



Titre: Algorithms for the Stochastic Bicycle Repositioning Problem and the
Title: Stochastic Vehicle Routing Problem

Auteur: Lucas Parada Pradenas
Author:

Date: 2024

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: Parada Pradenas, L. (2024). Algorithms for the Stochastic Bicycle Repositioning
Citation: Problem and the Stochastic Vehicle Routing Problem [Thèse de doctorat,
Polytechnique Montréal]. PolyPublie. <https://publications.polymtl.ca/61700/>

 **Document en libre accès dans PolyPublie**
Open Access document in PolyPublie

URL de PolyPublie: <https://publications.polymtl.ca/61700/>
PolyPublie URL:

**Directeurs de
recherche:** Michel Gendreau, & Jean-François Côté
Advisors:

Programme: Doctorat en Génie industriel
Program:

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

**Algorithms for the Stochastic Bicycle Repositioning Problem and the
Stochastic Vehicle Routing Problem**

LUCAS PARADA PRADENAS

Département de mathématiques et génie industriel

Thèse présentée en vue de l'obtention du diplôme de *Philosophiæ Doctor*
Génie industriel

Décembre 2024

© Lucas Parada Pradenas, 2024.

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Cette thèse intitulée :

**Algorithms for the Stochastic Bicycle Repositioning Problem and the
Stochastic Vehicle Routing Problem**

présentée par **Lucas PARADA PRADENAS**
en vue de l'obtention du diplôme de *Philosophiæ Doctor*
a été dûment acceptée par le jury d'examen constitué de :

Louis-Martin ROUSSEAU, président

Michel GENDREAU, membre et directeur de recherche

Jean-François CÔTÉ, membre et codirecteur de recherche

Walter REI, membre

Remy SPLIET, membre externe

DEDICATION

This thesis is dedicated with profound love and gratitude to my unbelievably beautiful wife Paula, whose endless love, patience, and encouragement have been my greatest strength throughout this journey. Her belief in me and her sacrifices made this achievement possible.

Secondly, I dedicate this work to my parents Lorena and Victor. Their unwavering support, guidance, and dedication have shaped me into who I am today. They instilled in me the values of hard work, perseverance, and the importance of education. Without their support, none of this would have been possible.

Thirdly, I extend this dedication to my brother Leandro, aunts Ivonne, Marcia, Arturo, Moni, Momo, and the rest of my family. Their understanding, encouragement, and moral support have been invaluable. They have always been there for me, providing a foundation of love and strength.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude to my two supervisors, Jean-Francois Côté and Michel Gendreau. Their guidance, support, and invaluable insights have been instrumental in completing this thesis.

Secondly, I would like to express my sincere gratitude to Deputy Rachel Bendayan's office for the legal immigration support they provided me. I want to mention Théo Rivière, whose proactive and efficient approach was instrumental in helping me resolve complex legal immigration issues. His assistance was crucial in navigating the intricacies of this process.

Lastly, I am grateful to the Centre Pauline Julienne and its francisation teachers. They have given me the tools to improve my French skills, significantly contributing to my academic and personal growth.

RÉSUMÉ

Un repositionnement efficace des vélos et les tournées des véhicules sont cruciaux pour les systèmes de transport et de logistique. La programmation mathématique présente des défis fondamentaux dans ce domaine. Alors que les problèmes de tournées de véhicules ont été largement étudiés depuis la fin des années 1950, le repositionnement des vélos est un domaine de recherche relativement récent dans lequel les chercheurs se sont déjà attaqués à divers problèmes combinatoires. L'objectif de ce dernier est d'assurer des niveaux de service élevés et de répondre à la demande des utilisateurs, également appelé maximisation des exigences de niveau de service.

Faire face à l'incertitude ajoute à la complexité des deux problèmes. Dans le domaine des tournées de véhicules, la variante la plus étudiée prend en compte l'incertitude des clients ayant des demandes aléatoires. Parallèlement, lors du repositionnement, il est crucial d'anticiper les demandes de voyage incertaines. Cela est particulièrement difficile dans les systèmes de partage de vélos réels, avec des milliers de stations et des demandes de déplacements quotidiens élevées. Actuellement, les méthodes de pointe ont du mal à gérer des systèmes de grande taille. Dans les tournées de véhicules avec des demandes stochastiques, les méthodes exactes ont du mal à résoudre les instances comportant plus d'une centaine de clients, de longues tournées et des flottes de grande taille.

Cette thèse comble les lacunes en proposant de nouvelles méthodes exactes qui se révèlent très efficaces en pratique. Bien que le problème des tournées de véhicules avec des demandes stochastiques repose sur une base théorique solide en termes de méthodes exactes, cela n'a pas été étendu au repositionnement des vélos dans des conditions d'incertitude. Nous exploitons cette base en proposant une nouvelle variante de la méthode en forme de L en nombre entière, et nous proposons des implémentations pour le problème de tournées de véhicules avec des demandes stochastiques et le repositionnement des vélos. De plus, nous abordons les exigences de niveau de service dans les systèmes de partage de vélos grandeur nature en dévoilant un nouveau

cadre centré autour d'une méthode exacte pour un problème nouvellement proposé. L'objectif est de montrer que ces nouvelles variantes algorithmiques exactes peuvent considérablement améliorer les solutions non seulement pour le problème de repositionnement stochastique des vélos, mais également pour le problème de tournées de véhicules avec des demandes stochastiques.

ABSTRACT

Efficient bicycle repositioning and vehicle routing are crucial for transportation and logistics systems. Mathematical programming presents fundamental challenges in this area. While vehicle routing problems have been extensively studied since the late 1950s, bicycle repositioning is a relatively newer area of research where already researchers have tackled various combinatorial problems. The goal of the latter is to ensure high service levels and meet user demand, also known as maximizing service level requirements.

Dealing with uncertainty adds complexity to both problems. In vehicle routing, the most studied variant considers uncertainty in customers with random demands. Meanwhile, in repositioning, it's crucial to anticipate uncertain trip demands. This is particularly challenging in real-life bicycle-sharing systems with thousands of stations and high daily trip demands. Currently, state-of-the-art methods struggle to handle large system sizes. In vehicle routing with stochastic demands, exact methods have difficulty solving instances with more than one hundred customers, long routes, and large fleet sizes.

This thesis bridges gaps by proposing new exact methods that turn out to be very efficient in practice. While the vehicle routing problem with stochastic demands has a solid theoretical foundation in exact methods, this has not been extended to bicycle repositioning under uncertainty. We leverage this foundation by proposing a new variant of the integer L-shaped method, and we propose implementations for the vehicle routing problem with stochastic demands and bicycle repositioning. Furthermore, we tackle service-level requirements in real-life-sized bicycle-sharing systems by unveiling a new framework centered around an exact method for a newly proposed problem. The aim is to showcase that these new exact algorithmic variants can significantly elevate solutions for not only the stochastic bicycle repositioning problem but also the vehicle routing problem with stochastic demands.

TABLE OF CONTENTS

DEDICATION	iii
ACKNOWLEDGEMENTS	iv
RÉSUMÉ	v
ABSTRACT	vii
TABLE OF CONTENTS	viii
LIST OF TABLES	xii
LIST OF FIGURES	xiv
LIST OF SYMBOLS AND ACRONYMS	xv
LIST OF APPENDICES	xvii
CHAPTER 1 INTRODUCTION	1
1.1 Challenges in the Stochastic Bicycle Repositioning Problem and Stochastic Vehicle Routing Problem	3
1.2 Objectives	4
1.2.1 Specific Objectives	4
CHAPTER 2 REVUE DE LITTÉRATURE / LITERATURE REVIEW	6
2.1 Related Literature: An Improved Integer L-Shaped Method for the Vehicle Routing Problem with Stochastic Demands	6
2.1.1 Origins and History of the Vehicle Routing Problem with Stochastic Demands	6
2.1.2 The Detour-to-Depot and Optimal Restocking Policies	7
2.1.3 Use of Exact Methods and the Integer L-Shaped Method	8

2.1.4	Towards an Improved Integer L-Shaped Method: A Disaggregated Monotonic Recourse	9
2.2	Related Literature: Service Level Requirements and Stochastic Bicycle Sharing Systems	11
2.2.1	Origins and History of Bicycle Sharing Systems	11
2.2.2	Service Level Requirements and Combinatorial Optimization Problems	11
2.2.3	Real-life Sized Bicycle Sharing Systems	13
2.2.4	Station-Based Repositioning of Bicycles	15
CHAPTER 3	THESIS ORGANIZATION	17
CHAPTER 4	ARTICLE 1: A disaggregated integer L-shaped method for stochastic vehicle routing problems with monotonic recourse	19
4.1	Introduction	20
4.2	Mathematical formulation	22
4.3	Monotonicity of the recourse function	25
4.4	Disaggregated integer L-shaped method	32
4.4.1	New optimality cuts	34
4.4.2	Lower bounding functionals	36
4.4.3	Recourse lower bounds	38
4.5	Implementation of the DL-shaped method	42
4.6	Computational experiments	43
4.6.1	Characterization of the instances	44
4.6.2	Lower bounds on the recourse	45
4.6.3	Results for the instances of the literature	47
4.6.4	Results for the new instances	49
4.7	Conclusion	50
CHAPTER 5	ARTICLE 2: An Integer L-Shaped Method for the Static Stochastic Bicycle Repositioning Problem	53
5.1	Introduction	53

5.2	Mathematical formulation	55
5.2.1	Recourse Calculation	57
5.2.2	A New Dynamic Programming Formulation for the Recourse .	59
5.2.3	Complexity	59
5.3	Lower Bounds	63
5.3.1	Lower Bounds for the Fleet Size	63
5.3.2	Lower Bounds for the Recourse	66
5.4	The DL-shaped Method	67
5.5	Valid Inequalities	69
5.5.1	Recourse Lower Bound Inequalities	69
5.5.2	Infeasible Path Inequalities	70
5.5.3	Infeasible Set Inequalities	70
5.5.4	Infeasible Arc Inequalities	71
5.6	Implementations of the DL-shaped Method	71
5.7	Computational Results	73
5.7.1	Characterization of the Instances	74
5.7.2	Results for Values of ϵ and δ	75
5.7.3	Effectiveness of the Valid Inequalities	78
5.7.4	Results for Lower Bounds on the Fleet Size	79
5.7.5	Results for the new instances	82
5.8	Conclusions	85
CHAPTER 6 ARTICLE 3: Service Level Requirements for Real-Life-Sized		
	Bicycle Sharing Systems	86
6.1	Introduction	87
6.2	Service Level Requirements	90
6.2.1	Problem Definition	91
6.3	The Target-Level Problem	92
6.3.1	A Scenario Formulation for the TLB	92
6.3.2	A Complete Formulation for the TLB	93
6.3.3	The Benders Decomposition Algorithm	95

6.3.4	A First Variant Formulation of the TLB	98
6.3.5	A Second Variant Formulation of the TLB	100
6.3.6	A Third Variant Formulation of the TLB	101
6.3.7	Optimizing the Station Capacity	102
6.4	The Bicycle Rebalancing Problem	104
6.4.1	Problem Definition	104
6.4.2	Cost of a Route	105
6.4.3	The Continue-to-Next Policy	105
6.4.4	The Restocking Trips Policy	106
6.4.5	Feasibility of a Route	107
6.4.6	Solution Method	108
6.5	Data Generation	109
6.5.1	Random Data Generation for the TLB	112
6.5.2	Characterization of the Instances	113
6.6	Computational Results	116
6.6.1	Results for the TLB	116
6.6.2	Results for the BRP	124
6.7	Conclusion	127
6.8	Acknowledgments	129
CHAPTER 7	CONCLUSION	130
7.1	Synthèse des travaux / Summary of Works	130
7.2	Limitations de la solution proposée / Limitations	131
7.3	Améliorations futures / Future Research	132
REFERENCES		133
APPENDICES		144
A.1	Proofs of Proposition 5, 6 and 7	144
A.2	Proof of Proposition 12	147

LIST OF TABLES

Table 4.1	Relative gap of lower bounds on the instances of [1]	46
Table 4.2	Relative gap of lower bounds on the CVRP instances	46
Table 4.3	Performance comparison with [1, 2] on the instances of [1] . . .	48
Table 4.4	Performance comparison with [3–5] on the CVRP instances. *	
	indicate that the authors use the OR policy.	48
Table 4.5	Solved instances and computation times of the DL-shaped method	
	for the new set	50
Table 5.1	Results for the instances of [6]	76
Table 5.2	Results for only the multi vehicle instances	77
Table 5.3	Convergence with and without the S-cuts and Infeasible Sets.	78
Table 5.4	Effectiveness of the vehicle lower bound K in a 5 minute-run.	80
Table 5.5	Lower Bound Values on Fleet Size for $\epsilon = 0.2$ and $\delta = 0.0$. . .	81
Table 5.6	Results for Multicut algorithm in the new instances.	83
Table 5.7	Results for the new instances.	83
Table 5.8	Convergence with and without the S-cuts and Infeasible Sets	
	for the new instances.	84
Table 5.9	Effectiveness of the vehicle lower bound K on the new instances.	84
Table 6.1	Summary of Instances with $ \Omega = 100$ for the TLB.	114
Table 6.2	Summary of Instances for the BRP.	116
Table 6.3	Performance of Models M1–M3 in Cities with 100 Scenarios. .	119
Table 6.4	%Gaps with UB_{400} of Models M1–M4 Using 100 Scenarios . .	120
Table 6.5	Average Percentage of Satisfied Trips with Fewer Available Bi-	
	cycles	121
Table 6.6	Increase or Decrease in Average Satisfied Trips with Respect	
	to M2 for Models M5, M6, and M7.	122
Table 6.7	Results for Models M5–M7 Using a Budget of 5% of $\sum_{i \in N} h_i$,	
	and 100 Scenarios	124
Table 6.8	BRP Results with $\delta = 1$	125

Table 6.9	BRP Results for All δ Values	126
Table 6.10	BRP Driver Counts for All δ Values.	128

LIST OF FIGURES

Figure 4.1	Largest solved and smallest unsolved instances in the literature with their filling coefficient	51
Figure 5.1	A Solution to the 1-Bicycle Repositioning Problem.	61
Figure 6.1	Min-Cost Flow Formulation $G^\omega = (V^\omega, A^\omega)$ for one scenario of the TLB with $n = 3$ trips and $t_{\max} = 4$	94
Figure 6.2	Histograms of Average Daily Trips in the [8h,22h] Interval of the Largest BSS in July 2023.	111
Figure 6.3	Comparison of results for Models M1–M3 with different scenarios: (a) Average % Gap with UB_{400} and (b) Average time taken	118
Figure 6.4	Montréal, New York, and Mexico City average percentage of satisfied trips with fewer available bicycles	122

LIST OF SYMBOLS AND ACRONYMS

CVRP	Capacitated vehicle routing problem
VRPSD	Vehicle routing problem with stochastic demand
BC	Branch-and-cut
BP	Branch-and-price
MDP	Markov decision process
CDF	Cumulative distribution function
RV	Random variable
DTD	Detour-to-depot
OR	Optimal restocking
LBF	Lower bounding functional
DL-shaped	Dissaggregated integer L-shaped
MP	Master problem
P-Cuts	Path cuts
S-Cuts	Set cuts
BRP	Bicycle repositioning problem
1-PTSD	one-commodity pickup and delivery traveling salesman problem
SPDVRP	single- commodity vehicle routing problem with split pickups and deliveries
SBRP	Static stochastic bicycle repositioning problem
CG	Column generation
TLB	Target-level problem
DP	Dynamic programming
NP	Non deterministic polynomial time
SLR	Service level requirements
OSC	Optimal station capacity problem
GBFS	General Bikeshare Feed Specification
CN	Continue-to-next

RT	Restocking trips
ALNS	Adaptative Large Neighborhood Search

LIST OF APPENDICES

Appendix A	Monotonicity Proofs for the Vehicle Routing Problem with Stochastic Demands.	144
------------	---	-----

CHAPTER 1 INTRODUCTION

Several complex combinatorial problems are of significant interest to the operations research community, particularly in the context of transportation and logistics systems. This thesis focuses on two such problems: the vehicle routing problem (VRP) and the bicycle repositioning problem (BRP). Both issues are particularly relevant in the context of bicycle sharing systems (BSS), which offer convenient, low-emission transportation options and serve as alternatives to traditional urban transit. BSS typically consist of stations where users can rent and return bicycles, playing a crucial role in facilitating short-distance travel and integrating with broader public transport networks.

Among the two problems studied in this thesis, vehicle routing problems (VRPs) have broader applicability, as they can model various logistics and delivery services, including goods distribution, parcel delivery, waste collection, and public transportation scheduling, among others. Effectively addressing vehicle routing problems can significantly improve operational efficiency, reduce environmental impact, and enhance customer satisfaction across multiple industries.

Additionally, we focus on specific routing and bicycle repositioning problems where the input data is only partially known and treated as random variables, placing them within the realm of stochastic optimization. This introduces an added layer of complexity, especially when evaluating decisions made under uncertainty, a concept referred to as recourse in the literature. The actual values of these variables may only be realized after initial decisions are made based on a specific set of scenarios, or they may evolve dynamically throughout the solution design process. The work of Tillman (1969) introduced the vehicle routing problem with stochastic demands (VRPSD), which is the most extensively studied variant in stochastic vehicle routing problems (SVRPs). Following the influential research by Benchimol et al. (2011), bicycle repositioning problems (BRPs) gained prominence in the mathematical programming community and, more recently, the stochastic bicycle repositioning

problems (SBRPs).

The main challenge of this thesis is to design efficient exact algorithms for both the VRPSD and SBRP. Both belong to the NP-hard class, and considerable efforts have been devoted to designing algorithms for them. Significant gaps remain, particularly in the SBRP, where rebalancing bicycles in large networks of thousands of stations to meet random demands is challenging for existing exact algorithms. Modern bike-sharing systems often experience daily trip demands nearing one hundred thousand, while current state-of-the-art methods are mostly heuristic and only address smaller scales.

For the VRPSD, the exact methods used in the literature are built upon a significantly more extensive theoretical background. They are divided into two categories in the literature: integer L-shaped methods and branch-and-price (B&P) methods. The integer L-shaped method can handle solutions with long routes (forty or more customers per route). It is compatible with various demand distributions, including continuous ones, and can manage different types of randomness sources such as uncertain customer presence and random customer demands. However, it is less efficient for instances requiring larger vehicle fleets. On the other hand, B&P methods have been proven to be effective for situations that involve a high volume of short routes. Nonetheless, they do have significant limitations when it comes to storing complete routes in the labeling algorithms for column generation pricing problems.

This thesis bridges the gaps in exact methods for both VRPSD and SBRP by proposing new algorithms that are highly efficient in practice. For the VRPSD, this thesis proposes the disaggregated integer L-shaped method, an enhancement of the original method of [7] that accelerates convergence by computing on-the-fly lower bounds for parts of a given solution. This new algorithm is then leveraged for an implementation of the SBRP. Finally, the thesis proposes a framework that maximizes service level requirements in real-life-sized BSS, built around the exact resolution of a newly proposed problem. We anticipate demonstrating that these proposed exact variants enhance solutions for the SBRP and offer significant improvements for a broader family of VRPSD problems.

1.1 Challenges in the Stochastic Bicycle Repositioning Problem and Stochastic Vehicle Routing Problem

The challenges in stochastic bicycle repositioning are extensive, focusing on the effectiveness of exact and heuristic solution methods, especially when dealing with large-scale or real-world instances. On the other hand, the challenges in the stochastic vehicle routing problem, particularly when dealing with the variant of random customer demands, are more specific. This is especially true when using exact methods to solve instances involving one hundred or more customers, requiring multi-vehicle solutions with lengthy routes covering several dozen customers. This dissertation addresses these challenges by proposing the following open questions in italics:

- **Lack of tailored, exact algorithms:** When addressing stochastic optimization problems, such as the SBRP, much of the existing specialized literature depends on decomposition techniques, like the integer L-shaped method. While these methods offer general frameworks adaptable to various problem families, they exhibit known shortcomings leading to poor convergence results. These limitations can be likened to bottlenecks that hinder the achievement of optimal solutions. Interestingly, these bottlenecks have been studied in the context of the VRPSD. The VRPSD literature has implemented several speed-up techniques to address the bottlenecks. Furthermore, many researchers have applied various versions of the VRP to solve the SBRP. It seems then that a good starting point would be to use this experience, experiment on the VRPSD as the base problem, and then seek to extend the results to the SBRP.

Can potential bottlenecks in current exact algorithms, particularly the integer L-shaped method, be identified?

Can the experience of the literature on VRPSD be employed to detect the bottlenecks and improve the integer L-shaped method?

How will the enhanced integer L-shaped algorithm perform in challenging instances for the VRPSD, specifically those requiring large vehicle fleets and long routes?

How can this improved integer L-shaped method be implemented for the SBRP?

- **Lack of capability to address real-life sized BSS:** Whether employing exact or heuristic methods, current approaches struggle to handle the scale of real-life BSS. Instances studied in the literature typically involve 50 to 400 stations and a few hundred daily origin-destination trips to model uncertain future customer demands. However, contemporary BSS in North America, for example, boasts several thousand stations and nearly one thousand daily trips on average, highlighting a substantial disparity between research methodologies and real-world complexities of service level requirement maximization.

Can methodologies be developed to generate and effectively solve real-life sized BSS instances, bridging this gap between theoretical advancements and practical applicability?

1.2 Objectives

This dissertation focuses on addressing the challenges of stochastic bicycle repositioning and vehicle routing problems and aims to advance solution methodologies for large-scale instances by pursuing three specific objectives:

1.2.1 Specific Objectives

1. To improve on the integer L-shaped method, particularly for the case of the VRPSD.
 - To identify and generalize the existing speed-ups in the VRPSD literature and propose new ones.
 - To propose new algorithmic components based on the speed-ups.
 - To propose new valid inequalities and theoretical bounds that improve the algorithm's convergence.
 - To test the new algorithm against state-of-the-art implementations of the integer L-shaped method on benchmark sets of instances.

- To propose a new challenging set of instances.
2. To implement the improved integer L-shaped method for the SBRP.
 - Propose suitable SBRP formulations for implementing the improved integer L-shaped method.
 - To propose new valid inequalities and theoretical bounds that improve the algorithm's convergence for the SBRP.
 - To test the new algorithm against the state-of-the-art ones on benchmark instances.
 - To propose a new challenging set of instances.
 3. To develop methodologies capable of generating and effectively solving real-life-sized instances of BSS, aiming to bridge the gap between theoretical advancements and practical applicability.
 - To identify key performance metrics that characterize real-life-sized BSS.
 - To propose new efficient formulations for these large-sized BSS.
 - To implement efficient exact and heuristic methods.
 - To propose new valid inequalities and theoretical bounds that improve the convergence of the algorithms.
 - To generate instances from real-life data of these large-sized BSS.

CHAPTER 2 REVUE DE LITTÉRATURE / LITERATURE REVIEW

This section reviews the literature on algorithms for the stochastic bicycle repositioning problem and the stochastic vehicle routing problem with uncertain demands. We specifically focus on the three main objectives outlined in Section 1.2. As mentioned in the introduction of this thesis, these three specific objectives aim to achieve the primary goal of designing efficient exact algorithms for both routing and bicycle repositioning problems under uncertainty.

Section 2.1 presents the literature on the first main objective of an improved integer L-shaped method for the VRPSD. Section 2.2 presents the literature associated with the second and third specific objectives: to implement the integer L-shaped method in the SBRP and develop methodologies for service level requirement maximization in real-life-sized BSS.

2.1 Related Literature: An Improved Integer L-Shaped Method for the Vehicle Routing Problem with Stochastic Demands

2.1.1 Origins and History of the Vehicle Routing Problem with Stochastic Demands

The stochastic vehicle routing problem with random customer demands and multiple depots by [8] is considered the first studied version of the VRPSD. Nowadays, the state-of-the-art exact methods can solve instances with slightly more than one hundred customers [5]. By this measure, since [8], the field has linearly increased over time in its solution capability. Although this is considerable progress for an NP-Hard problem like the VRPSD, [9] conclude that we are still in the early stages of the field, with room for significant improvement in exact solution methods. This dissertation focuses on the exact methods developed for the VRPSD.

The a priori and re-optimization paradigms are the two main modeling paradigms used in the literature for the VRPSD. The choice of either of these paradigms depends on when the solution process reveals the stochastic parameters. [10] formalized the a priori paradigm, which relies on designing a set of fixed routes. Re-optimization refers to formulating a Markov decision process (MDP) by [11], enabling the construction and adjustment of routes as information gradually unfolds. In recent decades, the increasing availability of information technologies has simplified the gathering of real-time information that helps decision-makers re-optimize a priori routes on the fly. Consequently, new paradigms have emerged, such as the dynamic and online vehicle routing problems, where the parameters update throughout the solution process. A comprehensive survey on the dynamic vehicle routing problem is available in [12].

2.1.2 The Detour-to-Depot and Optimal Restocking Policies

Cost computation policies for solutions to the VRPSD are referred to as recourse policies. In general, recourse refers to the corrective actions a decision-maker takes in response to unexpected events, which are represented as random variables. Therefore, the recourse policy for the VRPSD involves the decisions made after random customer demands are realized.

The most commonly discussed recourse policies in the literature are the Detour-to-Depot (DTD) and Optimal Restocking (OR) policies. Under the DTD policy, restocking trips are permitted only in the event of a failure, which occurs when a customer's actual demand exceeds the vehicle's remaining capacity. Studies of the VRPSD with this policy include the works of [13], [14], and [1]. For the OR policy, vehicles can perform preventive returns to the depot to avoid costly failures between any two given customers. [15] have shown that solving a Bellman equation can calculate the OR policy. Relevant works under this policy include [16], [17], [5], and [2]. Additionally, variants of these two policies have also been studied. For example, [18] propose a recourse policy based on pairwise sets of a priori routes. Also, in [19], an a priori planned tour is partly re-optimized using an MDP limited to a restricted set of states. [20] applies a different form of partial re-optimization,

allowing the order of successive customers in a planned route to be switched.

Several numerical experiments have been performed in the literature to compare the effectiveness of the proposed policies. Those of [16] show that the OR policy might yield some savings compared to the DTD policy, but they also noticed that the optimal a priori routes are mostly the same as those from the DTD policy. Their results also indicate that more instances can be solved using the DTD policy. [17] calculated the cost of the DTD policy based on an optimal OR policy solution, revealing an average increase in the optimal value of approximately 0.63%.

Additionally, according to [5], the advantage of implementing the OR policy instead of the DTD policy is minimal. However, it is possible to achieve an average savings of 1.6% by reducing the available routes and allowing the capacity to be exceeded. They also noted that the OR policy's benefit decreases when the demand variability increases. Finally, the partial re-optimization policy of [20] can further reduce costs compared to the OR policy by 0.73% on average. There are still open questions from these observations because the analyses are limited to relatively few instances of small sizes. In light of these results, the DTD policy is still worth studying.

2.1.3 Use of Exact Methods and the Integer L-Shaped Method

There are two primary types of exact methods found in the literature for solving the Vehicle Routing Problem with Stochastic Demand (VRPSD): integer L-shaped methods and branch-and-price (B&P) methods. The integer L-shaped method tackles the VRPSD using a branch-and-bound framework, incorporating on-the-fly separation of optimality and feasibility cuts. In contrast, the B&P method addresses the VRPSD by column generation within a similar branch-and-bound framework.

Both the integer L-shaped method and B&P possess distinct strengths and weaknesses when addressing the VRPSD. The integer L-shaped method is effective for managing long routes and diverse demand distributions; however, it becomes less efficient when dealing with scenarios that require more than three vehicles. Although several researchers have made significant contributions to developing the so-called

lower bounding functional (LBFs) [1, 2, 14] to address this limitation, solving instances with numerous vehicles still poses a challenge. Generally, an LBF for the VRPSD consists of an inequality that provides a lower bound on the recourse value of a set of solutions, along with an activation function that selectively activates the inequality.

On the other hand, B&P methods demonstrate efficiency for instances requiring many short routes, yet they also have notable limitations. For instance, they have primarily been applied to demands that follow Poisson distributions. The OR policy requires storing entire routes within the column generation subproblem labeling algorithm, which considerably restricts the length of the routes the method can accommodate. A recent study [21] has shown that, for various families of the OR policy, this pricing problem is strongly NP-hard.

Furthermore, the OR policy does not scale well when both demands and vehicle capacities are high since the size of the pricing subproblem is dependent on capacity. To address this limitation, some literature suggests dividing each instance’s capacity and expected demands by their greatest common divisor. This process generates subproblems of manageable size during the column generation procedures. Although the resulting instances are typically good approximations of the original problems, they are generally not equivalent and may yield different optimal solutions and values.

2.1.4 Towards an Improved Integer L-Shaped Method: A Disaggregated Monotonic Recourse

A key challenge in mathematical programs with recourse, such as the VRPSD, is efficiently computing and estimating the recourse function. One potential approach is to break down the recourse function into smaller, more manageable functions. This idea of disaggregating the recourse by separating it into independent disjoint functions was explored by [22]. The study shows that the recourse can be easily disaggregated when the second-stage linear programming matrix can be decomposed into blocks. In such cases, a variable is added in the first stage for each block. Then, in each iteration, the second-stage problem of each block is solved, and an optimality

cut containing only the first-stage variables of the block is added. Extending this approach to problems with non-decomposable second-stage matrices poses a significant challenge.

