages and disadvantages in both the tree-aligned and the misaligned partitioning strategies. The tree-aligned strategy violates as few

NACs as possible, but may fail to represent the entire scenario set accurately in each group subproblem. The misaligned strategy, on the other hand, represents a better portion of the entire scenario set in every group subproblem, but violates many NACs. To understand the characteristics of the scenario tree that may determine which of these two partitioning strategies result in better bounds, we experiment with a small DCAP instance.

We conjecture that when the major distinction between the scenario tree nodes is observed in initial stages (closer to the root node), better representing the scenario set in group subproblems is critical, so the misaligned partitioning strategy provides better bounds. Similarly, when the major distinction between the nodes is observed in final stages (closer to the leaves), both partitioning strategies would represent the scenario set in group subproblems well, and therefore violating fewer NACs becomes more important in bound quality. In that case, the tree-aligned partitioning strategy has a potential advantage in providing better bounds.

To test these ideas, we use a DCAP instance with only 12 scenarios, as depicted in Figure 4.6. The only uncertain parameter in the DCAP instances is the capacity consumption of each task on each resource. In order to better quantify the distinction between various scenario tree nodes, we design the problem instance so that the capacity consumption of every task on every resource is the same for a given scenario. For example, in scenario 1, all capacity consumptions are equal to a; while in scenario 8, all capacity consumptions are equal to $a + \Psi + 3\psi$. Ψ and ψ determine the difference between the two penultimate stage nodes and the difference between leaf nodes originating from the same penultimate stage node, respectively. Based on our conjecture, we expect that increasing values of Ψ would favor the misaligned partitioning strategy, and increasing values of ψ would favor the tree-aligned partitioning strategy.

We create three settings based on the overall capacity consumption: low, medium,



Note. The expression below each node represents the value of every uncertain parameter at the given stage and node.

Figure 4.6: Example scenario tree with $|\Xi^1 \times \Xi^2 \times \Xi^3| = 1 \times 2 \times 4$ for DCAP instances

and high. In each of these settings, we experiment with three Ψ values and three ψ values. For each (Ψ, ψ) pair, the value of a is calculated based on the total capacity consumption, A, with $a = \frac{A-4\Psi-12\psi}{8}$. For every setting and every (Ψ, ψ) pair, we generate all tree-aligned and all misaligned partitions, compute the corresponding partition bounds, and compare the average tree-aligned bound with the average misaligned bound. The results obtained from the described experimental setting are presented in Table 4.6. Values in the table represent the difference of average tree-aligned and average misaligned partition bounds, divided by the optimal objective. More specifically, $\delta^{T,M} = \frac{1}{OPT}\left(\frac{1}{|\Lambda^T|}\sum_{\Pi\in\Lambda^T} z^P(\Pi) - \frac{1}{|\Lambda^M|}\sum_{\Pi\in\Lambda^M} z^P(\Pi)\right)$, where Λ^T and Λ^M denote the collection of tree-aligned and misaligned partitions, respectively. Details of the percentage values in Table 4.6 are presented in Tables C.8, C.9, and C.10 of Appendix C.3.

Table 4.6: Difference of tree-aligned and misaligned partition bounds in a small DCAP 332 instance

(a) Low consumption ($A = 2.7$)

	ψ		
Ψ	0.01	0.05	0.09
0	0.02%	0.08%	0.16%
0.2	-11.87%	-8.86%	-4.92%
0.4	-23.11%	-22.77%	-22.45%

(b) Medium consumption ($A = 4.6$)

	ψ		
Ψ	0.01	0.11	0.21
0	0.01%	0.14%	0.22%
0.25	-15.35%	-10.02%	-2.04%
0.5	-20.43%	-17.69%	-7.89%

(c) High consumption ($A = 9.0$)

	ψ		
Ψ	0.01	0.15	0.3
0	0.01%	1.20%	1.67%
0.7	-0.08%	0.13%	1.21%
1.3	-2.13%	-1.30%	0.76%

Note. Plotted values, $\delta^{T,M} = \frac{1}{OPT}\left(\frac{1}{|\Lambda^T|}\sum_{\Pi\in\Lambda^T} z^P(\Pi) - \frac{1}{|\Lambda^M|}\sum_{\Pi\in\Lambda^M} z^P(\Pi)\right)$, are the percentage difference of average tree-aligned and misaligned partition bounds. Negative values indicate that the misaligned strategy gives the better bound.

Table 4.6 verifies that larger differences between leaf nodes (small Ψ, large ψ) provides an advantage for the tree-aligned partitioning strategy, and larger differences between earlier tree nodes (large Ψ, small ψ) provides an advantage for the misaligned partitioning strategy. Unfortunately, in the extant benchmark instances, the level of the scenario tree at which the major distinction between the scenarios occurs is not always clear. Indeed, the dependence, if any, of uncertainty parameter probability distributions on the tree node at which they are sampled is not clearly articulated in the problem descriptions. Therefore it is not easy to decide, a priori, which partitioning strategy would provide the best bounds for a specific problem instance. However, in the large DCAP instances ($L = 200, 300$, and 500) used in our experimental study, there is no evidence pointing to a major distinction between the earlier nodes of the scenario tree. The capacity consumption realizations at each leaf node seem to be independent of their parent node. Under these circumstances, as expected, the tree-aligned partitioning strategy provides better partition bounds.

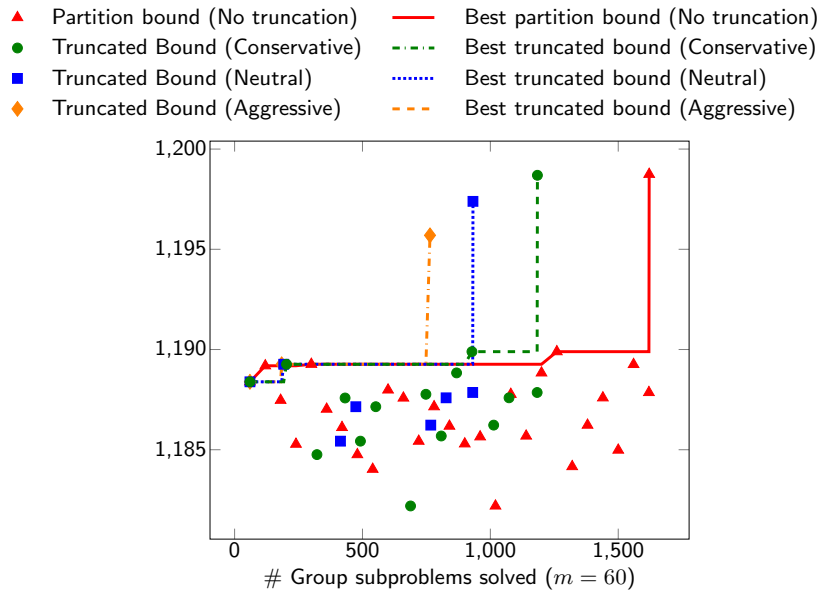
The method of recombining previously used scenario subsets into new, and possibly better partitions is described in Section 4.4.2. Figure 4.5 depicts the results of a computational experiment in which optimal recombination is attempted after every 10 partitions on a DCAP instance. It can be observed that the best partition bound is improved whenever a recombined partition can be obtained.

Even though optimal recombination of partitions have great potential to provide improved partition bounds, it is important to note that it may not always be possible to feasibly recombine the groups of existing partitions into a new partition. Clearly, the tree-aligned partitioning strategy provides some benefits in this area, since it inherently divides the scenario set into smaller subsets based on the scenario tree structure. (The successful recombinations while using a tree-aligned partitioning strategy can be observed in Figure 4.5.) However in other partitioning strategies, such as random sampling with no consideration of the scenario tree structure, the possibility to recombine the groups of existing partitions into a new partition may not be very likely. It is intuitive that smaller group sizes provide more opportunities for recombination, so the likelihood of obtaining a new recombined partition from the existing partitions is greater for small group sizes. We provide support for this intuition in Appendix C.4.

In cases where it is possible to generate recombined partitions from existing partitions, optimally recombined partition bounds tend to provide more improvement to the incumbent bound when the group size is small. This phenomenon can be observed in Figure 4.5, and explained with the observation that the distribution of partition bounds have higher variability for smaller group sizes.

As mentioned in Section 4.4.3, to further improve the efficiency of partition sampling, we suggest a truncated bound calculation strategy, where we cease the bound calculation for unpromising partitions part-way and start with a new partition.

Figure 4.7 demonstrates how bound truncation strategies affect the progression of

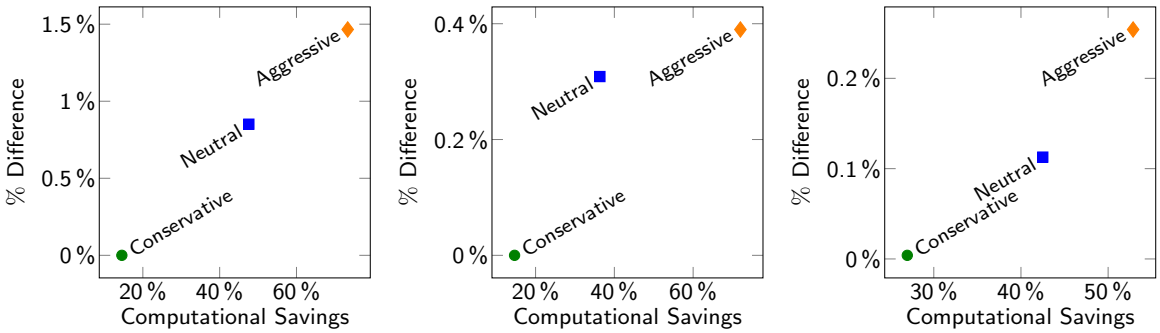


Note. The last partition bound of every truncation strategy belongs to an optimally recombined partition; and (α, β, γ) values of $(0.3, 0.02, 1.0)$, $(0.35, 0.03, 0.985)$ and $(0.4, 0.04, 0.96)$ are used to demonstrate *aggressive*, *neutral*, and *conservative* truncation strategies.

Figure 4.7: Partition bounds with different truncation strategies. DCAP 332 1x15x20, $q = 5$

the partition sampling algorithm on a DCAP instance with group size $q = 5$. It can be clearly seen that more aggressive truncation strategies result in less computations. Savings in computational effort comes at a cost of reduced bound quality in some cases, while in other cases bound quality remains the same. Figure 4.8 plots the savings in computational effort against the sacrifice in bound quality for different truncation strategies, where sacrifice in bound quality is expressed as the percentage difference between the bounds found using truncation strategies and the best partition bound without truncation.

Figures 4.8(a) - (b) and 4.8(c) show the tradeoff between computational savings and loss in bound quality for different truncation strategies on an SSLP instance and a DCAP instance, respectively. Using the conservative strategy, about 20% can be saved in computational effort while losing very little or nothing in terms of solution quality. Over 70% computational savings can be attained by eliminating partitions aggressively, while losing no more than 1.5% in bound quality. Similar results hold



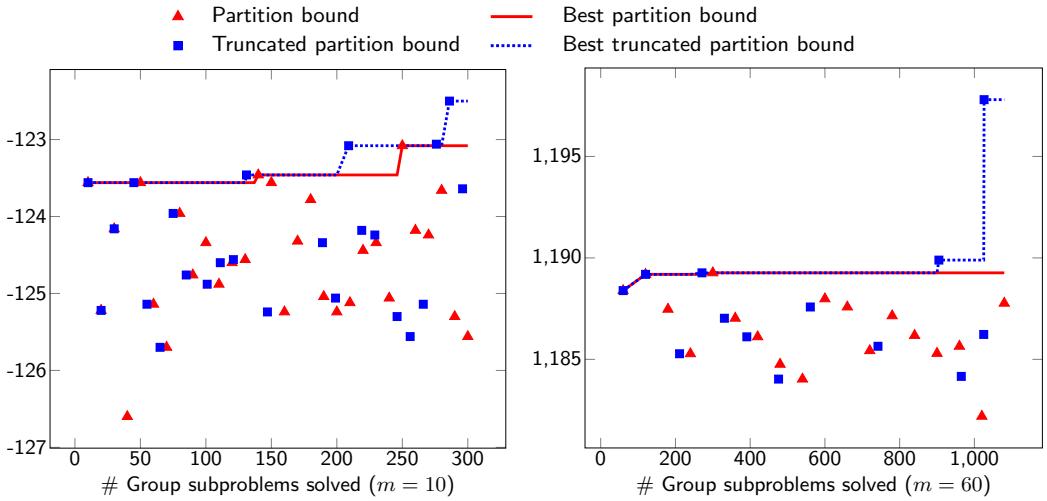
(a) SSLP 5-25-50, $q = 2$ **(b)** SSLP 5-25-50, $q = 5$ **(c)** DCAP 332 1x15x20, $q = 5$

Note. (α, β, γ) values of $(0.2, 0.02, 1.05)$, $(0.4, 0.04, 0.99)$ and $(0.5, 0.06, 0.95)$ are used to demonstrate *aggressive*, *neutral*, and *conservative* strategies for SSLP instances.

Figure 4.8: Computational savings vs. loss in bound quality for different truncation strategies

for other CAP, SSLP, and DCAP instances.

When truncation strategies are used, the saved computational effort can be used towards considering more partitions and therefore possibly improving the partition bound. To demonstrate this idea, we conduct experiments on some test instances, where new partitions are explored using the computational effort saved due to a conservative truncation strategy. The results of these experiments are presented in detail in Figure 4.9 for an SSLP and a DCAP instance. Figure 4.9(a) demonstrates an instance where 30 partitions ($30 \times 10 = 300$ group subproblems) are attempted originally, but the conservative truncation strategy results in solving only 256 group subproblems. The remaining computational effort, corresponding to 44 group subproblems, is used towards solving for 4 new partitions, one of which yielded a better partition bound than the best partition bound. A similar result can be observed in Figure 4.9(b). Clearly, it is not guaranteed that the truncation strategies will not eliminate a partition that would provide a better bound than the current best, or that the new partition bounds calculated using the saved computational effort will result in an improved partition bound. But the examples we provide substantiate the potential of the truncation approach to save computational effort or to yield better partition bounds.



(a) SSLP 5-25-50, $q = 5$ **(b)** DCAP 332 1x15x20, $q = 5$

Note. The last truncated partition bound on Figure (b) belongs to an optimally recombined partition.

Figure 4.9: Bound progression with conservative truncation vs. no truncation (same computational effort)

4.6 Conclusion

In this chapter, we focused on partition bounds, introduced expected partition bounds, and established that expected partition bounds form a hierarchy based on group size by drawing parallels between EP bounds and EGSO bounds introduced in [108].

We have shown that random sampling of partition bounds can be a simple, yet effective, approach to calculating dual bounds for SMIPs. On benchmark instances, it performs better than other sampling approaches for the same computational effort, and has the added benefit of providing a guaranteed bound, rather than one that is statistically likely. In practice, only a few partitions need to be sampled before the value of the best partition exceeds the expected value of group subproblems.

We extended the random partition sampling approach by introducing algorithmic enhancements: partition sampling based on scenario tree structure and optimally recombined partitions. We then introduced a heuristic method, "truncated bound calculation", for potentially improving computational performance of the algorithm. Finally, we demonstrated the effectiveness of these algorithmic enhancements in our

comprehensive computational study on a wide range of problem instances.

CHAPTER V

CONCLUSION

In this thesis, we studied emerging problems in the fields of fleet replacement and optimization under uncertainty. We explored various topics in empty repositioning, fleet replacement, and dual bounds in stochastic programming.

Naturally imbalanced freight flows force transportation service providers to reposition resources empty. When constructing empty repositioning plans, the reduction in lost (or outsourced) demand needs to be weighed against the cost of repositioning resources empty. This is especially challenging given the uncertainty of future demand. In Chapter 2, we proposed a robust optimization framework for fleet repositioning in a transportation network subject to demand uncertainty. We designed and implemented a robust rolling horizon framework for constructing effective empty repositioning plans. An extensive computational study demonstrates the benefits of explicitly accounting for uncertainty in future demand. Additionally, we provided robustness parameters specific to this problem for controlling the level of conservatism in the robust framework. We demonstrated with computational experiments that moderate values for these robustness parameters provide the best outcome in terms of service level, total transportation cost, and computational burden. Finally, we investigate practical strategies for reducing the complexity of managing the repositioning of empty resources in transportation service networks covering huge geographic areas.

In studying fleet management in transportation networks from a broader perspective, we then focused on fleet replacement strategies for introducing alternative fuel trucks (AFTs) into existing diesel fleets in long-haul trucking settings. In Chapter 3,

we concentrated on modeling fleet replacement decisions, while also taking into consideration structural changes to accommodate new fleet types and preserving feasible operations. We introduced a novel modeling framework that simultaneously decides on the number of trucks of each type to be purchased and retired at each period, terminals/periods at which maintenance facilities for AFTs should be opened, and the truck routes where the different fleet types should be deployed. The ability of the integrated model to capture interesting characteristics of the network and to make use of available information that is often overlooked by other fleet replacement strategies is demonstrated with computational experiments. In order to find optimal and good heuristic solutions to the integrated fleet management model, a Benders decomposition framework and a variable neighborhood search heuristic are introduced. The performance of these solution methodologies are demonstrated with a comprehensive computational study.