The second stage of vehicle routing problems does not present this block structure. One of the reasons for this is that the uncertain parameters are progressively revealed as the clients are visited. However, the first-stage solutions of vehicle routing problems have a detachable structure, as they form independent routes. Some works have introduced techniques to exploit this separability. [23] proposed to disaggregate the recourse by routes, [24] proposed to bound the recourse for sets of nodes, and [2] for unstructured sets. These methods utilize linear inequalities to introduce additional variables in the first stage to constrain certain aspects of the second stage. Numerical experiments show that these valid inequalities help reduce computation times. The superiority of our method in Chapter 4, over these previous works comes from new optimality cuts and new LBFs that we propose, which simultaneously bound the recourse for paths and sets of nodes that apply to a broad range of first-stage solutions.

A relevant concept central to the successful implementation of our method is the so-called monotonicity of the recourse function. The topic of monotonicity received little attention in the VRPSD literature. To our knowledge, the articles by [25] and [26] constitute the only theoretical works on the matter. They both give sufficient conditions for the probability of a failure to increase at each customer on a route, assuming independent and identically distributed (i.i.d.) demands. In Chapter 4, we demonstrate that this is sufficient for the recourse function to be monotonic under the DTD policy. We have also derived sufficient conditions for the monotonicity of the recourse function. These conditions are applicable in the case of independent but not identically distributed variables. Our findings establish that Poisson, normal, binomial, Erlang, and negative binomial demands exhibit monotonicity under specific conditions.

2.2 Related Literature: Service Level Requirements and Stochastic Bicycle Sharing Systems

2.2.1 Origins and History of Bicycle Sharing Systems

Since their introduction in the 1960s' [27], BSS have gained immense significance and scale in contemporary urban environments and will most likely continue to do so given their ability to reduce carbon emissions and offer a flexible, eco-friendly mode of transit (see the reviews of [28–31] for historical and contemporary evidence of the benefits above of the BSS). Over the past few decades, these systems have experienced remarkable growth, with some expanding to encompass thousands of bicycles and hundreds of stations, such as the BSS in Montréal, Canada [32]. Furthermore, recent trends suggest that the worldwide market size for BSS will continue to grow and nearly triple its market valuation by the end of the 2020s [33]. The fascination for BSS is also profound in the scientific community, evidenced by studies that draw parallels between the system's requirements and human needs, akin to Maslow's renowned hierarchy of needs [34]. And, it would seem that studies detailing the system need will begin to proliferate because there is now evidence that some BSS can fail quite spectacularly in some cities [35].

2.2.2 Service Level Requirements and Combinatorial Optimization Problems

A crucial aspect of ensuring the viability of a Bicycle Sharing System (BSS) is effective management of service level requirements (SLR). This aspect goes beyond the system itself, as BSS often function as integral components within larger, multi-modal, and integrated transportation networks [36]. In general terms, SLR refers to the minimum availability of bikes and/or empty docks at each station, which is essential for user satisfaction and system reliability.

In this context, the field of operations research has significantly contributed over the past two decades by identifying various SLR-related goals and challenges. This development traces back to the groundbreaking work of [37], which is considered the

first article to formulate a mathematical program for the bicycle repositioning problem. The challenges associated with SLR encompass several well-known combinatorial optimization problems, such as service network design [38], vehicle routing [39], inventory management [40], and many others. Furthermore, the mathematical programming community continues to evolve by developing and refining models and solution methodologies to address and improve these recognized challenges.

The many diverse combinatorial optimization problems in BSS are due to significant variations in their designs and features, tailored to meet the specific needs of users and cities. For instance, some BSS adopt free-floating bicycles, allowing users to pick up and drop off bikes anywhere in the service area for spontaneous trips. Such free-floating BSS gives rise to problems involving Markov decision processes [41]. In contrast, station-based BSS have fixed docking stations where bikes must be returned after use, giving rise to inventory routing problems and vehicle routing problem variants [42, 43]. Multi-modal integration with public transit networks further enhances user experiences, enabling the seamless combination of bikes with other transportation modes and resulting in multilayer network design problems for the different modes [36, 44]. Similar problems arise when integrating electric scooters into BSS, as their usage differs from bicycles, often serving for short trips in high-employment areas [45]. Another category of problems occurs when BSS design prohibits the repositioning vehicle fleet from using stations for temporary bicycle storage and transshipment [46] or when a limit is imposed on the number of times that a bicycle can be loaded and unloaded from a repositioning vehicle [47]. In all these problems, the objective depends on whether a repositioning time is considered, with goals typically centered on minimizing the total time for bicycle repositioning or routing costs of the repositioning fleet. Additional objectives often seek to minimize penalties for unsatisfied or partially satisfied demand at stations [48]. Furthermore, BSS features may include mobile apps, user accounts, and memberships, providing exclusive benefits and discounted rates to encourage desired user behaviors, such as bicycle returns to specific locations or adherence to designated routes. Such features give rise to pricing problems in BSS, where the objective aims to maximize utility or

profit [49].

2.2.3 Real-life Sized Bicycle Sharing Systems

Despite the mentioned interest in SLR, state-of-the-art methods, particularly exact ones, struggle to cope with the current dimensions posed by contemporary BSS. Major urban centers in North America feature expansive BSS networks, with thousands of stations, bicycles, and nearly one hundred thousand daily trips. Managing these systems involves intricate decision-making, encompassing bicycle repositioning strategies, pickup and delivery specifics, and strategic allocation of rebalancing vehicles. Formulating these problems typically leads to large models that exceed the capacity of commercial solvers for mixed-integer programming. Furthermore, integrating additional BSS features such as operational timelines, which entail fine-graining into minutes or hourly intervals at which rebalancing operations are necessary, amplifies the inherent difficulty.

The largest instances solved using exact methods typically involve 50 to 150 stations, which is significantly fewer than the number of stations in current real-life BSS. These problems are based on artificial data generated from BSS that conform to the GBFS and model various variations of the inventory routing problem [39, 50, 51], as well as vehicle routing problems [6]. The former differs from traditional vehicle routing problems in that they impose only one rebalancing visit per station. In contrast, inventory management overlooks vehicle route design. Furthermore, many of these references integrate additional BSS features such as operational timelines, which entail fine-graining into minutes or hourly intervals at which rebalancing operations are necessary. To solve them, the references implement decomposition approaches like the L-shaped method [6], Dual decomposition methods [51], or Benders decomposition [52]. Still, the resulting problems, even for the 50-150 stations, are so complex that few optimal solutions are reported with these exact methods.

An exception is the work of [53] because they address the problem of optimizing stations in real-life-sized US-based BSS, utilizing data from 2016. The authors' model calculates station capacity by reallocating bicycle docks in the network. However,

unlike many other references mentioned, the author’s model does not account for the rebalancing of bicycles by a capacitated vehicle fleet.

Besides this exception, the size disparity between real-life-sized BSS and the solved instance sizes illustrates the limitations in handling real-sized BSS by state-of-the-art exact methods, restricting their practical application and managerial insights. We detail the sizes of current BSS in Section 6.5, but one example is New York’s Citi Bike BSS, allegedly the largest in North America. In 2023, this system boasted over 2000 operational stations. Furthermore, according to [54], during July 2023, the system recorded an average of 117,762 trips daily, peaking at 142,879 trips on July 26. These sizes demonstrate the need for models catering to such expansive systems.

The latest heuristic methods also struggle with real-life sizes. For instance, [55] used a math heuristic to solve instances based on San Francisco’s Bay Wheels BSS, resulting in 340 stations. Our study developed instances from the same BSS, with over 500 stations using peak-usage summer data.

In this dissertation, we propose in Chapter 6 a two-step approach to efficiently manage SLR in real-life-sized BSS. Drawing inspiration from the two-step approach outlined by [50], we first introduce a novel problem denoted the target-level problem, which computes a target quantity of bicycles for each station. This computation considers the system’s unknown future bicycle trip demands, aiming to satisfy as many as possible. We solve this problem using the Benders decomposition algorithm with Benders cuts [56] as an optimality cut. The sub-problem in our decomposition algorithm is modeled as a min-cost flow problem, enabling us to utilize specialized solvers that implement polynomial algorithms for it. In the second step of our approach, we heuristically design vehicle routes to rebalance the bicycle quantities from initially given ones up to the targets. For this second step, called the bicycle rebalancing problem, we heuristically design vehicle routes to adjust the bicycle quantities to meet the targets. We utilize efficient dynamic programming algorithms to compute the solution cost and propose methodological enhancements to ensure the feasibility of the vehicle routes.

By decomposing the SLR into the target-level problem and the bicycle rebalancing

problem, we effectively overcome challenges associated with online station rebalancing, thus enhancing our ability to tackle larger BSS instances.

2.2.4 Station-Based Repositioning of Bicycles

In the specific case of station-based BSS and uncertain trip demands, we find the SBRP. The problem was proposed in [6], and the objective is to minimize the repositioning time while allowing the transshipment of bicycles between stations. The authors propose a two-stage mathematical program with a recourse in which each feasible first-stage solution consists of a set of routes. In the second stage, the unsatisfied demands of the stations are minimized over a finite set of scenarios. To solve the problem, the authors begin by computing an initial upper bound from a heuristic based on the nearest neighbor and constructive search of [57]. Using this initial upper bound, the authors implement five exact algorithms, including separation algorithms for subtour inequalities, Benders feasibility, and optimality cuts. This methodology is implemented for a benchmark set of 22 instances. The optimal solutions typically involve the use of just one vehicle.

The SBRP, as introduced by [6], extends the problem initially proposed by [58], known as the one-commodity pickup and delivery traveling salesman problem (1-PDTSP). Alternatively, one can view the SBRP as closely related to the single-commodity vehicle routing problem with split pickups and deliveries (SPDVRP) outlined by [59]. The main difference between the latter two is the fleet size and number of visits allowed to the stations. For the 1-PDTSP, exactly one vehicle needs to be used to visit each station exactly once, while in the SPDVRP, multiple station visits are allowed by a fleet of vehicles. Considering different features besides fleet size and number of station visits, many other closely related problems to the SBRP arise. We refer the interested reader to Table 1 in [59] for further details on these closely related problems.

In this dissertation, we extend the work of [6] in Chapter 5 by developing enhanced exact approaches. Notably, we compare the authors' most efficient exact approach, the so-called Multicut algorithm, with our methods and demonstrate the superior

performance of the latter. We show that our proposed approaches achieve more optimal solutions, exhibit better average gaps, and accomplish these results in a fraction of the computational time required by the Multicut algorithm.

Additional pertinent literature to the SBRP focuses on the OR policy for the VRPSD, previously detailed in Section 2.1. Surprisingly, this policy remains unimplemented in BSS literature despite its simplicity and potential to reduce vehicle fleet travel distance while minimizing unsatisfied station requirements. We breach this literature gap in this dissertation by proposing the OR policy in Chapter 6.

CHAPTER 3 THESIS ORGANIZATION

This section describes the structure of the rest of this dissertation, which focuses on the three contributions outlined in Chapters 4 to 6.

Chapter 4 details the first contribution of this dissertation. It begins with the abstract of the submitted article. Section 4.1 introduces the relevant topics, including stochastic optimization, two-stage mathematical programs with recourse, the integer L-shaped method, the VRPSD, lower bounding functions, and the monotonicity property for the recourse function. Section 4.2 presents the mathematical formulations for the VRPSD under the DTD policy. Section 4.3 formally introduces the monotonicity property and proves the monotonicity of the VRPSD recourse under the DTD policy for demands following Poisson, normal, binomial, Erlang, and negative binomial probability distribution functions. Section 4.4 proposes the improved disaggregated integer L-shaped method, along with new optimality cuts and lower bounding functionals, and introduces two new lower bounds for the recourse. Section 4.5 details the implementation steps for the new method applied to the VRPSD. Section 4.6 presents extensive numerical results obtained using the disaggregated integer L-shaped method and the new recourse lower bounds on benchmark sets of instances. This section also introduces a set of new challenging instances based on the method's success. Finally, Section 4.7 provides the conclusions of this contribution.

Chapter 5 outlines the second contribution of this dissertation. It starts with the abstract of the submitted article. Section 5.1 introduces the relevant topics, including station-based BSS, the SBRP formulation of [6], the exact methods used for similar SBRP variants, and the application of the disaggregated integer L-shaped method from the first contribution to the SBRP. Section 5.2 introduces the mathematical formulations for the SBRP as proposed by [6]. It also presents a new dynamic programming algorithm to solve the recourse problem and discusses the NP-hardness proofs for variants of the SBRP. Section 5.3 proposes new lower bounds for the recourse and the required vehicle fleet size, inspired by the NP-hard recourse problem

variants from Section 5.2. Section 5.5 introduces the valid inequalities added to the SBRP, including new ones based on the new lower bounds. Section 5.6 details the implementation steps for the disaggregated integer L-shaped method applied to the SBRP. Section 5.7 presents the results of our methods compared to those from the literature, demonstrating their success for several problem families and leading to the proposal of a new set of challenging instances. Finally, Section 5.8 summarizes the conclusions of this contribution.

Chapter 6 details the third contribution of this dissertation. It starts with the abstract of the submitted contribution. Section 6.1 introduces the relevant topics, including BSS, SLR, and the problems considered, which are the TLB and the BRP. Section 6.2 defines the SLR problem, while Sections 6.3 and 6.4 define the TLB and BRP problems, respectively, along with their formulations and solution strategies. Specifically, Section 6.3 introduces four different models for the TLB and a generalized problem known as the optimal station capacity problem, utilizing the Benders decomposition algorithm as the solution method. Section 6.4 presents two cost computation policies and a solution method based on the ALNS metaheuristic. Section 6.5 outlines the data-gathering process from real-life sources and artificial data generation. Section 6.6 provides the numerical results of our implementations. Finally, Section 6.7 presents the conclusions.

Chapter 7 summarizes the overall conclusions of the three contributions and discusses the results obtained in each contribution. It also offers general remarks on the limitations of the methodologies used.

CHAPTER 4 ARTICLE 1: A disaggregated integer L-shaped method for stochastic vehicle routing problems with monotonic recourse

Authors: Lucas Parada, Robin Legault, Jean-François Côté, and Michel Gendreau.
Submitted on February 8, 2024 and published on May 9 in European Journal of Operations Research, Volume 318 Issue 2, Pages 520-533, DOI:10.1016/j.ejor.2024.05.012

Abstract: This paper proposes a new integer L-shaped method for solving two-stage stochastic integer programs whose first-stage solutions can decompose into disjoint components, each one having a monotonic recourse function. In a minimization problem, the monotonicity property stipulates that the recourse cost of a component must always be higher or equal to that of any of its subcomponents. The method exploits new types of optimality cuts and lower bounding functionals that are valid under this property. The stochastic vehicle routing problem is particularly well suited to be solved by this approach, as its solutions can be decomposed into a set of routes. We consider the variant with stochastic demands in which the recourse policy consists of performing a return trip to the depot whenever a vehicle does not have sufficient capacity to accommodate a newly realized customer demand. This work shows that this policy can lead to a non-monotonic recourse function, but that the monotonicity holds when the customer demands are modeled by several commonly used families of probability distributions. We also present new problem-specific lower bounds on the recourse that strengthen the lower bounding functionals and significantly speed up the solving process. Computational experiments on instances from the literature show that the new approach achieves state-of-the-art results.

Keywords: Stochastic programming, integer L-shaped method, monotonic recourse, vehicle routing problem.

4.1 Introduction

In two-stage stochastic programs, first-stage decisions are taken before the realization of uncertain parameters. Once these parameters are revealed, second-stage decisions are taken to correct the first-stage solution. The objective is to find a first-stage solution that minimizes the cost of the first stage and the expected cost of the second stage. These problems have typically been solved using Benders decomposition [56], or equivalently the L-shaped method [60] when they have continuous variables and a large number of scenarios. For solving programs with binary variables in both stages, researchers historically relied on the integer L-shaped method of [7]. These methods can be regarded as branch-and-cut (BC) algorithms where the expectation function of the objective is relaxed and replaced by a variable that is linearly bounded with inequalities. Nowadays, the L-shaped and integer L-shaped methods are considered to be computationally impractical because of their slow convergence. Several techniques have been proposed in the literature to mitigate this problem. A common approach is to add lower bounding functionals (LBFs) [61] to the model. These inequalities bound the recourse for a much broader range of solutions than optimality cuts. However, many of the most impactful ones are problem-specific, which limits their range of application. Other methods like the logic-based Benders decomposition of [62] try to overcome the problem by providing cuts that include only a small subset of variables. These cuts are then effective for a large number of solutions. Other enhancements are reported in [63].

This paper introduces a new integer L-shaped method, named the disaggregated integer L-shaped method (DL-shaped method). This method is tailored for a specific class of two-stage stochastic programs in which, given a first-stage solution, the recourse cost can be expressed as a sum of independent recourse functions involving disjoint sets of first-stage variables. The DL-shaped method also requires each of the resulting recourse functions to be monotonic. In the case of a minimization problem, this property means that adding first-stage variables to a component cannot decrease its recourse cost. Many two-stage stochastic programs have such structure and property. For example, in the bin packing problem with uncertain weights, it

is possible to decompose the first-stage solution by bin. This problem is found in the allocation of surgeries to operating rooms [64]. Similarly, in job scheduling problems [65, 66], jobs are assigned to machines, and their first-stage solution can be decomposed by machine. In the latter application, the monotonicity of the objective function corresponds to the fact that assigning more jobs to a machine cannot decrease the expected makespan of its operations.

We propose an implementation of the DL-shaped method for the vehicle routing problem with stochastic demands (VRPSD). In this problem, each customer has to be visited exactly once by a fleet of capacitated vehicles. This structure makes it possible to decompose the recourse cost by route. Customer demands are modeled by independent random variables (RVs) with known distributions and are only observed when a vehicle arrives at the customer’s location. A failure occurs when a vehicle arrives at a customer’s location with a remaining capacity smaller than the customer’s demand. When this happens, a recourse action must be implemented to satisfy the current and next customer demands. This work applies the detour-to-depot (DTD) recourse policy, which performs a back-and-forth trip from the customer to the depot to restock the vehicle. We show sufficient conditions on the distribution of the demands for this policy to lead to a monotonic recourse function.

The contributions of this paper are as follows. First, we introduce the DL-shaped method in the context of the VRPSD. Second, we demonstrate that, although the DTD recourse function is not monotonic in general, it is for Poisson distributions and some normal, binomial, Erlang, and negative binomial distributions when the sum of the expected demands on each route respects the vehicle capacity. Third, we present several new lower bounds on the DTD recourse function that take advantage of its monotonicity and are used by the LBFs. To our knowledge, we perform the first numerical comparison of the different lower bounds on the recourse. Our numerical experiments show our new lower bounds greatly improve over the ones found in the literature and that our new method achieves state-of-the-art results on existing benchmark instances. This leads us to propose new instances that are more challenging than the existing ones from the literature.

The remainder of this chapter is organized as follows. Section 4.2 presents the mathematical formulation of the problem. Section 4.3 presents theoretical results on the monotonicity of the recourse function. Section 4.4 details the implementation of the new method as well as the new lower bounds for the problem. Section 4.6 presents the numerical results for our implementation, and lastly, Section 4.7 presents the conclusion.

4.2 Mathematical formulation

This section introduces the notation that will be used throughout the paper and the mathematical model of the problem.

The VRPSD is defined as follows. Let $G = \{N_0, E\}$ be an undirected graph where $N_0 = \{0, 1, \dots, n\}$ is the set of nodes, with node 0 being the depot, $N = \{1, \dots, n\}$ the set of customer nodes, and $E = \{(i, j) : i, j \in N, i < j\}$ the set of edges. Traveling on edge (i, j) incurs a travel cost of c_{ij} . Each customer $i \in N$ has a non-negative demand given by the RV ξ_i , with an expected value μ_i and a standard deviation σ_i . It is assumed that the demand variables are independently distributed. A fleet of identical vehicles is available to satisfy customer requests. Each vehicle has a capacity Q and must follow a route that starts and ends at the depot. Each customer must be visited exactly once. A route is defined as a sequence of customers starting and ending at the depot, and a sequence is feasible if the sum of the expected demands respects the vehicle capacity. This condition was added by [14] to avoid routes that systematically fail while having others that are underutilized. Let $M \subseteq \mathbb{N}$ be the set specifying the number of vehicles that can be used in a solution. For unlimited fleet problems, this set is given by $M = \{[(\sum_{i \in S} \mu_i)/Q], \dots, |N|\}$, and when exactly $\bar{m}$ vehicles must be used, $M = \{\bar{m}\}$.

Let $E(S)$ be the edges with both endpoints in S and $E(h)$ the set of edges linked with node h . The VRPSD can be formulated as follows. The variable x_{ij} indicates the number of times edge $(i, j) \in E$ is traversed. The variables associated with pairs of customers are binary, and those linking the depot to a customer can take the values

0, 1, or 2. The variable x_{0i} equals 2 if a vehicle serves a single customer $i \in E$. A binary variable z_m decides whether m vehicles are used in the solution. Let $\mathcal{Q}(x)$ denote the expected recourse cost of the first-stage solution $x = (x_{ij})$. The model is:

$$\min \sum_{(i,j) \in E} c_{ij}x_{ij} + \mathcal{Q}(x) \quad (4.1)$$

$$\text{s.t. } \sum_{i \in N} x_{0i} = \sum_{m \in M} 2mz_m, \quad (4.2)$$

$$\sum_{(i,j) \in E(h)} x_{ij} = 2, \quad h \in N, \quad (4.3)$$

$$\sum_{(i,j) \in E(S)} x_{ij} \leq |S| - \left\lceil \frac{\sum_{i \in S} \mu_i}{Q} \right\rceil, \quad S \subseteq N, \quad (4.4)$$

$$\sum_{m \in M} z_m = 1, \quad (4.5)$$

$$x_{ij} \in \{0, 1\}, \quad (i, j) \in E(N), \quad (4.6)$$

$$x_{ij} \in \{0, 1, 2\}, \quad (i, j) \in E(0), \quad (4.7)$$

$$z_m \in \{0, 1\}, \quad m \in M. \quad (4.8)$$

The objective (4.1) is to minimize the sum of travel costs plus the expected recourse cost. Constraint (4.2) imposes that m routes must be connected to the depot if m vehicles are used. Constraints (4.3) ensure that customers are visited exactly once. Constraints (4.4) impose that the expected demand on each route does not exceed the vehicle capacity and that the routes are connected to the depot. Constraint (4.5) ensures that the number of vehicles that are used in the solution is an element of M . Constraints (4.6)-(4.8) define the domain of the decision variables.

For the DTD recourse policy, it is possible to separate the calculation of the recourse by route. Let $\mathcal{R}^\nu$ be the set of routes in solution x^ν . Then, the expected recourse cost is defined as:

$$\mathcal{Q}(x^\nu) = \sum_{r \in \mathcal{R}^\nu} \mathcal{Q}(r), \quad (4.9)$$

where $\mathcal{Q}(r) = \min\{\mathcal{Q}^1(r), \mathcal{Q}^2(r)\}$ is the minimum expected recourse cost of both orientations of route $r = (0, i_1, \dots, i_t, 0)$. The computation of both orientations is required because model (4.1)-(4.8) does not capture routes orientation. The expected recourse cost is denoted by $\mathcal{Q}^1(r)$ and $\mathcal{Q}^2(r)$ in the two orientations. Under the DTD policy, a cost of $2c_{0i}$ is incurred when the demand of customer i cannot be satisfied. The expected recourse of route r , in the first orientation, is thus:

$$\mathcal{Q}^1(r) = 2 \sum_{j=1}^t \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j} \quad (4.10)$$

Note that a restocking trip occurs only when a customer's demand exceeds the vehicle's residual capacity. We exclude the case of going to the depot and then moving to the next customer when an exact stock-out occurs. If it is assumed that $\xi_i \leq Q$ for all i with probability 1, then the second summation of (4.11) can be limited to $j-1$ instead of going to ∞ . If demands can exceed the capacity of the vehicles, the probability $\mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ \leq \sum_{k=1}^j \xi_{i_k} \right)$ tends to zero as l increases and the summation can be stopped when it reaches a probability close to zero. In our implementation, the summation is stopped when the probability is lower than 0.0001. In the paper, the expected recourse cost $\mathcal{Q}(r)$ of a route $r = (0, p, 0)$ will equivalently be denoted by $\mathcal{Q}(p)$.

It is standard practice in the literature [11, 14] to express the expected recourse cost under the DTD policy in terms of the cumulative distribution function (CDF) of the demand, which has the advantage of being more direct to compute for usual probability distributions.

$$\mathcal{Q}^1(r) = 2 \sum_{j=1}^t \sum_{l=1}^{\infty} \left(\mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ \right) - \mathbb{P} \left(\sum_{k=1}^j \xi_{i_k} \leq lQ \right) \right) c_{0i_j} \quad (4.11)$$

The more intuitive and succinct expression $\mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right)$, which is equivalent to the difference $\mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ \right) - \mathbb{P} \left(\sum_{k=1}^j \xi_{i_k} \leq lQ \right)$ in the case of RVs with nonnegative support, is used throughout the paper for notational simplicity.

ity. However, as customary in the literature, we compute the recourse based on (4.11), including for instances using normally distributed demands. All the properties of the paper directly extend to formulation (4.11) by substituting the restocking probabilities in (4.10) by the corresponding difference of CDFs everywhere in the proofs.

4.3 Monotonicity of the recourse function

Our algorithm relies on new optimality cuts and lower bounds that require the recourse function to decrease monotonically when removing customers from a route. This section formalizes this property by the so-called monotonicity condition and monotonicity property, which respectively apply to sets of customers and instances of the VRPSD. Based on these definitions, we prove sufficient conditions for the recourse function to be monotonic under the DTD policy when the demands follow independent Poisson, normal, binomial, Erlang, and negative binomial distributions. We also provide counterexamples to illustrate that the recourse function is not monotonic in general.

Definition 1. Let $p = (i_1, \dots, i_t)$ be a path on graph G . We say that path $p' = (j_1, \dots, j_{t'})$ is a subsequence of p if it can be obtained by removing elements of p , without changing the ordering of the remaining elements. Formally, p' is thus a subsequence of p if $\{j_1, \dots, j_{t'}\} \subseteq \{i_1, \dots, i_t\}$ and, for any pair $(j_{a'}, j_{b'}) = (i_a, i_b)$, $a' < b'$ if and only if $a < b$.

Definition 2. Let $p = (i_1, \dots, i_t)$ be a path on graph G . A path p' that can be written as $p' = (i_a, \dots, i_b)$ for some $1 \leq a \leq b \leq t$ is a subpath of p . In other words, p' is a subsequence of p that can be obtained by removing elements only at the beginning and the end of p .

Definition 3. A set of customers $S \subseteq N$ is said to respect the monotonicity condition if, for any pair $(a, b) \in S \times S$ such that $a \neq b$ and for any subset $\tilde{S} \subseteq S \setminus \{a, b\}$, the

following inequality holds for any positive integer $l \in \mathbb{N}$:

$$\mathbb{P}(\tilde{\xi} + \xi_a \leq lQ < \tilde{\xi} + \xi_a + \xi_b) \geq \mathbb{P}(\tilde{\xi} \leq lQ < \tilde{\xi} + \xi_b), \quad (4.12)$$

where $\tilde{\xi} = \sum_{i \in \tilde{S}} \xi_i$.

Definition 4. An instance of the VRPSD with the DTD policy is said to have the monotonicity property if any set $S \subseteq N$ such that $\sum_{i \in S} \mu_i \leq Q$ respects the monotonicity condition.

The above definition can be applied by enumeration to verify whether an instance respects the monotonicity property, but it might be unpractical for large instances since the number of comparisons grows exponentially with $|N|$. In Proposition 1, we introduce a sufficient condition which will later be used to prove that the monotonicity property is respected if the demands are modeled by specific distributions.

Proposition 1. Let $S \subseteq N$ be a set of customers such that, for any pair $(a, b) \in S \times S$, $a \neq b$, and for any subset $\tilde{S} \subseteq S \setminus \{a, b\}$, the RVs $\tilde{\xi}$, ξ_a and ξ_b can respectively be rewritten as the sum of $\tilde{\rho} \in \mathbb{N}$, $\rho_a \in \mathbb{N}$ and $\rho_b \in \mathbb{N}$ i.i.d. RVs ζ_k , $k \in \mathbb{N}$. The following condition is sufficient for the set S to respect the monotonicity condition:

$$P_{lQ}(\tilde{\rho} + \rho_a + j) \geq P_{lQ}(\tilde{\rho} + j) \quad \forall l \in \mathbb{N}, \forall j \in \{1, \dots, \rho_b\}, \quad (4.13)$$

where $P_u(s) = \mathbb{P}(\sum_{k=1}^{s-1} \zeta_k \leq u < \sum_{k=1}^s \zeta_k)$ denotes the probability that exactly $s \in \mathbb{N}$ RVs ζ_k must be drawn for their cumulative sum to first exceed a given threshold $u \geq 0$.

Proof. Given the assumptions of the proposition, each original customer $i \in S$ can be seen as a group of ρ_i smaller customers with i.i.d. demands ζ_k , $k \in \mathbb{N}$ and sharing the same location.

From there, the left-hand side of inequality (4.12) can be rewritten as follows.

$$\begin{aligned}
\mathbb{P} \left(\tilde{\xi} + \xi_a \leq lQ < \tilde{\xi} + \xi_a + \xi_b \right) &= \mathbb{P} \left(\sum_{k=1}^{\tilde{\rho} + \rho_a} \zeta_k \leq lQ < \sum_{k=1}^{\tilde{\rho} + \rho_a + \rho_b} \zeta_k \right) \\
&= \sum_{j=1}^{\rho_b} \mathbb{P} \left(\sum_{k=1}^{\tilde{\rho} + \rho_a + j - 1} \zeta_k \leq lQ < \sum_{k=1}^{\tilde{\rho} + \rho_a + j} \zeta_k \right) \\
&= \sum_{j=1}^{\rho_b} P_{lQ}(\tilde{\rho} + \rho_a + j)
\end{aligned}$$

The second equality is obtained by decomposing the probability that the l -th restocking trip occurs between customers $\tilde{\rho} + \rho_a + 1$ and $\tilde{\rho} + \rho_a + \rho_b$ as the sum of the probabilities that it occurs at customer $\tilde{\rho} + \rho_a + j$, for $j=1, \dots, \rho_b$.

Doing the same for the right-hand side, we obtain the sufficient condition for inequality (4.12).

$$\begin{aligned}
&\mathbb{P} \left(\tilde{\xi} + \xi_a \leq lQ < \tilde{\xi} + \xi_a + \xi_b \right) \geq \mathbb{P} \left(\tilde{\xi} \leq lQ < \tilde{\xi} + \xi_b \right) \\
&\iff \sum_{j=1}^{\rho_b} P_{lQ}(\tilde{\rho} + \rho_a + j) \geq \sum_{j=1}^{\rho_b} P_{lQ}(\tilde{\rho} + j) \\
&\iff P_{lQ}(\tilde{\rho} + \rho_a + j) \geq P_{lQ}(\tilde{\rho} + j) \quad \forall j \in \{1, \dots, \rho_b\}
\end{aligned}$$

□

We now prove that, if the customers of a path respect the monotonicity condition, removing customers from this path cannot increase its recourse cost. This is shown in Proposition 2.

Proposition 2. *For a path $p = (i_1, \dots, i_t)$ composed of a set of customers respecting the monotonicity condition, the expected recourse cost decreases monotonically when customers are removed from p , i.e., $\mathcal{Q}(p) \geq \mathcal{Q}(p')$ for any subsequence p' of p .*

Proof.

$$\begin{aligned}
\mathcal{Q}(p) &= 2 \sum_{j=1}^t \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j} \\
&= 2 \sum_{\substack{j=1 \\ i_j \in p'}}^t \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j} + 2 \sum_{\substack{j=1 \\ i_j \in p \setminus p'}}^t \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j} \\
&\geq 2 \sum_{\substack{j=1 \\ i_j \in p'}}^t \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j} \\
&\geq 2 \sum_{\substack{j=1 \\ i_j \in p'}}^t \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{\substack{k=1 \\ i_k \in p'}}^{j-1} \xi_{i_k} \leq lQ < \sum_{\substack{k=1 \\ i_k \in p'}}^j \xi_{i_k} \right) c_{0i_j} \\
&= \mathcal{Q}(p')
\end{aligned}$$

The second inequality comes from the fact that, for any $l \geq 1$ and for any $j \in \{1, \dots, t\}$ such that $i_j \in p'$, the following inequality holds.

$$\mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) \geq \mathbb{P} \left(\sum_{\substack{k=1 \\ i_k \in p'}}^{j-1} \xi_{i_k} \leq lQ < \sum_{\substack{k=1 \\ i_k \in p'}}^j \xi_{i_k} \right) \quad (4.14)$$

To prove this, we first introduce some notation:

- $\tilde{S}^j = \{i_k \in p' : k \leq j-1\}$ is the set of customers that are visited before customer i_j in path p'
- $\tilde{\xi}^j = \sum_{i \in \tilde{S}^j} \xi_i$ is the total demand of these customers
- $\{a_1, \dots, a_s\} = \{i_k \in p : k \leq j-1, i_k \notin p'\}$ is the set of customers that are visited before customer i_j in path p , but are not visited in path p'

We can now prove inequality (4.14).

$$\mathbb{P} \left(\sum_{\substack{k=1 \\ i_k \in p'}}^{j-1} \xi_{i_k} \leq lQ < \sum_{\substack{k=1 \\ i_k \in p'}}^j \xi_{i_k} \right) = \mathbb{P} \left(\tilde{\xi}^j \leq lQ < \tilde{\xi}^j + \xi_{i_j} \right) \quad (4.15)$$

$$\leq \mathbb{P} \left(\tilde{\xi}^j + \xi_{a_1} \leq lQ < \tilde{\xi}^j + \xi_{a_1} + \xi_{i_j} \right) \quad (4.16)$$

$$\vdots \quad (4.17)$$

$$\leq \mathbb{P} \left(\tilde{\xi}^j + \xi_{a_1} + \cdots + \xi_{a_s} \leq lQ < \tilde{\xi}^j + \xi_{a_1} + \cdots + \xi_{a_s} + \xi_{i_j} \right) \quad (4.18)$$

$$= \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) \quad (4.19)$$

By hypothesis, the set $S = \{i_1, \dots, i_j\}$ respects the monotonicity condition. This allows us to apply inequality (4.12) s times for different pairs $(a, b) \in S \times S$ and subsets $\tilde{S} \subseteq S$ to obtain inequalities (4.16)-(4.18). At the q -th application of the inequality, $(a, b) = (a_q, i_j)$ and $\tilde{S} = \tilde{S}^j \cup \{a_1, \dots, a_{q-1}\}$. $\square$

Propositions 3, 4, 5, 6 and 7 show that, under some conditions, the monotonicity property is respected for customer demands following independent Poisson, normal, binomial, Erlang, and negative binomial distributions. We present next the proofs for the cases of Poisson and normal independently distributed customer demands. The remaining three proof are given in A.1.

Proposition 3. *Any set S of customers whose demands are given by independent Poisson RVs $\xi_i \sim \text{Poisson}(\lambda_i)$ with total expected value $\sum_{i \in S} \lambda_i \leq Q$ respects the monotonicity condition.*

Proof. Consider a pair $(a, b) \in S \times S$ such that $a \neq b$, a subset $\tilde{S} \subseteq S \setminus \{a, b\}$ with total expected demand $\tilde{\lambda} = \sum_{i \in \tilde{S}} \lambda_i$ and a positive integer $l \in \mathbb{N}$. To show that inequality (4.12) holds, we use the fact that the sum of s independent Poisson RVs with means $\lambda_1, \dots, \lambda_s$ is Poisson distributed with mean $\sum_{k=1}^s \lambda_k$. Also, we use

the fact that the CDF of a Poisson RV with mean λ evaluated at $lQ \in \mathbb{N}$ is given by $H(lQ, \lambda) = \frac{\Gamma(lQ+1, \lambda)}{(lQ)!}$, where $\Gamma(lQ+1, \lambda) = \int_{\lambda}^{\infty} t^{lQ} e^{-t} dt$ is the upper incomplete gamma function.

$$\mathbb{P}(\tilde{\xi} + \xi_a \leq lQ < \tilde{\xi} + \xi_a + \xi_b) \geq \mathbb{P}(\tilde{\xi} \leq lQ < \tilde{\xi} + \xi_b) \quad (4.20)$$

$$\iff \mathbb{P}(\tilde{\xi} + \xi_a \leq lQ) - \mathbb{P}(\tilde{\xi} + \xi_a + \xi_b \leq lQ) \geq \mathbb{P}(\tilde{\xi} \leq lQ) - \mathbb{P}(\tilde{\xi} + \xi_b \leq lQ) \quad (4.21)$$

$$\iff H(lQ, \tilde{\lambda} + \lambda_a) - H(lQ, \tilde{\lambda} + \lambda_a + \lambda_b) \geq H(lQ, \tilde{\lambda}) - H(lQ, \tilde{\lambda} + \lambda_b) \quad (4.22)$$

$$\iff \frac{\Gamma(lQ+1, \tilde{\lambda} + \lambda_a)}{(lQ)!} - \frac{\Gamma(lQ+1, \tilde{\lambda} + \lambda_a + \lambda_b)}{(lQ)!} \geq \frac{\Gamma(lQ+1, \tilde{\lambda})}{(lQ)!} - \frac{\Gamma(lQ+1, \tilde{\lambda} + \lambda_b)}{(lQ)!} \quad (4.23)$$

$$\iff \Gamma(lQ+1, \tilde{\lambda} + \lambda_a) - \Gamma(lQ+1, \tilde{\lambda} + \lambda_a + \lambda_b) \geq \Gamma(lQ+1, \tilde{\lambda}) - \Gamma(lQ+1, \tilde{\lambda} + \lambda_b) \quad (4.24)$$

$$\iff \int_{\tilde{\lambda} + \lambda_a}^{\infty} \lambda^{lQ} e^{-\lambda} d\lambda - \int_{\tilde{\lambda} + \lambda_a + \lambda_b}^{\infty} \lambda^{lQ} e^{-\lambda} d\lambda \geq \int_{\tilde{\lambda}}^{\infty} \lambda^{lQ} e^{-\lambda} d\lambda - \int_{\tilde{\lambda} + \lambda_b}^{\infty} \lambda^{lQ} e^{-\lambda} d\lambda \quad (4.25)$$

$$\iff \int_{\tilde{\lambda} + \lambda_a}^{\tilde{\lambda} + \lambda_a + \lambda_b} \lambda^{lQ} e^{-\lambda} d\lambda \geq \int_{\tilde{\lambda}}^{\tilde{\lambda} + \lambda_b} \lambda^{lQ} e^{-\lambda} d\lambda \quad (4.26)$$

$$\iff \int_{\tilde{\lambda}}^{\tilde{\lambda} + \lambda_b} (\lambda + \lambda_a)^{lQ} e^{-(\lambda + \lambda_a)} d\lambda \geq \int_{\tilde{\lambda}}^{\tilde{\lambda} + \lambda_b} \lambda^{lQ} e^{-\lambda} d\lambda \quad (4.27)$$

$$\iff \int_{\tilde{\lambda}}^{\tilde{\lambda} + \lambda_b} ((\lambda + \lambda_a)^{lQ} e^{-(\lambda + \lambda_a)} - \lambda^{lQ} e^{-\lambda}) d\lambda \geq 0 \quad (4.28)$$

$$\iff \int_{\tilde{\lambda}}^{\tilde{\lambda} + \lambda_b} (f_{lQ}(\lambda + \lambda_a) - f_{lQ}(\lambda)) d\lambda \geq 0, \quad (4.29)$$

where $f_{lQ}(\lambda) := \lambda^{lQ} e^{-\lambda}$. Since the inequalities $0 \leq \lambda \leq \lambda + \lambda_a \leq Q \leq lQ$ hold for any $\lambda \in [\tilde{\lambda}, \tilde{\lambda} + \lambda_b]$, it is sufficient to show that the function $f_{lQ}(\cdot)$ is non-decreasing on $[0, lQ]$ to conclude that the difference $f_{lQ}(\lambda + \lambda_a) - f_{lQ}(\lambda)$ is non-negative everywhere on the interval of integration. The derivative of function $f_{lQ}(\cdot)$ is given by:

$$\frac{\partial f_{lQ}(\lambda)}{\partial \lambda} = (lQ - \lambda) e^{-\lambda} \lambda^{lQ-1},$$

which is indeed non-negative $\forall \lambda \in [0, lQ]$. Inequality (4.29) is therefore respected.

□

Proposition 4. *Any set S of customers whose demands are given by independent normal RVs $\xi_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ with integer expected values $\mu_i \in \mathbb{N}$ and sharing a common coefficient of dispersion $D = \frac{\sigma_i^2}{\mu_i} \leq 1 \ \forall i \in S$ with total expected value $\sum_{i \in S} \mu_i \leq Q$ respects the monotonicity condition.*

Proof. Consider a pair $(a, b) \in S \times S$ such that $a \neq b$ and a subset $\tilde{S} \subseteq S \setminus \{a, b\}$ with total expected demand $\tilde{\mu} = \sum_{i \in \tilde{S}} \mu_i$. Using the notation of Proposition 1, $\tilde{\xi}$, ξ_a and ξ_b can respectively be decomposed as the sum of $\tilde{\rho} = \tilde{\mu}$, $\rho_a = \mu_a$ and $\rho_b = \mu_b$ i.i.d. RVs ξ_k , $k \in \mathbb{N}$, such that $\xi_k \sim \mathcal{N}(\mu=1, \sigma^2=D)$. From there, the sufficient condition (4.13) can be rewritten as follows.

$$P_{lQ}(\tilde{\mu} + \mu_a + j) \geq P_{lQ}(\tilde{\mu} + j) \quad \forall l \in \mathbb{N}, \forall j \in \{1, \dots, \mu_b\} \quad (4.30)$$

The coefficient of variation $c = \sigma/\mu$ of the i.i.d. RVs ξ_k is equal to $\sqrt{D}$. Since $D \leq 1$ by hypothesis, then $c \leq 1$. In this case, it follows from [26] that, for any $l \in \mathbb{N}$, the sequence $\{P_{lQ}(s)\}_{s=1}^{\lfloor lQ \rfloor}$ is monotonically increasing. Since $\tilde{\mu} + \mu_a + \mu_b \leq \sum_{i \in S} \mu_i \leq Q$ and all the expected demands are integer-valued, this allows to conclude that (4.30) holds. □

Proposition 5. *Any set S of customers whose demands are given by independent binomial RVs $\xi_i \sim \text{Bin}(n_i, p)$ sharing a common success probability p with total expected value $\sum_{i \in S} n_i p \leq Q$ respects the monotonicity condition.*

Proof. See A.1. □

Proposition 6. *Any set S of customers whose demands are given by independent Erlang RVs $\xi_i \sim \text{Erlang}(n_i, \lambda)$ sharing a common rate parameter λ with total expected value $\sum_{i \in S} n_i/\lambda \leq Q$ respects the monotonicity condition.*

Proof. See A.1. □

Proposition 7. *Any set S of customers whose demands are given by independent negative binomial RVs $\xi_i \sim NB(r_i, p)$ sharing a common success probability p with total expected value $\sum_{i \in S} r_i(1-p)/p \leq Q$ respects the monotonicity condition.*

Proof. See A.1. □

The next propositions present two counterexamples to illustrate that the recourse function is not monotonic in general.

Proposition 8. *Let $p = (i_1, \dots, i_t)$ be a path of customers whose demands are given by independent Poisson RVs $\xi_i \sim \text{Poisson}(\lambda_i)$ with total expected value $\sum_{i=1}^t \lambda_i > Q$, and p' a subpath of p . Inequality $\mathcal{Q}(p) \geq \mathcal{Q}(p')$ does not hold in general.*

Proof. Consider a problem with $Q = 20$, a path $p = (1, 2, 3)$, and $p' = (2, 3)$. Suppose the expected demand of each customer is given by $\lambda_1 = 5$, $\lambda_2 = 15$, and $\lambda_3 = 10$, respectively. If $c_{01} = c_{02} = 0$ and $c_{03} = 1$, then $\mathcal{Q}(p) \approx 1.11 < 1.47 \approx \mathcal{Q}(p')$. □

Proposition 9. *Let $p = (i_1, \dots, i_t)$ be a path of customers whose total expected demand respects the vehicle capacity and p' a subpath of p . Inequality $\mathcal{Q}(p) \geq \mathcal{Q}(p')$ does not hold in general.*

Proof. Consider a problem with $Q = 20$, a path $p = (1, 2, 3)$, and $p' = (2, 3)$. Suppose the demand of customer 1 is 5 with probability 1, while that of customers 2 and 3 is 6 with probability 0.9 and 16 with probability 0.1. If $c_{03} > c_{02}$, then $\mathcal{Q}(p) = (0.1)2c_{02} + (0.09)2c_{03} < (0.19)2c_{03} = \mathcal{Q}(p')$. □

4.4 Disaggregated integer L-shaped method

This section presents the DL-shaped method as a variant of the classical integer L-shaped method from [7]. First, an outline of the classical method is provided, and we motivate the need for our method by exposing the known shortcoming of the classical one. Next, Section 4.4.1 presents new optimality cuts, from which the

validity of our method results. Finally, Section 4.4.2 introduces the new LBFs we use to approximate the recourse cost.

The integer L-shaped method for two-stage recourse problems starts by constructing a so-called master problem (MP). The MP is obtained from the original problem by relaxing a set of complicating constraints. This set typically includes the integrality constraints and any set of constraints that is exponential in size, like the classical subtour elimination constraints. In addition, the recourse function is replaced by a continuous variable θ that bounds it from below, and is initially lower bounded by a finite value L . In the case of the VRPSD, the recourse cost is nonnegative and the trivial bound $L = 0$ can thus be used. The initial MP is defined by the non-relaxed constraints of the original program, the lower bound L on θ , and an objective function of the form $\min cx + \theta$, where c is a known cost vector with the same size as the solution vector. The problem is then solved through a branch-and-bound method that explores the set of feasible solutions of the original program and adds linear inequalities on the fly. The first MP that is solved in the method is denoted as the root node, and the following MPs are indexed by the iteration at which they appear in the solving process. The constraints that are added to the successive MPs are either feasibility or optimality cuts. The feasibility cuts are valid inequalities that are violated by the current master solution and are generated to prohibit infeasible solutions from being visited. The optimality cuts are supporting linear functions of $\mathcal{Q}(x)$ that are added whenever a feasible solution is found to improve the bound on θ . The feasibility cuts are generally problem-specific. In the context of the VRPSD, they typically correspond to subtour elimination constraints and rounded capacity inequalities. Optimality cuts have a general form regardless of the problem to which the integer L-shaped method is applied. Let x_1^ν be the set of positive variables in a feasible solution x^ν at iteration ν of the method. Then the following inequality:

$$\theta \geq (\mathcal{Q}(x^\nu) - L) \left(\sum_{i \in x_1^\nu} x_i - \sum_{i \notin x_1^\nu} x_i - |x_1^\nu| + 1 \right) + L, \quad (4.31)$$

where $\mathcal{Q}(x^\nu)$ is the expected recourse cost of solution x^ν , is a valid optimality cut.

It was shown by [7] that these optimality cuts will yield an optimal solution, if one exists, in a finite number of steps.

The known shortcoming of the integer L-shaped method is that its optimality cuts are active for a single solution. The optimality cut reduces to $\theta \geq Q(x^\nu)$ at solution x^ν , but for any other solution, it does not improve the inequality $\theta \geq L$. Consequently, without appropriate LBFs to increase the bound on θ , the classical method generally iterates through many first-stage solutions before converging to an optimal solution.

4.4.1 New optimality cuts

The main idea of the DL-shaped method is to replace the variable θ that bounds the recourse function by a sum of variables θ_i , one for each customer $i \in N$. The purpose of this transformation is to express the contribution of each customer to the cost of the second stage by the gradual addition of a new type of optimality cuts during the solving process. These cuts are added for feasible paths that are found, and the cut of each path only involves the variables associated with the arcs of the path and the θ_i variables of the customers of the path. This makes them active for each path that is active in the current solution, whether it is fractional or integer. They allow to bound the recourse function more tightly and effectively than the traditional optimality cuts, as those only bound one solution per cut. This gives the following objective function.

$$\min \sum_{(i,j) \in E} c_{ij}x_{ij} + \sum_{i \in N} \theta_i \quad (4.32)$$

The DL-shaped method starts by constructing a MP with the new objective function, relaxed integrality constraints, and rounded capacity inequalities. In each iteration of the method, when a feasible integer solution is found, an optimality cut is added for each route $r = (0, p, 0)$ having a positive recourse ($Q(r) > 0$), where $p = (i_1, \dots, i_t)$. Such a path p is composed of $|p| = t - 1$ edges. For a first-stage solution x , let us denote by $x(p) = \sum_{j=1}^{t-1} x_{i_j i_{j+1}}$ the number of these edges that are active. Also, let $N(r)$ be the set of customers that are visited in path p . The new optimality cut

follows.

$$\sum_{i \in N(r)} \theta_i \geq \mathcal{Q}(r) (x(p) - |p| + 1) \quad (4.33)$$

We refer to these cuts as path cuts (P-Cuts). Since they do not include the depot edges, as opposed to the route cuts of [23], these P-Cuts are more general. In the specific case of a route $r = (0, i, 0)$ visiting a single customer i , inequality (4.33) reduces to $\theta_i \geq \mathcal{Q}(r)$.

We now prove that the P-Cuts are valid optimality cuts. By [7], this implies that the DL-shaped method yields an optimal solution in a finite number of steps.

Proposition 10. *Equation (4.33) defines a valid set of optimality cuts if the problem has the monotonicity property.*

Proof. Let x^λ be a solution satisfying constraints (4.2)-(4.8) and let $\mathcal{R}^\lambda$ be the set of routes it visits. Let $\theta^*(x^\lambda)$ be the minimizer of $\sum_{i \in N} \theta_i^\lambda$ over the feasible set specified by the first-stage solution x^λ and by the constraints (4.33) that are contained in the MP at a given iteration. The MP is maximally constrained if (4.33) has been added to the model for each possible route r . Under this maximal set of optimality cuts, for each route $r \in \mathcal{R}^\lambda$, the sum $\sum_{i \in N(r)} \theta_i^\lambda$ is constrained to be larger than or equal to $\mathcal{Q}(\tilde{r})$ for each route $\tilde{r} = (0, \tilde{p}, 0)$ such that $\tilde{p}$ is a subpath of r . Any other route produces a trivial bound for r , as $(x^\lambda(\tilde{p}) - |\tilde{p}| + 1) \leq 0$ if $\tilde{p}$ is not a subpath of r . Since it is assumed that the problem respects the monotonicity property, it follows from Proposition 2 that the most restrictive constraint for r is $\sum_{i \in N(r)} \theta_i^\lambda \geq \mathcal{Q}(r)$. Hence, by definition of $\theta^*(x^\lambda)$, $\sum_{i \in N(r)} \theta_i^*(x^\lambda) = \mathcal{Q}(r)$ for each $r \in \mathcal{R}^\lambda$ and thus, $\sum_{i \in N} \theta_i^*(x^\lambda) = \sum_{r \in \mathcal{R}^\lambda} \sum_{i \in N(r)} \theta_i^*(x^\lambda) = \sum_{r \in \mathcal{R}^\lambda} \mathcal{Q}(r) = \mathcal{Q}(x^\lambda)$. The objective values of $(x^\lambda, \theta^*(x^\lambda))$ and x^λ for their respective problems are therefore identical when the optimality cuts associated with solution x^λ have been generated. Otherwise, the objective value of $(x^\lambda, \theta^*(x^\lambda))$ for the MP can only be less than or equal to that of x^λ for problem (4.1)-(4.8). $\square$

As shown in [7], the linear relaxation of (4.1)-(4.8) can be improved by introducing

a nontrivial lower bound on the recourse using a precomputed value L .

$$\sum_{i \in N} \theta_i \geq L \quad (4.34)$$

Unfortunately, such a bound is generally close to 0 for problems with an unlimited fleet, as using more vehicles reduces the probability of having a failure. In particular, the recourse cost is close to 0 when each vehicle only serves a single client. It can therefore be useful to compute a lower bound L_m on the recourse given that exactly $m \in M$ vehicles are being used. L_m is computed using the method defined in Section 4.4.3. This leads to the following valid lower bound inequalities.

$$\sum_{i \in N} \theta_i \geq \sum_{m \in M} L_m z_m \quad (4.35)$$

To conclude this section, we propose a different optimality cut that does not require the recourse to be monotonic. In the absence of monotonicity, the validity of our method can be preserved by adding the depot edges to cut (4.33). This way, the cut associated with a route $r = (0, i_1, \dots, i_t, 0)$ is only active when this route appears in a first-stage solution. However, the depot edge variables can not be simply added to (4.33) because they can take the value of two. This would lead the cut being active in other solutions. To avoid that, the coefficients of the non-depot edge variables are required to be equal to two. The modified optimality cut follows.

$$\sum_{i \in N(r)} \theta_i \geq \mathcal{Q}(r) \left(\sum_{j=1}^{t-1} 2x_{i_j i_{j+1}} + x_{0i_1} + x_{0i_t} - 2(t-1) - 1 \right) \quad (4.36)$$

4.4.2 Lower bounding functionals

This section presents a new type of LBFs that extends some ideas proposed in [24]. Similarly to the previously introduced optimality cuts, this type of cuts uses variables to bound the recourse for sets of customers S . The LBFs of [24] use a different set of variables than those used for optimality cuts, which has the disadvantage of making

the LBFs independent from the optimality cuts. This work improves over them by using the same θ_i variables for both types of cuts. The new LBFs bound the recourse using all the θ_i variables associated with a set of customers S . A cut is active whenever a path visits all the customers of this set consecutively. The new LBFs are referred to as the set cuts (S-Cuts) and are as follows:

$$\sum_{i \in S} \theta_i \geq L(S) \left(x(S) - |S| + \left\lceil \frac{\sum_{i \in S} \mu_i}{Q} \right\rceil + 1 \right), \quad (4.37)$$

where $x(S) = \sum_{(i,j) \in E(S)} x_{ij}$. The S-Cut (4.37) of a set S is based on the least possible recourse cost of visiting the customers of S . Such a lower bound $L(S)$ is valid if, for any set of feasible paths $\{p_1, \dots, p_m\}$ forming a partition of S , the following inequality is respected.

$$L(S) \leq \sum_{j=1}^m Q(p_j) \quad (4.38)$$

Proposition 11. *The S-Cut (4.37) is valid if the problem has the monotonicity property.*

Proof. Let x^λ be a solution satisfying constraints (4.2)-(4.8). If $x^\lambda(S) \leq |S| - \left\lceil \frac{\sum_{i \in S} \mu_i}{Q} \right\rceil - 1$, then the right-hand side of (4.37) is non-positive and the constraint is thus respected by any feasible non-negative solution $(x^\lambda, \theta^\lambda)$ to the MP. Otherwise, $x^\lambda(S) = |S| - \left\lceil \frac{\sum_{i \in S} \mu_i}{Q} \right\rceil$. This means that solution x^λ uses the smallest number of vehicles allowed by the capacity constraints to visit the customers of S and that, in each route $r \in \mathcal{R}^\lambda$, the customers of S are visited consecutively. In this case, the S-Cut reduces to $\sum_{i \in S} \theta_i \geq L(S)$. Let $\{p_1^\lambda, \dots, p_m^\lambda\}$ be the set of paths forming solution x^λ , where $m = \left\lceil \frac{\sum_{i \in S} \mu_i}{Q} \right\rceil$. To prove the validity of constraint (4.37), we have to demonstrate that $L(S) \leq \sum_{j=1}^m Q(p_j^\lambda)$. For each $j \in \{1, \dots, m\}$, the segment of path p_j^λ that is composed of customers of S is a subpath p_j of p_j^λ , and the set $\{p_1, \dots, p_m\}$ of these subpaths forms a partition of S . Since it is assumed that the problem respects the monotonicity property, Proposition 2 implies that $Q(p_j^\lambda) \geq Q(p_j) \forall j \in \{1, \dots, m\}$. By definition of the lower bound $L(S)$, the sequence of inequalities $L(S) \leq \sum_{j=1}^m Q(p_j) \leq \sum_{j=1}^m Q(p_j^\lambda)$ allows to conclude. $\square$

4.4.3 Recourse lower bounds

This section presents lower bounds on the recourse used by the S -Cuts for a given set of customers S to be served by m vehicles. Some lower bounds from the literature are first presented.

To our knowledge, the first lower bound on the recourse of a given set S is due to [14], where they suppose vehicles are only allowed to fail once. In the best case, the failures would happen for the customers of S that are the closest to the depot. Then, the sum of demands is seen as a continuous quantity that has to be distributed among the m vehicles. The problem is modeled as a non-linear program having for objective the minimization of the expected recourse.

A simpler bound, which is based on the assumption that there is one unique large vehicle with capacity mQ , is proposed by [16]. In the best case, the l -th failure would happen at the l -th customer closest to the depot. Then, the expected recourse cost can be easily calculated by summing the probability of the l -th failure multiplied by twice the l -th closest distance to the depot among the customers of S .

In the following, we propose three new bounds, $L_1(S)$, $L_2(S)$ and $L_3(S)$, which have the advantage of providing tighter values than those of [14] and [16]. All these bounds are compared later in Section 4.6.

Single-route lower bound.

The lower bound $L_1(S)$ introduced in this section applies to sets of customers that respect the monotonicity condition and can be assigned to a single vehicle. Let $\mathcal{P}^1(S)$ denote the set of all possible paths that visit the customers of S consecutively. Proposition 12 shows that sorting these customers in non-increasing order of their distance to the depot produces the path of $\mathcal{P}^1(S)$ that minimizes the recourse cost. The proof is given in A.2. By the monotonicity condition, the probability that a restocking trip occurs at a given customer increases with its position in the path. The idea of this bound is thus to place customers that are far from the depot at the beginning of the path to avoid costly restocking trips.