In Chapter 4, we focused on finding dual bounds in multistage stochastic programming problems. In this study, we defined "Expected Partition ($EP(q)$) bounds as the expected value of the partition bound over all partitions with group size q, and showed that they form a hierarchy based on the group size, q. By using the fact that we can obtain dual bounds with as few as a single partition of the scenario set, we presented a scenario sampling method that finds dual bounds by randomly sampling partitions of the scenario set. In order to improve the efficiency of this sampling method, we suggested two enhancements to the approach: sampling partitions that align with the multistage scenario tree structure, and use of an auxiliary optimization problem to discover new best bounds based on the values of group subproblems already computed. We established the effectiveness of these ideas with computational experiments on benchmark problems. Finally, we provided a heuristic to save computational effort by ceasing computation of a partition part-way through, if it appears unpromising.

The topics covered in this thesis can be extended in various research directions. Currently, there is a global trend in increasing the use of AFVs in order to reduce greenhouse gas emissions. For example, in the U.S., federal and state governments are working towards the goal of cutting petroleum use by 2.5 billion gallons per year by 2020 by regulating commercial fleets, as well as offering incentives and tax breaks. [71] Taking into account the government regulations and funding opportunities, along with the positive environmental and economic impacts, switching to AFVs is the logical course of action for various transportation agencies. The work in Chapter 3 of this thesis focuses on long-haul trucking, and it has overarching applications in many different environments, such as last-mile delivery, city buses, street sweepers, garbage trucks, school buses, airport and port ground operations. Extending this work is particularly important, since different fleet types require different sets of conditions to hold. For example, some of the unique considerations in city bus systems are *(i)* the need to operate in a rigid schedule, *(ii)* the presence of government incentives, and *(iii)* the availability of Online Electric Vehicle (OLEV) technology and en-route charging options. Addressing these new requirements would necessitate investigating various modeling choices and potentially invoke new solution methodologies.

Another natural extension to the integrated fleet management framework and related solution methodologies would be the the introduction of parameter uncertainty. This modeling framework has the potential to be useful in fleet replacement settings where there are uncertain parameters. It would be particularly interesting to consider uncertain fuel prices in order to find fleet replacement strategies that are robust to volatile fuel prices.

The ideas in scenario set partition bounds have the potential to be extended in several directions. For any partition, the partition bound can be viewed as the special case of a bound obtained by Lagrangian relaxation of the NACs between scenarios from different subsets of the partition: the partition bound is simply the case with

zero Lagrange multipliers. Any method that provides better Lagrange multipliers for the partition can be used to improve the partition bound. This suggests that sampling of partitions may be combined with Lagrangian relaxation methods, to alter both the partition and the Lagrange multipliers, in tandem.

The calculation of partition bounds is naturally amenable to parallel implementation, and effective parallel codes could be developed, in the future. These would be particularly helpful for cases in which the number of scenarios is very large.

Finally, we mention that more systematic re-sampling approaches could be explored. These might be designed, for example, so that new partitions re-use some previously solved group subproblems, while still exploring re-groupings of the scenarios via randomness. Another idea would be to record the degree of NAC violation in partition bound solutions, by scenario pair, and seek to group scenarios that exhibit large violations when assigned to different group subproblems in a partition.

APPENDIX A

SUPPLEMENTARY MATERIAL FOR CHAPTER II

A.1 Proofs of Propositions and Theorems

Proof (Proposition 1): By construction of the network and the associated net supply vector $\mathbf{I}$, a feasible flow in $\mathcal{G}_W(\mathbf{x}, \boldsymbol{\delta})$ defines a set of recovery moves $\mathbf{w} \in \mathbb{Z}^{|\mathcal{A}|}$ that would recover the feasibility of $\mathbf{x}$ after the realization of $\boldsymbol{\delta}$. For repositioning arcs $a \in \mathcal{A}_R$, $w(a)$ would be the flow on arc a minus $(x(a) + \delta(a))$, and for inventory arcs $a \in \mathcal{A}_I$, $w(a)$ is simply the flow on the corresponding arc in $\mathcal{G}_W(\mathbf{x}, \boldsymbol{\delta})$. Also, a feasible flow on $\mathcal{G}_W(\mathbf{x}, \boldsymbol{\delta})$ can be constructed using a feasible solution $\mathbf{x}$ that can be recovered using recovery actions $\mathbf{w}$ after perturbation $\boldsymbol{\delta}$. For $a \in \mathcal{A}_R$, the flow is $x(a) + \delta(a) + w(a)$; and for $a \in \mathcal{A}_I$, the flow is $w(a)$. □

Proof (Proposition 2): See [46], Proposition 2. □

Proof (Theorem 1): Let $\mathbf{x}$ be a feasible solution to **NP** and $\boldsymbol{\delta} \in \varphi_k$ be a perturbation vector. Then, by definition on $\mathbf{I}$,

$$\sum_{v \in U} I(v) = \sum_{a \in \Delta^{out}(U) \cup \mathcal{A}_I} x(a) - \sum_{\substack{i \in U \\ j \in \mathcal{N} \setminus U}} \delta(i,j) + \sum_{\substack{i \in \mathcal{N} \setminus U \\ j \in U}} \delta(i,j)$$

for every $U \in \mathcal{U}_W$. Thus, by Propositions 1 and 2, $\mathbf{x}$ is feasible for **RP** if and only if

$$\sum_{v \in U} I(v) = \sum_{a \in \Delta^{out}(U) \cup \mathcal{A}_I} x(a) - \sum_{\substack{i \in U \\ j \in \mathcal{N} \setminus U}} \delta(i,j) + \sum_{\substack{i \in \mathcal{N} \setminus U \\ j \in U}} \delta(i,j) \geq 0 \qquad \forall U \in \mathcal{U}_W \quad \text{(A.1)}$$

holds for each $\boldsymbol{\delta}$. Since $\boldsymbol{\delta} \in \varphi_k$, $\sum_{\substack{i \in U \\ j \in \mathcal{N} \setminus U}} \delta(i,j) - \sum_{\substack{i \in \mathcal{N} \setminus U \\ j \in U}} \delta(i,j)$ can be bounded:

$$\sum_{\substack{i\in U\\ j\in\mathcal{N}\setminus U}} \delta(i,j) - \sum_{\substack{i\in\mathcal{N}\setminus U\\ j\in U}} \delta(i,j) \leq \sum_{\substack{i\in U\\ j\in\mathcal{N}\setminus U}} kl^{+}(i,j) - \sum_{\substack{i\in\mathcal{N}\setminus U\\ j\in U}} -kl^{-}(i,j) = \vartheta(U,k)$$

Note that this bound is tight for at least one $\boldsymbol{\delta} \in \varphi_k$ (namely, $\delta(a) = l(a) + kl^{+}(a)$ for each $a \in \Delta^{out}(U)$, and $\delta(a) = l(a) - kl^{-}(a)$ for each $a \in \Delta^{in}(U)$).

Thus, condition (A.1) simplifies to

$$\sum_{a\in\Delta^{out}(U)\cup\mathcal{A}_I} x(a) \geq \vartheta(U,k) \qquad \forall U \in \mathcal{U}_W.$$

□

A.2 Detailed Computational Results

Table A.1: Data associated with Figure 2.1a

Robustness Characteristics	% Outsourcing	Transportation Cost	Solution Time (seconds)
Non-Robust	2.507%	1,252,846.400	261.294
Robust; $(R, N) = (2, 2)$	1.922%	1,260,953.866	248.504
Robust; $(R, N) = (4, 2)$	1.719%	1,262,446.133	250.722
Robust; $(R, N) = (6, 2)$	0.888%	1,271,464.133	253.241
Robust; $(R, N) = (8, 2)$	0.774%	1,273,461.066	252.775
Robust; $(R, N) = (10, 2)$	0.636%	1,276,436.600	253.428
Robust; $(R, N) = (16, 2)$	0.574%	1,278,341.000	256.996
Robust; $(R, N) = (20, 2)$	0.603%	1,277,732.000	263.060

Table A.2: Data associated with Figure 2.1b

Robustness Characteristics	% Outsourcing	Transportation Cost	Solution Time (seconds)
Non-Robust	2.658%	2,516,557.862	5,374.510
Robust; $(R, N) = (2, 2)$	1.095%	2,546,511.466	5,244.896
Robust; $(R, N) = (6, 2)$	0.641%	2,555,547.266	4,282.854
Robust; $(R, N) = (10, 2)$	0.597%	2,571,598.733	4,492.900

Table A.3: Data associated with Figure 2.1c

Robustness Characteristics	% Outsourcing	Transportation Cost	Solution Time (seconds)
Non-Robust	3.362%	3,404,887.866	9,824.288
Robust; $(R, N) = (2, 2)$	1.355%	3,440,173.000	10,241.292
Robust; $(R, N) = (4, 2)$	1.154%	3,445,266.400	10,369.630
Robust; $(R, N) = (6, 2)$	0.751%	3460,009.533	10,385.561

Table A.4: Data associated with Figure 2.2

Robustness Characteristics	% Outsourcing	Transportation Cost	Solution Time (seconds)
Non-Robust	2.5079%	1,252,846.400	261.294
Robust; $(R, N) = (6, 2)$	0.8887%	1,271,464.133	253.241
Robust; $(R, N) = (6, 3)$	0.8888%	1,271,560.466	313.371
Robust; $(R, N) = (6, 4)$	0.8623%	1,271,845.533	6,595.112
Robust; $(R, N) = (6, 5)$	0.8348%	1,272,501.400	176,915.330

Table A.5: Data associated with Figure 2.3a

Robustness Characteristics	k	% Outsourcing	Transportation Cost	Solution Time (seconds)
Non-Robust	N/A	2.507%	1,252,846.400	261.294
Robust; $(R, N) = (6, 3)$	1	0.195%	1,295,388.600	317.826
Robust; $(R, N) = (8, 3)$	1	0.164%	1,300,076.733	410.635
Robust; $(R, N) = (10, 3)$	1	0.065%	1,314,678.666	624.784
Robust; $(R, N) = (6, 3)$	$1/2$	0.476%	1,278,676.666	311.738
Robust; $(R, N) = (8, 3)$	$1/2$	0.412%	1,280,963.333	406.185
Robust; $(R, N) = (10, 3)$	$1/2$	0.260%	1,286,487.066	608.169
Robust; $(R, N) = (6, 3)$	$1/3$	0.888%	1,271,560.466	313.371
Robust; $(R, N) = (8, 3)$	$1/3$	0.769%	1,273,698.200	403.862
Robust; $(R, N) = (10, 3)$	$1/3$	0.629%	1,276,555.200	617.180
Robust; $(R, N) = (6, 3)$	$1/4$	1.200%	1,267,420.200	310.658
Robust; $(R, N) = (8, 3)$	$1/4$	1.108%	1,269,179.866	405.012
Robust; $(R, N) = (10, 3)$	$1/4$	0.955%	1,271,038.800	615.430

Table A.6: Data associated with Figure 2.3b

Robustness Characteristics	k	% Outsourcing	Transportation Cost
Non-Robust	N/A	3.362%	3,404,887.866
Robust; $(R, N) = (2, 2)$	$1/2$	1.317%	3,442,661.400
Robust; $(R, N) = (4, 2)$	$1/2$	1.088%	3,448,070.333
Robust; $(R, N) = (6, 2)$	$1/2$	0.238%	3,488,445.000
Robust; $(R, N) = (2, 2)$	$1/3$	1.355%	3,440,173.000
Robust; $(R, N) = (4, 2)$	$1/3$	1.154%	3,445,266.400
Robust; $(R, N) = (6, 2)$	$1/3$	0.751%	3,460,009.533
Robust; $(R, N) = (2, 2)$	$1/4$	1.717%	3,430,121.066
Robust; $(R, N) = (4, 2)$	$1/4$	1.500%	3,441,054.533
Robust; $(R, N) = (6, 2)$	$1/4$	1.129%	3,449,934.066

Table A.7: Data associated with Figure 2.5

Empty Network Structure	Robustness Characteristics	% Outsourcing	Transportation Cost
5 Sharing groups	Non-Robust	2.507%	1,252,846.400
	Robust; $(R, N) = (2, 2)$	1.922%	1,260,953.866
	Robust; $(R, N) = (4, 2)$	1.719%	1,262,446.133
	Robust; $(R, N) = (6, 2)$	0.888%	1,271,464.133
	Robust; $(R, N) = (8, 2)$	0.774%	1,273,461.066
	Robust; $(R, N) = (10, 2)$	0.636%	1,276,436.600
5 Sharing groups - balanced	Non-Robust	2.189%	1,184,865.066
	Robust; $(R, N) = (2, 2)$	1.232 %	1,195,691.800
	Robust; $(R, N) = (4, 2)$	1.108%	1,195,410.000
	Robust; $(R, N) = (6, 2)$	0.988%	1,197,751.866
	Robust; $(R, N) = (8, 2)$	0.525 %	1,208,772.866
	Robust; $(R, N) = (10, 2)$	0.469%	1,209,949.800

Table A.8: Data associated with Figure 2.6

Empty Network Structure	Robustness Characteristics	% Outsourcing	Transportation Cost
3 Sharing groups - balanced	Non-Robust	2.610%	1,232,041.066
	Robust; $(R, N) = (4, 2)$	2.276%	1,223,992.000
	Robust; $(R, N) = (6, 2)$	1.329%	1,249,781.066
	Robust; $(R, N) = (8, 2)$	0.918%	1,260,972.933
	Robust; $(R, N) = (10, 2)$	0.778%	1,267,246.933
5 Sharing groups - balanced	Non-Robust	2.189%	1,184,865.066
	Robust; $(R, N) = (4, 2)$	1.108%	1,195,410.000
	Robust; $(R, N) = (6, 2)$	0.988%	1,197,751.866
	Robust; $(R, N) = (8, 2)$	0.525%	1,208,772.866
	Robust; $(R, N) = (10, 2)$	0.469%	1,209,949.800
7 Sharing groups - balanced	Non-Robust	2.371%	1,189,464.533
	Robust; $(R, N) = (4, 2)$	1.713%	1,194,920.600
	Robust; $(R, N) = (6, 2)$	0.952%	1,207,463.866
	Robust; $(R, N) = (8, 2)$	0.578%	1,216,283.200
	Robust; $(R, N) = (10, 2)$	0.546%	1,219,565.200

APPENDIX B

SUPPLEMENTARY MATERIAL FOR CHAPTER III

B.1 Detailed Computational Results

Table B.1: Problem instances used in computational experiments

Maint. Fac. Cost	Truck Costs	AFT Targets	Initial Fleet	Demand
Expensive	Expensive AFT	Constant	No	Constant
Expensive	Expensive AFT	Constant	Yes	Constant
Expensive	Expensive AFT	Increasing	Yes	Constant
Expensive	Expensive AFT	Constant	Yes	Region Swap
Expensive	Expensive AFT	Constant	Yes	Increasing
Expensive	Expensive AFT	Constant	Yes	Volatile
Expensive	Cheap AFT	Constant	No	Constant
Expensive	Cheap AFT	Constant	Yes	Constant
Expensive	Cheap AFT	Increasing	Yes	Constant
Expensive	Cheap AFT	Constant	Yes	Region Swap
Expensive	Cheap AFT	Constant	Yes	Increasing
Expensive	Cheap AFT	Constant	Yes	Volatile
Expensive	Changing	Constant	No	Constant
Expensive	Changing	Constant	Yes	Constant
Expensive	Changing	Increasing	Yes	Constant
Expensive	Changing	Constant	Yes	Region Swap
Cheap	Expensive AFT	Constant	No	Constant
Cheap	Expensive AFT	Constant	Yes	Constant
Cheap	Expensive AFT	Increasing	Yes	Constant
Cheap	Expensive AFT	Constant	Yes	Region Swap
Cheap	Expensive AFT	Constant	Yes	Increasing
Cheap	Expensive AFT	Constant	Yes	Volatile
Cheap	Cheap AFT	Constant	No	Constant
Cheap	Cheap AFT	Constant	Yes	Constant
Cheap	Cheap AFT	Increasing	Yes	Constant
Cheap	Cheap AFT	Constant	Yes	Region Swap
Cheap	Cheap AFT	Constant	Yes	Increasing
Cheap	Cheap AFT	Constant	Yes	Volatile
Cheap	Changing	Constant	No	Constant
Cheap	Changing	Constant	Yes	Constant
Cheap	Changing	Increasing	Yes	Constant
Cheap	Changing	Constant	Yes	Region Swap

Expensive maintenance facilities: $C_{jt} = 50P_t^D + \sum_{\tau=t}^{T} P_\tau^D + \epsilon_j$,
Cheap maintenance facilities: $C_{jt} = P_t^D + \sum_{\tau=t}^{T} P_\tau^D + \epsilon_j$
Cheap AFT: $P_t^A = 1.475P_t^D$, $c_{\tau t}^A = 0.8c_{\tau t}^D$, Expensive AFT: $P_t^A = 1.478P_t^D$,
$c_{\tau t}^A = 0.8c_{\tau t}^D$,
Changing truck costs: $P_t^A = 1.5P_t^D$, $c_{\tau t}^A/c_{\tau t}^D := 0.972 \rightarrow 0.785$
Constant AFT targets: 0.3, Increasing AFT targets: $0.3 \rightarrow 0.8$