Proposition 12. *Let $S \subseteq N$ be a set of customers respecting the monotonicity condition. Let $p^* = (i_1^*, i_2^*, \dots, i_{|S|}^*) \in \mathcal{P}^1(S)$ be a path such that $c_{0i_1^*} \geq c_{0i_2^*} \geq \dots \geq c_{0i_{|S|}^*}$. The expected recourse $\mathcal{Q}(p^*)$ of path p^* is a valid lower bound on the expected recourse $\mathcal{Q}(p)$ of any path $p \in \mathcal{P}^1(S)$.*

Proof. See A.2. □

The definition of the single-route lower bound follows.

$$L_1(S) = \mathcal{Q}(p^*) \tag{4.39}$$

Dynamic programming algorithm.

The lower bound proposed in this section applies to sets of customers S that can be assigned to $m \geq 2$ vehicles. It is inspired by the lower bound of [14], where the demands are seen as continuous quantities that must be distributed among the vehicles. We assume that Q , μ_i and σ_i are integer. In the case a μ_i or a σ_i is fractional, we round it down, if Q is fractional, we round it up. This new bound, which requires the monotonicity of the recourse function, improves over [14] in two ways. First, the value of the lower bound can be greatly increased by considering that multiple failures can happen for a vehicle. This is done by computing for each vehicle k and for each quantity of expected demand $q \in [0, Q]$ the least-cost sequence of customers by taking advantage of Proposition 12. The second improvement is that our search for an assignment of the demands can be restricted to integer solutions through the use of a new dynamic programming algorithm. The new bound is computed in two steps.

In the first step, the least-cost sequence for assigning $q_h \in [0, Q]$ units of expected demand to a vehicle using the $|S| - h + 1$ farthest customers of S is computed. To do so, the customers of S are sorted by non-decreasing distance to the depot. The least-cost sequence is computed by dynamic programming over a set of $|S|$ stages indexed by $h = 1$ to $|S|$. The state q_h represents the number of units of expected

demand assigned in stage h . In state q_h , we denote the total demand of the assigned customers by the RV ξ_{q_h} . Its distribution is given by $\xi_{q_h} \sim \text{Poisson}(q_h)$ if the demands are modeled as Poisson variables and by $\xi_{q_h} \sim \mathcal{N}(q_h, \text{var}(q_h))$ if the demands are normally distributed. In the case of normal demands, we set $\text{var}(q_h) = \min\{\sum_{i=h}^{|S|} \sigma_i^2 z_i \mid \sum_{i=h}^{|S|} \mu_i z_i = q_h, z_i \in \{0, 1\}\}$ to underestimate the probability of failure, which is the smallest possible variance for the total demand of the assigned customers. Function $G_h(q_h)$ finds the value of the least-cost sequence.

$$G_h(q_h) = \begin{cases} \min \{G_{h+1}(q_h), G_{h+1}(q_h - \mu_h) + \\ 2c_{0h} \sum_{l=1}^{\infty} \mathbb{P}(\xi_{q_h - \mu_h} \leq lQ < \xi_{q_h})\}, & \text{for } h < |S| \text{ and } q_h \geq \mu_h, \\ G_{h+1}(q_h), & \text{for } h < |S| \text{ and } q_h < \mu_h, \\ 0, & \text{for } h = |S| \text{ and } q_h = 0, \\ 2c_{0h} \mathbb{P}(\xi_{q_h} > Q), & \text{for } h = |S| \text{ and } q_h = \mu_h, \\ \infty, & \text{for } h = |S| \text{ and } q_h \notin \{0, \mu_h\}. \end{cases}$$

The first case selects the lowest-cost assignment between the previous stage in state q_h and the previous stage in state $q_h - \mu_h$ plus the expected recourse cost paid at customer h . The second case indicates that customer h cannot be used to fulfill $q_h < \mu_h$ units of expected demand, only those after h can thus be used. The last three cases cover the initial stage. The farthest customer $h = |S|$ can be either visited or not, which corresponds to assigning either 0 or μ_h units of expected demand to the vehicle.

In the second step, the bound is obtained by computing the least-cost assignment of the $\sum_{i \in S} \mu_i$ units of expected demand to the vehicles. This is done by defining another dynamic program that uses the previously computed functions $G_h(q)$ and only allows the h -th customer to be assigned to vehicle k if $k \leq h$. This dynamic

program is defined over m stages indexed by $k = 1$ to m . The state q_k represents the number of units of expected demand that are assigned to the first k vehicles. At stage k , variable x_k decides which quantity is assigned to vehicle k . Then, function $H_k(q_k)$ finds the optimal assignment of q_k units to the first k vehicles. It is defined as follows.

$$H_k(q_k) = \begin{cases} \infty, & \text{if } q_k > kQ, \\ G_k(q_k), & \text{for } k = 1 \text{ and } q_k \leq Q, \\ \min_{0 \leq x_k \leq \min(q_k, Q)} \{G_k(x_k) + H_{k-1}(q_k - x_k)\}, & \text{if } 2 \leq k \leq m. \end{cases}$$

Finally, the expected recourse cost of the optimal assignment of the demands is given by:

$$L_2(S) = H_m(D), \quad (4.40)$$

where $D = \sum_{i \in S} \mu_i$. This value can be computed in $O(mDQ)$.

A set-covering formulation.

The last lower bound is based on a set-covering formulation, where the linear relaxation is solved by column generation. Each column represents a path that visits some customers of S and respects the expected capacity constraint. The problem is to find a set of feasible paths of cardinality m that visits all customers of S and minimizes the total expected recourse cost. Let $\mathcal{P}(S)$ be the set of all feasible paths visiting only customers of S . We define z_p as a binary variable indicating whether path $p \in \mathcal{P}(S)$ is taken or not. Also, the binary coefficient b_{jp} indicates whether customer j is in path p . The formulation is as follows:

$$L_3(S) = \min \sum_{p \in \mathcal{P}(S)} \mathcal{Q}(p) z_p \quad (4.41)$$

$$\text{s.t. } \sum_{p \in \mathcal{P}(S)} b_{ip} z_p \geq 1, \quad i \in S, \quad (4.42)$$

$$\sum_{p \in \mathcal{P}(S)} z_p = m, \quad (4.43)$$

$$z_p \in \{0, 1\}, \quad p \in \mathcal{P}(S). \quad (4.44)$$

The objective function (4.41) minimizes the sum of the expected recourse costs of the selected paths. Constraints (4.42) state that each customer of S must be visited. Constraint (4.43) ensures that exactly m paths are selected, and constraints (4.44) define the domain of the variables.

As the set of paths $\mathcal{P}(S)$ is too large to be generated up front, we rely on column generation to solve the problem. An initial set of columns is first generated to obtain a feasible solution for the linear relaxation. Then, a pricing problem is solved to find a column having a negative reduced cost. If such a column is found, it is added to the model, and the linear relaxation is solved again. This process continues until no more negative reduced cost columns can be found.

The pricing problem of finding the least-cost path can be formulated using the dynamic program of the previous section. The first equation, when adding customer h , is modified by subtracting the value of the dual variable associated with the constraint (4.42) of customer h . A negative cost column is found if the value of the least-cost sequence is less than the value of the dual variable associated with constraint (4.43).

4.5 Implementation of the DL-shaped method

This section details the steps performed when executing the DL-shaped method.

1. Compute $L_m = L_3(N)$ for each number of vehicles $m \in M$.
2. Build model (4.2)-(4.8) with the objective function (4.32) and add constraint (4.35).
3. Solve the linear relaxation of the model.

4. Verify if there are any violated subtour or capacity constraints (4.4) using the CVRPSEP package of [67]. Add the violated inequalities (4.4) to the model. For each identified set S , check if its associated S-Cut is violated, and if so, add the inequality to the model.
5. If the solution is fractional, enumerate its connected components. For each component, verify if an S-Cut is violated, and if one is found, add the S-Cut to the model. If the number of nodes in the component equals the number of non-zero valued edges plus one, then the component is a path. Then, check if the P-Cut associated with the path is violated, and if so, add it to the model.
6. Go back to Step 3 if violated inequalities were found in Steps 4 or 5. Otherwise, branch on a fractional edge variable if the solution is fractional and return to Step 3.
7. For each route in the solution, add the violated P&S-Cuts to the model. Return to Step 3 if violated inequalities were found.

The procedure stops when all the nodes of the branch-and-bound tree have been explored. The incumbent solution is then optimal. In our implementation, the S-Cut associated with a set of customers S is generated using the lower bound $L_1(S)$ if the sum of their expected demands does not exceed the vehicle capacity. Otherwise, the lower bound $L_2(S)$ is used, with $m = \lceil (\sum_{i \in S} \mu_i) / Q \rceil$. To compute the P&S-Cuts of Step 7, we iterate on the routes of the current solution and generate all the subpaths in which at most five customers are removed from the original route. The recourse cost and the single-route lower bound L_1 are then computed for each subpath. The five most violated P-Cuts and the five most violated S-Cuts are added to the model.

4.6 Computational experiments

This section analyzes the performance of the DL-shaped method through computational experiments. All the results were produced with a C++ implementation

with Cplex 12.10 on a machine with an AMD Rome 7532 @ 2.40GHz CPU. For all the computational experiments, we defined a maximum time of one hour per instance. A quick upper bound is also provided to the solver by applying the adaptive large neighborhood search of [68]. The detailed results can be found at <https://sites.google.com/view/jfcote/>.

Section 4.6.1 describes the existing instances that are used in our experiments and introduces a new set of instances. In Section 4.6.2, the new lower bounds on the recourse are compared with classical bounds from the literature. Sections 4.6.3 compare our algorithm to previous methods on two different instance sets. Finally, Section 4.6.4 presents the performance of our method on the new set of instances and summarizes the key takeaways of our experiments.

4.6.1 Characterization of the instances

Computational experiments are based on three sets of instances: two from the literature and a new one. The first set is taken from [1] and contains 270 instances. In each instance, n ranges from 40 to 80, and the demands are normally distributed, with parameters (μ_i, σ_i) such that $\mu_i = 3\sigma_i$ and $\mu_i \in \{1, \dots, 10\}$. The number of vehicles is fixed to $\bar{m} \in \{2, 3, 4\}$ and the filling coefficient $\bar{f} = \frac{\sum_{i \in N} \mu_i}{\bar{m}Q}$ ranges from 0.80 to 0.95. The DL-shaped method can be used on this set since it respects the monotonicity property. We verified that inequality (4.12) holds for each capacity Q in this set of instances, for each number of restocking trips $l \in \{1, 2, 3\}$, and for each possible distribution of $\xi_a \sim \mathcal{N}(\mu_a, \sigma_a^2)$, $\xi_b \sim \mathcal{N}(\mu_b, \sigma_b^2)$, and $\tilde{\xi} \sim \mathcal{N}(\tilde{\mu}, \tilde{\sigma}^2)$. We ignored the case $l \geq 4$ since the probability of more than three restocking trips occurring is negligible.

The second set is taken from classical CVRP instances and contains the 95 instances of the sets A, B, E, F, M, and P of the CVRPLIB repository [69]. In these instances, the fleets are unlimited, and the customer demands are modeled as Poisson RVs. The expectation μ_i of each customer $i \in N$ is set to the deterministic demand of the CVRP instance. Also, as done by [3], [4] and [5], demands and the capacity of each instance are divided by their greatest common divisor.

For the new set, we generated 1980 instances using the method of [1]. To obtain a challenging set of instances, we produced ten instances for each combination of the following problem parameters: $n \in \{20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120\}$, $\bar{m} \in \{2, 3, 4, 5, 6, 7\}$, and $\bar{f} \in \{0.85, 0.90, 0.95\}$. The coefficient of dispersion of the demand of each customer $i \in N$ is given by $D = \sigma_i^2 / \mu_i = 1$. By Proposition 4, these instances therefore respect the monotonicity property.

4.6.2 Lower bounds on the recourse

This section compares our new lower bounds on the recourse to those previously used in the literature based on the relative gap they provide when applied to an optimal first-stage solution. This measure is defined as follows.

$$\text{Gap}\% = \frac{\text{Recourse in the best or optimal solution} - \text{lower bound}}{\text{Recourse in the best or optimal Solution}} \times 100$$

For the first set of instances of [1], the results are grouped by number of customers, whereas the results are presented separately for each set A, B, E, M, and P for the CVRP instances. In Tables 4.1 and 4.2, the columns LLVH02 and LSG18 present the gaps obtained from the implementation of the lower bound from [14] and [16], respectively. The performance of our three lower bounds, L_1 , L_2 , and L_3 is also reported. Column L_1 presents the gaps obtained from computing $\sum_{r \in \mathcal{R}^*} L_1(r)$, where $\mathcal{R}^*$ is the set of routes obtained in either the optimal or best solution found at the end of the DL-shaped method. Columns L_2 and L_3 are obtained by computing $L_2(N)$ and $L_3(N)$. The average computing time of $L_3(N)$ is also reported. The computing time of all the other bounds is negligible.

The lower bound of [16] is systematically the weakest, followed by that of [14]. Their average gap increase with both the fleet size and the number of customers. The bound of [16] assumes a single large vehicle and therefore becomes a poor approximation of the recourse as the fleet size increases. Regarding the bound of [14], it suffers from an underestimation of the variance of the total demand on a route, which worsens as the number of customers grows.

Table 4.1 Relative gap of lower bounds on the instances of [1]

n	#	LLVH02	LSG18	L_1	L_2	L_3	L_3 (s)
40	30	99.9	99.9	3.5	89.7	60.0	0.9
50	60	96.0	99.6	6.9	85.1	55.1	2.9
60	90	88.6	97.2	7.4	78.5	50.8	6.4
70	60	86.3	96.7	9.0	73.8	44.2	13.7
80	30	76.6	94.5	10.2	61.4	30.4	21.7
Average		89.5	97.6	7.4	77.7	48.1	8.8

Table 4.2 Relative gap of lower bounds on the CVRP instances

Set	#	LLVH02	LSG18	L_1	L_2	L_3	L_3 (s)
A	27	67.6	100.0	1.9	67.0	64.9	0.5
B	23	58.4	99.9	0.5	59.6	55.9	0.5
E	13	60.4	99.5	4.0	59.4	56.9	0.7
M	3	97.6	100.0	3.2	100.0	93.8	6.3
P	5	50.8	99.8	2.2	49.3	43.3	7.0
F	24	38.5	95.5	2.0	37.2	35.2	0.4
Average		57.1	98.8	1.9	56.8	53.9	1.0

These results demonstrate that our new bounds significantly improve over the previous bounds from the literature. The set covering bound L_3 achieves an average gap of approximately 50% on average on both sets of instances. The DP bound L_2 is generally slightly less tight than L_3 but is significantly cheaper to compute. The good computational performance of L_2 can be attributed to the fact that the least-cost sequence can be computed by sorting the customers according to their distance to the depot. The single-route lower bound L_1 is by far the tightest, with an average gap of 7.4% for the first set and 1.4% for the second. For a given set of customers, the quality of bounds L_1 and L_2 directly depends on the similarity of the least-cost sequence for the recourse cost provided by Proposition 12 and the total least-cost sequence that takes the first-stage routing cost into account. In particular, the performance of bound L_1 reported in Table 4.2 indicates the substantial similarity of these sequences for the CVRP instances.

4.6.3 Results for the instances of the literature

This section reports results on the two sets of instances of the literature. It compares the efficiency of the DL-shaped method against other BC methods and against BP methods.

Table 4.3 presents the results against the integer L-shaped methods of [1] and [2]. The performance of the three methods is presented under the columns JRGL14, HS23, and DL-Shaped of Table 4.3. Each row of Table 4.3 contains 30 instances with the same number of customers n and the same fleet size $\bar{m}$. The number of optimal solutions found by each method and their average computing time are reported, as well as the average optimality gap of the unsolved instances.

Table 4.3 indicates that the DL-shaped method achieves state-of-the-art results by solving to optimality all the 270 instances of the first set, while [1] and [2] respectively solve 102 and 246 instances. Furthermore, our average solving time is 17.1 seconds per instance, compared to 2711.4 and 185.72 seconds for the instances that were solved to optimality by [1] and [2], respectively. These results highlight the advantage of disaggregating the recourse in the context of the integer L-shaped method, as well as the capacity of our LBFs to approximate the recourse function efficiently.

In Table 4.4, the DL-shaped method is compared to the algorithms of [3] (CL07), [4] (GDG14), [5] (FHM20), and [2] (HS23) on the second set of instances.

A rigorous comparison of these algorithms is difficult to achieve, as the previous works do not all report results for the complete set of instances, and some use the OR policy instead of the DTD policy. Nevertheless, the fact that the DL-shaped method can solve more than a third of the 95 instances of this set is a new milestone for a BC approach. Indeed, the previous integer L-shaped algorithms from the literature have not been successful in solving these instances. [1, 16, 17] did not report any results, and HS23 can only solve 14 of them under the OR policy. Our method achieves comparable performance as GDG14 for the B and E sets and generally produces solutions of good quality, with the average gaps being at most 10% for each set. Also, the DL-shaped method can solve two of the three instances of set F. In one of those,

Table 4.3 Performance comparison with [1, 2] on the instances of [1]

		JRGL14			HS23			DL-Shaped	
n	$\bar{m}$	Opt	Sec	Gap%	Opt	Sec	Gap%	Opt	Sec
40	4	9	1240.7	1.5	28	91.9	1.6	30	2.0
50	3	16	6918.0	0.7	29	101.5	2.9	30	9.8
50	4	5	1360.8	1.9	25	224.8	1.7	30	30.9
60	2	24	1393.0	0.4	30	72.2	-	30	1.2
60	3	6	2766.0	0.7	27	191.4	1.2	30	16.7
60	4	3	4922.0	2.0	25	262.1	1.9	30	25.3
70	2	17	2577.5	0.5	30	99.2	-	30	2.0
70	3	9	1753.3	1.5	24	563.7	1.5	30	46.2
80	2	13	1809.2	0.5	28	193.7	1.0	30	19.8
Total or avg.		102	2711.4	1.2	246	185.7	1.6	270	17.1

Table 4.4 Performance comparison with [3–5] on the CVRP instances. * indicate that the authors use the OR policy.

Set	#	CL07	GDG14	FHM20*	HS23*	DL-Shaped	Gap%	Time (s)
A	27	6/19	22/26	15/27	1	9	6.8	202.1
B	23	-	7/23	-	-	6	4.3	1067.1
E	13	2/3	5/11	3/8	6	7	7.3	127.2
F	3	-	-	-	-	2	1.8	18.7
M	5	-	1/1	-	-	0	9.9	-
P	24	11/18	20/22	17/23	7	11	6.0	143.9
Total or avg.	95	19/40	55/83	35/58	14	35	6.1	306.7
Max.	n/m	6.3	18.5	10.5	10.5	25		

the vehicle capacity is 30,000. For comparison, the highest vehicle capacity is 8,000 among the instances solved to optimality by HS23, and 3,000 for FHM20. These results illustrate the advantage of the DL-shaped method over previous algorithms from the literature for instances with very high vehicle capacities and, more generally, that BC methods are less sensitive than BP methods to large vehicle capacities and expected customer demands. The last aspect in which our method stands out is the length of the routes that can be handled. The last row of Table 4.4 reports the maximal ratio of customers to vehicles among the instances that can be solved to optimality by each method. This value is 25 for the DL-shaped method, more than twice that of FHM20 and HS23. It was achieved on an instance of set P that contains 100 customers and whose optimal solution comprises four routes, the longest of which visits 29 customers.

4.6.4 Results for the new instances

This section presents the results for our new set of instances. For each number of customers and vehicles, Table ?? reports the number of instances, out of 30, that could be solved within the time limit and their average solving time. In total, 1175 instances were solved, and the average optimality gap of the 805 unsolved instances was less than 5% in each group. Even for as many as 120 customers, instances with two vehicles were generally solved easily, with an average time of one minute per solved instance. Although the difficulty of the new instances appears to increase with the number of vehicles, some of the instances that were solved by the DL-shaped method are significantly larger than those previously found in the literature. These include two instances with 50 customers and seven vehicles and one instance with 120 customers and five vehicles.

Figure 4.1 summarizes the parameters of the largest instances that could be solved and the smallest instances that remained unsolved by different BC methods from the literature. For each number of customers and each number of vehicles, the highest filling coefficient $\bar{f}$ for which instances have been solved and the smallest filling coefficient leading to instances that could not be solved are reported. To

Table 4.5 Solved instances and computation times of the DL-shaped method for the new set

$n \backslash \bar{m}$	Number of optimal solutions							Average computing times (s)						
	2	3	4	5	6	7	All	2	3	4	5	6	7	All
20	30	30	30	30	30	30	180	0.0	0.4	1.3	7.9	32.4	10.7	8.8
30	30	30	30	30	24	20	164	0.1	0.6	5.6	276.1	807.3	863.9	275.2
40	30	30	30	28	13	3	134	0.3	1.4	42.2	498.0	867.0	1718.3	236.5
50	30	30	28	23	4	2	117	0.6	45.1	232.2	516.1	913.6	1297.1	222.1
60	30	30	24	17	3	0	104	3.9	12.3	275.3	1114.8	1926.9	-	306.0
70	30	30	22	7	1	0	90	1.6	286.1	272.3	1653.7	400.0	-	295.5
80	30	30	19	10	2	0	91	17.7	198.1	1024.2	1455.7	1667.5	-	481.6
90	30	28	19	6	1	0	84	15.8	233.3	710.4	611.6	2184.9	-	313.8
100	30	25	18	6	0	0	79	16.3	396.9	956.0	1382.5	-	-	454.6
110	30	26	9	5	0	0	70	101.9	520.4	234.7	1484.3	-	-	373.2
120	29	21	11	1	0	0	62	62.2	306.3	317.4	1468.2	-	-	212.8
All	329	310	240	163	78	55	1175	19.9	170.1	318.1	615.2	602.2	460.9	262.3

increase readability, we only present what we consider to be the most important configurations per paper. We also indicate whether the customer demands were modeled as deterministic (D), Poisson RVs (P), or normal RVs (N) in each set of instances. Results from [1] are omitted because they are slightly inferior to those of [2].

Although the figure should be taken with a grain of salt, these results indicate that the DL-shaped method constitutes a significant advancement in the ability to solve instances with a high number of customers and vehicles, and a high filling coefficient. Although previous methods from the literature can solve some instances with a relatively high number of customers, they also fail to solve instances with as few as 30 or 40 customers.

4.7 Conclusion

This paper presents a new approach for solving a class of stochastic integer programs in which the first-stage solutions can be decomposed into disjoint components. The

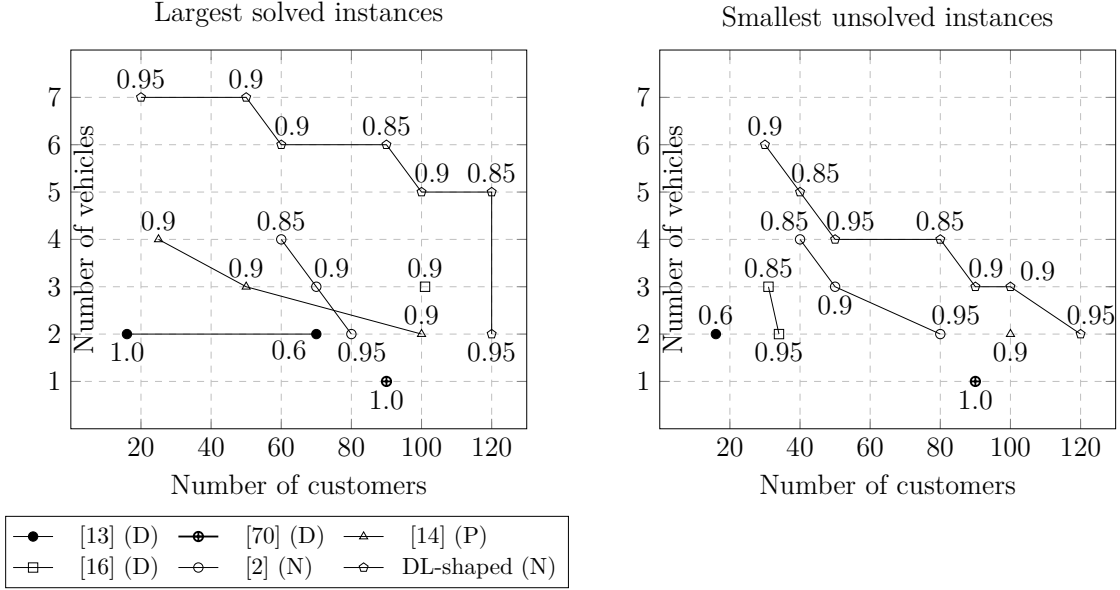


Figure 4.1 Largest solved and smallest unsolved instances in the literature with their filling coefficient

method is applied to the vehicle routing problem with stochastic demands under the detour-to-depot recourse policy. Our computational experiments show that it achieves state-of-the-art results on instances from the literature. Our approach is based on new optimality cuts and exploits new lower bounds on the recourse that are used by new lower bounding functionals. Computational experiments show that our lower bounds improve over the existing ones from the literature.

Regarding future lines of research, the method could be applied to other variants of the stochastic vehicle routing problem and other two-stage stochastic integer programs. Since the efficiency of this method depends on the availability of tight bounds on disjoint components of the recourse, this work may motivate the derivation of such bounds for other problems of interest. Finally, as our method requires the recourse function to be monotonic, it would be interesting to conduct a systematic study of this property for other two-stage stochastic programs.

Acknowledgments

We thank Digital Research Alliance of Canada for providing high-performance computing facilities. Financial support for this work was provided by the Canadian Natural Sciences and Engineering Research Council (NSERC) under Grant 2021-04037. This support is gratefully acknowledged.

CHAPTER 5 ARTICLE 2: An Integer L-Shaped Method for the Static Stochastic Bicycle Repositioning Problem

Authors: Lucas Parada, Jean-François Côté, and Michel Gendreau.

Submitted on July 18, 2024 in European Journal of Operations Research

Abstract: This work deals with the static stochastic bicycle repositioning problem by proposing new formulations, an exact solution method, new valid inequalities, and new lower bounds. The problem is NP-hard, and it arises in the context of bicycle-sharing systems that need to ensure quality of service. The latter is understood as the availability of bicycles and docks to park them in a network of stations. Each station has a random request that can be in the form of either pickups or deliveries of bicycles. The static nature of the problem is because none of the input parameters change during the solution process, and stochasticity is given exclusively in the form of random variables for the station requests. The problem is formulated as a variant of the stochastic vehicle routing problem, and we propose an implementation of the integer L-shaped method to solve it. To speed up the resolution, we create customized lower-bounding functionals based on a complexity result we developed for the recourse problem. The results of computational experiments show that our implementation is superior to those in the literature, which leads us to generate and propose a new, challenging benchmark set of instances.

Keywords: Bicycle repositioning, two-stage mathematical programs with recourse, integer l-shaped method, lower-bounding functionals

5.1 Introduction

The static stochastic bicycle repositioning problem (SBRP) arises in bicycle sharing systems (BSS), a particular case of shared mobility systems when the means of

transport is a set of bicycles [27]. A BSS typically comprises a network of stations with docks (to park the bicycles), requests for bicycles at each station, and a capacitated vehicle fleet to reposition them by picking them up or delivering them. These requests are not known in advance, and they are revealed when a vehicle arrives at a station. There is also a centralized depot from which bicycles can be picked up or delivered. To be an effective means of transportation, the BSS must ensure the quality of service, which is achieved by making available both bicycles and docks [40]. The problem is to determine a set of repositioning decisions for the fleet of vehicles to follow such that requests are exactly or partially satisfied, incurring a non-negative penalty in the latter case. Following the definitions of [12] for static and dynamic stochastic processes, we characterize this repositioning problem as static because it involves data with partially known values represented as random variables whose realizations are only revealed during the visit to the stations.

The SBRP, which has received little attention from the literature on exact methods, is an NP-hard problem, as it can be seen to generalize stochastic vehicle routing problems with recourse by imposing the conditions that each station be visited exactly once, and all vehicle routes must start and end at the depot. The few exact methods that have been implemented for the SBRP have relied on decomposing the problem into simpler sub-problems by mathematical programming techniques. However, these same methods have not taken into account successful techniques that speed up convergence to the optimal solution, like the lower bounding functionals (LBFs) of [61]. Without such techniques, numerical results show that considerable computational times are required to obtain good feasible solutions for the SBRP, and few optimal solutions have been reported.

Recently, the disaggregated integer L-shaped (DL-shaped) method was proposed by [71] as an efficient method to solve stochastic vehicle routing problems, and this paper proposes an implementation of the method to solve the SBRP. The method decomposes solutions into a finite number of disjoint components and then builds on the integer L-shaped method of [7] to introduce different and more efficient LBFs and optimality cuts. These new LBFs are linear inequalities built from lower bounds of

the recourse value of a component, which help to bind the recourse for a large number of solutions [1, 14]. Adding such inequalities in the search process has been shown to reduce the number of enumerated solutions [24]. The suitability of the DL-shaped method for the SBRP arises from how the latter is formulated as a stochastic vehicle routing problem, with the a priori paradigm of [72]. This formulation minimizes the vehicle travel cost plus the expected recourse value. The recourse value is a linear program that minimizes penalties for partially unsatisfied requests, subject to a set of bicycle flow constraints in a capacitated network.

The contributions of this paper are as follows. First, we propose a new dynamic programming (DP) formulation to compute the recourse value. This new formulation has been shown to be very efficient. Next, we propose an implementation of the DL-shaped method for the SBRP, which uses new valid inequalities and new recourse lower bounds. These bounds were inspired by the 1-bicycle repositioning problem, a variant of the recourse problem that does not require a set of a priori routes. We demonstrate that this variant is NP-hard to solve. Lastly, we propose a new set of challenging instances that require several vehicles, unlike the existing benchmark set where the majority of instances need a single vehicle.

The rest of the paper is structured as follows. The problem is mathematically formulated in Section 5.2. Section 5.3 provides new lower bounds for the fleet size and for the recourse. Section 5.4 provides the theoretical and practical aspects behind the DL-shaped method. Section 5.5 describes valid inequalities from the literature and new ones that are used by the DL-shaped method, and Section 5.6 details the implementation of the method. Section 5.7 presents the computational results of our study. Lastly, we present our conclusions in Section 5.8.

5.2 Mathematical formulation

This Section presents the notation that will be used throughout this paper, as well as a mathematical formulation of the problem. Section 5.2.1 provides the definition of the recourse and Section 5.2.2 presents a novel DP formulation to compute the

recourse. Section 5.2.3 proposes a generalization of the recourse and shows the complexity of this new problem.

The SBRP is defined as follows. Let $G = (V, A)$ be a directed graph where $V = \{0, 1, \dots, n\}$ is the set of nodes with node 0 being the depot. $N = \{1, \dots, n\}$ are the station nodes, and $A = \{(i, j) \in V \times V : i \neq j\}$ is the set of arcs. Each arc $(i, j) \in A$ has an associated travel time c_{ij} . Each station $i \in N$ has a maximum capacity of H_i bicycles as well as a desired balanced level d_i (with $H_i \geq d_i$). The random station requests are modelled via a vector (q_i^ξ) for each station i and scenario ξ , in the finite set of all scenarios Ξ . q_i^ξ can take positive or negative values to denote pickups or deliveries respectively. Each scenario ξ occurs with a given probability p^ξ such that $\sum_{\xi \in \Xi} p^\xi = 1$. The scenario request needs to be exactly or partially satisfied, in the latter case incurring a penalty cost of Δ for each unsatisfied bicycle. These partially unsatisfied bicycles cannot exceed the quantities $w_i^+ = \min\{\lceil \delta H_i \rceil, H_i - d_i\}$ and $w_i^- = \min\{\lceil \delta H_i \rceil, d_i\}$, with δ given as an input parameter, and w_i^+ , w_i^- denoting the maximum unsatisfied pickups and deliveries at station i .

Let x_{ij} be a binary variable that indicates whether arc $(i, j) \in A$ is traversed or not, and z be a strictly positive integer variable denoting the number of vehicles to be used. Let $\mathcal{Q}(x)$ denote the total expected recourse cost of solution x . The model follows:

$$\min \sum_{(i,j) \in A} c_{ij}x_{ij} + \mathcal{Q}(x) \quad (5.1)$$

$$\text{s. to } \sum_{i \in V \setminus \{0\}} x_{0i} = z, \quad (5.2)$$

$$\sum_{i \in V} x_{ij} = 1 \quad \forall j \in N, \quad (5.3)$$

$$\sum_{i \in V} x_{ji} = 1 \quad \forall j \in N, \quad (5.4)$$

$$x(S) \leq |S| - 1 \quad S \subseteq N, |S| \geq 2, \quad (5.5)$$

$$x(P) \leq |P| - 1 \quad P \in P^{inf}, \quad (5.6)$$

$$x_{ij} \in \{0, 1\} \quad (i, j) \in A, \quad (5.7)$$

$$z \in \mathbb{Z}^+. \quad (5.8)$$

The objective in (5.1) is to minimize total travel time plus the total expected recourse cost. Constraints (5.2)–(5.4) are connectivity constraints. Constraints (5.5) are sub-tour elimination constraints where $x(S) = \sum_{i,j \in S} x_{ij}$. For computational efficiency, 2-cycles are directly added to the model, whereas 3-cycles and above are dynamically added. Constraints (5.6) are infeasible path constraints that are dynamically added in the resolution, where P denotes a sequence of stations (a path) of the form $P = (i_1, \dots, i_j, \dots, i_t)$, $i_j \in N$, $t \leq n$, $x(P) = \sum_{j=1}^{t-1} x_{j(j+1)}$, and P^{inf} is the set of all infeasible paths. Lastly, equations (5.7)–(5.8) define the domain and the nature of the variables.

5.2.1 Recourse Calculation

The recourse can be calculated as follows. For a given solution x^ν , let $\mathcal{R}^\nu$ be the set of routes, where each one starts and ends at the depot. Then, the expected recourse cost is defined as:

$$\mathcal{Q}(x^\nu) = \sum_{r \in \mathcal{R}^\nu} \mathcal{Q}(r), \quad (5.9)$$

where $\mathcal{Q}(r)$ is the cost of the expected number of unsatisfied bicycles in route $r = (0, i_1, \dots, i_t, 0)$ and is defined as follows:

$$\mathcal{Q}(r) = \Delta \sum_{\xi \in \Xi} p^\xi \mathcal{Q}(\xi, r). \quad (5.10)$$

$\mathcal{Q}(\xi, r)$ is the number of unsatisfied bicycles for route r and scenario ξ , and can be calculated by solving the following linear program proposed by [6]. Let $N(r)$ be the set of stations of route r , and $\pi^+(i)$ be the successor of station i in route r . The number of bicycles inside the vehicle is represented by the variable f_i^ξ , while the bicycles that were not picked up are counted by the variable $w_i^{\xi+}$, and those that were not delivered are counted in $w_i^{\xi-}$. The linear program follows:

$$\mathcal{Q}(\xi, r) = \min \sum_{i \in N(r)} (w_i^{\xi+} + w_i^{\xi-}) \quad (5.11)$$

$$\text{s. to } f_i^\xi = f_{\pi^+(i)}^\xi + q_i^\xi + w_i^{\xi-} - w_i^{\xi+} \quad i \in N(r), \quad (5.12)$$

$$0 \leq w_i^{\xi+} \leq w_i^+ \quad i \in N(r), \quad (5.13)$$

$$0 \leq w_i^{\xi-} \leq w_i^- \quad i \in N(r), \quad (5.14)$$

$$0 \leq f_i^\xi \leq Q \quad i \in N(r). \quad (5.15)$$

The objective (5.11) minimizes the recourse cost of route r by summing all $w_i^{\xi+}$ and $w_i^{\xi-}$ variables. These excess and shortage variables are bounded in constraints (5.13) and (5.14) with $w_i^+ = \min\{\lceil \delta H_i \rceil, H_i - d_i\}$ and $w_i^- = \min\{\lceil \delta H_i \rceil, d_i\}$. Constraints (5.12) and (5.15) are flow conservation of bicycles at each station. Throughout this

paper we adopt the convention that the expected recourse cost of a route of stations $\mathcal{Q}(r) = (0, P, 0)$ will be equivalently denoted as $\mathcal{Q}(P)$, with P being a path. Lastly, whenever the program in (5.11)–(5.15) is infeasible for a given scenario, an infeasible path inequality (5.6) is added to the model.

5.2.2 A New Dynamic Programming Formulation for the Recourse

This Section proposes a DP formulation that can solve the recourse model of Section 5.2.1 in pseudo polynomial time, which turns out to be very efficient in practice.

The recourse problem of route $r = \{0, i_1, \dots, i_t, 0\}$ and scenario ξ is divided in $t + 1$ stages where each stage $k = 1, \dots, t + 1$, corresponds to a visit to a station following the sequence in r . The requests at the stations are denoted by q_k^ξ and can be positive or negative in the case of a pickup or a delivery. Define the variable $x_k^\xi \in [0, Q]$ as the vehicle load after repositioning operations at stage k , and set $x_{t+1}^\xi = 0$. Additionally, for $k = 1, \dots, t$, define the variables $0 \leq w_k^{\xi+} \leq w_k^+$ and $0 \leq w_k^{\xi-} \leq w_k^-$, for the missed pickups and missed deliveries, similar to equations (5.13) and (5.14). Let $H_k^\xi(x_k^\xi)$ be the cost function. The DP algorithm follows:

$$H_k^\xi(x_k^\xi) = \min_{\substack{w_k^{\xi-} \in [0, w_k^-] \\ w_k^{\xi+} \in [0, w_k^+] \\ 0 \leq x_k^\xi + w_k^{\xi-} - w_k^{\xi+} \leq Q - q_k^\xi}} \left\{ w_k^{\xi+} + w_k^{\xi-} + H_{k+1}^\xi(q_k^\xi + x_k^\xi + w_k^{\xi-} - w_k^{\xi+}) \right\} \quad (5.16)$$

The optimal cost is obtained by $\min_{0 \leq x_0 \leq Q} \{H_0^\xi(x_0)\}$.

Solving this DP algorithm has a pseudo polynomial time complexity of $O(nQw_{max})$ in the worst case, where w_{max} is the maximal value of the constants w_k^- and w_k^+ .

5.2.3 Complexity

In [71], the authors demonstrated that the DL-shaped method can be enhanced by incorporating LBFs to gradually bound the second stage cost from below. They also

proposed a new type of LBFs that are based on sets of nodes, which are used in conjunction with their optimality cut. The LBF of a given set of nodes requires calculating a lower bound on the recourse. The problem of finding the highest lower bound can be formulated as finding the route that leads to the lowest recourse. [71] had an easy way to find such a route given the properties of their problem. In this section, we show that finding such a route for the SBRP is an NP-hard problem.

To demonstrate our results, we first introduce a modification of the recourse model from Section 5.2.1. Suppose we want to determine if a specific set of stations can be satisfied by a single vehicle, without considering the routing cost. This setting resembles the recourse model, with the exception that no routes are provided as input. We refer to this problem as the 1-bicycle repositioning problem (1-BRP).

Consider the following 1-BRP instance. Define $N = \{i_1, \dots, i_{13}\}$ stations with one scenario and bicycle requests of $\{3, 6, 6, 7, 7, 7, 8, 8, 8, 9, 22, -22, -22\}$, and a vehicle capacity of $Q = 22$. Assume, without loss of generality, that only stations i_1 and i_{11} allow missed bicycles equaling their requested values of 3 and 22. Figure 5.1 shows a solution to the instance as a sequence of stations starting and ending in the depot. Different colors (grey and red) indicate pickup and delivery requests without missed bicycles, while white denotes stations with missed bicycles. The number of unsatisfied requests is shown by the corresponding w_1^+ , w_{11}^+ values. The residual capacity of the vehicle is also displayed above the arcs.

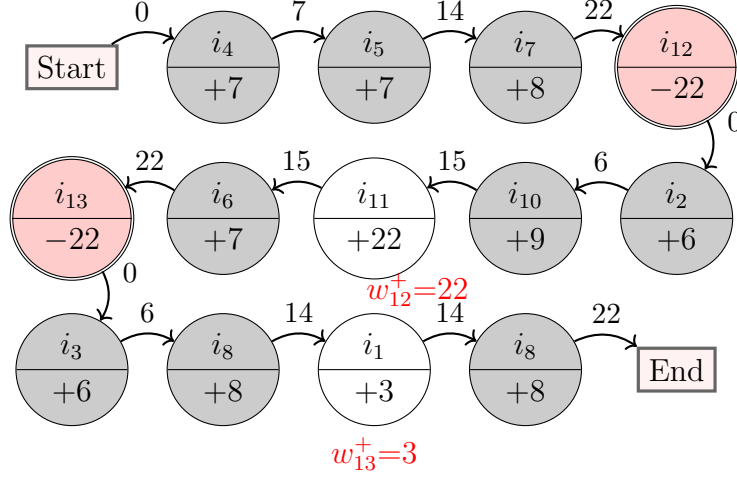


Figure 5.1 A Solution to the 1-Bicycle Repositioning Problem.

A further examination of Figure 5.1 shows that the sequence of stations is not unique. Stations with the same number of bicycle requests, such as i_4 , i_5 , or i_7 , can be swapped, creating different sequences. Notably, stations i_1 and i_{11} can be moved anywhere in the sequence, and the resulting number of missed bicycles will remain the same. However, the solution structure remains unique, consisting of groups of three stations with no unmet bicycle requests, totaling 22 requests, followed by a station with a negative request. This unique triplet arrangement can be used to prove the NP-completeness of the 1-BRP by using a transformation from the following problem.

Definition 5. *3-Partition.* Let B be a finite set of $3s$ non-negative integers $\{b_1, b_2, \dots, b_{3s}\}$ and Q a non-negative integer such that the following two constraints $\sum_{i=1}^{3s} b_i = sQ$ and $\frac{Q}{4} < b_i < \frac{Q}{2}$, for all $1 \leq i \leq 3s$ are satisfied. The goal is to partition B into disjoint sets of triplets, such that the sum of each triplet is Q .

Proposition 13. *Finding a feasible solution for the 1-BRP with a recourse smaller than or equal to k , where $k \geq 0$ and is an integer, is strongly NP-complete.*

Proof. For a given 3-Partition instance, we can create an instance of the 1-BRP

problem in polynomial time using the following transformation. From the $3s$ integers, we generate a set of $4s - 1 + \lceil k/Q \rceil$ stations, where station requests associated with the $3s$ integers are assigned in a way that satisfies the two constraints of 3-Partition. Next, there are $s - 1$ stations with a request of $-Q$. The remaining $\lceil k/Q \rceil$ stations have a request of $k \pmod{Q}$ for the first one and Q for the rest. These $\lceil k/Q \rceil$ stations allow maximum missed requests equaling their demand. All the other stations do not allow missed requests. The vehicle capacity is set to the value of Q .

In a feasible 1-BRP solution created from 3-Partition, the stations are arranged so that there are groups of stations consisting of a 3-Partition triplet plus a negative request station. The remaining stations that allow missed requests can be sequenced anywhere, as they will always have maximum missed requests regardless of their position. Therefore, the 1-BRP has a feasible solution if and only if 3-Partition has one, and since 3-Partition is strongly NP-complete the same holds true for the 1-BRP. $\square$

Figure 5.1 is an example of a recourse of $k = 25$ missed bicycles and a 3-Partition instance $\{6, 6, 7, 7, 7, 8, 8, 8, 9\}$ with $s = 3$. This solution has an equivalent 3-Partition solution given by the three triplets of gray stations.

In Proposition 13, we encounter a decision problem related to the 1-BRP. However, our focus is on finding the minimum recourse sequence, as this would establish a lower bound for the SBRP. The latter is the 1-BRP optimization problem. In other words, while we have a certificate of NP-completeness for the decision problem, we are concerned with the optimization problem's complexity. It's evident that a polynomial algorithm for the optimization problem would also address the decision version, constituting a Turing reduction between the two problems [73]. This, coupled with the NP-completeness certificate, serves as proof that the 1-BRP optimization problem is NP-hard.

5.3 Lower Bounds

This Section proposes lower bounds for the fleet size and the expected recourse cost that are used to develop new valid inequalities for the SBRP. Section 5.3.1 presents the lower bounds for the fleet size and Section 5.3.2 presents the lower bounds for the recourse.

5.3.1 Lower Bounds for the Fleet Size

This section proposes two lower bounds on the fleet size required to serve a set of stations $S \subseteq N$. Section 5.2.3 demonstrates that finding the least-recourse feasible route to serve a given set S of stations is an NP-hard problem. We thus turn our attention to proposing conditions that determine whether S can be feasibly served or not. These conditions are denoted as feasibility conditions, and the first bound proposed in this section uses them by enumerating fleet sizes starting from one vehicle. The second bound embeds the conditions in a set covering formulation. We begin by proposing the feasibility conditions.

The feasibility conditions build on the following idea. For one scenario, a set of pickup request stations can be paired together with a set of delivery request stations to reduce the unsatisfied request of the fleet that visits those two sets. Assume that these two sets of stations are visited by a fleet of l vehicles and consequently the total fleet capacity is lQ . Then, for the set of pickup requests, we determine if the sum of the maximum unsatisfied pickup quantities, the sum of delivery requests, and the fleet capacity is less than the sum of the pickup requests. If this condition does not hold, it implies that the set S is infeasible. Similarly, for the delivery requests, we evaluate whether the sum of the maximum unsatisfied delivery quantities, the sum of pickup requests, and the fleet capacity is less than the sum of the delivery requests. If this condition is not satisfied, the set S is also deemed infeasible. These two relations for pickups and deliveries are consequently referred to as feasibility conditions and they are as follows:

$$\sum_{\substack{i \in S \\ q_i^\xi > 0}} q_i^\xi \leq lQ + \sum_{i \in S} w_i^+ + \sum_{\substack{i \in S \\ q_i^\xi < 0}} |q_i^\xi|, \quad (5.17)$$

$$\sum_{\substack{i \in S \\ q_i^\xi < 0}} |q_i^\xi| \leq lQ + \sum_{i \in S} w_i^- + \sum_{\substack{i \in S \\ q_i^\xi > 0}} q_i^\xi, \quad (5.18)$$

where equation (5.17) is the pickup feasibility condition and (5.18) the delivery feasibility condition. Both conditions need to be satisfied for all scenarios.

In cases where the set S is found to be infeasible based on either the pickup or delivery condition, we increase l until both conditions are satisfied.

The conditions can be used to find a lower bound of the fleet size to satisfy the requests for the entire set of stations N . Let l_N^+, l_N^- be such that:

$$l_N^+ = \min_{l \geq 1} \left\{ l : \sum_{\substack{i \in N \\ q_i^\xi > 0}} q_i^\xi \leq lQ + \sum_{i \in N} w_i^+ + \sum_{\substack{i \in N \\ q_i^\xi < 0}} |q_i^\xi|, \forall \xi \in \Xi \right\}, \quad (5.19)$$

$$l_N^- = \min_{l \geq 1} \left\{ l : \sum_{\substack{i \in N \\ q_i^\xi < 0}} |q_i^\xi| \leq lQ + \sum_{i \in N} w_i^- + \sum_{\substack{i \in N \\ q_i^\xi > 0}} q_i^\xi, \forall \xi \in \Xi \right\}. \quad (5.20)$$

After computing equations (5.19), (5.20) for all scenarios, the following lower bound K_0 on the fleet size arises:

$$K_0 = \max\{l_N^+, l_N^-\}. \quad (5.21)$$

The second lower bound on the fleet size proposed in this Section is given by the following set covering formulation. Let $\mathcal{K}$ be a collection of sets of stations that satisfy conditions (5.17) and (5.18) for one vehicle:

$$\mathcal{K} = \left\{ S \subseteq N \mid \sum_{\substack{i \in S \\ q_i^\xi < 0}} |q_i^\xi| \leq Q + \sum_{i \in S} w_i^- + \sum_{\substack{i \in S \\ q_i^\xi > 0}} q_i^\xi, \text{ and} \right. \\ \left. \sum_{\substack{i \in S \\ q_i^\xi > 0}} q_i^\xi \leq Q + \sum_{i \in S} w_i^+ + \sum_{\substack{i \in S \\ q_i^\xi < 0}} |q_i^\xi|, \xi \in \Xi \right\}.$$

Let ζ_k be a binary variable that takes a value of 1 if set $k \in \mathcal{K}$ is taken and 0 if it is not, and b_{ik} be a parameter taking a value of 1 if station i is in set k or 0 if it is not. The set covering formulation follows:

$$\min \sum_{k \in \mathcal{K}} \zeta_k \tag{5.22}$$

$$\text{s.t. } \sum_{k \in \mathcal{K}} b_{ik} \zeta_k \geq 1 \quad \forall i \in N, \tag{5.23}$$

$$0 \leq \zeta_k \leq 1, \quad k \in \mathcal{K}. \tag{5.24}$$

The lower bound on the fleet size is given by K_1 , computed as the ceiling of the optimal solution to the problem in (5.22)–(5.24).

The size of $\mathcal{K}$ is exponential, and solving (5.22)–(5.24) to optimality is impractical. The following column generation (CG) approach is applied. Let α_i be the dual variable associated with the covering constraints (5.23) and z_i a variable taking a value of 1 whenever station i is in subset k . The pricing problem is as follows:

$$\max \sum_{i \in N} \alpha_i z_i \quad (5.25)$$

$$\text{s.t. } \sum_{\substack{i \in N \\ q_i^\xi < 0}} |q_i^\xi| z_i - \sum_{i \in N} w_i^- z_i - \sum_{\substack{i \in N \\ q_i^\xi > 0}} q_i^\xi z_i \leq Q \quad \xi \in \Xi, \quad (5.26)$$

$$\sum_{\substack{i \in N \\ q_i^\xi > 0}} q_i^\xi z_i - \sum_{i \in N} w_i^+ z_i - \sum_{\substack{i \in N \\ q_i^\xi < 0}} |q_i^\xi| z_i \leq Q \quad \xi \in \Xi, \quad (5.27)$$

$$z_i \in \{0, 1\} \quad i \in N. \quad (5.28)$$

The objective (5.25) maximizes the value of the reduced cost of the entering variable z_i to the set covering problem. Equations (5.26)–(5.27) are the feasibility conditions for one vehicle. Equations (5.28) are variable definitions.

Lastly, we can compare both lower bounds K_0 , K_1 in the following expression:

$$K = \max\{K_0, K_1\}. \quad (5.29)$$

We set K as the lower bound on the fleet size.

5.3.2 Lower Bounds for the Recourse

This Section proposes two recourse lower bounds for the recourse of the SBRP. The bounds use the results of Section 5.3.1, namely the fleet size required for a set of stations $S \subseteq N$.

Let $l_S = \max\{l_S^-, l_S^+\}$ be the fleet size required to serve S , given by the feasibility conditions (5.17) and (5.18). Then, for a given scenario ξ , a lower bound of the recourse of S is given as follows:

$$L_0(\xi, S, l_S) = \Delta \max\{0, |\sum_{i \in S} q_i^\xi| - l_S Q\}, \quad (5.30)$$

where the quantity $|\sum_{i \in S} q_i^\xi| - l_S Q$ is the maximum unsatisfied requests in S given that l_S vehicles are minimally required. Next, by setting $S = N$ in equation (5.30), and considering the fleet size lower bound K , the following lower bound on the recourse arises if K vehicles are used:

$$L_1(K) = \sum_{\xi \in \Xi} p_\xi L_0(\xi, N, K), \quad (5.31)$$

5.4 The DL-shaped Method

This Section presents the theoretical and practical aspects of the method. First, the recourse function $\mathcal{Q}(x)$ is relaxed and bounded from below by the sum of a set of continuous non-negative variables θ_i defined for each station i . The authors refer to the inclusion of these variables in the model as a way to disaggregate the recourse. A master program is then constructed with the following objective function:

$$\min \sum_{(i,j) \in A} c_{ij} x_{ij} + \sum_{i \in N} \theta_i. \quad (5.32)$$

The value of θ_i variables can be viewed as their contribution to the cost of the second stage, and this value is optimized by dynamically adding optimality cuts from [71], denoted as path cuts (P-cuts). A P-cut is added for each feasible path of stations found during the resolution, and the cut binds the x_{ij} variables of the path to the θ_i variables of the stations in that path. Consequently, a P-cut will be active whenever the corresponding path is active in a solution. The P-cut follows:

$$\sum_{i \in N(P)} \theta_i \geq \mathcal{Q}(P) (x(P) - |P| + 1), \quad (5.33)$$

where $N(P)$ is the set of stations in P . In [71], it was proved that if the recourse function $\mathcal{Q}(x)$ respects the *monotonicity condition*, then the method will yield an optimal solution, if one exists, in a finite number of iterations of the method. This

property requires that the expected recourse cost of a path P must be no less than that of any path P' , where P' is a subpath of P obtained by the removal of stations at the start or at the end of P . Consequently, P' is denoted as a subpath of P and its definition follows:

Definition 6. Let $P = (i_1, \dots, i_t)$ be a path of stations that yields a feasible solution to the recourse problem with respect to equations (5.11)–(5.15) for all scenarios. P' is a subpath of P if it can be written as $P' = (i_a, \dots, i_b)$ for some $1 \leq a \leq b \leq t$, and it also yields a feasible solution to the recourse problem in equations (5.11)–(5.15) for all scenarios.

Next, denote the set of subpaths of path P as $\mathcal{P}$. As [6] state, removing the first and/or last stations of a feasible path yields a feasible subpath, and this means that all the subpaths in $\mathcal{P}$ are feasible. We extend the definition of the monotonicity condition of [71] to the SBRP as follows:

Definition 7. An instance of the SBRP satisfies the monotonicity condition if, for any feasible path $P \subseteq N$, the following inequality holds:

$$\mathcal{Q}(P) \geq \mathcal{Q}(P'), \quad P' \in \mathcal{P}, \quad (5.34)$$

Proposition 14 proves that the recourse function of the SBRP respects the monotonicity condition.

Proposition 14. The SBRP respects the monotonicity condition.

Proof. The expected recourse cost for both $\mathcal{Q}(P)$ and $\mathcal{Q}(P')$ is computed using the program in equations (5.11)–(5.15). As P has at least one additional station compared to P' , the program $\mathcal{Q}(P)$ will include one extra flow variable and two penalty variables, all of which are non-negative. Furthermore, the objective in equation (5.11) is monotonically increasing with respect to the number of $w_i^{\xi+}$ and $w_i^{\xi-}$ variables. This implies that the additional penalty variables in $\mathcal{Q}(P)$ cannot reduce the objective's value compared to the program $\mathcal{Q}(P')$. It follows that the cost $\mathcal{Q}(P)$ is necessarily greater than or equal to $\mathcal{Q}(P')$. $\square$

5.5 Valid Inequalities

This Section builds on the theoretical properties described in Section 5.3 and on the disaggregation of the recourse in Section 5.4 to develop new valid inequalities for the SBRP.

5.5.1 Recourse Lower Bound Inequalities

To improve convergence of the method, the DL-shaped method relies on a new type of LBF, called set cuts (S-cuts) [71]. In the SBRP, these S-cuts take a set S of stations as input, the minimum number of vehicles required to serve S as $l_S = \max\{l_S^-, l_S^+\}$, and a lower bound on the expected recourse of this set, which we denote as $L(S) = \sum_{\xi \in \Xi} p^\xi L_0(\xi, S, l_S)$. The result is a cut that bounds the set whenever a path that consecutively visits the stations of S is active in a given solution.

In Section 5.4, the LBFs were mentioned as a means to accelerate the convergence of the DL-shaped method to the optimal solution. Then, for a set $S \subseteq N$, inequality:

$$\sum_{i \in S} \theta_i \geq L(S) (x(S) - |S| + l_S + 1), \quad (5.35)$$

is a S-cut that is valid for the SBRP. In the inequality, l_S is the minimum number of vehicles required to serve S . We refer to [71] for the proof that the S-cuts are valid inequalities.

Next, the following inequality:

$$\sum_{i \in N} \theta_i \geq L_1(K) - \Delta Q(z - K) \quad (5.36)$$

is valid for the SBRP. Validity of this inequality arises from the fact that $L_1(K)$ was proposed in Section 5.3.2 as a lower bound on the recourse when K vehicles are used to satisfy all the stations in N . If $z > K$, this means that there is at least $Q(z - K)$ additional bicycle requests that can be satisfied. The recourse can be penalized by

a factor of Δ times the additional bicycle requests served.

From equation (5.30), another valid inequality can be developed if all arcs in G are individually considered. Let $\sigma_{ij}^\xi = \max\{0, |q_i^\xi + q_j^\xi| - Q\}$ be the recourse cost associated with traversing arc (i, j) in scenario ξ . Then, the inequality:

$$\sum_{i \in N} \theta_i \geq \Delta \sum_{(i,j) \in A} \sum_{\xi \in \Xi} p^\xi \sigma_{ij}^\xi x_{ij}, \quad (5.37)$$

is valid for the SBRP. Validity of this inequality is given by first defining the set $S = \{i, j\}$ for each arc $(i, j) \in A$. In the SBRP, at most one vehicle can traverse each arc, so equation (5.30) for S reduces to $L_0(\xi, S, 1) = \max\{0, |q_i^\xi + q_j^\xi| - Q\}$. It follows that any solution using a given set of arcs cannot have a smaller expected recourse cost than the sum of the lower bounds on each arc.

Equations (5.36) and (5.37) are added at the root node of the DL-shaped method.

5.5.2 Infeasible Path Inequalities

The infeasible path inequality (5.6) is a valid inequality that is dynamically added whenever there is a path P in the resolution that is infeasible for the recourse problem in (5.11)–(5.15), for at least one scenario. In this case, we search for the smallest infeasible subpath and add an inequality (5.6) for it.

5.5.3 Infeasible Set Inequalities

The conditions defined by equations (5.17) and (5.18) can be extended to find sets of stations that do not satisfy either condition. First, suppose that an infeasible set of stations $S \subset N$ is given. From equations (5.19) and (5.20), compute $l_S = \max\{l_S^-, l_S^+\}$ as the minimum number of vehicles required to satisfy S . The infeasible set inequality follows:

$$x(S) \leq |S| - l_S, \quad (5.38)$$

Validity of this inequality is due to the fact that l_S is a lower bound of the fleet required to visit the stations in S . Then, equation (5.38) takes the form of the classical rounded capacity constraints, which are valid inequalities for variants of the capacitated vehicle routing problem and, in particular, for the SBRP.