Table B.2: Solution times in 10-terminal instances: Direct solution (OPT) vs. Benders' variants

Maint. Fac. Cost	Truck Costs	AFT Targets	Initial Fleet	Demand	OPT	BEN-0	BEN-1	BEN-2	BEN-3	BEN-4	#(BEN <OPT)
Expensive	Expensive AFT	Constant	No	Constant	16.09	49.71	98.55	94.15	103.19	21.44	0
Expensive	Expensive AFT	Constant	Yes	Constant	125.91	16.94	22.59	102.78	400.09	103.09	4
Expensive	Expensive AFT	Increasing	Yes	Constant	209.87	1271.61	1317.45	1468.13	1526.06	733.62	0
Expensive	Expensive AFT	Constant	Yes	Region Swap	121.45	16.80	23.45	91.89	87.59	92.36	5
Expensive	Expensive AFT	Constant	Yes	Increasing	140.73	91.35	132.45	104.84	48.59	32.80	5
Expensive	Expensive AFT	Constant	Yes	Volatile	144.96	62.52	39.66	39.19	570.13	183.33	3
Expensive	Cheap AFT	Constant	No	Constant	100.11	206.28	360.34	165.05	150.38	169.72	0
Expensive	Cheap AFT	Constant	Yes	Constant	226.89	79.87	231.21	199.72	235.05	204.13	3
Expensive	Cheap AFT	Increasing	Yes	Constant	2263.80	749.10	1636.10	1970.21	1539.27	1928.13	5
Expensive	Cheap AFT	Constant	Yes	Region Swap	132.98	77.67	190.69	179.48	174.36	177.84	1
Expensive	Cheap AFT	Constant	Yes	Increasing	323.72	190.93	297.09	312.60	319.03	306.24	5
Expensive	Cheap AFT	Constant	Yes	Volatile	35.37	62.99	194.73	180.86	204.49	183.37	0
Expensive	Changing	Constant	No	Constant	1240.84	65.98	133.56	139.69	144.06	138.57	5
Expensive	Changing	Constant	Yes	Constant	350.13	280.61	190.99	187.71	186.85	191.54	5
Expensive	Changing	Increasing	Yes	Constant	1664.22	2099.96	2650.91	3086.30	3296.56	2913.26	0
Expensive	Changing	Constant	Yes	Region Swap	1440.20	1546.49	216.50	443.93	210.47	265.26	4
Cheap	Expensive AFT	Constant	No	Constant	26.55	142.96	374.57	396.30	400.35	396.04	0
Cheap	Expensive AFT	Constant	Yes	Constant	98.03	187.39	475.54	465.62	507.19	455.42	0
Cheap	Expensive AFT	Increasing	Yes	Constant	4852.49	4203.07	9016.39	9000.16	17129.36	5166.59	1
Cheap	Expensive AFT	Constant	Yes	Region Swap	148.24	94.86	163.19	167.96	194.51	247.87	1
Cheap	Expensive AFT	Constant	Yes	Increasing	113.33	870.50	832.52	704.33	815.87	719.18	0
Cheap	Expensive AFT	Constant	Yes	Volatile	177.20	120.31	337.34	332.48	315.63	347.45	1
Cheap	Cheap AFT	Constant	No	Constant	108.64	62.50	406.50	347.28	349.56	354.59	1
Cheap	Cheap AFT	Constant	Yes	Constant	630.12	490.42	1245.47	1331.03	1333.29	1045.59	1
Cheap	Cheap AFT	Increasing	Yes	Constant	2397.09	5456.12	8817.30	6547.95	6511.95	6397.37	0
Cheap	Cheap AFT	Constant	Yes	Region Swap	707.70	529.58	2005.79	1829.67	1912.95	1668.11	1
Cheap	Cheap AFT	Constant	Yes	Increasing	2287.09	11343.54	7967.07	9222.27	10444.48	8367.91	0
Cheap	Cheap AFT	Constant	Yes	Volatile	710.17	4872.37	4700.57	3714.33	3847.28	4467.57	0
Cheap	Changing	Constant	No	Constant	625.02	1188.40	2506.48	188.64	249.19	303.83	3
Cheap	Changing	Constant	Yes	Constant	376.29	250.99	351.83	387.16	383.31	387.91	2
Cheap	Changing	Increasing	Yes	Constant	19467.71	4786.67	12474.85	11241.55	20778.20	13750.60	4
Cheap	Changing	Constant	Yes	Region Swap	452.27	3413.96	1593.91	1436.22	1630.37	1589.61	0
				Average	**1303.60**	**1402.58**	**1906.42**	**1752.48**	**2374.99**	**1665.95**	**1.88**

OPT, BEN-0, BEN-1, BEN-2, BEN-3, and BEN-4 represent the direct solution approach, Benders' decomposition with no Pareto-optimal cuts, with Parato-optimal cuts using core points $\boldsymbol{y_1^0}$, $\boldsymbol{y_2^0}$, $\boldsymbol{y_3^0}$, and $\boldsymbol{y_4^{0,k}}$, respectively.

Table B.3: Number of Benders' iterations in 10 terminal instances

Maint. Fac. Cost	Truck Costs	AFT Targets	Initial Fleet	Demand	BEN-0	BEN-1	BEN-2	BEN-3	BEN-4
Expensive	Expensive AFT	Constant	No	Constant	6	4	4	4	4
Expensive	Expensive AFT	Constant	Yes	Constant	6	4	4	4	4
Expensive	Expensive AFT	Increasing	Yes	Constant	76	67	73	71	63
Expensive	Expensive AFT	Constant	Yes	Region Swap	5	4	4	4	4
Expensive	Expensive AFT	Constant	Yes	Increasing	8	4	5	2	5
Expensive	Expensive AFT	Constant	Yes	Volatile	8	7	7	7	7
Expensive	Cheap AFT	Constant	No	Constant	9	9	9	9	9
Expensive	Cheap AFT	Constant	Yes	Constant	10	9	9	10	9
Expensive	Cheap AFT	Increasing	Yes	Constant	81	74	76	71	75
Expensive	Cheap AFT	Constant	Yes	Region Swap	9	7	6	6	6
Expensive	Cheap AFT	Constant	Yes	Increasing	17	9	9	9	9
Expensive	Cheap AFT	Constant	Yes	Volatile	8	7	7	7	7
Expensive	Changing	Constant	No	Constant	9	9	9	9	9
Expensive	Changing	Constant	Yes	Constant	9	10	9	9	9
Expensive	Changing	Increasing	Yes	Constant	170	116	118	120	122
Expensive	Changing	Constant	Yes	Region Swap	132	7	6	6	8
Cheap	Expensive AFT	Constant	No	Constant	6	5	5	5	5
Cheap	Expensive AFT	Constant	Yes	Constant	6	5	5	5	5
Cheap	Expensive AFT	Increasing	Yes	Constant	233	164	169	158	163
Cheap	Expensive AFT	Constant	Yes	Region Swap	5	3	3	3	4
Cheap	Expensive AFT	Constant	Yes	Increasing	34	14	13	14	13
Cheap	Expensive AFT	Constant	Yes	Volatile	8	7	7	7	7
Cheap	Cheap AFT	Constant	No	Constant	18	12	9	10	11
Cheap	Cheap AFT	Constant	Yes	Constant	34	23	24	24	20
Cheap	Cheap AFT	Increasing	Yes	Constant	262	190	187	184	181
Cheap	Cheap AFT	Constant	Yes	Region Swap	39	35	33	32	33
Cheap	Cheap AFT	Constant	Yes	Increasing	394	134	126	139	122
Cheap	Cheap AFT	Constant	Yes	Volatile	222	72	73	75	90
Cheap	Changing	Constant	No	Constant	33	15	15	14	15
Cheap	Changing	Constant	Yes	Constant	32	16	17	17	17
Cheap	Cheap AFT	Increasing	Yes	Constant	420	321	323	327	319
Cheap	Changing	Constant	Yes	Region Swap	267	55	51	49	57
				Average	**80.50**	**44.31**	**44.22**	**44.09**	**44.13**

OPT, BEN-0, BEN-1, BEN-2, BEN-3, and BEN-4 represent the direct solution approach, Benders' decomposition with no Pareto-optimal cuts, with Parato-optimal cuts using core points $\boldsymbol{y_1^0}$, $\boldsymbol{y_2^0}$, $\boldsymbol{y_3^0}$, and $\boldsymbol{y_4^{0,k}}$, respectively.

Table B.4: Optimality gaps after 1 hour in 30-terminal instances for Gurobi (OPT) and Benders' variants

Maint. Fac. Cost	Truck Costs	AFT Targets	Initial Fleet	Demand	OPT	BEN-0	BEN-1	BEN-2	BEN-3	BEN-4	#(BEN <OPT)
Expensive	Expensive AFT	Constant	No	Constant	0.000%	0.010%	0.000%	0.000%	0.000%	0.000%	4
Expensive	Expensive AFT	Constant	Yes	Constant	0.000%	0.011%	0.000%	0.000%	0.000%	0.000%	4
Expensive	Expensive AFT	Increasing	Yes	Constant	3.854%	N/A	N/A	16.093%	N/A	N/A	0
Expensive	Expensive AFT	Constant	Yes	Region Swap	0.000%	0.166%	0.000%	0.000%	0.000%	0.000%	4
Expensive	Expensive AFT	Constant	Yes	Increasing	0.000%	0.030%	0.000%	0.000%	0.000%	0.000%	4
Expensive	Expensive AFT	Constant	Yes	Volatile	0.000%	0.165%	0.000%	0.000%	0.000%	0.000%	4
Expensive	Cheap AFT	Constant	No	Constant	2.046%	0.165%	0.000%	0.000%	0.000%	0.000%	5
Expensive	Cheap AFT	Constant	Yes	Constant	0.000%	0.000%	0.000%	0.000%	0.182%	0.000%	4
Expensive	Cheap AFT	Increasing	Yes	Constant	4.721%	N/A	N/A	N/A	N/A	N/A	0
Expensive	Cheap AFT	Constant	Yes	Region Swap	0.106%	0.478%	0.106%	0.106%	0.106%	0.106%	4
Expensive	Cheap AFT	Constant	Yes	Increasing	0.051%	0.127%	0.127%	0.127%	0.127%	0.127%	0
Expensive	Cheap AFT	Constant	Yes	Volatile	0.173%	0.553%	0.173%	0.173%	0.173%	0.173%	4
Cheap	Expensive AFT	Constant	No	Constant	0.007%	0.010%	0.000%	0.000%	0.000%	0.000%	4
Cheap	Expensive AFT	Constant	Yes	Constant	0.000%	0.011%	0.000%	0.000%	0.000%	0.000%	4
Cheap	Expensive AFT	Increasing	Yes	Constant	1.373%	8.232%	8.232%	8.232%	8.232%	8.232%	0
Cheap	Expensive AFT	Constant	Yes	Region Swap	0.000%	0.168%	0.000%	0.096%	0.096%	0.096%	1
Cheap	Expensive AFT	Constant	Yes	Increasing	0.000%	0.031%	0.000%	0.000%	0.000%	0.000%	4
Cheap	Expensive AFT	Constant	Yes	Volatile	0.086%	0.253%	0.086%	0.086%	0.086%	0.086%	4
Cheap	Cheap AFT	Constant	No	Constant	0.411%	0.738%	0.160%	0.288%	0.261%	0.430%	3
Cheap	Cheap AFT	Constant	Yes	Constant	0.588%	0.775%	0.049%	0.049%	0.153%	0.153%	4
Cheap	Cheap AFT	Increasing	Yes	Constant	1.559%	N/A	N/A	N/A	N/A	N/A	0
Cheap	Cheap AFT	Constant	Yes	Region Swap	0.844%	0.897%	0.155%	0.155%	0.155%	0.155%	4
Cheap	Cheap AFT	Constant	Yes	Increasing	0.849%	0.924%	0.449%	0.518%	0.328%	0.324%	4
Cheap	Cheap AFT	Constant	Yes	Volatile	0.020%	1.066%	0.320%	0.684%	0.320%	0.684%	0
				Average	**0.695%**	**0.705%**	**0.469%**	**1.209%**	**0.487%**	**0.503%**	**2.88**

Optimality gaps are calculated with respect to the best bound found in all solution methods (OPT and all BEN variants).
The cells with "N/A" represent instances where the given solution approach could not find a feasible solution in 1 hour.

Table B.5: VNS Optimality gaps and solution time (seconds) performance in 10-terminal instances

Maint. Fac. Cost	Truck Costs	AFT Targets	Initial Fleet	Demand	OPT Obj.	VNS Obj.	OPT Time	VNS Time	VNS Opt. Gap	VNS % time saved
Expensive	Expensive AFT	Constant	No	Constant	118506.1	118506.1	54.42	109.67	0.000%	<0
Expensive	Expensive AFT	Constant	Yes	Constant	104499.2	104499.2	87.09	72.83	0.000%	16.4%
Expensive	Expensive AFT	Increasing	Yes	Constant	112208.8	112214.6	1303.80	466.22	0.005%	64.2%
Expensive	Expensive AFT	Constant	Yes	Region Swap	96193.7	96203.1	62.86	75.72	0.010%	<0
Expensive	Expensive AFT	Constant	Yes	Increasing	228527.7	228540.2	139.26	85.38	0.005%	38.7%
Expensive	Expensive AFT	Constant	Yes	Volatile	153794.3	153794.3	86.16	150.43	0.000%	<0
Expensive	Cheap AFT	Constant	No	Constant	120262.8	120262.8	217.57	112.39	0.000%	48.3%
Expensive	Cheap AFT	Constant	Yes	Constant	106259.0	106259.0	121.94	85.42	0.000%	30.0%
Expensive	Cheap AFT	Increasing	Yes	Constant	112631.1	112664.4	1295.04	325.07	0.029%	74.9%
Expensive	Cheap AFT	Constant	Yes	Region Swap	97707.7	97707.7	352.88	81.25	0.000%	77.0%
Expensive	Cheap AFT	Constant	Yes	Increasing	232220.0	232220.0	355.51	265.26	0.000%	25.4%
Expensive	Cheap AFT	Constant	Yes	Volatile	156661.1	156661.1	579.19	177.67	0.000%	69.3%
Expensive	Changing	Constant	No	Constant	124374.6	124374.6	1407.27	401.67	0.000%	71.5%
Expensive	Changing	Constant	Yes	Constant	110355.8	110355.8	2226.62	484.27	0.000%	78.3%
Expensive	Changing	Increasing	Yes	Constant	114341.8	114548.4	1792.51	375.37	0.181%	79.1%
Expensive	Changing	Constant	Yes	Region Swap	100957.9	100957.9	345.32	84.04	0.000%	75.7%
Cheap	Expensive AFT	Constant	No	Constant	113116.1	113116.1	26.55	53.81	0.000%	<0
Cheap	Expensive AFT	Constant	Yes	Constant	99109.2	99109.2	98.03	231.70	0.000%	<0
Cheap	Expensive AFT	Increasing	Yes	Constant	101477.1	101482.8	4852.49	422.16	0.006%	91.3%
Cheap	Expensive AFT	Constant	Yes	Region Swap	90803.7	90805.9	148.24	148.36	0.002%	<0
Cheap	Expensive AFT	Constant	Yes	Increasing	223137.7	223137.7	113.33	190.13	0.000%	<0
Cheap	Expensive AFT	Constant	Yes	Volatile	148404.3	148404.3	177.20	610.00	0.000%	<0
Cheap	Cheap AFT	Constant	No	Constant	114764.0	114872.8	108.64	81.96	0.095%	24.6%
Cheap	Cheap AFT	Constant	Yes	Constant	100738.1	100869.0	630.12	250.56	0.130%	60.2%
Cheap	Cheap AFT	Increasing	Yes	Constant	101899.4	102272.6	2397.09	200.53	0.366%	91.6%
Cheap	Cheap AFT	Constant	Yes	Region Swap	92258.6	92317.7	707.70	393.94	0.064%	44.3%
Cheap	Cheap AFT	Constant	Yes	Increasing	226395.2	226830.0	2287.09	296.56	0.192%	87.0%
Cheap	Cheap AFT	Constant	Yes	Volatile	151172.0	151324.8	710.17	133.87	0.101%	81.1%
Cheap	Changing	Constant	No	Constant	115516.2	116887.7	625.02	678.25	1.187%	<0
Cheap	Changing	Constant	Yes	Constant	101471.7	102571.4	376.29	573.07	1.084%	<0
Cheap	Changing	Increasing	Yes	Constant	102713.0	103816.6	19467.71	574.37	1.075%	97.0%
Cheap	Changing	Constant	Yes	Region Swap	93217.6	95567.9	452.27	164.55	2.521%	63.6%
						Average	**1362.67**	**261.14**	**0.220%**	**18.2%**

Table B.6: VNS Optimality gaps and solution time (seconds) performance in 30-terminal instances