There are two ways in which we identify infeasible sets of stations. First, starting from a given route $r = (0, i_1, \dots, i_t, 0)$, denote set $S = \{i_1, \dots, i_t\}$ and compute l_S . If l_S is strictly greater than one, then this means that the route requires at least one more vehicle and the set S is infeasible. Second, any set of stations violating a subtour elimination constraint is also an infeasible set.

5.5.4 Infeasible Arc Inequalities

The infeasible arc inequalities were proposed in [6] as a direct consequence of properties taken from the traveling salesman problem with pickups and deliveries, as proposed by [58]. These properties define a set of infeasible arcs A^{inf} and the inequalities are as follows:

$$x_{ij} + \sum_{h \in N} x_{jh} \leq 1, \quad (i, j) \in A^{inf}, \quad (5.39)$$

$$\sum_{h \in N} x_{hi} + x_{ij} \leq 1, \quad (i, j) \in A^{inf}. \quad (5.40)$$

We include these inequalities at the root node of our branch and cut framework.

5.6 Implementations of the DL-shaped Method

To solve the SBRP we implemented three different variants of the DL-shaped method proposed by [71]. The variants differ based on the choice of one of three types of optimality cut separation routines that we propose.

The DL-shaped method solves the problem like a branch-and-cut algorithm. A model is created in mixed-integer solver that takes care of solving linear relaxations of

the model and perform branching to find integer solutions. At each node of the branch-and-bound tree, user-defined methods are called by the solver to identify violated inequalities. In those methods, we search for violated subtour elimination, S-cuts, infeasible sets, infeasible paths, and optimality cuts. The solver initialization includes the following two steps:

- a) Compute a lower bound on the fleet size as k and on the expected recourse cost as $L_1(K)$
- b) Build the model with objective (5.32) and constraints (5.2)–(5.8), (5.36), (5.37), (5.39), (5.40). Provide an upper bound value to the problem, if one is given.

For Step (b) we provide an upper bound computed by means of the Adaptive Large Neighborhood Search (ALNS) heuristic of [68].

Next, upon an incumbent solution x'' found by the solver, the following four steps are executed:

- c) Build a graph $G(x'')$ with all the non-zero valued arcs in x'' and enumerate all connected components of $G(x'')$.
- d) For each connect component S of $G(x'')$, check if it violates subtour inequality (5.5). If so, compute the minimum number of vehicles required to satisfy S , namely $l_S = \max\{l_S^-, l_S^+\}$ from equations (5.19) and (5.20), and add an infeasible set inequality (5.38) to the model.
- e) For each connect component S with a violated subtour or infeasible set inequality, compute recourse lower bound given by (5.30). If its S-cut (5.35) is violated, add it to the model.
- f) If no violated inequality has been found in steps (c)–(e) and the solution x'' is integral, check the feasibility of each path of the solution by computing the end load feasibility window of [6]. Add an infeasible path inequality (5.6) for each infeasible path.

If no inequality is added in steps (a)-(f) then solution x^ν is feasible and we proceed with one of three optimality cut separation routines:

1. *P&S cuts*: For each route in solution x^ν , we enumerate all subpaths of contiguous stations of length two or more. For each subpath P , check if the set S that is composed of the stations in P violates a S-cut, and if so, add it to the model. Also, compute the recourse cost of subpath P using (5.16). Check if P violates a P-cut, and if so, add it to the model. We only add the 5 most violated cuts of each kind.
2. *Benders cuts*: Find a violated Benders according to equation (18) in [6]. To do so, formulate and solve the linear relaxation of the dual of the subproblem given by equations (5.11)–(5.15). The cut is given by the optimal value of the dual variables.
3. *Hybrid cuts*: Find a mixture of violated P&S cuts and Benders cuts. To do so, we invoke the Benders cuts separation routine whenever the solution uses one vehicle and the P&S cuts separation routine otherwise. This approach was inspired by initial computational experiments that showed the effectiveness of the Benders cut whenever the solution uses exactly one vehicle, and the P&S cuts for solutions using multiple vehicles.

The procedure stops when all the nodes from the branch-and-bound tree have been explore. In this case, the incumbent solution is the optimal one.

5.7 Computational Results

This Section presents the results obtained for our implementations of the DL-shaped method in the set of instances of [6], and a newly generated set of challenging instances. The DL-shaped method was implemented in C++ with Cplex 12.10, and experiments were run in a CPU with the following characteristics: Intel E5-2683 v4 Broadwell @ 2.1Ghz. A maximum run time of 1 h was set for each instance.

Section 5.7.1 describes the existing instances for the SBRP and introduces the new set of instances. In Section 5.7.2, the results of our DL-shaped implementations are compared with the best results of [6]. Section 5.7.3 shows the effectiveness of our new valid inequalities. Section 5.7.4 presents the results of our vehicle lower bounds K_0 and K_1 . Finally, Section 5.7.5 presents the results for the new set of instances. These latter results are compared with our implementation of the Multicut algorithm of [6], the best algorithm of the authors.

We adopt the following formulas to present our results. For an individual instance, the gap percentage is computed as $(UB - LB)/LB \times 100$, where UB and LB represent the upper and lower bounds. When calculating the average gaps for instances within a same class, we use the non-optimal solutions as the denominator. [6] use different conventions, so to ensure comparability with their results, we re-compute their numerical values according to our adopted formulas

5.7.1 Characterization of the Instances

[6] proposed 22 instances that were developed from real world data gathered from BSS of mostly American cities. Each instance varies from 20 to 100 stations and has $|\Xi| = \{30, 91\}$ scenario realizations of the requests. The authors use $\Delta = \lceil \epsilon c_{min} \rceil$ where ϵ is a parameter and $c_{min} = \min_{(i,j) \in A} \{c_{ij}\}$. The parameters ϵ, δ are each considered to have the following values: $\{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. With the various parameter combinations, a total of 242 instances are defined across 11 distinct ϵ, δ configurations. The majority of these 242 instances require one vehicle to be solved, and the scenarios for a given instance are generally similar to one another in terms of the station requests.

Our numerical results demonstrate that instances with similar scenario requests and requiring only one vehicle, such as those from [6], are easier to solve compared to those with very different scenario requests and multiple vehicles. The main reason why is because the resolution of the latter encounters a higher number of infeasible solutions, necessitating efficient feasibility checks and recourse bounding procedures by the solution algorithm. Inspired by this fact we generated 100 new instances,

with different scenario requests and requiring multiple vehicles.

Specifically, to generate the new set of 100 instances we placed a given number of $|V|$ stations (including the depot) in a $[0,100] \times [0,100]$ area. To obtain a challenging set, station requests were randomly defined in the interval $[-Q, Q]$ within a fixed number of $|\Xi| = 20$ scenarios. Lastly, we fixed $\epsilon = \delta = 0.2$ for this new set.

5.7.2 Results for Values of ϵ and δ

Results for pairs ϵ, δ are presented in Table 5.1. The rows show each of the ϵ, δ configurations that were tested (11 in total). Each configuration shows the values of the Multicut algorithm and our values that were obtained by solving with the three types of optimality cuts detailed in Section 5.4. The first column gives the values of ‘ δ ’ or ‘ ϵ ’, according to the respective parameter value. When ‘ δ ’ varies, $\epsilon = 0.2$ is set, and conversely, when ‘ ϵ ’ varies, $\delta = 0.2$ is set. ‘Ins’ denotes the number of instances for each pair ϵ, δ . ‘Opt’ denotes the number of optimal solutions found, ‘Gap%’ denotes the average percentage gap of the non-optimal instances, and ‘Time’ denotes the average time in seconds to reach either the optimal solution or the best solution found.

Table 5.1 Results for the instances of [6]

		Multicut			P&S cuts			Benders cuts			Hybrid cuts		
δ	Ins	Opt	Gap%	Time	Opt	Gap%	Time	Opt	Gap%	Time	Opt	Gap%	Time
0.0	22	2	203.9	3287.5	7	38.2	2427.3	7	38.2	2521.7	7	38.1	2490.1
0.2	22	11	34.2	1847.9	13	22.3	1436.5	13	22.2	1519.0	13	22.2	1542.4
0.4	22	17	2.8	872.2	18	1.9	939.4	20	3.0	761.9	18	1.5	796.6
0.6	22	20	0.3	476.0	17	0.9	1072.7	19	1.1	922.3	20	1.6	820.1
0.8	22	20	0.3	473.3	16	0.9	1265.2	19	1.1	830.0	19	1.1	921.0
1.0	22	20	0.3	461.3	16	0.9	1303.7	20	1.5	952.2	19	1.1	917.2
Total/Avg.	132	90	40.3	1236.4	87	10.8	1407.5	98	11.2	1251.2	96	10.9	1247.9
ϵ		Opt	Gap%	Time	Opt	Gap%	Time	Opt	Gap%	Time	Opt	Gap%	Time
0.0	22	13	38.5	1690.1	14	23.9	1249.5	14	23.9	1306.6	14	24.0	1292.3
0.2	22	11	34.2	1847.9	13	22.3	1436.5	13	22.2	1519.0	13	22.2	1542.4
0.4	22	10	32.9	2003.3	12	20.1	1688.1	13	23.4	1609.2	12	20.3	1577.2
0.6	22	10	35.4	2063.2	9	16.9	2080.8	11	19.8	1771.6	10	18.5	1914.6
0.8	22	10	37.2	2220.8	9	18.1	2074.6	10	19.0	2044.4	9	18.5	2032.9
1.0	22	8	33.0	2303.5	9	18.4	2123.6	10	19.7	2094.8	9	18.6	2054.3
Total/Avg.	132	62	35.2	2021.5	66	20.0	1775.5	71	21.3	1724.3	67	20.3	1735.6
Global Tot.	264	152	37.8	1628.9	153	15.4	1591.5	169	16.3	1487.7	163	15.6	1491.8

Our methods are superior when compared to the Multicut algorithm. For example, for Benders cuts, with respect to the 11 possible parameter pairs, our implementation of the DL-shaped method improves results from the Multicut algorithm in 7 of these parameter pairs. Specifically, the improved parameter pairs are $\epsilon = 0.2$, $\delta = \{0.0, 0.2, 0.4\}$ and $\epsilon = \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$, $\delta = 0.2$.

When we compare the performance of our three types of optimality cuts, the Benders cut performs best, although this is partly owing to our available instance set, which consists mostly of instances requiring one vehicle to solve. To show this, Table 5.2 presents the results of the DL-shaped method and Multicut algorithm only for the instances whose known solution requires more than one vehicle.

Table 5.2 Results for only the multi vehicle instances

		Multicut			P&S cuts			Benders cuts			Hybrid cuts		
δ	Ins	Opt	Gap%	Time	Opt	Gap%	Time	Opt	Gap%	Time	Opt	Gap%	Time
0.0	6	1	64.6	3054.1	6	0.0	305.1	6	0.0	276.4	6	0.0	264.1
0.2	3	3	0.0	258.8	3	0.0	121.0	3	0.0	37.2	3	0.0	119.2
0.4	1	0	3.3	3600.0	1	0.0	3207.4	1	0.0	3411.3	0	0.0	3590.0
0.6	-	-	-	-	-	-	-	-	-	-	-	-	-
0.8	-	-	-	-	-	-	-	-	-	-	-	-	-
1.0	-	-	-	-	-	-	-	-	-	-	-	-	-
	10	4	22.6	2304.3	10	0.0	1211.2	10	0.0	1241.6	9	0.0	1324.5
ϵ	Ins	Opt	Gap%	Time	Opt	Gap%	Time	Opt	Gap%	Time	Opt	Gap%	Time
0.0	3	3	0.0	27.1	3	0.0	0.4	3	0.0	0.4	3	0.0	0.5
0.2	3	3	0.0	258.8	3	0.0	121.0	3	0.0	37.2	3	0.0	119.2
0.4	3	2	0.5	1275.0	2	1.5	1242.9	3	0.0	576.6	2	1.5	1202.7
0.6	3	3	0.0	220.7	2	2.4	865.9	3	0.0	308.3	2	2.3	1207.3
0.8	3	3	0.0	920.1	2	4.7	900.3	2	1.4	1200.1	2	4.7	1209.5
1.0	2	2	0.0	57.1	2	0.0	268.2	2	0.0	34.9	2	0.0	184.4
Total/Avg.	17	16	0.1	459.8	14	1.4	566.4	16	0.2	359.6	14	1.4	653.9
Global Tot.	27	20	7.6	1074.6	24	1.0	781.3	26	0.2	653.6	23	1.0	877.4

The column headers in Table 5.2 are the same as Table 5.1, with the exception of a different meaning for the columns ‘Ins’ and ‘Opt’. In Table 5.2, these columns count the number of instances whose known optimal solution uses more than one vehicle, and the number of these latter instances that are optimal for each type of optimality cut or the Multicut algorithm. A value of ‘-’ indicates that no multi vehicle solution is known for the given ϵ, δ pair.

These multi vehicle instances are few in comparison to the total number of solved instances (10 and 17 for the different ϵ, δ pairs in comparison to the 98 and 71 that the Benders cut solve in Table 5.1), but they show that our three optimality cuts perform similarly, as all three of them achieve a similar number of optimal solutions. Additionally, the superiority of our methods compared to the Multicut algorithm is preserved, as all three optimality cuts achieve significantly more optimal solutions

for the different values of δ .

5.7.3 Effectiveness of the Valid Inequalities

This Section analyzes the effectiveness of our proposed valid inequalities. We created 8 different configurations of the DL-shaped method to test. They are divided into two groups of 4 configurations each, where the first group uses the Benders cuts and the second one uses the P-cuts. Each configuration can be differentiated by the use or not of the S-cut (5.35) and by the use or not of the Infeasible Set inequality (5.38). Results are presented in Table 5.3 on the 242 instances of [6]. Each instance was run for a maximum of 5 minutes. Column ‘Opt’ indicates the number of optimal solutions found by the configuration, ‘Gap%’ denotes the average percentage gap of the non-optimal instances, ‘Time’ denotes the average time to either the optimal solution or the best one found within the time limit of 5 minutes, and ‘Inf.Set’, ‘Inf.Path’, ‘S-cuts’, ‘Benders’ and ‘P-cut’ are the average number of cuts of type of equations (5.38), (5.6), (5.35), Benders optimality and (5.33).

Table 5.3 Convergence with and without the S-cuts and Infeasible Sets.

Optimality Cut: Benders								
S-Cut	Inf. Set	Opt	Gap	Time	Inf. Sets	Inf. Paths	S-Cuts	Benders
Yes	Yes	125	26.0	197.7	321.9	0.0	422.5	35.7
No	Yes	118	28.5	211.7	470.3	0.0	0.0	63.3
Yes	No	116	25.1	215.7	0.0	382.4	356.2	31.3
No	No	111	29.0	220.4	0.0	464.3	0.0	49.5
Optimality Cut: P-cuts								
S-Cut	Inf. Set	Opt	Gap	Time	Inf. Sets	Inf. Paths	S-Cuts	P-Cuts
Yes	Yes	119	23.4	168.6	365.3	0.0	549.7	481.1
No	Yes	105	22.1	184.3	775.6	0.0	0.0	691.0
Yes	No	110	23.6	181.0	0.0	305.0	431.2	363.2
No	No	102	21.5	186.4	0.0	509.3	0.0	186.4

As shown in Table 5.3, the use of S-cuts and infeasible sets accelerates the convergence of the DL-shaped method. Specifically, when employing the Benders optimality cuts, the utilization of S-cuts alongside infeasible sets resulted in a notable improvement in convergence. Instances where both S-cuts and infeasible sets were implemented showcased a reduced average gap percentage and a higher number of optimal solutions within the stipulated time frame compared to runs where these cuts were not utilized. Similarly, in the context of P-cuts, the incorporation of S-cuts and infeasible sets also demonstrated a similar trend. The average gap percentage decreased, and the number of optimal solutions increased notably in instances where these cuts were employed in conjunction.

Lastly, we note that when a route is infeasible, it seems there is always an infeasible set of stations that can be found. This is an interesting results, because infeasible set inequalities are stronger than infeasible path inequalities. We noted this behavior also in our other results.

5.7.4 Results for Lower Bounds on the Fleet Size

This Section analyzes the effectiveness of the two new lower bounds K_0 and K_1 . To do so, we first present results without these two bounds and we then present the values of K_0 and K_1 for the instances with $\delta = 0.0$, because these are the instances that require the largest fleet sizes in the set of [6].

The results without the lower bound K are given in Table 5.4 for the 242 instances of [6] with the following columns. The column ‘ K ’ indicates the lower bound used and the columns ‘Ins’ and ‘Opt’ count the number of instances and optimal solution in the set. The column ‘Gap%’ is the average percentage gap of the non-optimal instances and the column ‘Time’ is the average time to either the optimal solution or the best one found within the time limit of 5 minutes.

Table 5.4 Effectiveness of the vehicle lower bound K in a 5 minute-run.

K	Opt	Gap%	Time
$\max\{K_0, K_1\}$	125	26.0	197.7
1	117	34.6	213.2

The effect of K is significant. When utilizing the maximum value between K_0 and K_1 as the lower bound, a notable increase in the number of optimal solutions is observed, with a corresponding decrease in the average percentage gap for non-optimal instances. Next, for the specific case of $\delta = 0.0$ the results are presented in Table 5.5. In this Table the column ‘Opt’ indicates with a value of 1 that the instance was solved to optimality, and 0 otherwise. The columns ‘ K_0 ’, and ‘ K_1 ’ denote the number of vehicles resulting from both lower bounds on fleet size. A value of ‘-’ for the ‘ K_1 ’ column denotes that the linear relaxation could not be solved during the time limit for the column generation algorithm. The column ‘#veh.’ denotes the number of vehicles used by the best solution found at the end of the method. For the cases where the best solution is non optimal, the value in the column ‘#veh.’ is an upper bound on the required fleet size. Solutions might exist with fewer vehicles, but these may be more costly.

Table 5.5 Lower Bound Values on Fleet Size for $\epsilon = 0.2$ and $\delta = 0.0$.

Instance	Multicut		DL-Shaped			
	#veh.	Opt	K_0	K_1	#veh.	Opt
Reggio Emilia	1	1	1	1	1	1
Washington (20)1	3	0	2	2	2	0
Washington (20)2	6	0	5	8	8	1
Chicago (20)1	8	0	6	6	8	1
Chicago (20)2	7	0	3	5	6	1
Washington (30)1	3	0	3	3	3	1
Washington (30)2	3	1	3	3	3	1
Chicago (30)1	13	0	7	8	13	1
Chicago (30)2	5	0	4	4	4	0
Washington (40)1	7	0	4	4	4	0
Washington (40)2	12	0	7	10	10	0
Chicago (40)1	10	0	4	4	7	0
Chicago (40)2	8	0	3	3	6	0
Washington (50)1	12	0	7	8	8	0
Washington (50)2	17	0	6	11	13	0
Chicago (50)	13	0	6	6	8	0
Washington (66)	4	0	3	3	3	0
Chicago (66)	22	0	10	10	13	0
Washington(80)1	6	0	3	3	4	0
Washington(80)2	33	0	11	-	30	0
Washington(90)	36	0	12	-	33	0
Washington(100)	14	0	7	-	7	0
Total	243	2			194	7

The total number of optimal solutions, presented in the last row of the Table, is the smallest compared to any other ϵ , δ configuration, making it the most challenging to solve. However, thanks to K_0 , K_1 , the DL-Shaped algorithm achieves significantly more optimal solutions than Multicut. Moreover, for many instances, the values of K_0 , K_1 are close to the number of vehicles used by the best solution found. Table 5.5 also indicates that further research is needed in the SBRP for cases where the lower bounds of the fleet size deviate significantly from the best solution found.

5.7.5 Results for the new instances

This Section presents the results for the new set of instances and compares them to our implementation of the Multicut algorithm of [6]. All of these instances were solved with the pair $\epsilon = \delta = 0.2$, and the results are presented in tables 5.6 and 5.9. Additionally we also analyze the effectiveness of the valid inequalities and vehicle lower bounds in tables 5.8 and 5.9. To implement the Multicut algorithm, we followed Section 4.3 in the authors' work.

Tables 5.6 and 5.7 present the comparison between our implementation of the Multicut algorithm and the variant of our DL-shaped method with P&S cuts as follows. Column ' n ' denotes the number of stations while the column ' $\#veh.$ ' denotes the average number of vehicles used by optimal solutions or the best solutions found. Column 'Opt' gives the number of optimal solutions while column 'Gap%' give the percentage gap of the non-optimal instances. Column 'Time' gives the average time to either the optimal solution or the best one found within the time limit of 3600 seconds. For the results with the Multicut algorithm the columns 'Benders Feas.' and 'Benders Opt' denote the average number of Benders feasibility and optimality cuts. For the results with the DL-shaped the columns 'Inf.Set', 'S-cuts' and 'P-Cuts' denote the average number of cuts of type of equations (5.38), (5.35) and (5.33). A '-' under the 'Opt' header indicates that no optimal solution was found.

Table 5.6 Results for Multicut algorithm in the new instances.

Multicut							
n	Ins	#veh.	Opt	Gap%	Time	Bender Feas.	Benders Opt
30	20	2.1	12	20.1	1413.8	1142.2	451.0
40	20	2.1	5	42.8	2834.4	1969.8	422.0
50	20	1.9	3	51.8	3225.8	1342.3	485.0
60	20	1.5	-	71.1	3600.0	1087.8	340.0
70	20	1.3	-	54.4	3600.0	883.9	162.0
Total/Average	100	1.8	20	48.0	2934.8	1285.2	372.0

Table 5.7 Results for the new instances.

DL-Shaped								
n	Ins	#veh.	Opt	Gap%	Time	Inf. Set	S-Cuts	P-Cuts
30	20	2.6	20	0.0	66.0	100.6	211.8	41.6
40	20	2.5	18	3.3	729.0	279.1	657.2	58.7
50	20	2.4	12	6.8	1759.4	578.6	1330.6	300.4
60	20	2.8	10	8.7	2621.4	632.7	1898.7	377.2
70	20	1.8	2	2.5	3442.2	770.3	1705.5	680.2
Total/Average	100	2.4	62	4.3	1723.6	472.2	1160.8	291.6

The DL-shaped method consistently outperformed Multicut algorithm throughout this new instance set by achieving three times more optimal solutions in, on average, less computation time. Our approach yielded average gaps of approximately 4%, significantly smaller than the Multicut algorithm’s average gaps, which were more than ten times larger. Notably our algorithm was able to solve instances with up to 70 stations while the Multicut can only solve up to one instance with 50 stations.

This superior performance is attributed to our efficient algorithms for handling infeasible set inequalities and S-cuts. Unlike Multicut, which potentially generates multiple Benders feasibility cuts for each infeasible solution, our method efficiently

returns a single cut for the smallest infeasible component found, preventing rapid problem size growth. Moreover, our DL-shaped method’s ability to generate S-cuts for both feasible and infeasible solutions enables more effective bounding of the recourse in comparison to solely bounding it in feasible solutions via Benders cuts.

Next, to analyze the effectiveness of our new valid inequalities and vehicle lower bounds in the new set of 100 instances we present in tables 5.8 and 5.9 the results of a 5-minute run with each of the new valid inequalities as well as the vehicle lower bound K .

Table 5.8 Convergence with and without the S-cuts and Infeasible Sets for the new instances.

Configuration		Optimality Cut: P-cuts						
S-Cut	Inf. Set	Opt	Gap	Time	Inf. Sets	Inf. Paths	S-Cuts	Benders
Yes	Yes	36	11.3	214.1	572.1	0.0	789.8	163.6
No	Yes	34	7.8	218.3	736.1	0.0	0.0	377.6
Yes	No	27	12.2	236.9	0.0	628.4	737.0	131.4
No	No	31	7.9	227.7	0.0	931.0	0.0	318.4

Table 5.9 Effectiveness of the vehicle lower bound K on the new instances.

K	Opt	Gap%	Time
$\max\{K_0, K_1\}$	36	11.3	214.1
1	32	11.7	225.8

Table 5.8 demonstrates the impact of utilizing S-cuts and infeasible sets on convergence. When both are employed, we observe a significant improvement in convergence rates compared to when either one or none are utilized. This is particularly evident in the higher number of optimal solutions achieved, along with reduced average gaps and computation times.

Similarly, Table 5.9 showcases the effectiveness of the vehicle lower bounds. By

incorporating both K_0 and K_1 into the resolution instead of setting $K = 1$, we attain more optimal solutions at lesser average gaps and computation times.

In summary, within this new benchmark set, the utilization of both S-cuts and infeasible sets, along with the incorporation of the vehicle lower bounds, leads to the most favorable outcomes in terms of convergence to optimal solutions, reduced average gaps, and computation times.

5.8 Conclusions

In this article, we have presented a new formulation, valid cuts and lower bounds for the stochastic bicycle repositioning problem. We implemented the disaggregated integer L-shaped method to solve the problem exactly. The idea of this method is to disaggregate the recourse into components and bound each of these components in a dynamic fashion using optimality cuts and lower bounding functionals. A novel aspect of our implementation is that we successfully included a mixture of more than one type of optimality cut. This was the case when we included both path cuts and Benders cuts to produce so-called Hybrid cuts. Additionally, a key aspect of the method is that it develops and includes (also in a dynamic fashion) lower bounding functionals to accelerate convergence to the optimal solution. We build our lower bounding functionals based on novel recourse lower bounds that we propose.

In terms of our findings, we achieve state-of-the-art results for solving a benchmark set, and this leads us to propose a challenging set of instances. Among the factors that contribute to our success are the new dynamic programming formulation for the recourse, the infeasible set inequalities and the lower bounds on fleet size. On this latter topic, future research should focus on deriving tighter bounds for the fleet size.

CHAPTER 6 ARTICLE 3: Service Level Requirements for Real-Life-Sized Bicycle Sharing Systems

Authors: Lucas Parada, Jean-François Côté, and Michel Gendreau.

Submitted on December 18, 2024 in Transportation Science

Abstract: This paper presents a two-step approach to managing Service Level Requirements (SLR) in real-life Bicycle Sharing Systems (BSS). SLR is a general concept that broadly describes how to effectively manage a BSS to improve user satisfaction while minimizing system operation costs. The two steps involve two proposed problems: first, the Target-Level Problem computes target bicycle quantities for stations, maximizing trip satisfaction. Second, the Bicycle Rebalancing Problem designs vehicle routes to adjust bicycle quantities. SLR literature includes several variants of these problems, but to our knowledge, very few exact approaches such as the one we propose can successfully handle real-life BSS, which comprise thousands of stations, tens of thousands of bicycles, and nearly one hundred thousand daily trip demands. We gather real-life data from large-size systems for our computational experiments, and build artificial data from simulations of Poisson processes based on nine major real-world BSS in Boston, Chicago, Madrid, Mexico City, Montréal, New York, San Francisco, Toronto, and Washington DC. From a managerial perspective, our numerical results provide the decision-makers of BSS with several insights related to bicycle and station usage throughout the network of stations. Our contributions include novel methodological enhancements, problem formulations, and an instance generator for BSS data.

Keywords: Service Level requirements, target-level problem, bicycle repositioning problem, Benders decomposition.

6.1 Introduction

Bicycle sharing systems (BSS) are integral to urban transportation, intertwining with broader transit networks to enhance overall effectiveness in urban mobility [74]. Similar to larger urban transit frameworks, BSS present a fertile ground for various combinatorial optimization problems [38, 39], with a common goal of optimizing system efficiency by maximizing bike availability while minimizing operational costs and user inconvenience. System efficiency is defined by Service Level Requirements (SLR) [40], with a key objective of strategically redistributing bicycles among stations to meet fluctuating daily demand. To achieve success, managers must tackle critical questions. They need to determine the number of bicycles and their technology (electric, non-electric, or hybrid), as well as the optimal station locations and capacities. Additionally, they must establish effective pricing policies and assess their impact on demand patterns, determine the number and capacity of vehicles needed for redistribution, and develop efficient redistribution schedules and strategies. Our work addresses the question of calculating station capacity, required bicycle quantities, vehicle capacity, fleet size, redistribution schedules, and strategy.

This paper proposes a novel two-step approach to strategically redistribute bicycles within SLR. In the first step, we introduce a problem to efficiently compute target quantities of bicycles, such that these amounts maximize the number of future uncertain trips satisfied. The second step focuses on designing vehicle routes to rebalance station quantities from initial values to the target quantities. By breaking down this goal within SLR into two more minor problems, our work demonstrates the ability to manage large-scale BSS efficiently.

The first problem that we propose is the Target-Level Problem (TLB), for which we present several two-stage stochastic programs with recourse, a technique recognized for its effectiveness in managing combinatorial optimization problems under uncertainty [75]. In the TLB, a decision-maker allocates bicycles across multiple stations before the day begins, anticipating random trip demands with varying start and end times that are given in several samples. The objective is to determine quantities

(target levels) that maximize the satisfaction of incoming trips. The proposed two-stage formulation involves integer decisions regarding the initial allocation of bikes across the station set. The second stage adapts to revealed daily demand across a finite number of scenarios, optimizing bicycle flows to enhance trip satisfaction. This formulation is similar to a sample average approximation (SAA) problem [76]. We recognize that modern-day solvers are unable to solve the problem, even for small-sized samples.