Maint. Fac. Cost	Truck Costs	AFT Targets	Initial Fleet	Demand	VNS Obj.	Best Bound	VNS Time	VNS Opt. Gap	Direct Sol. Opt. Gap (1 hr)	Better
Expensive	Expensive AFT	Constant	No	Constant	652391.1	652391.1	395.74	0.000%	0.000%	VNS
Expensive	Expensive AFT	Constant	Yes	Constant	577748.7	577748.7	144.18	0.000%	0.000%	OPT
Expensive	Expensive AFT	Increasing	Yes	Constant	596137.8	595492.7	4847.25	0.108%	3.085%	VNS
Expensive	Expensive AFT	Constant	Yes	Region Swap	630216.6	630216.6	351.66	0.000%	0.000%	VNS
Expensive	Expensive AFT	Constant	Yes	Increasing	801703.0	801703.0	411.39	0.000%	0.000%	VNS
Expensive	Expensive AFT	Constant	Yes	Volatile	604707.8	604032.7	123.75	0.112%	0.000%	OPT
Expensive	Cheap AFT	Constant	No	Constant	664908.4	663834.1	220.65	0.162%	2.046%	VNS
Expensive	Cheap AFT	Constant	Yes	Constant	590266.0	589191.7	517.03	0.182%	0.000%	OPT
Expensive	Cheap AFT	Increasing	Yes	Constant	599712.1	594186.9	2681.45	0.930%	3.694%	VNS
Expensive	Cheap AFT	Constant	Yes	Region Swap	644928.1	642308.2	1694.84	0.408%	0.087%	OPT
Expensive	Cheap AFT	Constant	Yes	Increasing	819964.0	819544.9	476.41	0.051%	0.051%	OPT
Expensive	Cheap AFT	Constant	Yes	Volatile	619369.3	616831.7	295.54	0.411%	0.173%	OPT
Cheap	Expensive AFT	Constant	No	Constant	647001.1	647001.1	1584.22	0.000%	0.007%	VNS
Cheap	Expensive AFT	Constant	Yes	Constant	572358.7	572358.7	2525.47	0.000%	0.000%	OPT
Cheap	Expensive AFT	Increasing	Yes	Constant	580064.2	579449.4	6797.80	0.106%	1.085%	VNS
Cheap	Expensive AFT	Constant	Yes	Region Swap	624826.6	624826.6	1205.11	0.000%	0.000%	VNS
Cheap	Expensive AFT	Constant	Yes	Increasing	796313.0	796313.0	335.26	0.000%	0.000%	VNS
Cheap	Expensive AFT	Constant	Yes	Volatile	598642.7	598126.4	2798.00	0.086%	0.086%	OPT
Cheap	Cheap AFT	Constant	No	Constant	659518.4	654768.2	219.86	0.725%	0.397%	OPT
Cheap	Cheap AFT	Constant	Yes	Constant	584876.0	580367.5	334.70	0.777%	0.588%	OPT
Cheap	Cheap AFT	Increasing	Yes	Constant	583650.5	580408.9	8464.80	0.558%	1.574%	VNS
Cheap	Cheap AFT	Constant	Yes	Region Swap	639538.1	634352.3	285.41	0.817%	0.811%	OPT
Cheap	Cheap AFT	Constant	Yes	Increasing	811137.7	808599.8	2447.23	0.314%	0.733%	VNS
Cheap	Cheap AFT	Constant	Yes	Volatile	610664.6	608321.8	3454.22	0.385%	0.020%	OPT
						Average	**1775.5**	**0.256%**	**0.602%**	

APPENDIX C

SUPPLEMENTARY MATERIAL FOR CHAPTER IV

C.1 Preliminary Lemmas and Theorems

Lemma C.1 (Sandıkçı et al. [108], Lemma 1)**.** *Given integers q_1 and q_2 with $1 \leq q_1 \leq q_2 \leq L = |\mathcal{S}|$ and and a set $S \subseteq \Omega_{q_2}$,*

$$\sum_{S' \in \Omega_{q_1} \cap 2^S} \rho(S') = \tbinom{q_2-1}{q_1-1} \rho(S).$$

Lemma C.2 ([108], Lemma 2)**.** *Given integers q_1, q_2 and q_3 with $1 \leq q_1 \leq q_2 \leq q_3 \leq L = |\mathcal{S}|$, a set $S \subseteq \Omega_{q_3}$, and a function $v(\cdot) : \Omega_{q_1} \cap 2^S \to \mathbb{R}$,*

$$\sum_{S' \in \Omega_{q_2} \cap 2^S} \sum_{S'' \in \Omega_{q_1} \cap 2^{S'}} \rho(S'')v(S'') = \begin{pmatrix} q_3 - q_1 \\ q_2 - q_1 \end{pmatrix} \sum_{\hat{S} \in \Omega_{q_1} \cap 2^S} \rho(\hat{S})v(\hat{S}).$$

Proof of the first two lemmas from [108] follow in a straightforward way from counting arguments. For example, if $\gamma(s)$ denotes any quantity dependent on scenario $s \in \mathcal{S}$, and $\mathcal{C}$ is any collection of subsets of $\mathcal{D} \subseteq \mathcal{S}$, then

$$\sum_{S \subseteq \mathcal{C}} \sum_{s \in S} \gamma(s) = \sum_{s \in \mathcal{D}} \eta(\mathcal{C}, s)\gamma(s),$$

where $\eta(\mathcal{C}, s)$ is the number of sets in the collection $\mathcal{C}$ that contain s. In the case that $\mathcal{C} = \Omega_q$ and $\mathcal{D} = \mathcal{S}$, then $\eta(\mathcal{C}, s) = \binom{L-1}{q-1}$ for all s, since for each s, this is the number of ways that the remaining $q-1$ scenarios needed to form a subset of size q containing s can be chosen from the L scenarios other than s. From such arguments and from definitions the following results hold. Note that 2^S denotes the power set of S, *i.e.*, the set of all subsets of S.

Corollary C.1 ([108], Corollary 1)**.** *Given integers q_1, q_2 with $1 \leq q_1 \leq q_2 \leq L = |\mathcal{S}|$, a set $S \subseteq \Omega_{q_2}$, and a function $v(\cdot) : \Omega_1 \cap 2^S \to \mathbb{R}$,*

$$\sum_{S' \in \Omega_{q_1} \cap 2^S} \sum_{s \in S'} p_s v(\{s\}) = \begin{pmatrix} q_2 - 1 \\ q_1 - 1 \end{pmatrix} \sum_{s \in S} p_s v(\{s\}).$$

The following result requires the observation that if $((x_t^s)_{t=1}^T)_{s \in S}$ is feasible for $GR(S)$, where $S \subseteq \mathcal{S}$, then $((x_t^s)_{t=1}^T)_{s \in S'}$ is feasible for $GR(S')$ for any proper subset $S' \subseteq S$. This is obviously true in the multistage case by inspection of the definition of the group subproblem.

Lemma C.3 ([108], Lemma 3)**.** *Given an integer q with $1 \leq q \leq L = |\mathcal{S}|$ and a set $S \subseteq \Omega_q$,*

$$(q-1)\rho(S)v_{GR}(S) \geq \sum_{\hat{S} \in \Omega_{q-1} \cap 2^S} \rho(\hat{S}) v_{GR}(\hat{S}).$$

Proof. Let $((\tilde{x}_t^s)_{t=1}^T)s \in S$ denote an optimal solution to $GR(S)$. Then it is obvious from the group subproblem definition that, for any $S' \in \Omega_{q-1} \cap 2^S$, $((\tilde{x}_t^s)_{t=1}^T)s \in S'$ is feasible for $GR(S')$ and thus

$$\frac{1}{\rho(S')} \sum_{s \in S'} p_s \sum_{t=1}^T F_t^s(\tilde{x}_t^s) \geq z_{GR}(S').$$

Multiplying both sides by $\rho(S')$ and summing over all $S' \in \Omega_{q-1} \cap 2^S$, we obtain

$$\sum_{S' \in \Omega_{q-1} \cap 2^S} \sum_{s \in S'} p_s \sum_{t=1}^T F_t^s(\tilde{x}_t^s) \geq \sum_{S' \in \Omega_{q-1} \cap 2^S} \rho(S') z_{GR}(S').$$

Applying Corollary C.1 to the left-hand side of this inequality, with $q_2 = q$, $q_1 = q-1$, and $v(\{s\}) = \sum_{t=1}^T F_t^s(\tilde{x}_t^s)$, we obtain

$$(q-1) \sum_{s \in S} p_s \sum_{t=1}^T F_t^s(\tilde{x}_t^s) = (q-1)\rho(S)v_{GR}(S) \geq \sum_{S' \in \Omega_{q-1} \cap 2^S} \rho(S') z_{GR}(S'),$$

as required. □

Proof. ***[Theorem 4.1.]*** $z_{WS} = EGSO(1)$ and $EGSO(L) = z_{MSP}$ follow immediately from definitions and earlier observations. Now consider any integer q with $1 \leq q \leq L-1$. For any $S \in \Omega_{q+1}$, we have, from Lemma C.3, that

$$q\rho(S)v_{GR}(S) \geq \sum_{\hat{S}\in\Omega_q\cap 2^S} \rho(\hat{S})v_{GR}(\hat{S}).$$

Summing over all $S \in \Omega_{q+1}$ yields

$$q \sum_{S\in\Omega_{q+1}} \rho(S)v_{GR}(S) \geq \sum_{S\in\Omega_{q+1}} \sum_{\hat{S}\in\Omega_q\cap 2^S} \rho(\hat{S})v_{GR}(\hat{S}),$$

which, by the definition of $EGSO(q+1)$, implies

$$q\binom{L-1}{q} EGSO(q+1) \geq \sum_{S\in\Omega_{q+1}} \sum_{\hat{S}\in\Omega_q\cap 2^S} \rho(\hat{S})v_{GR}(\hat{S}).$$

Applying Lemma C.2 with $q_3 = L$, $q_2 = q+1$ and $q_1 = q$ to the right-hand side of this inequality, we obtain

$$q\binom{L-1}{q} EGSO(q+1) \geq \begin{pmatrix} L-q \\ 1 \end{pmatrix} \sum_{\hat{S}\in\Omega_q} \rho(\hat{S})v_{GR}(\hat{S}) = (L-q)\binom{L-1}{q-1} EGSO(q),$$

which, noting $L-q \geq 1$, is precisely $EGSO(q+1) \geq EGSO(q)$, as required. □

Proof. ***[Lemma 4.1.]*** Define $p, r \in \mathbb{Z}_+$ so that $L = pq + r$, and $r < q$, *i.e.*, let $p = \lfloor \frac{L}{q} \rfloor$ and $r = L - q\lfloor \frac{L}{q} \rfloor = L - qp$. Recall from Definition 4.1 that $m' = q\lceil \frac{L}{q} \rceil - L$ and $m = \lceil \frac{L}{q} \rceil - m'$.

When $q|L$, $m' = 0$ and $m = \frac{L}{q} = p \geq 1$ (since $q \leq L$ is assumed).

Otherwise, when $q \nmid L$, $\lceil \frac{L}{q} \rceil = \lfloor \frac{L}{q} \rfloor + 1 = p + 1$, so

$$m' = q\left\lceil \frac{L}{q} \right\rceil - L = q(p+1) - (pq+r) = q - r \quad \text{and} \quad m = \left\lceil \frac{L}{q} \right\rceil - m' = p + 1 - (q-r).$$

In this case, $m \geq 1$ if and only if

$$p+1-(q-r) \geq 1 \iff p \geq q-r \iff pq+r \geq (q-r)q+r \iff L \geq (q-r)q+r.$$

□

C.2 Test Instances

The stochastic capacitated facility location problem (CAP) considered in this paper is described in [83]. The first stage decisions determine which facilities to open, and the second stage decisions give the fraction of customer demand to be satisfied by each open facility. Given that y_i denotes whether or not facility i is opened, and x_{ij} denotes the amount of flow from facility i to customer j, the formulation is as follows:

$$
\begin{array}{lll}
\min & \sum_{i\in I} f_i y_i + \frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}}\sum_{i\in I}\sum_{j\in J} q_{ij}x_{ij}^s & \\
\text{s.t.} & \sum_{i\in I} x_{ij}^s \geq \lambda_j^s & j\in J; s\in\mathcal{S} \\
& \sum_{j\in J} x_{ij}^s \leq C_i y_i & i\in I; s\in\mathcal{S} \\
& \sum_{i\in I} C_i y_i \geq \max_{s\in\mathcal{S}} \sum_{j\in J} \lambda_j^s & \\
& y_i \in \{0,1\} & i\in I \\
& x_{ij}^s \geq 0 & i\in I; j\in J; s\in\mathcal{S}
\end{array}
$$

where f_i and q_{ij} represent costs of opening facility i and satisfying one unit of demand associated with customer j using facility i, respectively. Furthermore, C_i represents capacity of facility i, and λ_j^s denotes the demand of customer j in scenario s. Note that the third constraint set, which consists of a single constraint, enforces relatively complete recourse by making the second stage feasible for each fixed feasible first stage solution.

The stochastic server location problem (SSLP) considered in this paper is described in [93], and the instances used in experiments come from [4]. The first stage decisions concern installation of servers at possible locations, and the second stage decisions define assignment of clients to servers. Decision variables x_i and y_{ij}^s denote whether or not a server is installed at location i and whether or not client j is assigned to be served by the server at location i under scenario s, respectively. Furthermore, y_{j0}^s is an artificial continuous variable corresponding to the unsatisfied demand of

client j under scenario s. The formulation is as follows:

$$
\begin{aligned}
\min \quad & \textstyle\sum_{i\in I} c_i x_i - \sum_{s\in\mathcal{S}} p_s \left(\sum_{i\in I} \sum_{j\in J} q^s_{ij} y^s_{ij} - \sum_{j\in J} q^s_{j0} y^s_{j0} \right) & & \\
\text{s.t.} \quad & \textstyle\sum_{i\in I} x_i \le v & & i \in I \\
& \textstyle\sum_{j\in J} d_{ij} y^s_{ij} - y^s_{j0} \le u x_i & & i \in I; s \in \mathcal{S} \\
& \textstyle\sum_{i\in I} y^s_{ij} = h^s_j & & j \in J; s \in \mathcal{S} \\
& x_i \in \{0,1\} & & i \in I \\
& y^s_{ij} \in \{0,1\} & & i \in I; j \in J; s \in \mathcal{S} \\
& y^s_{j0} \ge 0 & & j \in J; s \in \mathcal{S}
\end{aligned}
$$

where c_i denotes the cost of placing a server at location i, q^s_{ij} denotes the revenue associated with assigning client j to the server at location i under scenario s, q^s_{j0} denotes the loss of revenue associated with losing one unit of demand from client j under scenario s, and d_{ij} denotes unit consumption of the capacity of location i by client j. u and v denote server capacity and maximum number of servers that can be installed, respectively. Lastly, p_s corresponds to the occurrence probability of scenario s, and h^s_j defines whether or not client j is present in scenario s.

The dynamic capacity acquisition and allocation problem (DCAP) is described in [3], and the 3-stage instances used in computational experiments of this paper are acquired from [135]. The capacity acquisition decisions are made at stages 1 and 2; and based on the acquired capacities of the resources, allocation decisions are made at stages 2 and 3. Decision variable x^t_{is} denote the acquired capacity of resource i at stage t under scenario s, while z^t_{is} represents whether or not additional capacity for resource i is acquired at stage t under scenario s. w^t_{is} is the auxiliary cross-stage accumulated capacity variable, y^t_{ijs} is the binary variable denoting whether or not resource i is assigned to task j at stage t under scenario s, and y^t_{0js} is the binary variable denoting whether or not task j is outsourced at stage t under scenario s.

The H-stage formulation is as follows:

$$\min \quad \sum_{s\in\mathcal{S}} p_s \sum_{t\in T\setminus\{H\}} \left[\sum_{i\in I}(f_i^t z_{is}^t + c_i^t x_{is}^t + \sum_{j\in J} a_{ij}^{t+1} y_{ijs}^{t+1}) + \sum_{j\in J} a_{0j}^{t+1} y_{0js}^{t+1}\right]$$

$$\text{s.t.} \quad \sum_{j\in J} d_{js}^{t+1} y_{ijs}^{t+1} \le w_{is}^t \qquad i\in I; s\in\mathcal{S}; t\in T\setminus\{H\}$$

$$w_{is}^1 = x_{is}^1 \qquad i\in I; s\in\mathcal{S}$$

$$w_{is}^t = w_{is}^{t-1} + x_{is}^t \qquad i\in I; s\in\mathcal{S}; t\in T\setminus\{1,H\}$$

$$x_{is}^t \le C_i^t z_{is}^t \qquad i\in I; s\in\mathcal{S}; t\in T\setminus\{H\}$$

$$\sum_{i\in I\cup\{0\}} y_{ijs}^t = 1 \qquad j\in J; s\in\mathcal{S}; t\in T\setminus\{1\}$$

$$(x_{is^1}^t, z_{is^1}^t, w_{is^1}^t) = (x_{is^2}^t, z_{is^2}^t, w_{is^2}^t) \qquad i\in I; s^1, s^2\in\mathcal{S}; s^1\ne s^2;$$
$$\mathcal{H}(t,s^1) = \mathcal{H}(t,s^2), t\in T\setminus\{H\}$$

$$y_{ijs^1}^t = y_{ijs^2}^t \qquad i\in I\cup\{0\}; j\in J; s^1, s^2\in\mathcal{S}; s^1\ne s^2;$$
$$\mathcal{H}(t,s^1) = \mathcal{H}(t,s^2), t\in T\setminus\{1\}$$

$$x_{is}^t, w_{is}^t \ge 0 \qquad i\in I; s\in\mathcal{S}; t\in T\setminus\{H\}$$

$$z_{is}^t \in \{0,1\} \qquad i\in I; s\in\mathcal{S}; t\in T\setminus\{H\}$$

$$y_{ijs}^t \in \{0,1\} \qquad i\in I\cup\{0\}; j\in J; s\in\mathcal{S}; t\in T\setminus\{1\}$$

where f_i^t and c_i^t represent fixed and variable costs of capacity acquisition for resource i at stage t; and a_{ij}^t and a_{0j}^t represent the cost of allocating resource i to task j and the cost of outsourcing task j, respectively, at stage t. Furthermore, d_{js}^t and C_i^t denote the capacity requirement of task j at stage t under scenario s and the maximum capacity of resource i at stage t, respectively.