To solve the TLB, we employ the Benders decomposition algorithm [56], a recognized exact method when applied to stochastic integer programs. This method is integrated into a branch-and-bound framework, where an iterative process involves formulating and solving a master problem alongside a corresponding sub-problem. This process generates optimality cuts dynamically using a specific formulation of the scenario sub-problem, structured as a min-cost flow problem. While min-cost flow formulations as sub-problems in BSS have been utilized in the literature [77], to our knowledge, they have not been successfully integrated into an exact approach like ours before. Finally, we assess the quality of our TLB solution by evaluating it on larger sample sizes using standard SAA methodologies [78].

The second problem, the Bicycle Rebalancing Problem (BRP), is a variant of the one-commodity pickup and delivery vehicle routing problems, drawing inspiration from the stochastic bicycle repositioning problem outlined by [6]. In the BRP, a decision-maker devises vehicle routes to minimize fleet travel distance and reduce missed bicycle pickups and deliveries around the target levels computed in the TLB. Throughout the resolution of the BRP, numerous infeasible solutions may arise. We contribute to the existing literature by proposing methodological enhancements in heuristics to ensure feasibility during solution optimization. Furthermore, we propose a restocking policy inspired by the vehicle routing with stochastic demands restocking policy that has already been amply studied in the literature (see [79]).

We conduct numerical computations for the TLB and BRP utilizing a newly curated benchmark instance set constructed from real-world historical data from major BSS. This real-world data draws from a common source extensively used in BSS litera-

ture: the General Bikeshare Feed Specification (GBFS, [80]). This source is a set of standards maintained by a developer community, which numerous BSS adhere to. Interested readers can find examples of studies using the same source to build instances for mathematical programs in [50], [53], and [6]. In our study, we model real-world data points according to Poisson processes and devise an instance generator capable of incorporating data from any BSS adhering to these standards. As a result, our benchmark instance set represents, to the best of our knowledge, some of the largest instances for a mathematical program found in BSS literature. This comprehensive set facilitates a rigorous evaluation of the efficiency of the TLB formulations, providing valuable insights into their performance and practical applicability within BSS environments. One notable finding is that certain BSS can fulfill most trip demands with considerably fewer bicycles in inventory. In specific instances, the BSS required as little as 50% of the available bicycles. This finding is particularly significant considering real-world BSS often maintain tens of thousands of bicycles in their inventory. Such insight could lead to substantial cost savings for system operators.

This paper makes several significant contributions to the field. Firstly, we introduce a novel combinatorial optimization problem prevalent in major BSS. Secondly, we present several two-stage mathematical programs with recourse formulations. Thirdly, we propose a Benders decomposition algorithm alongside the station and scenario-based computation of bicycles’ lower and upper bounds. We show that our implementation of the decomposition algorithm can handle real-life-sized instances. Fourthly, we propose a bicycle rebalancing problem alongside methodological enhancements to meet the previously computed bicycle target levels efficiently. Additionally, the paper introduces an instance generator capable of effectively handling data from any BSS conforming to the GBFS standards.

The remainder of this paper is organized as follows. Sections 6.2, 6.3 and 6.4 present the problem definitions, formulations and solution methods for SLR, TLB and BRP respectively. Section 6.5 outlines the processes of gathering and generating data. Section 6.6 presents the numerical results for our implementations, and lastly, Section

7 presents the conclusions.

6.2 Service Level Requirements

In this section, we lay out the problem definition for our SLR maximization problem. Our model is based on a real-life-sized BSS, but we have abstracted some features to propose our framework. Below, we present our assumptions:

1. The trips between each pair of stations follow a Poisson process.
2. The trip demands are processed on a first-come, first-served basis.
3. The BSS is always online, with variable trip demand rates that can be approximated at different times.
4. The quantities of bicycles at the stations do not affect the trip demands.

While these assumptions may not fully capture the complexities of real-life BSS operations, we believe they do not significantly impact our SLR maximization framework. The first and second assumptions build on previous work by [50], specifying that a trip is the fundamental demand unit in the BSS, and multiple trip demands are allowed. A trip is when a single user picks up and returns a bicycle, possibly at the same station at a later time. For this to happen, a bicycle must be available for pick-up and a dock for delivery. The third assumption is that trip demands occur around the clock at different rates, which can be estimated for time intervals, such as hourly. Considering real-life trips shown in our exploratory experiments, we propose that station bicycle quantities be decided by 22:00 at night, before the next day's demand surge starts at 8:00 am. We define the rebalancing process as occurring between determining station bicycle quantities and the next day's demands. Without loss of generality, we assume that the demands between 22:00 and 8:00 are negligible.

Assumption four is the most restrictive as it deals with lost demand, but it allows us to build a model that can approximate the number of satisfied trips. In the context of

BSS, lost demand is the failure of the system to provide a bicycle or a dock to satisfy a trip and the subsequent likelihood that users will choose alternate transportation modes. The latter might produce, for example, a cascade effect of users searching for bicycles or docks at neighboring stations or simply turning to alternate urban transportation modes, thus temporarily or permanently modifying station demands. In a first-come-first-served system such as BSS, a suitable model for such situations should consider the probability of users waiting for a bicycle when none are available, going to a nearby station to find one, or using a different mode of transportation altogether. Such a model is beyond the scope of our study. Nevertheless, we refer the interested reader to [81] for examples of the complexities and intricacies of lost demand cost function modeling (in a different context than BSS).

With the enforcement of assumption four, we can compute an approximation of satisfied trips by proposing a model that accepts or rejects a trip based on the presence of bicycles or docks.

6.2.1 Problem Definition

Before the start of each operational day, we must determine the ideal number of bicycles (target levels) at each station in the network to accommodate as many future trips as possible. The network set N consists of n stations and a depot indexed 0. Each station has a capacity represented by h_i bicycles and an initial quantity of bicycles, denoted as h_{i0} . The trips occur according to a Poisson process within a time period T , beginning at the start of the operational day and comprising discrete time intervals $\{0, 1, \dots, t_{max}\}$. The trips can be sampled based on the rate of trips between stations i and j at time t , represented as $\lambda_{ij}^t > 0$. Each sample of trips, denoted as Ω , comprises a set of equiprobable scenarios.

After determining the target levels, we design routes for a fleet of K homogeneous vehicles, each with a capacity of Q , to rebalance stations from initial quantities h_{i0} to target levels. This rebalancing occurs overnight, before the beginning of T .

The problem has two specific objectives: (1) to calculate bicycle target levels to

maximize the number of satisfied future trips, and (2) to design vehicle routes for rebalancing the stations. To achieve these objectives, it is important to consider how many bicycles are needed, whether station capacities can be optimized, and how many drivers are required to rebalance the stations.

We divide the latter problem definition into two problems, one for each of the two objectives of the decision-maker. The problem of computing the bicycle target levels is the Target-Level Problem (TLB), and the problem of designing the rebalancing routes is the Bicycle Rebalancing Problem (BRP).

6.3 The Target-Level Problem

This section presents the Target-Level Problem (TLB). A formulation for a single scenario of the TLB is presented in Section 6.3.1, and the complete problem formulation is presented in Section 6.3.2. Section 6.3.3 presents the exact decomposition method to solve the problem. Five different models are presented in Sections 6.3.4–6.3.6, and a generalized problem to optimize station capacities is presented in Section 6.3.7.

6.3.1 A Scenario Formulation for the TLB

In the TLB, each scenario ω can be formulated as a min-cost flow problem on a directed graph $G^\omega(V^\omega, A^\omega)$. The set of nodes V^ω comprises one node for each station and time, including one node for each station at time $t = 0$ to denote station target levels, and a source and sink node to denote bicycles moving from and to the depot. The demands of the source and sink nodes are Q_{tot} and $-Q_{\text{tot}}$ while all other nodes have zero demands. The set of arcs A^ω comprises arcs that connect each node in the network, where the capacity of each arc is the station capacity h_i and the cost is 0. We denote these arcs as holding arcs. Additionally, there is one arc for each trip with a cost of -1, giving a reward of 1 unit and a capacity of 1. We accordingly denote these arcs as trip arcs.

The objective of the min-cost flow is to minimize the total cost, effectively maximizing

the total reward by optimizing the flow of bicycles through the network, accounting for station capacities and the rewards associated with trips. The set of nodes V^ω can be significantly reduced by only considering nodes where events happen, where an event is either an incoming trip, an outgoing trip, or both. Figure 6.1 shows an example of this reduction with three stations, $t_{\max} = 4$, and six trips denoted by green arcs. The target levels at each of the three stations are the outgoing flows from the source denoted by y_i , $i = \{1, 2, 3\}$.

6.3.2 A Complete Formulation for the TLB

The complete formulation of the TLB for a given sample Ω is based on the min-cost flow formulation from Section 6.3.1. Let $e = (pickup_e, delivery_e, start_e, end_e)$ denote a trip, where $pickup_e$, $delivery_e$ are the pickup, delivery stations and $start_e$, end_e are the pickup, delivery times. Let E^ω be the set of trips in scenario ω . Also, let $O^\omega(i, t)$ be the set of trips originating from station i having a pickup time of t , and $U^\omega(i, t)$ be the set of trips that terminate at station i having a delivery time of t .

Let y_i be an integer variable to represent the target level of bicycles of station i , $f_{it\omega}$ a continuous variable for the number of bicycles at station i in scenario ω at the end of time t , and let v_e be a continuous variable to represent whether trip e can be satisfied or not. The following mixed-integer program computes the target levels:

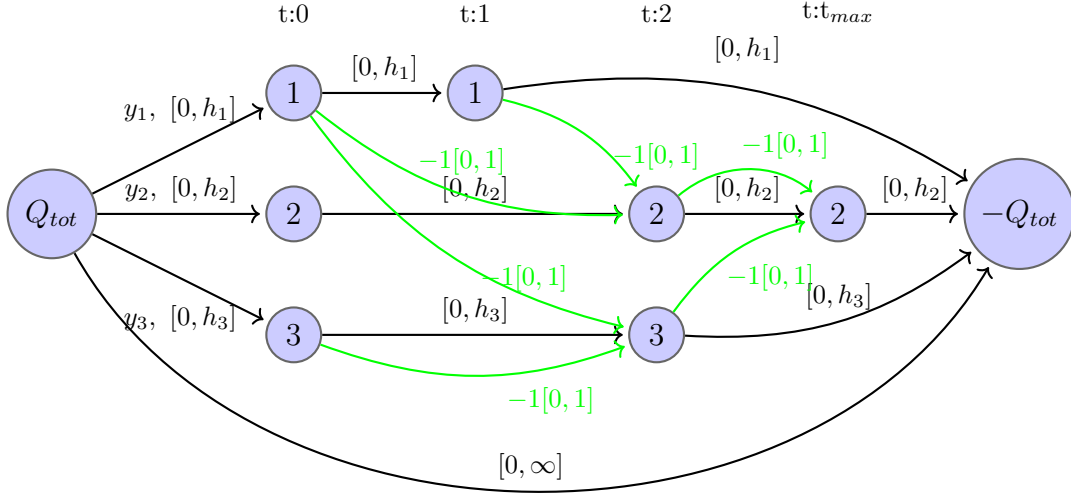


Figure 6.1 Min-Cost Flow Formulation $G^\omega = (V^\omega, A^\omega)$ for one scenario of the TLB with $n = 3$ trips and $t_{\max} = 4$.

$$(TLB) \max \sum_{\omega \in \Omega} \mathbb{P}(\omega) \sum_{e \in E_\omega} v_e \quad (6.1)$$

$$\text{s.t.} \quad \sum_{i \in N} y_i \leq Q_{\text{tot}}, \quad (6.2)$$

$$f_{i0\omega} = y_i, \quad i \in N, \omega \in \Omega \quad (6.3)$$

$$f_{it\omega} = f_{it-1\omega} + \sum_{r \in U^\omega(i,t)} v_e - \sum_{e \in O^\omega(i,t)} v_e, \quad i \in N, t \in T \setminus \{0\}, \omega \in \Omega \quad (6.4)$$

$$0 \leq y_i \leq h_i \text{ and integer}, \quad i \in N, \quad (6.5)$$

$$0 \leq f_{it\omega} \leq h_i, \quad i \in N, t \in T, \omega \in \Omega, \quad (6.6)$$

$$0 \leq v_e \leq 1, \quad e \in E^\omega, \omega \in \Omega. \quad (6.7)$$

The objective in (6.1) maximizes the expected satisfied trips in the sample. Constraint (6.2) ensures that the quantity of bicycles used by the targets does not exceed

the available number of bicycles. Constraints (6.3)–(6.4) impose the min-cost flow problem of Section 6.3.1 for all scenarios in the sample Ω . In these constraints, and following the guidelines of the net demand process, simultaneous pickup and delivery at a station do not alter the number of bicycles at that station. Constraints (6.5)–(6.7) are definitions of variables.

Furthermore, an upper bound can be derived for the TLB using the assumption that in each min-cost flow problem, we have the flexibility to select which trips to fulfill. Denote c^ω , $u^{*\omega}$ as the cost and optimal flow vector of the arcs in A^ω . The formula for calculating the upper bound of the TLB, denoted UB, is as follows:

$$UB = \sum_{\omega \in \Omega} \mathbb{P}(\omega) c^\omega u^{*\omega}. \quad (6.8)$$

Lastly, the problem formulated in (6.1)–(6.7) can be referred to as a sample average approximation (SAA) problem [76]. In traditional approaches to such problems, the quality of a solution is assessed using much larger samples than Ω . It is also commonly understood that the quality improves with increasing sample sizes in the resolution [78, 82]. However, unlike other SAA discrete optimization problems, solving real-life sized samples for the TLB requires significant computational resources, as we show in our results in Section 6.6. Nonetheless, we evaluate the quality of our solutions using large samples.

6.3.3 The Benders Decomposition Algorithm

To solve the TLB we utilize the Benders decomposition algorithm, a well-known method for solving two-stage stochastic programming problems with integer variables in the first stage and continuous variables in the second stage [56]. The algorithm introduces a continuous variable $\theta > 0$ as an underestimator of the expected recourse cost. Then, a master problem is formulated with the first-stage decisions, the recourse underestimator, and a finite set of optimality cuts. These cuts are generated on the fly using separation algorithms whenever a feasible first-stage solution is found, typically with the branch-and-check method of [83], a known variant of the broader

branch-and-cut family of algorithms.

For the TLB, we implement a branch-and-cut approach with Benders cuts [56] as optimality cuts. We begin by formulating it as a two-stage mathematical program with recourse as follows:

$$\begin{aligned} \max \quad & \mathcal{Q}(y) \\ \text{s.t.} \quad & (6.2), (6.5). \end{aligned} \tag{6.9}$$

In this formulation, $\mathcal{Q}(y)$ represents the expected recourse cost function. To calculate it, we first provide a first-stage solution $y^\nu = (y_i^\nu)$, followed by a variant of the min-cost flow problem in G^ω denoted as $G_1^\omega = (V_1^\omega, A_1^\omega)$. To create G_1^ω , begin by adjusting the demand at the source to $Q_{tot} - \sum_{i \in N} y_i^\nu$. Second, adjust the source-outgoing arcs to the $|n|$ station nodes by imposing lower and upper capacities of $[y_i^\nu, y_i^\nu]$. The remainder of the nodes in V^ω and arcs in A^ω stay the same. Next, denote $c_1^\omega, u_1^{*\omega}$ as the cost and optimal flow vector of the arcs in A_1^ω . Then, the expected recourse cost of first stage solution y^ν is given by the following formula:

$$\mathcal{Q}(y^\nu) = \sum_{\omega \in \Omega} \mathbb{P}(\omega) c_1^\omega u_1^{*\omega}. \tag{6.10}$$

With a procedure to compute the expected recourse cost, the master program of the Benders decomposition method follows:

$$(M1) \max \theta \tag{6.11}$$

$$\text{s.t.} \quad (6.2), (6.5),$$

$$\theta \leq UB, \tag{6.12}$$

$$\text{Optimality cuts,}$$

$$\theta \geq 0. \tag{6.13}$$

We refer to the model in equations (6.2),(6.5),(6.11)–(6.13), as TLB Model 1 or simply M1.

Separation of Benders Cuts for the TLB

To separate Benders cuts, we construct and solve the min-cost flow problem associated with G_1^ω for all ω . The following notation is introduced. For $i \in N$, let $\gamma_i^{*\omega}$ and $\epsilon_i^{*\omega}$ represent the optimal values of the dual variables associated with the upper and lower flow capacity constraints of the source-outgoing arcs to the station nodes in G_1^ω . Furthermore, for $(i, j) \in A_1^\omega \setminus A_1^\omega(N)$, let $\gamma_{ij}^{*\omega}$ be the optimal values of the dual variables associated with the upper capacity constraints of all arcs in $A_1^\omega \setminus A_1^\omega(N)$, where $A_1^\omega(N)$ is the set of source-outgoing arcs to the stations. Additionally, denote the set of the trip arcs of A_1^ω as $A_{1E\omega}^\omega$. Lastly, for $i \in V_1^\omega$, let $\pi^{*\omega}(i)$ represent the optimal value of the dual variables associated with the node constraints, with $\pi^{*\omega}(source)$ and $\pi^{*\omega}(sink)$ being the values associated with the source and sink nodes. The Benders optimality cut follows:

$$\begin{aligned} \theta \leq \sum_{\omega \in \Omega} \mathbb{P}(\omega) & \left(Q_{\text{tot}}(\pi(source) - \pi(sink)) + \sum_{i \in N} y_i(\gamma_i^{*\omega} - \epsilon_i^{*\omega}) \right. \\ & \left. + \sum_{(g,j) \in A_{1E\omega}^\omega} \gamma_{gj}^{*\omega} + \sum_{(g,j) \in A_1^\omega \setminus (A_{1E\omega}^\omega \cup A_1^\omega(N))} h_i \gamma_{gj}^{*\omega} \right) \end{aligned} \quad (6.14)$$

To improve the quality of the generated cut, and taking advantage of the min-cost flow structure of our sub-problem, we implemented the Pareto-optimal cut separation method introduced by [84]. This method is an adaptation of the approach outlined in [85], which involves constructing and solving the dual of the sub-problem for solution $\bar{y}$ and adding an extra constraint in the form of a Benders cut, also at solution $\bar{y}$. The customization of [84] has the following advantage over [85]: instead of solving the dual, this approach allows for the formulation and solution of a slightly different min-cost flow problem with increased capacity for all arcs. This increased

arc capacity is determined by a parameter derived from the dual variable associated with the Benders cut constraint. In our case, we set this parameter as Q_{tot} . As a result, the arc capacities are significantly higher, which makes the sub-problem slower to solve. Consequently, achieving a higher quality cut may be impractical, as it would demand repetitive solutions to a min-cost flow sub-problem with significantly higher capacity.

6.3.4 A First Variant Formulation of the TLB

Solving M1 can be inefficient due to the poor convergence of the classical optimality cuts, especially when the second-stage information is not considered in the first-stage problem [86,87]. To address this, we propose a second model, M2, for the TLB. This model utilizes a set of lower bounds for bicycles at stations and in scenarios, assuming that each station has a bicycle quantity interval that maximizes the number of satisfied trips. Additionally, the model introduces variables to measure the deviation from the bounds. These variables not only convey information from the second-stage model but also allow us to accommodate scenarios where the exact number of bicycles may not strictly adhere to the predefined bounds.

Let lb_i^ω and ub_i^ω represent the lower and upper bounds of the number of bicycles required at station i in scenario ω . Additionally, let lb^ω and ub^ω denote the lower and upper bounds on the total number of bicycles required in scenario ω . Furthermore, let l^ω and m^ω be the variables that measure the violation of the total scenario bicycles, and let l_i^ω and m_i^ω represent the variables that measure the violation of the station scenario bicycles. The master problem for the decomposition method follows:

$$(M2) \max \theta \tag{6.15}$$

$$\text{s.t. } (6.2), (6.5), (6.13), (6.14),$$

$$\sum_{i \in N} y_i + l^\omega \geq lb^\omega, \quad \omega \in \Omega, \tag{6.16}$$

$$\sum_{i \in N} y_i - m^\omega \leq ub^\omega, \quad \omega \in \Omega, \tag{6.17}$$

$$\theta \leq UB - \sum_{\omega \in \Omega} \mathbb{P}(\omega)(l^\omega + m^\omega), \tag{6.18}$$

$$y_i + l_i^\omega \geq lb_i^\omega, \quad \omega \in \Omega, i \in N, \tag{6.19}$$

$$y_i - m_i^\omega \leq ub_i^\omega, \quad \omega \in \Omega, i \in N, \tag{6.20}$$

$$\theta \leq UB - \sum_{\omega \in \Omega} \mathbb{P}(\omega) \sum_{i \in N} (l_i^\omega + m_i^\omega), \tag{6.21}$$

$$l^\omega, m^\omega \geq 0, \quad \omega \in \Omega, \tag{6.22}$$

$$l_i^\omega, m_i^\omega \geq 0, \quad \omega \in \Omega, i \in N. \tag{6.23}$$

Constraints (6.16)–(6.17) bind the total bicycles used in a scenario, and constraints (6.19)–(6.20) bind the bicycles used by the individual stations in different scenarios. Constraints (6.18) and (6.21) bind the expected recourse cost to the violation variables l^ω , m^ω , l_i^ω and m_i^ω . Constraints (6.22)–(6.23) are definitions of variables.

We perform the following calculations to get the station and scenario lower and upper bounds. For scenario ω , obtaining lower and upper bounds lb^ω and ub^ω involves modifying G^ω as follows. Modify the cost of the source-to-sink arc to 1 and solve the min-cost flow problem. The summation of optimal flow values from the source to stations gives the lower bound on required bicycles for scenario ω (lb^ω). To compute the upper bound (ub^ω), repeat the procedure with the modified arc cost as -1 instead of 1.

For station i within scenario ω , determining lower and upper bounds lb_i^ω and ub_i^ω involves similar modifications to G^ω . Adjust the cost of the source-to-station arc i to 1 and solve the flow problem. The optimal flow value in this arc gives the bicycle

lower bound for scenario ω for station i (lb_i^ω). To compute the upper bound (ub_i^ω), repeat the procedure with the modified arc cost as -1 instead of 1 .

We need to solve a total of $2|\Omega|(N+1)$ min-cost flow problems in order to compute the bounds.

Finally, it's important to consider that when solving M2, the optimal solution of M1 may be cut off by the assumption that there is a linear relationship between the number of missing bikes and the number of unsatisfied trips. In practice, one bike might be used for several trips.

6.3.5 A Second Variant Formulation of the TLB

Solving M2 can be computationally expensive due to the requirement to compute all station and scenario bounds lb_i^ω , ub_i^ω , lb^ω and ub^ω . We thus propose a third model, M3, that requires the computation of fewer bounds. Let l_i , m_i be the variables that measure how the station bicycles violate the given upper and lower bounds. The formulation of M3 follows:

$$(M3) \quad \max \theta \tag{6.24}$$

$$\text{s.t.} \quad (6.2), (6.5), (6.13), (6.14),$$

$$y_i + l_i \geq \max_{\omega \in \Omega} lb_i^\omega, \quad i \in N, \tag{6.25}$$

$$y_i - m_i \leq \min_{\omega \in \Omega} ub_i^\omega, \quad i \in N, \tag{6.26}$$

$$\theta \leq UB - \sum_{i \in N} (l_i + m_i), \tag{6.27}$$

$$l_i, m_i \geq 0. \tag{6.28}$$

Constraints (6.25)–(6.26) bind the total bicycles used to a single upper and lower bound for each station. Constraint (6.27) binds the expected recourse cost to the violation variables l_i and m_i . Constraints (6.28) are definitions of variables.

The maximal station lower bound, $\max_{\omega \in \Omega} lb_i^\omega$, and minimal station upper bound, $\min_{\omega \in \Omega} ub_i^\omega$, are determined through an heuristic process. This heuristic takes a collection of bicycle flows for all scenarios as input and seeks the smallest and largest flows. In practice, the collection of flows can be obtained when solving the min-cost flow in G^ω , $\forall \omega \in \Omega$. If the smallest and largest flow values of station i are 0 or h_i , respectively, the procedure to compute station-scenario bounds of M2 is invoked for i . Otherwise, the maximal and minimal bounds are set to the maximum and minimum flows across all scenarios.

6.3.6 A Third Variant Formulation of the TLB

A decision-maker might intuitively aim to keep target levels near half capacity to balance both high pickup and delivery demands. The following mathematical program implements this logic:

$$\begin{aligned} \min_y \quad & \sum_{i \in N} (y_i - 0.5h_i)^2 \\ \text{s. to} \quad & (6.2), (6.5). \end{aligned} \tag{6.29}$$

The program in (6.29), (6.2), (6.5) is quadratic and hence cannot be embedded in the decomposition method that we propose for models M1 – M3. We thus propose the following dynamic programming (DP) algorithm to compute target levels under the logic of halving the station capacities. We denote this problem as M4 and the cost function that calculates the optimal target levels as $H_i^{M4}(z)$, for all stations i and all available bicycle quantities z . The recursion to compute the optimal target levels follows:

$$H_i^{M4}(z) = \begin{cases} 0, & \text{if } i = n + 1, \\ \min_{0 \leq y_i \leq \min\{z, 0.5h_i\}} \left\{ (y_i - 0.5h_i)^2 + H_{i+1}^{M4}(z - y_i) \right\}, & \text{if } i = 1, \dots, n. \end{cases} \quad (6.30)$$

The minimum cost is obtained by calculating $H_1^{M4}(Q_{tot})$ and the time complexity is $O(n \sum_{i=1}^n h_i)$.

6.3.7 Optimizing the Station Capacity

This section presents an approach for determining optimal station capacities. We suppose that we are given a non-negative integer budget B representing the maximum number of docks that can be added to the network of stations. Let h_i^+ be integer variables denoting the capacity increase of station i . Then, the formulation of the Station Capacity Problem (SCP), which we denote as Model 5, or M5, follows:

$$(M5) \quad \max \theta \quad (6.31)$$

$$\text{s.t.} \quad (6.2), (6.13),$$

$$\sum_{i \in N} h_i^+ \leq B, \quad (6.32)$$

$$0 \leq y_i - h_i^+ \leq h_i, \quad (6.33)$$

$$h_i^+, y_i \geq 0, \text{ and integer}, \quad (6.34)$$

Optimality Cuts.

Equation (6.31) aims to maximize the expected number of satisfied trips. Inequality (6.32) represents the budget constraint, while inequality (6.33) is a generalization of inequality (6.5). Inequalities (6.34) introduce the new variable definitions.

Notably, the optimality cuts for this generalized problem differ from those of the

TLB due to the altered second-stage formulation resulting from potentially varied station capacities. To separate these optimality cuts, we construct the graph $G_2^\omega = (V_2^\omega, A_2^\omega)$ for each ω , following a similar procedure to the construction of G_1^ω . Let $h_i^{\nu+}$ denote the values of the increase variables, respectively, at a given solution ν . From G_1^ω , adjust the capacity of the holding arcs associated with station i from $[0, h_i]$ to $[0, h_i + h_i^{\nu+}]$. The construction of the remaining arcs and all nodes in G_2^ω follows the same principles as those in G_1^ω .

To separate Benders cuts for M5, we use a procedure similar to models M1–M3 and introduce analogous notation to the dual of G_1^ω . Let $\zeta_i^{*\omega}$ and $\iota_i^{*\omega}$ represent the optimal dual variables for upper and lower capacity constraints of flow for the source-outgoing to station arcs. $\iota_{ij}^{*\omega}$ represents the optimal dual variables for flow capacity constraints not connecting to the source. Lastly, $\eta^{*\omega}(i)$ represents the optimal dual variables associated with the node constraints, with $\eta^{*\omega}(source)$ and $\eta^{*\omega}(sink)$ as the values for the source and sink nodes. The Benders cut for M5 follows:

$$\begin{aligned} \theta \leq \sum_{\omega \in \Omega} \mathbb{P}(\omega) & \left(Q_{\text{tot}}(\eta^{*\omega}(source) - \eta^{*\omega}(sink)) + \sum_{i \in N} y_i(\iota_i^{*\omega} - \zeta_i^{*\omega}) \right. \\ & \left. + \sum_{(g,j) \in A_{2E\omega}^\omega} \iota_{gj}^{*\omega} + \sum_{(g,j) \in A_2^\omega \setminus (A_{2E\omega}^\omega \cup A_2^\omega(N))} (h_i + h_i^+) \iota_{gj}^{*\omega} \right), \end{aligned} \quad (6.35)$$

where $A_{2E\omega}^\omega$ represents the set of trips arcs and $A_2^\omega(N)$ the set of source-outgoing arcs to the station nodes.

We incorporate inequality (6.35) into the model presented in (6.31)–(6.34).

Lastly, an upper bound UB^{SCP} for M5 can be computed by performing the following modifications to $G^\omega(V^\omega, A^\omega)$. Modify the capacity of each holding arc from h_i to ∞ or, in practice, to Q_{tot} . The rest of the graph remains the same, and UB^{SCP} is given by averaging the scenario optimal solutions.

We also propose two variants for M5, following the same logic as the variants of

M1 with the station and scenario upper and lower bounds. Specifically, we add more constraints to the station capacity constraints in equations (6.32)–(6.34) by incorporating the inequalities from M2, resulting in a model we refer to as Model 6, or M6. Alternatively, by combining the inequalities from M3 into the station capacity constraints in equations (6.32)–(6.34), we derive a model referred to as Model 7, or M7.