C.3 Results of Computational Experiments

K^{max} partitions are solved for every instance, and every partition requires solving m group subproblems, where m equals total number of scenarios, L, divided by group size, q. (K^{max} is 30 for CAP and SSLP instances, and 50 for DCAP instances.) OPT denotes the optimal objective over all scenarios. Δ_k^P and Δ_k^{SAA} denote optimality gaps associated with the partition bound and the lower end of the 95% confidence interval around the SAA estimate, respectively, after k partitions are completed. The optimality gaps are calculated relative to the wait-and-see solution, $z_{WS} = \sum_{s \in \mathcal{S}} z_{GR}(\{s\})$.

$$\Delta_k^P = \frac{(OPT - z_{WS}) - (max_{i=1,\ldots,k}\{z^{P_i}\} - z_{WS})}{OPT - z_{WS}}$$

$$\Delta_k^{SAA} = \frac{(OPT - z_{WS}) - \left(\hat{z}_{km}^{SAA} - t_{0.025,km-1}\sqrt{\frac{\hat{s}_{SAA}^2}{km}} - z_{WS}\right)}{OPT - z_{WS}}, \text{ where } m = \frac{L}{q}$$

$$\delta_k^{SAA,P} = \frac{\Delta_k^{SAA} - \Delta_k^P}{\Delta_k^{SAA}}$$

Table C.1: Partition bound vs. lower end of the 95% SAA confidence interval, CAP instances, $q = 5$.

				# Cont.	# Bin.	# Cons-	$k = 10$			$k = 20$			$k = 30$		
Instance	L	OPT	z_{WS}	Vars.	Vars.	traints	Δ_{10}^{P}	Δ_{10}^{SAA}	$\delta_{10}^{SAA,P}$	Δ_{20}^{P}	Δ_{20}^{SAA}	$\delta_{20}^{SAA,P}$	Δ_{30}^{P}	Δ_{30}^{SAA}	$\delta_{30}^{SAA,P}$
CAP 101	240	734,354	720,186	6,250	25	376	14.23%	49.14%	**71.05%**	14.23%	58.77%	**75.79%**	14.23%	44.26%	**67.86%**
CAP 102	240	788,961	769,943	6,250	25	376	18.30%	56.15%	**67.41%**	18.30%	35.97%	**49.13%**	18.30%	36.83%	**50.32%**
CAP 103	240	823,775	808,880	6,250	25	376	21.61%	73.22%	**70.48%**	21.61%	50.25%	**56.99%**	21.61%	43.59%	**50.42%**
CAP 104	240	841,151	831,881	6,250	25	376	5.77%	65.31%	**91.16%**	5.77%	46.72%	**87.65%**	5.65%	40.89%	**86.18%**
CAP 111	240	763,265	749,675	12,500	50	501	9.12%	42.35%	**78.46%**	8.97%	36.02%	**75.09%**	8.97%	28.06%	**68.03%**
CAP 112	240	840,190	815,795	12,500	50	501	19.61%	37.55%	**47.77%**	18.55%	33.21%	**44.15%**	17.33%	26.38%	**34.32%**
CAP 113	240	916,658	890,418	12,500	50	501	20.87%	38.89%	**46.32%**	19.89%	42.89%	**53.62%**	19.89%	33.54%	**40.69%**
CAP 114	240	1,024,707	1,003,174	12,500	50	501	17.17%	25.94%	**33.82%**	15.53%	23.37%	**33.56%**	15.53%	23.02%	**32.54%**
CAP 121	240	720,520	707,503	12,500	50	501	12.54%	41.20%	**69.57%**	11.39%	21.98%	**48.18%**	11.12%	17.83%	**37.61%**
CAP 122	240	777,722	761,099	12,500	50	501	16.28%	28.06%	**41.97%**	15.01%	24.51%	**38.76%**	15.01%	34.40%	**56.37%**
CAP 123	240	824,148	805,682	12,500	50	501	18.90%	43.72%	**56.77%**	17.31%	32.23%	**46.30%**	17.16%	32.03%	**46.43%**
CAP 124	240	859,846	838,223	12,500	50	501	16.03%	34.52%	**53.56%**	16.03%	39.01%	**58.91%**	16.03%	34.82%	**53.96%**
CAP 131	240	723,212	709,403	12,500	50	501	13.93%	67.40%	**79.33%**	13.93%	47.31%	**70.55%**	13.93%	38.62%	**63.93%**
CAP 132	240	778,244	763,527	12,500	50	501	17.46%	42.68%	**59.08%**	16.10%	29.32%	**45.10%**	16.10%	32.77%	**50.87%**
CAP 133	240	815,298	796,921	12,500	50	501	19.52%	36.75%	**46.87%**	19.52%	35.40%	**44.85%**	19.52%	34.70%	**43.75%**
CAP 134	240	849,237	836,685	12,500	50	501	7.52%	28.35%	**73.48%**	7.52%	7.60%	**1.12%**	7.52%	15.81%	**52.46%**
Mean							15.55%	44.45%	61.70%	14.98%	35.29%	51.86%	14.87%	32.35%	52.23%
Std. dev.							4.75%	14.32%	15.86%	4.56%	12.58%	20.18%	4.53%	8.34%	14.08%

Note. 240-scenario instances of CAP 101-104 have 300,000 continuous variables, 25 binary variables, and 18,001 constraints; and 240-scenario instances of CAP 111-134 have 600,000 continuous variables, 50 binary variables, and 24,001 constraints.

Table C.2: Partition bound vs. lower end of the 95% SAA confidence interval, CAP instances, $q = 10$.

Instance	L	OPT	z_{WS}	# Cont. Vars.	# Bin. Vars.	# Constraints	$k = 10$			$k = 20$			$k = 30$		
							Δ_{10}^{P}	Δ_{10}^{SAA}	$\delta_{10}^{SAA,P}$	Δ_{20}^{P}	Δ_{20}^{SAA}	$\delta_{20}^{SAA,P}$	Δ_{30}^{P}	Δ_{30}^{SAA}	$\delta_{30}^{SAA,P}$
CAP 101	240	734,354	720,186	12,500	25	751	7.07%	15.54%	**54.48%**	6.98%	17.35%	**59.78%**	6.98%	19.68%	**64.55%**
CAP 102	240	788,961	769,943	12,500	25	751	8.06%	37.20%	**78.34%**	8.06%	24.51%	**67.12%**	8.06%	20.36%	**60.42%**
CAP 103	240	823,775	808,880	12,500	25	751	12.72%	32.83%	**61.27%**	12.72%	29.87%	**57.42%**	12.72%	27.21%	**53.27%**
CAP 104	240	841,151	831,881	12,500	25	751	0.00%	60.04%	**100.00%**	0.00%	34.36%	**100.00%**	0.00%	31.64%	**100.00%**
CAP 111	240	763,265	749,675	25,000	50	1,001	3.45%	17.56%	**80.35%**	3.06%	25.93%	**88.19%**	3.06%	17.21%	**82.20%**
CAP 112	240	840,190	815,795	25,000	50	1,001	7.65%	18.80%	**59.30%**	7.65%	26.42%	**71.04%**	7.65%	27.86%	**72.53%**
CAP 113	240	916,658	890,418	25,000	50	1,001	10.83%	34.05%	**68.20%**	10.51%	26.79%	**60.78%**	10.51%	26.63%	**60.54%**
CAP 114	240	1,024,707	1,003,174	25,000	50	1,001	6.68%	50.18%	**86.69%**	6.53%	38.80%	**83.17%**	6.45%	34.01%	**81.03%**
CAP 121	240	720,520	707,503	25,000	50	1,001	4.08%	39.59%	**89.70%**	4.08%	33.67%	**87.89%**	4.08%	27.16%	**84.99%**
CAP 122	240	777,722	761,099	25,000	50	1,001	6.06%	40.30%	**84.97%**	6.06%	29.98%	**79.80%**	5.95%	30.65%	**80.58%**
CAP 123	240	824,148	805,682	25,000	50	1,001	7.82%	30.77%	**74.60%**	6.98%	23.32%	**70.09%**	6.98%	16.49%	**57.70%**
CAP 124	240	859,846	838,223	25,000	50	1,001	8.82%	44.03%	**79.96%**	8.15%	28.68%	**71.58%**	7.82%	24.32%	**67.85%**
CAP 131	240	723,212	709,403	25,000	50	1,001	5.34%	25.47%	**79.05%**	5.34%	26.96%	**80.21%**	5.27%	28.87%	**81.73%**
CAP 132	240	778,244	763,527	25,000	50	1,001	7.36%	21.64%	**65.98%**	7.36%	18.73%	**60.70%**	7.12%	12.85%	**44.58%**
CAP 133	240	815,298	796,921	25,000	50	1,001	11.03%	29.69%	**62.86%**	10.05%	23.92%	**58.01%**	10.05%	18.72%	**46.35%**
CAP 134	240	849,237	836,685	25,000	50	1,001	0.58%	71.30%	**99.18%**	0.58%	62.12%	**99.06%**	0.58%	47.67%	**98.78%**
Mean							6.72%	35.56%	76.56%	6.51%	29.46%	74.68%	6.45%	25.71%	71.07%
Std. dev.							3.49%	15.43%	13.75%	3.38%	10.28%	14.16%	3.37%	8.44%	16.96%

Table C.3: Partition bound vs. lower end of the 95% SAA confidence interval, CAP instances, $q = 15$.

				# Cont.	# Bin.	# Cons-	$k = 10$			$k = 20$			$k = 30$		
Instance	L	OPT	z_{WS}	Vars.	Vars.	traints	Δ_{10}^{P}	Δ_{10}^{SAA}	$\delta_{10}^{SAA,P}$	Δ_{20}^{P}	Δ_{20}^{SAA}	$\delta_{20}^{SAA,P}$	Δ_{30}^{P}	Δ_{30}^{SAA}	$\delta_{30}^{SAA,P}$
CAP 101	240	734,354	720,186	18,750	25	1,126	4.41%	28.97%	**84.76%**	4.41%	30.28%	**85.42%**	3.91%	26.99%	**85.51%**
CAP 102	240	788,961	769,943	18,750	25	1,126	3.74%	32.03%	**88.32%**	3.74%	15.64%	**76.09%**	3.74%	14.89%	**74.89%**
CAP 103	240	823,775	808,880	18,750	25	1,126	9.33%	21.34%	**56.29%**	9.33%	26.76%	**65.14%**	9.33%	17.13%	**45.54%**
CAP 104	240	841,151	831,881	18,750	25	1,126	0.00%	45.52%	**100.00%**	0.00%	25.66%	**100.00%**	0.00%	20.04%	**100.00%**
CAP 111	240	763,265	749,675	37,500	50	1,501	1.39%	45.33%	**96.93%**	1.34%	32.17%	**95.82%**	1.34%	30.42%	**95.58%**
CAP 112	240	840,190	815,795	37,500	50	1,501	5.36%	35.52%	**84.92%**	5.36%	31.96%	**83.24%**	5.36%	27.57%	**80.57%**
CAP 113	240	916,658	890,418	37,500	50	1,501	6.77%	44.97%	**84.95%**	6.77%	26.51%	**74.46%**	6.12%	19.05%	**67.88%**
CAP 114	240	1,024,707	1,003,174	37,500	50	1,501	2.70%	23.50%	**88.52%**	2.65%	18.51%	**85.68%**	2.65%	16.43%	**83.86%**
CAP 121	240	720,520	707,503	37,500	50	1,501	2.14%	14.28%	**85.03%**	2.14%	-0.85%	**N/A**	2.14%	7.71%	**72.27%**
CAP 122	240	777,722	761,099	37,500	50	1,501	2.50%	28.79%	**91.33%**	2.50%	19.34%	**87.09%**	2.50%	16.10%	**84.49%**
CAP 123	240	824,148	805,682	37,500	50	1,501	2.88%	29.87%	**90.35%**	2.88%	19.21%	**85.00%**	2.88%	21.39%	**86.53%**
CAP 124	240	859,846	838,223	37,500	50	1,501	5.30%	25.78%	**79.44%**	5.30%	20.53%	**74.19%**	4.66%	17.38%	**73.19%**
CAP 131	240	723,212	709,403	37,500	50	1,501	2.94%	57.95%	**94.92%**	2.94%	51.73%	**94.31%**	2.85%	38.91%	**92.68%**
CAP 132	240	778,244	763,527	37,500	50	1,501	4.54%	56.09%	**91.91%**	4.14%	22.39%	**81.51%**	3.87%	16.78%	**76.96%**
CAP 133	240	815,298	796,921	37,500	50	1,501	7.04%	26.77%	**73.70%**	5.18%	22.55%	**77.02%**	5.18%	14.26%	**63.66%**
CAP 134	240	849,237	836,685	37,500	50	1,501	0.15%	5.32%	**97.27%**	0.15%	-2.23%	**N/A**	0.00%	1.73%	**100.00%**
Mean							3.82%	32.63%	86.79%	3.68%	22.51%	83.21%	3.53%	19.17%	80.23%
Std. dev.							2.53%	14.35%	10.61%	2.41%	12.61%	9.49%	2.34%	8.80%	14.27%

Table C.4: Partition bound vs. lower end of the 95% SAA confidence interval, CAP instances, $q = 20$.

Instance	L	OPT	z_{WS}	# Cont. Vars.	# Bin. Vars.	# Constraints	$k = 10$			$k = 20$			$k = 30$		
							Δ_{10}^{P}	Δ_{10}^{SAA}	$\delta_{10}^{SAA,P}$	Δ_{20}^{P}	Δ_{20}^{SAA}	$\delta_{20}^{SAA,P}$	Δ_{30}^{P}	Δ_{30}^{SAA}	$\delta_{30}^{SAA,P}$
CAP 101	240	734,354	720,186	25,000	25	1,501	3.98%	36.21%	**89.00%**	3.46%	15.63%	**77.84%**	3.28%	12.60%	**73.95%**
CAP 102	240	788,961	769,943	25,000	25	1,501	3.31%	29.87%	**88.92%**	3.28%	12.86%	**74.48%**	3.20%	9.96%	**67.84%**
CAP 103	240	823,775	808,880	25,000	25	1,501	7.88%	40.66%	**80.61%**	7.88%	18.96%	**58.42%**	6.90%	15.53%	**55.59%**
CAP 104	240	841,151	831,881	25,000	25	1,501	0.00%	69.85%	**100.00%**	0.00%	40.50%	**100.00%**	0.00%	27.85%	**100.00%**
CAP 111	240	763,265	749,675	50,000	50	2,001	1.06%	35.61%	**97.03%**	0.93%	20.45%	**95.44%**	0.26%	20.70%	**98.72%**
CAP 112	240	840,190	815,795	50,000	50	2,001	4.86%	43.12%	**88.74%**	4.86%	25.16%	**80.70%**	4.86%	16.29%	**70.19%**
CAP 113	240	916,658	890,418	50,000	50	2,001	3.69%	25.64%	**85.59%**	3.69%	11.75%	**68.56%**	3.69%	21.29%	**82.65%**
CAP 114	240	1,024,707	1,003,174	50,000	50	2,001	1.07%	26.31%	**95.95%**	1.07%	20.55%	**94.81%**	1.07%	16.13%	**93.39%**
CAP 121	240	720,520	707,503	50,000	50	2,001	1.64%	55.38%	**97.04%**	1.64%	38.48%	**95.74%**	1.64%	30.19%	**94.56%**
CAP 122	240	777,722	761,099	50,000	50	2,001	1.46%	10.72%	**86.42%**	1.46%	3.64%	**59.99%**	1.46%	5.68%	**74.38%**
CAP 123	240	824,148	805,682	50,000	50	2,001	1.51%	22.25%	**93.23%**	1.51%	14.07%	**89.30%**	1.51%	13.05%	**88.45%**
CAP 124	240	859,846	838,223	50,000	50	2,001	4.12%	30.08%	**86.31%**	3.43%	24.96%	**86.25%**	3.43%	24.39%	**85.93%**
CAP 131	240	723,212	709,403	50,000	50	2,001	1.45%	60.04%	**97.58%**	1.45%	30.46%	**95.23%**	1.45%	25.41%	**94.28%**
CAP 132	240	778,244	763,527	50,000	50	2,001	2.28%	36.82%	**93.82%**	2.28%	32.14%	**92.92%**	2.28%	27.70%	**91.78%**
CAP 133	240	815,298	796,921	50,000	50	2,001	3.39%	22.59%	**85.01%**	3.39%	20.72%	**83.66%**	3.39%	18.15%	**81.34%**
CAP 134	240	849,237	836,685	50,000	50	2,001	0.00%	4.03%	**100.00%**	0.00%	-3.91%	**N/A**	0.00%	-10.72%	**N/A**
Mean							2.61%	34.32%	91.58%	2.52%	20.40%	83.55%	2.40%	17.14%	83.54%
Std. dev.							2.04%	17.19%	6.00%	2.00%	11.78%	13.33%	1.87%	10.16%	12.88%

Table C.5: Best partition bound vs. lower limit of the 95% SAA confidence interval on SSLP instances.

Instance	L	OPT	z_{WS}	q	# Cont. Var.	# Bin. Var.	# Constraints	$k=10$ Δ_{10}^{P}	$k=10$ Δ_{10}^{SAA}	$k=10$ $\delta_{10}^{SAA,P}$	$k=20$ Δ_{20}^{P}	$k=20$ Δ_{20}^{SAA}	$k=20$ $\delta_{20}^{SAA,P}$	$k=30$ Δ_{30}^{P}	$k=30$ Δ_{30}^{SAA}	$k=30$ $\delta_{30}^{SAA,P}$
SSLP 5-25-50	50	-121.6	-134.3	2	50	255	65	48.82%	106.99%	**54.37%**	41.60%	80.94%	**48.60%**	41.60%	73.36%	**43.29%**
				5	125	630	155	15.38%	28.86%	**46.69%**	14.60%	18.54%	**21.27%**	11.62%	19.97%	**41.83%**
				10	250	1255	305	1.26%	73.90%	**98.30%**	0.00%	38.38%	**100.00%**	0.00%	30.50%	**100.00%**
				25	625	3130	755	0.00%	52.13%	**100.00%**	0.00%	29.61%	**100.00%**	0.00%	21.19%	**100.00%**
SSLP 10-50-100	100	-354.2	-371.4	2	100	1010	130	46.74%	80.53%	**41.95%**	45.29%	74.21%	**38.97%**	45.29%	67.90%	**33.30%**
				5	250	2510	310	13.49%	38.40%	**64.87%**	9.01%	26.56%	**66.07%**	9.01%	25.51%	**64.67%**

Note. The full SSLP 5-25-50 instance have 1250 continuous variables, 6255 binary variables, and 1505 constraints; and the full SSLP 10-50-100 instance have 5000 continuous variables, 50,010 binary variables, and 6010 constraints.

Table C.6: Best partition bound vs. lower limit of the 95% SAA confidence interval on DCAP 332 instances.

Instance	L	OPT	z_{WS}	q	# Cont. Var.	# Bin. Var.	Partition Selection	# Cons-traints	$k=10$ Δ_{10}^{P}	Δ_{10}^{SAA}	$\delta_{10}^{SAA,P}$	$k=20$ Δ_{20}^{P}	Δ_{20}^{SAA}	$\delta_{20}^{SAA,P}$	$k=30$ Δ_{30}^{P}	Δ_{30}^{SAA}	$\delta_{30}^{SAA,P}$
DCAP 332 1x10x20	200	918.69	850.83	2	24	60	Tree	78.0	63.73%		**51.32%**	63.46%		**45.28%**	63.42%		**40.75%**
							Random	59.8	67.31%	130.91%	**48.58%**	59.27%	115.97%	**48.89%**	57.93%	107.03%	**45.87%**
							Misaligned	57.0	74.44%		**43.13%**	73.71%		**36.44%**	73.68%		**31.16%**
				5	60	150	Tree	240.0	25.34%		**76.04%**	25.10%		**75.31%**	25.05%		**71.85%**
							Random	174.3	62.77%	105.77%	**40.66%**	61.86%	101.66%	**39.16%**	61.86%	89.00%	**30.50%**
							Misaligned	156.0	70.04%		**33.78%**	70.04%		**31.11%**	70.02%		**21.32%**
				10	120	300	Tree	510.0	5.06%		**94.43%**	4.99%		**95.06%**	4.96%		**94.14%**
							Random	392.8	51.17%	90.97%	**43.75%**	51.17%	101.09%	**49.38%**	51.17%	84.57%	**39.50%**
							Misaligned	321.0	69.94%		**23.12%**	69.94%		**30.82%**	69.93%		**17.31%**
				20	240	600	Tree	1050.0	1.25%		**98.36%**	1.25%		**98.12%**	1.25%		**97.89%**
							Random	884.7	41.86%	76.70%	**45.43%**	40.05%	66.81%	**40.05%**	40.05%	59.46%	**32.63%**
							Misaligned	861.0	48.86%		**36.29%**	47.89%		**28.32%**	47.89%		**19.46%**
				40	480	1200	Tree	2109.0	0.06%		**99.94%**	0.05%		**99.94%**	0.05%		**99.93%**
							Random	1943.1	24.02%	96.20%	**75.03%**	23.39%	82.44%	**71.63%**	23.39%	69.54%	**66.37%**
							Misaligned	1941.0	30.25%		**68.55%**	27.40%		**66.76%**	27.40%		**60.59%**
DCAP 332 1x15x20	300	1212.54	1130.14	2	24	60	Tree	78.0	59.65%		**53.99%**	59.45%		**47.07%**	59.41%		**39.98%**
							Random	59.6	78.04%	129.65%	**39.80%**	68.83%	112.34%	**38.73%**	64.29%	98.99%	**35.05%**
							Misaligned	57.4	86.04%		**33.63%**	84.17%		**25.08%**	82.78%		**16.38%**
				5	60	150	Tree	240.0	19.56%		**73.07%**	16.75%		**79.01%**	16.65%		**79.29%**
							Random	168.7	67.48%	72.65%	**7.12%**	67.48%	79.77%	**15.41%**	67.27%	80.38%	**16.31%**
							Misaligned	156.0	73.46%		**N/A**	73.46%		**7.91%**	73.06%		**9.10%**
				10	120	300	Tree	510.0	3.42%		**95.96%**	3.34%		**94.95%**	3.32%		**94.29%**
							Random	371.5	52.20%	84.61%	**38.31%**	52.20%	66.09%	**21.02%**	52.20%	58.14%	**10.23%**
							Misaligned	332.8	60.13%		**28.93%**	60.09%		**9.08%**	59.87%		**N/A**
				20	240	600	Tree	1050.0	2.11%		**96.96%**	2.11%		**96.55%**	2.11%		**96.27%**
							Random	832.4	41.82%	69.51%	**39.84%**	41.82%	61.19%	**31.66%**	41.82%	56.59%	**26.10%**
							Misaligned	767.4	50.07%		**27.97%**	50.07%		**18.18%**	50.07%		**11.53%**
				30	360	900	Tree	1569.0	0.48%		**99.38%**	0.44%		**99.22%**	0.43%		**99.23%**
							Random	1333.0	34.45%	78.45%	**56.09%**	34.20%	55.81%	**38.73%**	34.01%	55.87%	**39.13%**
							Misaligned	1296.0	41.53%		**47.07%**	41.53%		**25.60%**	41.53%		**25.68%**
DCAP 332 1x25x20	500	1691.62	1646.88	2	24	60	Tree	78.0	61.05%		**48.60%**	60.49%		**44.27%**	60.42%		**45.45%**
							Random	58.9	78.66%	118.79%	**33.79%**	72.05%	108.55%	**33.62%**	68.64%	110.77%	**38.03%**
							Misaligned	57.1	82.33%		**30.69%**	81.08%		**25.30%**	80.48%		**27.34%**
				5	60	150	Tree	240.0	25.90%		**79.95%**	24.14%		**73.65%**	23.89%		**69.39%**
							Random	163.7	68.89%	129.13%	**46.65%**	68.89%	91.61%	**24.80%**	68.89%	78.05%	**11.73%**
							Misaligned	156.0	72.05%		**44.20%**	72.05%		**21.35%**	71.69%		**8.15%**
				10	120	300	Tree	510.0	10.89%		**90.63%**	10.73%		**89.08%**	10.67%		**87.67%**
							Random	353.8	55.77%	116.22%	**52.01%**	55.29%	98.26%	**43.73%**	54.42%	86.56%	**37.14%**
							Misaligned	325.2	59.10%		**49.15%**	59.10%		**39.86%**	59.10%		**31.73%**
				20	240	600	Tree	1050.0	6.53%		**79.92%**	6.53%		**89.73%**	6.53%		**90.10%**
							Random	775.1	45.27%	32.51%	**N/A**	45.17%	63.59%	**28.96%**	44.89%	65.96%	**31.94%**
							Misaligned	693.5	52.95%		**N/A**	52.74%		**17.07%**	52.74%		**20.05%**
				50	600	1500	Tree	2628.0	2.36%		**97.00%**	2.22%		**95.88%**	2.21%		**95.85%**
							Random	2226.5	32.53%	78.62%	**58.62%**	31.51%	53.90%	**41.54%**	31.43%	53.16%	**40.88%**
							Misaligned	2166.0	38.24%		**51.36%**	36.79%		**31.75%**	36.79%		**30.79%**

Note. The full DCAP 332 1x10x20 instance has 2,400 continuous variables, 6,000 binary variables, and 10,581 constraints; the full DCAP 332 1x15x20 instance has 3,600 continuous variables, 9,000 binary variables, and 15,876 constraints; and the full DCAP 332 1x25x20 instance has 6,000 continuous variables, 15,000 binary variables, and 26,466 constraints.

Table C.7: Best partition bound vs. lower limit of the 95% SAA confidence interval on DCAP 342 instances.

Instance	L	OPT	z_{WS}	q	# Cont. Var.	# Bin. Var.	Partition Selection	# Cons-traints	$k=10$ Δ_{10}^{P}	Δ_{10}^{SAA}	$\delta_{10}^{SAA,P}$	$k=20$ Δ_{20}^{P}	Δ_{20}^{SAA}	$\delta_{20}^{SAA,P}$	$k=30$ Δ_{30}^{P}	Δ_{30}^{SAA}	$\delta_{30}^{SAA,P}$
DCAP 342 1x10x20	200	1817.92	1780.54	2	24	76	Tree	86.0	39.31%		**81.16%**	38.25%		**77.40%**	38.21%		**71.40%**
							Random	65.2	65.93%	208.63%	**68.40%**	50.05%	169.22%	**70.43%**	42.53%	133.61%	**68.17%**
							Misaligned	61.0	85.26%		**59.13%**	85.02%		**49.76%**	84.93%		**36.43%**
				5	60	190	Tree	266.0	6.24%		**95.19%**	3.87%		**97.58%**	3.85%		**96.94%**
							Random	187.8	70.81%	129.79%	**45.44%**	68.39%	159.91%	**57.23%**	68.39%	125.94%	**45.70%**
							Misaligned	166.0	84.10%		**35.20%**	83.80%		**47.59%**	83.80%		**33.46%**
				10	120	380	Tree	566.0	1.42%		**98.78%**	1.26%		**98.69%**	1.20%		**98.65%**
							Random	426.1	51.54%	116.45%	**55.74%**	51.54%	96.67%	**46.69%**	51.54%	88.84%	**41.99%**
							Misaligned	341.0	83.54%		**28.26%**	83.51%		**13.62%**	83.49%		**6.03%**
				20	240	760	Tree	1166.0	0.29%		**99.84%**	0.29%		**99.70%**	0.29%		**99.66%**
							Random	967.6	36.60%	183.77%	**80.08%**	35.60%	95.09%	**62.56%**	35.60%	84.30%	**57.77%**
							Misaligned	941.0	46.89%		**74.48%**	45.47%		**52.18%**	45.47%		**46.06%**
				40	480	1520	Tree	2341.0	0.24%		**99.69%**	0.24%		**99.68%**	0.24%		**99.59%**
							Random	2143.3	18.37%	78.42%	**76.58%**	16.24%	76.95%	**78.89%**	16.24%	59.67%	**72.78%**
							Misaligned	2141.0	19.96%		**74.55%**	13.21%		**82.83%**	13.21%		**77.85%**
DCAP 342 1x15x20	300	1897.17	1852.91	2	24	76	Tree	86.0	42.49%		**71.92%**	41.95%		**72.80%**	41.85%		**71.00%**
							Random	64.0	67.14%	151.28%	**55.62%**	53.67%	154.19%	**65.19%**	49.18%	144.34%	**65.93%**
							Misaligned	61.3	77.51%		**48.76%**	76.56%		**50.35%**	74.93%		**48.08%**
				5	60	190	Tree	266.0	11.73%		**89.44%**	11.51%		**87.48%**	11.40%		**87.93%**
							Random	181.3	67.93%	111.04%	**38.83%**	67.93%	91.94%	**26.12%**	67.93%	94.45%	**28.08%**
							Misaligned	166.0	77.20%		**30.48%**	77.17%		**16.06%**	77.17%		**18.30%**
				10	120	380	Tree	566.0	2.98%		**98.14%**	2.85%		**97.32%**	2.83%		**97.06%**
							Random	402.7	59.76%	160.18%	**62.69%**	59.76%	106.20%	**43.73%**	58.23%	96.00%	**39.34%**
							Misaligned	354.8	73.23%		**54.29%**	72.57%		**31.66%**	72.40%		**24.58%**
				20	240	760	Tree	1166.0	1.76%		**98.66%**	1.76%		**97.84%**	1.76%		**98.02%**
							Random	906.8	43.67%	131.29%	**66.74%**	43.67%	81.79%	**46.61%**	43.67%	88.81%	**50.83%**
							Misaligned	830.5	59.77%		**54.47%**	58.61%		**28.34%**	58.61%		**34.01%**
				30	360	1140	Tree	1741.0	1.14%		**98.84%**	1.13%		**98.76%**	1.11%		**98.66%**
							Random	1458.2	38.17%	98.36%	**61.19%**	36.91%	91.27%	**59.55%**	36.80%	82.95%	**55.63%**
							Misaligned	1416.0	47.07%		**52.14%**	44.36%		**51.40%**	44.36%		**46.52%**
DCAP 342 1x25x20	500	1675.73	1636.25	2	24	76	Tree	86.0	47.77%		**56.85%**	47.28%		**57.89%**	47.21%		**57.56%**
							Random	63.2	70.84%	110.69%	**36.00%**	64.19%	112.27%	**42.82%**	57.18%	111.23%	**48.59%**
							Misaligned	61.2	80.10%		**27.63%**	79.01%		**29.63%**	77.94%		**29.93%**
				5	60	190	Tree	266.0	11.37%		**90.56%**	9.90%		**90.99%**	9.46%		**89.97%**
							Random	174.9	74.46%	120.48%	**38.20%**	74.07%	109.92%	**32.62%**	74.07%	94.30%	**21.46%**
							Misaligned	166.0	79.93%		**33.66%**	79.93%		**27.28%**	79.93%		**15.24%**
				10	120	380	Tree	566.0	1.95%		**98.28%**	1.84%		**98.20%**	1.79%		**98.04%**
							Random	379.4	68.19%	113.02%	**39.66%**	68.00%	102.34%	**33.55%**	67.82%	90.97%	**25.45%**
							Misaligned	346.2	77.93%		**31.05%**	77.44%		**24.33%**	77.44%		**14.88%**
				20	240	760	Tree	1166.0	0.56%		**99.59%**	0.56%		**99.53%**	0.56%		**99.38%**
							Random	837.1	58.22%	137.77%	**57.74%**	57.25%	118.77%	**51.80%**	57.25%	90.66%	**36.85%**
							Misaligned	741.5	71.65%		**48.00%**	71.40%		**39.89%**	71.40%		**21.25%**
				50	600	1900	Tree	2916.0	0.37%		**99.65%**	0.31%		**99.54%**	0.31%		**99.54%**
							Random	2438.9	39.31%	104.49%	**62.38%**	37.04%	68.00%	**45.53%**	37.04%	66.90%	**44.63%**
							Misaligned	2366.0	48.16%		**53.92%**	47.22%		**30.57%**	46.42%		**30.62%**

Note. The full DCAP 342 1x10x20 instance has 2,400 continuous variables, 7,600 binary variables, and 11,741 constraints; the full DCAP 342 1x15x20 instance has 3,600 continuous variables, 11,400 binary variables, and 17,616 constraints; and the full DCAP 342 1x25x20 instance has 6,000 continuous variables, 19,000 binary variables, and 29,366 constraints.

Table C.8: Tree-aligned vs. misaligned partition bounds in DCAP 332 1x4x2 with $A = 8a + 4\Lambda + 12\lambda = 2.7$

	$\Lambda = 0.01$			$\Lambda = 0.05$			$\Lambda = 0.09$		
λ	OPT	Tree-aligned	Misaligned	OPT	Tree-aligned	Misaligned	OPT	Tree-aligned	Misaligned
0	101.01	100.92	100.90	102.00	101.52	101.43	102.98	102.10	101.94
0.2	102.48	90.20	102.37	111.29	94.75	104.61	112.14	99.39	104.90
0.4	144.92	111.26	144.75	145.94	111.86	145.09	146.96	112.44	145.43

Table C.9: Tree-aligned vs. misaligned partition bounds in DCAP 332 1x4x2 with $A = 8a + 4\Lambda + 12\lambda = 4.6$

	$\Lambda = 0.01$			$\Lambda = 0.11$			$\Lambda = 0.21$		
λ	OPT	Tree-aligned	Misaligned	OPT	Tree-aligned	Misaligned	OPT	Tree-aligned	Misaligned
0	147.00	146.90	146.88	149.47	148.35	148.14	151.27	149.31	148.97
0.25	149.00	125.90	148.77	151.01	134.16	149.30	192.42	155.81	159.74
0.5	149.81	119.04	149.64	152.36	123.70	150.64	193.78	146.06	161.35

Table C.10: Tree-aligned vs. misaligned partition bounds in DCAP 332 1x4x2 with $A = 8a + 4\Lambda + 12\lambda = 9.0$

	$\Lambda = 0.01$			$\Lambda = 0.15$			$\Lambda = 0.3$		
λ	OPT	Tree-aligned	Misaligned	OPT	Tree-aligned	Misaligned	OPT	Tree-aligned	Misaligned
0	2,031.38	2,031.21	2,030.99	2,036.85	2,034.25	2,009.77	2,042.14	2,019.73	1,985.60
0.7	1,096.40	1,095.31	1,096.16	1,099.97	1,097.92	1,096.44	1,145.38	1,120.88	1,107.05
1.3	1,099.37	1,075.65	1,099.10	1,102.99	1,085.18	1,099.49	1,412.90	1,395.65	1,384.91

C.4 Probability of Recombination

When a number of partitions are randomly sampled, recombination may not succeed, as it may not be possible to feasibly recombine their groups into a *new partition*, different from all existing partitions. In order to better understand the likelihood that recombination succeeds in this sense, *i.e.*, that groups in existing partitions can be combined to form a new partition, we consider the simplest case: generating a recombined partition from only two existing partitions. More specifically, we assume that we have a randomly sampled partition Π, and examine the probability that a second randomly sampled partition $\hat{\Pi}$ can be feasibly recombined with Π to create a new partition that is different from both Π and $\hat{\Pi}$.

In order for this to occur, the groups in Π and $\hat{\Pi}$ must be able to be partitioned in two, say $\Pi = \Pi_1 \cup \Pi_2$ and $\hat{\Pi} = \hat{\Pi}_1 \cup \hat{\Pi}_2$, with the union of all groups in Π_1 precisely the set of scenarios in $\hat{\Pi}_1$ (which implies the same for Π_2 and $\hat{\Pi}_2$, respectively). In this case, provided $\Pi_1 \neq \hat{\Pi}_1$ and $\Pi_2 \neq \hat{\Pi}_2$, it must be that $\hat{\Pi}_1 \cup \Pi_2$ (and $\Pi_1 \cup \hat{\Pi}_2$) is a new partition. For this to be possible, it must be that $|\Pi_1| = |\hat{\Pi}_1| \geq 2$ (and $|\Pi_2| = |\hat{\Pi}_2| \geq 2$). In order to provide insight into the relationship between group size (q) and the probability of finding new partitions using recombination, we focus here on the simplest case, where $|\Pi_1| = |\hat{\Pi}_1| = 2$. We also assume that q divides evenly into the number of scenarios, L, and use $k = L/q$.

Given Π a q-partition of $\mathcal{S}$, let ρ denote the probability that a second, randomly sampled, q-partition $\hat{\Pi}$, has the property that, for some distinct groups $\hat{S}, \hat{S}' \in \hat{\Pi}$, there exist $S, S' \in \Pi$ with $\hat{S} \cup \hat{S}' = S \cup S'$, $\{\hat{S}, \hat{S}'\} \neq \{S, S'\}$ and $\hat{\Pi} \setminus \{\hat{S}, \hat{S}'\} \neq \Pi \setminus \{S, S'\}$. In other words, ρ is the probability that for randomly sampled $\hat{\Pi}$, there exist $\Pi_1, \Pi_2, \hat{\Pi}_1, \hat{\Pi}_2$ with $\Pi = \Pi_1 \cup \Pi_2$, $\hat{\Pi} = \hat{\Pi}_1 \cup \hat{\Pi}_2$, $|\Pi_1| = |\hat{\Pi}_1| = 2$, $\Pi_1 \neq \hat{\Pi}_1$ and $\Pi_2 \neq \hat{\Pi}_2$, enabling recombination to obtain a new partition. Counting arguments

allow us to deduce upper and lower bounds on ρ:

$$\binom{k}{2} M \geq \rho \geq (k-1)M, \tag{C.1}$$

where

$$M = \frac{(F(2q,q)-1)(F(L-2q,q)-1)}{F(L,q)}$$

for $F(A,q) = \frac{A!}{(q!)^c c!}$, the number of q-partitions of a scenario set with A scenarios and $c = A/q$, integer.

We explain why these bounds on ρ are valid with the aid of a 12-scenario example with $q = 2$. Suppose $\Pi = \{\{1,2\},\{3,4\},\{5,6\},\{7,8\},\{9,10\},\{11,12\}\}$.

If we fix a choice of distinct groups $S, S' \in \Pi$, then the number of different q-partitions of $\mathcal{S}$ that can be obtained by repartitioning $S \cup S'$ and repartitioning $\mathcal{S} \setminus (S \cup S')$ to yield $\hat{\Pi}_1$ and $\hat{\Pi}_2$, respectively, with $\hat{\Pi}_1 \neq \{S, S'\}$ and $\hat{\Pi}_2 \neq \Pi \setminus \{S, S'\}$, is exactly $(F(2q,q)-1)(F(L-2q,q)-1)$, since $F(2q,q)$ is the number of ways to partition $S \cup S'$ and $F(L-2q,q)$ is the number of ways to partition $\mathcal{S} \setminus (S \cup S')$. The resulting q-partitions are distinct and each can be successfully recombined with Π.

However, different choices of $S, S' \in \Pi$, followed by repartitioning as described above, may yield the same q-partitions. For example, if we take $\{S, S'\} = \{\{1,2\},\{3,4\}\}$, one repartitioning is $\hat{\Pi} = \{\{1,3\},\{2,4\},\{5,7\},\{6,8\},\{9,11\},\{10,12\}\}$. But this q-partition may also be found by repartitioning with $\{S, S'\} = \{\{5,6\},\{7,8\}\}$, or with $\{S, S'\} = \{\{9,10\},\{11,12\}\}$.

Fortunately, there is a set of choices for $S, S' \in \Pi$ for which no repartitioning can coincide, and that is one in which there is a common group. Specifically, if we write $\Pi = \{S_1, \ldots, S_k\}$, and take $S = S_1$, then each choice of $S' = S_2, \ldots, S_k$ must yield different q-partitions. For example, fixing $S = \{1,2\}$ and $S' = \{3,4\}$ and repartitioning as just described must yield different q-partitions from fixing $S = \{1,2\}$ and $S' = \{5,6\}$ before repartitioning, since in the former case scenario 1 can only appear in a group with scenario 3 or scenario 4, where as in the latter case, scenario

1 can only appear in a group with scenario 5 or scenario 6. Thus the number of different q-partitions $\hat{\Pi}$ with the property we seek must be at least $(k-1)(F(2q,q)-1)(F(L-2q,q)$. The lower bound on ρ follows.

Taking all possible pairs of distinct $S, S' \in \Pi$ followed by repartitioning as described above clearly yields an upper bound on the number of different q-partitions $\hat{\Pi}$ with the desired property.

Table C.11 presents the upper and lower bounds for the recombination probability, ρ, when the original problem has 240 scenarios, which is the case for all CAP instances used in the computational studies of this paper. Clearly, the recombination probability is the highest for the smallest group size, $q = 2$, and it decreases with increasing group size. This is an indication that finding a recombined partition that is different from existing partitions is more likely in smaller group sizes.

Table C.11: Upper and lower bounds for ρ when $L = 240$

q	Lower bound	Upper bound
2	0.0042	0.3782
3	9.015e-07	4.007e-05
4	4.181e-10	1.291e-08
5	3.640e-13	8.806e-12
6	5.265e-16	1.055e-14
10	8.671e-26	1.041e-24
12	5.517e-30	5.517e-29
15	1.242e-35	9.937e-35
20	1.148e-43	6.891e-43
30	8.054e-56	3.222e-55
60	1.980e-70	3.960e-70

REFERENCES

[1] Ahmed, S., "A scenario decomposition algorithm for 0–1 stochastic programs," *Operations Research Letters*, vol. 41, no. 6, pp. 565–569, 2013.

[2] Ahmed, S., "Two stage stochastic integer programming: a brief introduction," *Wiley Encyclopedia of Operations Research and Management Science*, 2010.

[3] Ahmed, S. and Garcia, R., "Dynamic capacity acquisition and assignment under uncertainty," *Annals of Operations Research*, vol. 124, no. 1-4, pp. 267–283, 2003.

[4] Ahmed, S., Garcia, R., Kong, N., Ntaimo, L., Parija, G., Qui, F., and Sen, S., "Siplib: A stochastic integer programming test problem library," http://www.isye.gatech.edu/ sahmed/siplib, 2015, accessed: 2015-12-21.

[5] Aldasoro, U., Escudero, L. F., Merino, M., and Pérez, G., "An algorithmic framework for solving large-scale multistage stochastic mixed 0–1 problems with nonsymmetric scenario trees. part ii: Parallelization," *Computers & Operations Research*, vol. 40, no. 12, pp. 2950–2960, 2013.

[6] Ansaripoor, A. H., Oliveira, F. S., and Liret, A., "A risk management system for sustainable fleet replacement," *European Journal of Operational Research*, vol. 237, no. 2, pp. 701–712, 2014.

[7] Atamtürk, A. and Zhang, M., "Two-Stage Robust Network Flow and Design Under Demand Uncertainty," *Operations Research*, vol. 55, no. 4, pp. 662–673, Aug. 2007.

[8] ATRI, "An Analysis of the Operational Costs of Trucking: A 2014 Update," tech. rep., 2014.

[9] Bae, S. H., Sarkis, J., and Yoo, C. S., "Greening transportation fleets: Insights from a two-stage game theoretic model," *Transportation Research Part E: Logistics and Transportation Review*, vol. 47, no. 6, pp. 793–807, 2011.

[10] Beier, E., Venkatachalam, S., Corolli, L., and Ntaimo, L., "Stage- and scenario-wise fenchel decomposition for stochastic mixed 0-1 programs with special structure," *Computers & Operations Research*, vol. 59, pp. 94–103, 2015.

[11] Bellman, R., "Equipment replacement policy," *Journal of the society for industrial and applied Mathematics*, vol. 3, no. 3, pp. 133–136, 1955.

[12] Ben-Tal, a., Goryashko, A., Guslitzer, E., and Nemirovski, A., "Adjustable robust solutions of uncertain linear programs," *Mathematical Programming*, vol. 99, no. 2, pp. 351–376, Mar. 2004.

[13] Ben-Tal, A. and Nemirovski, A., "Robust Convex Optimization," *Mathematics of Operations Research*, vol. 23, no. 4, pp. 769–805, Nov. 1998.

[14] Ben-Tal, A., Golany, B., Nemirovski, A., and Vial, J.-P., "Retailer-Supplier Flexible Commitments Contracts: A Robust Optimization Approach," *Manufacturing Service Operations Management*, vol. 7, no. 3, pp. 248–271, Jul. 2005.

[15] Ben-Tal, A. and Nemirovski, A., "Robust solutions of Linear Programming problems contaminated with uncertain data," *Mathematical Programming*, vol. 88, no. 3, pp. 411–424, Sep. 2000.

[16] Bertsimas, D. and Sim, M., "The price of robustness," *Operations research*, vol. 52, no. 1, pp. 35–53, 2004.

[17] Bertsimas, D. and Sim, M., "Robust discrete optimization and network flows," *Mathematical Programming*, vol. 98, no. 1-3, pp. 49–71, Sep. 2003.

[18] Bertsimas, D. and Thiele, A., "A Robust Optimization Approach to Inventory Theory," *Operations Research*, vol. 54, no. 1, pp. 150–168, Jan. 2006.

[19] Birge, J. R., "Aggregation bounds in stochastic linear programming," *Mathematical Programming*, vol. 31, no. 1, pp. 25–41, 1985.

[20] Birge, J. R. and Louveaux, F., *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

[21] Bodur, M., Dash, S., Günlük, O., and Luedtke, J., "Strengthened benders cuts for stochastic integer programs with continuous recourse," tech. rep., Technical report. available as Optimization Online 2014-03-4263, 2014.

[22] BTS, "Freight in America: A New national picture," tech. rep., 2006.

[23] BTS, "Freight Transportation: Global Highlights 2010," tech. rep., 2010.

[24] BTS, "Freight Facts and Figures 2015," tech. rep., 2015.

[25] Büyüktahtakin, I. E. and Hartman, J. C., "A mixed-integer programming approach to the parallel replacement problem under technological change," *International Journal of Production Research*, vol. 54, no. 3, pp. 680–695, 2016.

[26] Carøe, C. C. and Schultz, R., "Dual decomposition in stochastic integer programming," *Operations Research Letters*, vol. 24, no. 1, pp. 37–45, 1999.

[27] Chen, Y. and Iyengar, G., "A new robust cycle-based inventory control policy," 2012. [Online]. Available: http://www.optimization-online.org/DB_FILE/2011/10/3210.pdf

[28] Cheung, K., Gade, D., Silva-Monroy, C., Ryan, S. M., Watson, J.-P., Wets, R. J.-B., and Woodruff, D. L., "Toward scalable stochastic unit commitment," *Energy Systems*, pp. 1–22, 2015.

[29] Cheung, R. and Chen, C., "A Two-Stage Stochastic Network Model and Solution Methods for the Dynamic Empty Container Allocation Problem," *Transportation science*, vol. 32, no. 2, pp. 142–162, 1998.

[30] Cheung, R. and Powell, W., "An algorithm for multistage dynamic networks with random arc capacities, with an application to dynamic fleet management," *Operations Research*, vol. 44, no. 6, pp. 951–963, 1996.

[31] Chocteau, V., Drake, D., Kleindorfer, P. R., Orsato, R., and Roset, A., "Collaborative Innovation for Sustainable Fleet Operations: The Electric Vehicle Adoption Decision," *SSRN Electronic Journal*, pp. 1–32, 2011.

[32] Chung, B. D., Yao, T., Xie, C., and Thorsen, A., "Robust Optimization Model for a Dynamic Network Design Problem Under Demand Uncertainty," *Networks and Spatial Economics*, vol. 11, no. 2, pp. 371–389, Sep. 2010.

[33] Crainic, T. G., Fu, X., Gendreau, M., Rei, W., and Wallace, S. W., "Progressive hedging-based metaheuristics for stochastic network design," *Networks*, vol. 58, no. 2, pp. 114–124, 2011.

[34] Crainic, T., Gendreau, M., and Dejax, P., "Dynamic stochastic models for the allocation of empty containers," *Operations Research*, vol. 41, no. 1, pp. 102–126, 1993.

[35] D'Aversa, J. S. and Shapiro, J. F., "Optimal Machine Maintenance and Replacement by Linear Programming and Enumeration," *The Journal of the Operational Research Society*, vol. 29, no. 8, pp. 759–768, 1978.

[36] Di Francesco, M., Crainic, T. G., and Zuddas, P., "The effect of multi-scenario policies on empty container repositioning," *Transportation Research Part E: Logistics and Transportation Review*, vol. 45, no. 5, pp. 758–770, Sep. 2009.

[37] DOE, "Alternative Fuels Data Center: Coca-Cola Charges Forward With Hybrid Delivery Trucks," 2012.

[38] DOE, "Alternative Fuels Data Center: Largest U.S. Port Complex Embraces LNG for Heavy-Duty Trucks," 2012.

[39] DOE, "Alternative Fuels Data Center: Kentucky Trucking Company Adds CNG Vehicles to Its Fleet," 2014.

[40] DOE, "Alternative Fuels Data Center: Coca-Cola Continues to Expand Its Heavy-Duty Hybrid Fleet in Atlanta," 2015.

[41] DOE, "Alternative Fuels Data Center: Indiana Beverage Company Invests in Alternative Fuels," 2015.

[42] Dreyfus, S. E., "A Generalized Equipment Replacement Study," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 3, pp. 425–435, 1960.

[43] Du, Y. and Hall, R., "Fleet Sizing and Empty Equipment Redistribution for Center-terminal Transportation Networks," *Management Science*, vol. 43, no. 2, pp. 145–157, 1997.

[44] EC, "Reducing emissions from transport - European Comission," 2016.

[45] EPA, "Fast Facts: U.S. Transportation Sector Greenhouse Gas Emissions 1999-2013," tech. rep., 2015.

[46] Erera, A., Morales, J., and Savelsbergh, M., "Robust optimization for empty repositioning problems," *Operations Research*, pp. 1–32, 2009.

[47] Escudero, L. F., Garín, M. A., Pérez, G., and Unzueta, A., "Scenario cluster decomposition of the lagrangian dual in two-stage stochastic mixed 0–1 optimization," *Computers & Operations Research*, vol. 40, no. 1, pp. 362–377, 2013.

[48] Escudero, L. F., Garín, M. A., Merino, M., and Pérez, G., "On bfc-msmip strategies for scenario cluster partitioning, and twin node family branching selection and bounding for multistage stochastic mixed integer programming," *Computers & Operations Research*, vol. 37, no. 4, pp. 738–753, 2010.

[49] Escudero, L. F., Garín, M. A., Merino, M., and Pérez, G., "An algorithmic framework for solving large-scale multistage stochastic mixed 0–1 problems with nonsymmetric scenario trees," *Computers & Operations Research*, vol. 39, no. 5, pp. 1133–1144, 2012.

[50] Escudero, L. F., Garín, M. A., and Unzueta, A., "Cluster lagrangean decomposition in multistage stochastic optimization," *Computers & Operations Research*, vol. 67, pp. 48–62, 2016.

[51] Espinoza, D. and Moreno, E., "A primal-dual aggregation algorithm for minimizing conditional value-at-risk in linear programs," *Computational Optimization and Applications*, vol. 59, no. 3, pp. 617–638, 2014.

[52] Fan, W. D., Machemehl, R. B., and Gemar, M. D., "Optimization of Equipment Replacement," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2292, pp. 160–170, Dec. 2012.

[53] Feng, W. and Figliozzi, M., "Impacts of Economic, Technological and Operational Factors on the Economic Competitiveness of Electric Commercial Vehicles in Fleet Replacement," in *91st Annual Meeting of the Transportation Research Board*, 2012.

[54] Fetter, R. B., "A Linear Programming Model for Long Range Capacity Planning," *Management Science*, vol. 7, no. 4, pp. 372–378, 1961.

[55] Figliozzi, M. A., Boudart, J. A., and Feng, W., "Economic and Environmental Optimization of Vehicle Fleets," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2252, pp. 1–6, 2011.

[56] Frantzeskakis, L. F. and Powell, W. B., "A Successive Linear Approximation Procedure for Stochastic, Dynamic Vehicle Allocation Problems," *Transportation Science*, vol. 24, no. 1, pp. 40–57, Feb. 1990.

[57] Gabrel, V., Lacroix, M., Murat, C., and Remli, N., "Robust location transportation problems under uncertain demands," *Discrete Applied Mathematics*, Oct. 2011.

[58] Godfrey, G. A. and Powell, W. B., "An Adaptive Dynamic Programming Algorithm for Dynamic Fleet Management, I: Single Period Travel Times," *Transportation Science*, vol. 36, no. 1, pp. 21–39, Feb. 2002.

[59] Godfrey, G. A. and Powell, W. B., "An adaptive dynamic programming algorithm for dynamic fleet management, II: Multi-period travel times," *Transportation Science*, vol. 36, no. 1, pp. 40–54, 2002.

[60] Gong, J. and Wu, C., "A transit fleet replacement model for emissions reduction," in *Proceedings of 2011 IEEE International Conference on Service Operations, Logistics and Informatics, SOLI 2011*, 2011, pp. 22–25.

[61] Guo, G., Hackebeil, G., Ryan, S. M., Watson, J.-P., and Woodruff, D. L., "Integration of progressive hedging and dual decomposition in stochastic integer programs," *Operations Research Letters*, vol. 43, no. 3, pp. 311–316, 2015.

[62] Hall, R. W. and Zhong, H., "Decentralized inventory control policies for equipment management in a many-to-many network," *Transportation Research Part A: Policy and Practice*, vol. 36, no. 10, pp. 849–865, 2002.

[63] Hartman, J. C., "An economic replacement model with probabilistic asset utilization," *IIE Transactions (Institute of Industrial Engineers)*, vol. 33, no. 9, pp. 717–727, 2001.

[64] Hartman, J. C., "A General Procedure for Incorporating Asset Utilization Decisions Into Replacement Analysis," *The Engineering Economist*, vol. 44, no. 3, pp. 217–238, 1999.

[65] Hartman, J. C., "Parallel replacement problem with demand and capital budgeting constraints," *Naval Research Logistics*, vol. 47, no. 1, pp. 40–56, 2000.

[66] Hartman, J. C., "Multiple asset replacement analysis under variable utilization and stochastic demand," *European Journal of Operational Research*, vol. 159, no. 1, pp. 145–165, 2004.

[67] Hartman, J. C. and Murphy, A., "Finite-horizon equipment replacement analysis," *IIE Transactions*, vol. 38, no. 5, pp. 409–419, 2006.

[68] Hartman, J. C. and Rogers, J., "Dynamic programming approaches for equipment replacement problems with continuous and discontinuous technological change," *IMA Journal of Management Mathematics*, vol. 17, no. 2, pp. 143–158, 2006.

[69] Jabali, O. and Erdogan, G., "Continuous Approximation Models for the Fleet Replacement and Composition Problem," tech. rep., CIRRELT, 2015.

[70] Jin, D. and Kite-Powell, H. L., "Optimal fleet utilization and replacement," *Transportation Research Part E: Logistics and Transportation Review*, vol. 36, no. 1, pp. 3–20, 2000.

[71] Johnson, C. and Singer, M., "Clean cities 2014 annual metrics report," tech. rep. NREL/TP-5400-65265, National Renewable Energy Laboratory, 2014.

[72] Karabakal, N., Lohmann, J. R., and Bean, J. C., "Parallel Replacement under Capital Rationing Constraints," *Management Science*, vol. 40, no. 3, pp. 305–319, 1994.

[73] Keles, P. and Hartman, J. C., "Case Study: Bus Fleet Replacement," *The Engineering Economist*, vol. 49, no. March, pp. 253–278, 2004.

[74] Kim, K. and Zavala, V. M., "Algorithmic innovations and software for the dual decomposition method applied to stochastic mixed-integer programs," *Optimization Online*, 2015.

[75] Klein Haneveld, W. K., Stougie, L., and van der Vlerk, M. H., "Simple integer recourse models: convexity and convex approximations," *Mathematical programming*, vol. 108, no. 2-3, pp. 435–473, 2006.

[76] Klein Haneveld, W. K. and van der Vlerk, M. H., "Stochastic integer programming:general models and algorithms," *Annals of Operations Research*, vol. 85, no. 0, pp. 39–57, 1999.

[77] Kleindorfer, P. R., Neboian, A., Roset, A., and Spinler, S., "Fleet renewal with electric vehicles at la Poste," *Interfaces*, vol. 42, no. 5, pp. 465–477, 2012.

[78] Kleywegt, A. J., Shapiro, A., and Homem-de Mello, T., "The sample average approximation method for stochastic discrete optimization," *SIAM Journal on Optimization*, vol. 12, no. 2, pp. 479–502, 2002.

[79] Kouvelis, P. and Yu, G., *Robust Discrete Optimization and Its Applications*. Dordrecht, Germany: Kluwer Academic, 1997.

[80] Laporte, G., Marín, A., Mesa, J. A., and Perea, F., "Designing robust rapid transit networks with alternative routes," *Journal of Advanced Transportation*, vol. 45, pp. 54–65, 2011.

[81] Leddon, C. and Wrathall, E., "Scheduling empty freight car fleets on the Louisville and Nashville railroad," in *Second International Symposium on the Use of Cybernetics on the Railways*, Montreal, Canada, 1967, pp. 1–6.

[82] Løkketangen, A. and Woodruff, D. L., "Progressive hedging and tabu search applied to mixed integer (0, 1) multistage stochastic programming," *Journal of Heuristics*, vol. 2, no. 2, pp. 111–128, 1996.

[83] Louveaux, F., "Discrete stochastic location models," *Annals of Operations research*, vol. 6, no. 2, pp. 21–34, 1986.

[84] Lubin, M., Martin, K., Petra, C. G., and Sandikçi, B., "On parallelizing dual decomposition in stochastic integer programming," *Operations Research Letters*, vol. 41, no. 3, pp. 252–258, 2013.

[85] Maggioni, F., Allevi, E., and Bertocchi, M., "Bounds in multistage linear stochastic programming," *Journal of Optimization Theory and Applications*, vol. 163, no. 1, pp. 200–229, 2014.

[86] Maggioni, F., Allevi, E., and Bertocchi, M., "Monotonic bounds in multistage mixed-integer linear stochastic programming," *Computational Management Science*, vol. 13, no. 3, pp. 423–457, 2016.

[87] Magnanti, T. L. and Wong, R. T., "Accelerating benders decomposition: Algorithmic enhancement and model selection criteria," *Operations research*, vol. 29, no. 3, pp. 464–484, 1981.

[88] Mak, W.-K., Morton, D. P., and Wood, R. K., "Monte carlo bounding techniques for determining solution quality in stochastic programs," *Operations Research Letters*, vol. 24, no. 1, pp. 47–56, 1999.

[89] Misra, S., "Linear programming of empty wagon disposition," *Rail International*, 1972.

[90] Nair, S. K. and Hopp, W. J., "A model for equipment replacement due to technological obsolescence," *European Journal of Operational Research*, vol. 63, no. 2, pp. 207–221, 1992.

[91] New, C. C., "Transport Fleet Planning for Multi-Period Operations," *Operational Research Quarterly*, vol. 26, no. 1, pp. 151–166, 1975.

[92] Norkin, V. I., Pflug, G. C., and Ruszczyński, A., "A branch and bound method for stochastic global optimization," *Mathematical programming*, vol. 83, no. 1-3, pp. 425–450, 1998.

[93] Ntaimo, L. and Sen, S., "The million-variable "march" for stochastic combinatorial optimization," *Journal of Global Optimization*, vol. 32, no. 3, pp. 385–400, 2005.

[94] Ntaimo, L. and Sen, S., "A comparative study of decomposition algorithms for stochastic combinatorial optimization," *Computational Optimization and Applications*, vol. 40, no. 3, pp. 299–319, 2008.

[95] Papadakos, N., "Practical enhancements to the magnantiwong method," *Operations Research Letters*, vol. 36, no. 4, pp. 444 – 449, 2008.

[96] Parthanadee, P., Buddhakulsomsiri, J., and Charnsethikul, P., "A study of replacement rules for a parallel fleet replacement problem based on user preference utilization pattern and alternative fuel considerations," *Computers and Industrial Engineering*, vol. 63, no. 1, pp. 46–57, 2012.

[97] Pedraza-Martinez, A. J. and Van Wassenhove, L. N., "Vehicle replacement in the international committee of the red cross," *Production and Operations Management*, vol. 22, no. 2, pp. 365–376, 2013.

[98] Powell, W. B., "Clearing the jungle of stochastic optimization," *Informs TutORials*, 2014.

[99] Powell, W. B., *Approximate Dynamic Programming: Solving the curses of dimensionality*. John Wiley & Sons, 2007, vol. 703.

[100] Powell, W., "A stochastic model of the dynamic vehicle allocation problem," *Transportation science*, vol. 20, no. 2, pp. 117–129, 1986.

[101] Powell, W., "An operational planning model for the dynamic vehicle allocation problem with uncertain demands," *Transportation Research Part B: Methodological*, vol. 21, no. 3, pp. 217–232, 1987.

[102] Powell, W. and Carvalho, T., "Dynamic control of logistics queueing network for large-scale fleet management," *Transportation Science*, 1998.

[103] Rajagopalan, S., "Capacity Expansion and Equipment Replacement: A Unified Approach," *Operations Research*, vol. 46, no. 6, pp. 846–857, 1998.

[104] Rockafellar, R. T. and Wets, R. J.-B., "Scenarios and policy aggregation in optimization under uncertainty," *Mathematics of operations research*, vol. 16, no. 1, pp. 119–147, 1991.

[105] Ruszczynski, A. P. and Shapiro, A., *Stochastic programming*. Elsevier Amsterdam, 2003, vol. 10.

[106] Ryan, K., Rajan, D., and Ahmed, S., "Scenario decomposition for 0-1 stochastic programs: Improvements and asynchronous implementation," *Optimization Online*, 2015.

[107] Sahinidis, N. V., "Optimization under uncertainty: state-of-the-art and opportunities," *Computers & Chemical Engineering*, vol. 28, no. 6, pp. 971–983, 2004.

[108] Sandikçi, B., Kong, N., and Schaefer, A. J., "A hierarchy of bounds for stochastic mixed-integer programs," *Mathematical Programming*, vol. 138, no. 1-2, pp. 253–272, 2013.

[109] Sandikçi, B. and Özaltin, O. Y., "A scalable bounding method for multistage stochastic programs," *SIAM Journal on Optimization*, forthcoming.

[110] Santoso, T., Ahmed, S., Goetschalckx, M., and Shapiro, A., "A stochastic programming approach for supply chain network design under uncertainty," *European Journal of Operational Research*, vol. 167, no. 1, pp. 96–115, Nov. 2005.

[111] Schultz, R., "Stochastic programming with integer variables," *Mathematical Programming*, vol. 97, no. 1-2, pp. 285–309, 2003.

[112] See, C.-T. and Sim, M., "Robust Approximation to Multiperiod Inventory Management," *Operations Research*, vol. 58, no. 3, pp. 583–594, 2010.

[113] Sen, S. and Higle, J. L., "The c 3 theorem and a d 2 algorithm for large scale stochastic mixed-integer programming: set convexification," *Mathematical Programming*, vol. 104, no. 1, pp. 1–20, 2005.

[114] Sen, S. and Sherali, H. D., "Decomposition with branch-and-cut approaches for two-stage stochastic mixed-integer programming," *Mathematical Programming*, vol. 106, no. 2, pp. 203–223, 2006.

[115] Sen, S. and Zhou, Z., "Multistage stochastic decomposition: a bridge between stochastic programming and approximate dynamic programming," *SIAM Journal on Optimization*, vol. 24, no. 1, pp. 127–153, 2014.

[116] Sethi, S. P. and Chand, S., "Planning Horizon Procedures for Machine Replacement Models," *Management Science*, vol. 25, no. 2, pp. 140–151, 1979.

[117] Shapiro, A., "On complexity of multistage stochastic programs," *Operations Research Letters*, vol. 34, no. 1, pp. 1–8, 2006.

[118] Shapiro, A., Dentcheva, D. *et al.*, *Lectures on stochastic programming: modeling and theory*. SIAM, 2014, vol. 16.

[119] Simms, B. W., Lamarre, B. G., Jardine, a. K. S., and Boudreau, a., "Optimal buy, operate and sell policies for fleets of vehicles," *European Journal of Operational Research*, vol. 15, no. 2, pp. 183–195, 1984.

[120] Sloan, T. W., "Green renewal: incorporating environmental factors in equipment replacement decisions under technological change," *Journal of Cleaner Production*, vol. 19, no. 2-3, pp. 173–186, Jan. 2011.

[121] Solyali, O., Cordeau, J.-F., and Laporte, G., "Robust Inventory Routing Under Demand Uncertainty," *Transportation Science*, vol. 46, no. 3, pp. 327–340, 2012.

[122] Solyali, O., Cordeau, J.-F., and Laporte, G., "The Impact of Modeling on Robust Inventory Management Under Demand Uncertainty," *Management Science*, vol. 62, no. 4, pp. 1188–1201, 2016.

[123] Song, Y. and Luedtke, J., "An adaptive partition-based approach for solving two-stage stochastic programs with fixed recourse," *SIAM Journal on Optimization*, vol. 25, no. 3, pp. 1344–1367, 2015.

[124] Soyster, A., "Convex Programming with Set-Inclusive Constraints and Applications to Inexact Linear Programming," *Operations research*, vol. 21, no. 5, pp. 1154–1157, 1973.

[125] Stasko, T. H. and Gao, O. H., "Reducing transit fleet emissions through vehicle retrofits, replacements, and usage changes over multiple time periods," *Transportation Research Part D: Transport and Environment*, vol. 15, no. 5, pp. 254–262, Jul. 2010.

[126] Terborgh, G., *Dynamic Equipment Policy*. New York, NY: McGraw-Hill, 1949.

[127] van der Vlerk, M. H., "Stochastic integer programming bibliography," World Wide Web, http://www.eco.rug.nl/mally/biblio/sip.html, 1996-2007.

[128] van der Vlerk, M. H., "Convex approximations for a class of mixed-integer recourse models," *Annals of Operations Research*, vol. 177, no. 1, pp. 139–150, 2010.

[129] Veliz, F. B., Watson, J.-P., Weintraub, A., Wets, R. J.-B., and Woodruff, D. L., "Stochastic optimization models in forest planning: a progressive hedging solution approach," *Annals of Operations Research*, pp. 1–16, 2014.

[130] Wagner, H. M., *Principles of operation research*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1975.

[131] Watson, J.-P. and Woodruff, D. L., "Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems," *Computational Management Science*, vol. 8, no. 4, pp. 355–370, 2011.

[132] White, W., "Dynamic transshipment networks: An algorithm and its application to the distribution of empty containers," *Networks*, 1972.

[133] Wu, P., Hartman, J. C., and Wilson, G. R., "An Integrated Model and Solution Approach for Fleet Sizing with Heterogeneous Assets," *Transportation Science*, vol. 39, no. 1, pp. 87–103, Feb. 2005.

[134] Zenarosa, G. L., Prokopyev, O. A., and Schaefer, A. J., "Scenario-tree decomposition: bounds for multistage stochastic mixed-integer programs," *Optimization Online*, 2014.

[135] Zenarosa, G. L., Prokopyev, O. A., and Schaefer, A. J., "M-smiplib: A multistage stochastic mixed-integer programming test set library," http://www.cs.cmu.edu/ gzen/m-smiplib/, 2014, accessed: 2016-10-17.

[136] Zhang, G., Smilowitz, K., and Erera, A., "Dynamic planning for urban drayage operations," *Transportation Research Part E: Logistics and Transportation Review*, vol. 47, no. 5, pp. 764–777, Sep. 2011.