The following modifications are made to G_1^ω to formulate the min-cost flow graphs of the station and scenario lower and upper bounds in M6 and M7. The capacity of all holding arcs is modified to infinity, or in practice, to Q_{tot} , while the remaining arcs and nodes remain unchanged.

6.4 The Bicycle Rebalancing Problem

This section presents the Bicycle Rebalancing problem (BRP). Section 6.4.1 presents the problem formulation, while the policies to compute the cost of a route are given in sections 6.4.3 and 6.4.4. Lastly, Section 6.4.6 presents an heuristic solution method.

Throughout this section, we follow the notation of the TLB given in Section 6.3, while introducing new notation as needed.

6.4.1 Problem Definition

The BRP is defined as the problem of designing vehicle routes that start and end at the depot, to efficiently redistribute bicycle quantities among a set of stations. This redistribution is managed by a fleet of homogenous vehicles, each with a capacity denoted as Q . The network of stations is represented by a graph $G = (V, A)$, where $V = N \cup \{0\}$ denotes the set of stations N and the depot labeled as 0, and A denotes the arcs that connect each station with a given distance d_{ij} , and $(i, j) \in A$. The goal is to redistribute the bicycles at each station exactly once, all while ensuring that the total distance traveled on any given route remains below a predefined distance D .

Denoting y_i^* as the target level of station i , computed by solving the TLB in Section 6.3, the station's demand is represented by $q_i \in \mathbb{Z}$, with $q_i = y_i^* - h_{i0}$, where h_{i0} is the initial bicycle occupation. A rebalanced station has its demand either fully or partially satisfied within the interval $[q_i - (h_i - h_{i0}), q_i + h_{i0}]$. Here, $(h_i - h_{i0})$, h_{i0} , represent maximum missed deliveries and pickups respectively. To scale the missed bicycle quantities to the distance traveled by the vehicle fleet, an integer parameter $\delta \geq 1$ is defined with units of [kilometers/missed bicycles].

Without loss of generality, and to ensure that all nodes in V can feasibly be visited by one vehicle, we suppose that an instance of the BRP must satisfy the conditions $|q_i| \leq h_i$ and $|q_i| \leq Q$ for all $i \in V$.

6.4.2 Cost of a Route

In the BRP, the cost of a route comprises the total distance traveled and the cost incurred due to deviations from target-level bicycle quantities at each station. We propose two distinct cost computation functions based on policies designed to enhance the service level requirements of the bicycle-sharing system. The first policy forces the driver to visit all stations on the route sequentially. In contrast, the second policy allows performing restocking trips to the depot depending on the residual capacity of the vehicle. These two policies are denoted respectively as the continue-to-next (CN) policy and the restocking trips (RT) policy, and they are presented in Sections 6.4.3 and 6.4.4.

6.4.3 The Continue-to-Next Policy

The cost computation for a route $r = \{0, i_1, \dots, i_t, 0\}$ is divided into $t + 1$ stages. Each stage k corresponds to a station visit in the sequence r , with each station visit having a corresponding request denoted by q_k .

We introduce the variables $x_k \in [0, Q]$ to represent the vehicle load when arriving at station k , where $x_{t+1} = 0$. Let $\bar{w}_k^+ = h_{i0}$ and $\bar{w}_k^- = h_i - h_{i0}$ denote the maximum missed pickup and delivery quantities at stage k , respectively.

Define $H_k^C(x_k)$ as the cost function of the CN policy. Then, the following DP algorithm computes $H_k^C(x_k)$ as follows:

$$H_k^C(x_k) = \begin{cases} 0, & k = t + 1, \\ \min_{\substack{-\bar{w}_k^- \leq w_k \leq \bar{w}_k^+ \\ 0 \leq x_k + w_k \leq Q - q_k}} \left\{ \delta |w_k| + H_{k+1}^C(q_k + x_k + w_k) \right\}, & k = 1, \dots, t. \end{cases} \quad (6.36)$$

The cost of r is given by $\sum_{k=0}^t d_{i_k i_{k+1}} + d_{i_t 0} + \min_{0 \leq x_0 \leq Q} \{H_1^C(x_0)\}$.

Solving this DP algorithm has a pseudo-polynomial time complexity of $O(nQw_{max})$ in the worst case, where $w_{max} = \max_{1 \leq k \leq t} \{\bar{w}_k^-, \bar{w}_k^+\}$.

6.4.4 The Restocking Trips Policy

The cost of performing a restocking trip on route r is calculated as follows. At each stage k , the vehicle must decide whether to continue to the next station or to make a restocking trip to the depot to drop off or pick up bicycles. The state variable d_k represents the vehicle's remaining autonomy and is used to determine whether to continue or restock. The formula to calculate d_k is $d_k = d_{k-1} - r_k$. Here, $r_k = c_{k-1,0} + c_{0k} - c_{k-1,k}$ represents the additional distance incurred during the restocking trip between stations $k-1$ and k , and $d_0 = D - \sum_{(i,j) \in r} c_{ij}$. Let $H_k^R(x_k, d_k)$ denote the cost of performing a restocking trip at stage k , given by the following DP algorithm:

$$H_k^R(x_k, d_k) = \begin{cases} 0, & k = t + 1, \\ \min \left\{ r_k + \min_{\substack{-\bar{w}_k^- \leq w_k \leq \bar{w}_k^+ \\ 0 \leq x_k + w_k \leq Q - q_k}} \{ \delta |w_k| \} + \min_{0 \leq y \leq Q} \{ H_{k+1}^R(y, d_k - r_k) \} \right\}, \\ \min_{\substack{-\bar{w}_k^- \leq w_k \leq \bar{w}_k^+ \\ 0 \leq x_k + w_k \leq Q - q_k}} \left(\delta |w_k| + H_{k+1}^R(q_k + x_k + w_k, d_k) \right), & k = 1, \dots, t. \end{cases} \quad (6.37)$$

The two terms in this DP sum the cost of missing w_k bicycles with the cost of moving to the next station on the route with minimal residual bicycle quantities y in the vehicle and available distance to restock $d_k - r_k$.

With $H_k^R(x_k, d_k)$ defined, the cost is given by $\sum_{k=0}^t d_{i_k i_{k+1}} + d_{i_t 0} + \min_{0 \leq x_0 \leq Q} H_1^R(x_0, D)$.

6.4.5 Feasibility of a Route

Under both cost policies described in Section 6.4.2, a route may be infeasible depending on the sequencing of the stations. In this section, we specifically focus on the CN policy to propose conditions that determine whether a given route of stations is feasible.

Let $r = \{0, i_1, \dots, i_t, 0\}$ be a route and S be the set of stations comprising route r . We propose the following two feasibility conditions that need to be simultaneously satisfied for r :

$$\sum_{\substack{i \in S \\ q_i > 0}} q_i \leq Q + \sum_{i \in S} \bar{w}_i^+ + \sum_{\substack{i \in S \\ q_i < 0}} |q_i|, \quad (6.38)$$

$$\sum_{\substack{i \in S \\ q_i < 0}} |q_i| \leq Q + \sum_{i \in S} \bar{w}_i^- + \sum_{\substack{i \in S \\ q_i > 0}} q_i, \quad (6.39)$$

Equations (6.38), (6.39) are conditions for the pickup and delivery request in r . A

route becomes infeasible if it violates either one of the conditions, as shown in [88].

6.4.6 Solution Method

To solve the BRP, we implement the Adaptive Large Neighborhood Search (ALNS) metaheuristic as proposed by [68]. Our approach includes sequential insertion, related removal, and random removal operators.

To accelerate the computations of the DP in the CN and RT policies, we propose a lower bound L_{HC} on $H_k^C(x_k)$ for a given set $S \subseteq N$. This lower bound is calculated by summing the requests in S and comparing the total with Q as follows:

$$L_{HC} = \delta \max\{0, |\sum_{i \in S} q_i| - Q\}. \quad (6.40)$$

Using this lower bound, the sequential insertion operator can determine if a route satisfies the condition $H_1^C(x_0) = 0$ by sequentially applying equation (6.40) to the stations on the route until a non-zero cost is found. This method allows for faster sorting of candidate insertion moves without explicitly computing $H_1^C(x_0)$.

For the restocking trips policy, we optimize the number of recursive calls by using a linear relaxation of a knapsack problem at each station k to determine the maximum feasible number of restocking trips, given the available distance d_k . In this problem, the knapsack items represent restocking trips at stations $k + 1, \dots, t$, with their respective distances as weights.

Initial exploratory experiments made with ALNS determined that an effective strategy is to run 50,000 ALNS iterations with the CN policy (using the DP for $H_k^C(x_k)$), then take the best solution from this run and perform 5,000 ALNS iterations with the RT policy (using the DP for $H_k^R(x_k, d_k)$). This approach is applied in our computational experiments presented in section 6.6.2.

6.5 Data Generation

This section describes the sources of real-life BSS data and the algorithms used to generate artificial data for our benchmark TLB and BRP instances. The data for our proposed instances were collected from nine real-life BSS namely: Blue Bikes from Boston, US; Divvy Bikes from Chicago, US; BiciMAD from Madrid, SP; Ecobici from Mexico City, MX; BIXI from Montréal, CA; Citi Bike from New York, US; Bay Wheels from San Francisco, US; Bike Share Toronto from Toronto, CA; and Capital Bike Share from Washington DC, US. Section 6.5.1 details the instance generator we propose, and Section 6.5.2 describes the instances resulting from the generator.

The data for all bike-sharing systems (BSS) comes from two main sources: JSON schemas and a set of historical trip data. The JSON schemas strictly follow the General Bikeshare Feed Specification (GBFS), standardizing the data format for real-time information sharing about BSS. The defined endpoints and data structures provide important up-to-date system details such as station locations, bicycle availability, and status updates. Historical trip data is provided by the BSS operator, city administrators (as in Toronto and Madrid), or both.

The standard data types and formats maintained by GBFS are essential to prevent issues with historical trip data that do not adhere to the guidelines. In Madrid, the historical trip data is missing crucial fields required by the JSON schemas, such as the ‘station_id’ field, which is necessary for identifying starting and ending stations for these historical trips. All parties involved in curating trip data must strictly adhere to the guidelines, and proper data curation contributes significantly to managerial insight; without it, not only may the quality of BSS service be hindered, but also the conclusions obtained by SLR maximization projects may lack correct applicability.

Based on the JSON data files and historical trips, we conducted an initial exploratory analysis and found significant differences in size and usage among the nine BSS. For instance, Citi Bike, Bixi, and Ecobici were the three BSS with the largest number of trips from April to July 2023, each recording more than three million trips. In contrast, the BSS in Washington DC, and Toronto were the smallest, with fewer

than a million trips each during the same period. Notably, Citi Bike (New York), the largest BSS, had more than 10 million trips during those months.

We used data from July 2023, the busiest month for all the BSS mentioned, to ensure comparability in our initial analysis. Figure 6.2 depicts histograms showing the average number of trips in July 2023 for the three largest BSS and Madrid. Madrid is noteworthy as the only non-North American city in our study. It is important to note that as of the current date, the Empresa Municipal de Transportes de Madrid has not yet released the data for July 2023, and the data presented here are from July 2022.

The average number of trips in Figure 6.2 is measured by the hourly count of trips that started between 8:00 and 22:00, divided by the number of days in July 2023 (31 days).

The data depicted in Figure 6.2 clearly illustrates the system usage patterns throughout an average day and reveals the disparities in size among the four locations. It is evident that Montréal, New York, and Madrid experience peak usage between 16:00 and 19:00, signifying a surge in demand during the end-of-workday period. Conversely, Mexico City experiences a more evenly distributed demand throughout the day, with a noticeable decline after 19:00. In terms of size, New York shows significant demand during its low-demand hours from 20:00 to 22:00, exceeding the peak demand in Mexico City at any time of the day and matching many peak hours in Montréal. Specifically, New York experiences demand for more than ten thousand trips between 17:00 and 18:00. Montréal's peak occurs at nearly five thousand trip demands at 17:00. In comparison, Mexico City peaks at less than three thousand trips at various hours throughout the day. At the same time, Madrid has the smallest peak of all the histograms. This difference is partly due to the smaller size of Madrid's BSS, partly to data curators not adhering to the GBFS standards, thereby invalidating many historical trips for our study due to the absence of assigned starting and ending stations.

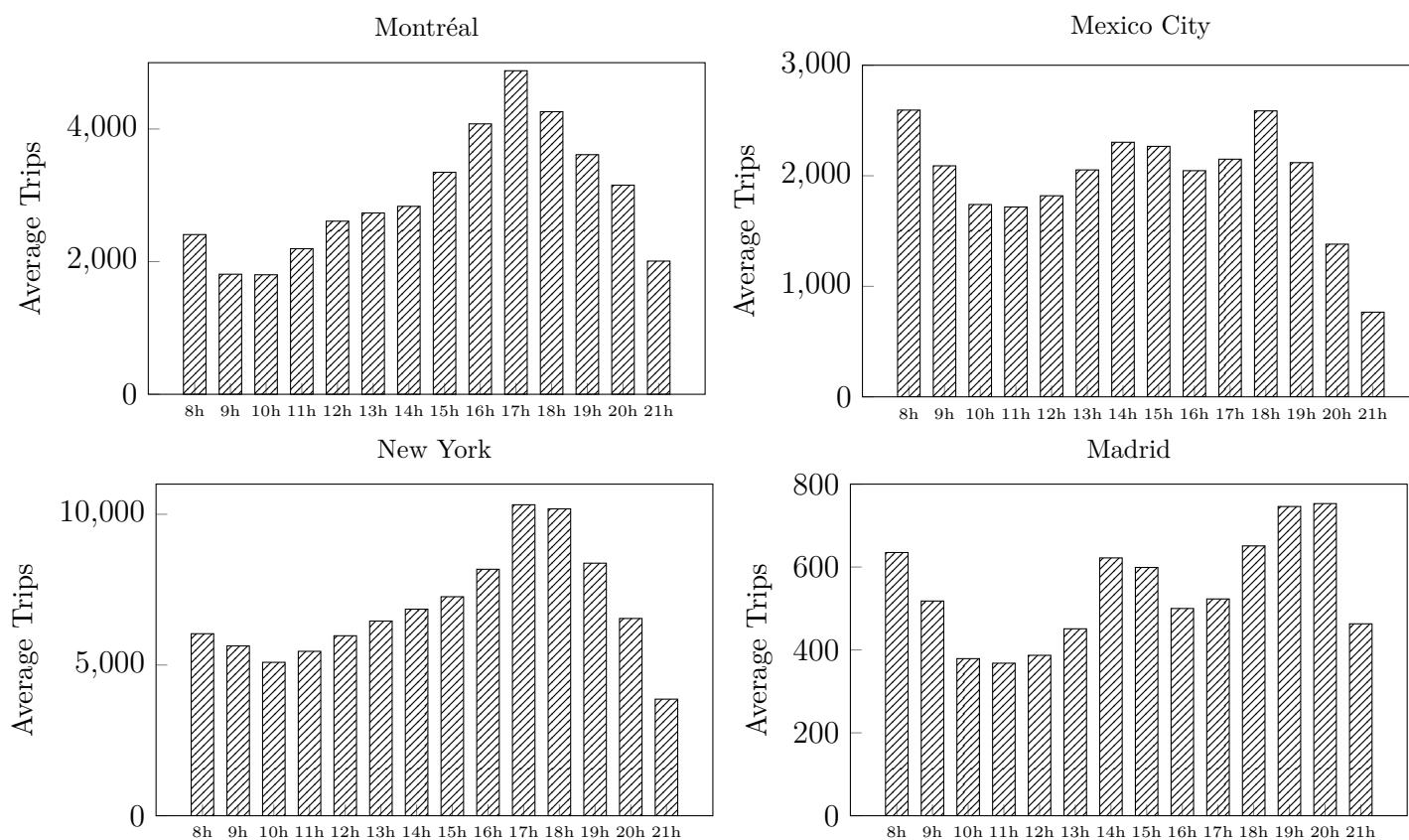


Figure 6.2 Histograms of Average Daily Trips in the [8h,22h] Interval of the Largest BSS in July 2023.

6.5.1 Random Data Generation for the TLB

To construct our instance set, we implemented in C++ an instance generator whose pseudo-code is provided in Algorithm 1. The generator conceptualizes the trips within a BSS as conforming to a Poisson process [89]. The input to the generator is a set of real-world trips from a BSS and a number of scenarios for generating random, artificial trips. Each scenario resembles a simulated operational day of the BSS. The output of the generator is an instance of the TLB.

The Poisson process is performed in steps 5–6 of the generator. Let λ represent the rate of occurrence of a given pickup and delivery station pair at a given hour of the day. These pickup and delivery pairs are denoted as OD pairs in the pseudocode. Under the Poisson process framework, the time until the initial OD pair event follows an exponential distribution with rate λ . Denote T_1 as the time of the first occurrence of the pair and $N(t)$ as the total number of events that have occurred before time t . Then, the formula to compute T_1 follows:

$$T_1 = \min\{t > 0 : N(t) = 1\} = e^{-\lambda t}. \quad (6.41)$$

In practice, we need to compute the time to the first random event by using the cumulative distribution (CDF) of the exponential distribution $F(t)$ as follows:

$$F(t) = 1 - e^{-\lambda t} = u, \quad u \sim Uniform[0, 1], \quad (6.42)$$

$$\implies t = -\frac{\ln(1 - u)}{\lambda}. \quad (6.43)$$

This method enables computation of the time to the first occurrence based on the uniform distribution within the range $[0, 1]$, utilizing the inverse transform sampling technique associated with the exponential distribution. To generate the uniform random numbers, we build on the generators of [90] and [91]. Next, the ‘If’ statement in step 7 verifies that the generated trip does not go beyond the current hour, since

the rate of occurrences changes outside this interval.

To compute a random duration for the newly generated OD pair, we assume it is a random variable following a beta distribution $Beta(\alpha, \beta)$ with given α, β parameters. This is done in steps (11)–(12). This choice is motivated by the versatility of the beta distribution in modeling non-negative random values, which are appropriate for trip durations observed in empirical BSS data. We use the method of moments to compute α, β [89]. Next, to compute the beta probabilities for a given α, β , we implement the algorithms from [92] and [93].

The last step of the algorithm, step 16, updates the time to t_{next} , and then the generator proceeds to loop through scenarios and hours, effectively generating artificial trips for each scenario within specified hour ranges.

We implemented Algorithm 1 with real-life trips selected from the 8:00 to 22:00 interval.

6.5.2 Characterization of the Instances

In this section, we outline the instances generated for the TLB and the BRP. Each instance represents a specific city; hence, when convenient, we refer to it as a ‘city’ for simplicity.

For the TLB, we generate five instances, with 15, 25, 50, 75, and 100 scenarios respectively, for each of the nine cities, totaling 45 instances. Table 6.1 provides an overview of the largest instances (with 100 scenarios), detailing key metrics such as the number of real-life stations (N), total available bicycles (Q_{tot}), and the sizes of upper bound wait-and-see scenario graphs G^ω , represented by the average number of MCF arcs and nodes.

The table reveals significant variation in instance sizes across cities, as measured by the average number of arcs and nodes. In line with the analysis of the real-life data presented in the histograms in Figure 6.2, the instances of New York, Montréal, and Mexico City are the largest.

Algorithm 1: Random Instance Generator

```

1 foreach scenario  $\omega$  do
2   foreach hour  $h$  do
3      $t = 0$ ;
4     foreach OD pair do
5       Compute  $\lambda =$  rate of the the OD pair in  $h$ ;
6       Compute  $t_{next} = t + \frac{-\ln(1-u)}{\lambda}$ ,  $u \sim Uniform[0, 1]$ ;
7       if  $t_{next} > 60$  then
8         break
9       end
10      Generate a trip with start time  $t_{next}$ ;
11      Compute  $\alpha, \beta$ ;
12      Generate the trip duration  $dur$  as a realization from a r.v. following a
         $B(\alpha, \beta)$  distribution;
13      if  $t_{next} + dur \leq End\_of\_day$  then
14        Store the trip in the container for scenario  $\omega$ , with start time  $t_{next}$ 
        and end time  $t_{next} + dur$ ;
15      end
16      Update  $t = t_{next}$ ;
17    end
18  end
19 end

```

Table 6.1 Summary of Instances with $|\Omega| = 100$ for the TLB.

City	N	Q_{tot}	Average Arcs	Average Nodes
Boston	423	4,000	22,198.8	12,085.5
Chicago	1,678	15,242	36,475.3	19,518.4
Madrid	610	7,500	18,639.7	10,437.1
Mexico City	661	6,800	54,876.5	26,436.9
Montréal	800	10,000	79,547.1	36,832.5
New York	2,175	32,000	179,221.2	80,816.4
San Francisco	554	7,000	12,795.1	7,656.3
Toronto	791	9,000	41,809.1	22,341.3
Washington DC	738	6,000	26,976.6	15,184.1

Additionally, we generated a different set of 400 scenarios, denoted as Ω_{400} , for each city. This set is used to compute an unbiased estimator through the expected recourse cost of a given solution in Ω_{400} .

For the BRP instances, we propose one instance for each city, totaling nine instances. In each city, we construct the graph (V, A) where the set of nodes V is constructed from the real-life locations of the same N stations as the TLB instances. The arc distances $(i, j) \in A$ are calculated using the Haversine distance formula between node pairs $i, j \in V$. We gathered JSON files from GBFS to obtain the initial station occupation h_{i0} .

The rebalancing vehicle capacity is set at $Q = 40$, consistent with current real-life BSS capacities [94]. However, a small percentage of nodes have demand levels (q_i) exceeding this threshold ($q_i > 40$), making it impossible to find a feasible solution that visits each node exactly once. Specifically, in New York, Montreal, and Toronto instances, fewer than 10% of stations have $h_i > 40$. Depending on the target levels and initial occupation, the resulting nodes may have demands greater than vehicle capacity. We address this by creating multiple nodes corresponding to real-life stations whose demand surpasses vehicle capacity, such that each resulting node manages a portion of the total demand. This process does not impede our approach as we only need feasible routes for redistributing bicycles in node-assigned stations, rather than optimal routes. Additionally, we remove nodes with $q_i = 0$ from each instance, as they were not significantly used as transshipment nodes.

Table 6.2 summarizes the BRP instances with the following columns: ‘N’ indicates the number of real-life stations, ‘|V|’ shows the number of generated nodes, and ‘ $\sum_{i \in N} q_i$ ’ represents the total demand, reflecting the number of bicycles that need to be delivered to or picked up from the depot.

The table shows the activity of different BSS, with New York having the highest activity. Chicago has relatively low activity, with nearly a third of its stations not requiring rebalancing. Nearly 500 nodes out of 1,677 stations had a q_i -value of 0 and were removed.

Table 6.2 Summary of Instances for the BRP.

City	N	V	$\sum_{i \in N} q_i$
Boston	423	392	100
Chicago	1,677	1,158	49
Madrid	610	586	181
Mexico City	661	626	100
Montréal	800	762	100
New York	2,175	2,111	119
San Francisco	554	528	49
Toronto	791	724	100
Washington DC	738	693	149

6.6 Computational Results

This section presents the results for implementing the decomposition methods for the TLB and the ALNS metaheuristic for the BRP. Both were implemented in C++ with Cplex 22.10 for the decomposition method. Experiments were run on a CPU with Intel E5-2683 v4 @ 2.1 GHz. For the TLB, we implemented the code in parallel using the OpenMP library, and experiments were run using 64 cores. This section has been divided into Sections 6.6.1 and 6.6.2 for the results of the TLB and BRP, respectively.

For the TLB, we use the following formula to compute percentage gaps given an upper bound UB :

$$\%Gap = \frac{UB - LB}{LB} \times 100,$$

where LB denotes the lower bound, which is the value of a given solution to any of our models.

6.6.1 Results for the TLB

We present the results for the TLB models M1–M4 in Section 6.6.1. Additionally, we present the results of changing the number of available bicycles in Section 6.6.1, and the results for the OSC models M5–M7 in Section 6.6.1.

Results for Models M1–M4

The results of solving models M1–M3 are shown in Figure 6.3. This figure displays the average percentage gaps of an unbiased estimator with respect to UB_{400} after solving all three models for the nine cities and all scenarios (15, 50, 75, and 100). The unbiased estimators were calculated as the average cost of a TLB model solution in G^ω , $\forall \omega \in \Omega_{400}$, using the Benders decomposition algorithm with a time limit of 1,800 seconds. UB_{400} is the average optimal cost of G^ω , $\forall \omega \in \Omega$. Each data point on the plotted lines corresponds to the average gap of all nine cities for a given scenario. The figure also displays the total averaged computational times for each model and scenario, which are the sum of the Benders algorithm and the time required to compute bounds in M2 and M3. Finally, for these experiments, we did not use the Pareto-optimal Benders cut of [84], since initial exploratory experiments did not show any significant improvements in solution quality.

Figure 6.3 shows that M2 consistently outperforms M1 and M3, and that M3 improves as the number of scenarios increases while M1 worsens, possibly because fewer first-stage solutions are visited. This suggests that the station and scenario bounds help convergence for M2 and M3, but not for M1. In the plot of computation times, M1 and M3 are displayed as an average, since they are not significantly different from each other. The average times for M2 increase with the complexity of station scenario bounds for larger cities. This information is further detailed in Table 6.4.

The figure also indicates that the best results are achieved with 100 scenarios. Therefore, we utilize instances with 100 scenarios for further analysis in Table 6.3.

Table 6.3 presents a detailed comparison of the Benders decomposition algorithm for models M1–M3 after selecting an appropriate number of scenarios. The columns include ‘Opt’ for optimal solutions (with 1 indicating an optimal solution and 0 non-optimal), ‘%CplexGap’ for the Cplex gap at a 1,800-second limit, ‘Time[s]’ for the time to reach the best or optimal solution, and ‘Cuts’ for the number of Benders optimality cuts added to the model.

Table 6.3 confirms M2’s superiority over M1 and M3, as it is the only one of the

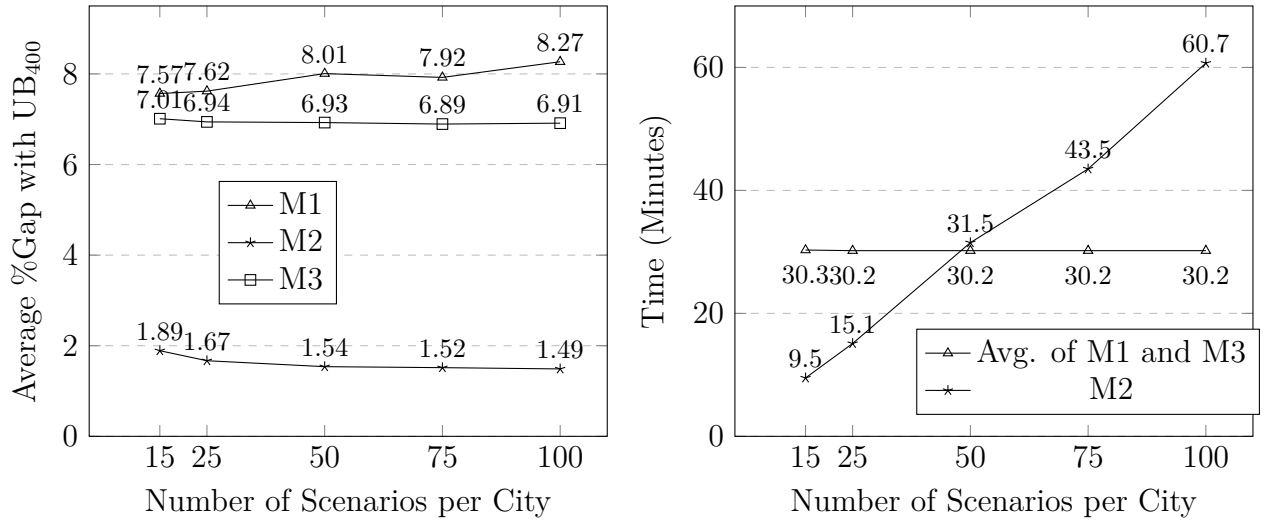


Figure 6.3 Comparison of results for Models M1–M3 with different scenarios: (a) Average % Gap with UB_{400} and (b) Average time taken

three models to solve all cities to optimality. This table highlights the advantage of using the complete station and scenario bounds in M2, shown by the large number of cuts added to models M1 and M3. These cuts suggest that M1 and M3 are unlikely to converge to optimality regardless of the time limit. Initial experiments with an extended time limit of four hours instead of half an hour corroborate this. M2, in contrast, converged for all cities with zero optimality cuts needed, and a closer inspection of our results showed that convergence was attained at the root node in all cases.

We conclude the analysis of the TLB models in Table 6.4 by comparing the performance of models M1–M4 in achieving the best gap in the 400 scenario set of Ω_{400} . From a BSS managerial standpoint, this table is the most relevant in this section as it answers the question of which model should be used to determine the bicycle target levels that maximize average trip satisfaction. The table is structured as follows. For each city, the column ‘ UB_{400} ’ denotes the average optimal cost of G^ω , $\omega \in \Omega_{400}$. Columns ‘M1’ to ‘M4’ under the header ‘%Gap’ denote the gap of models M1–M4 with respect to UB_{400} . The gaps for models M1–M3 are calculated using the best

Table 6.3 Performance of Models M1–M3 in Cities with 100 Scenarios.

City	M1				M2				M3			
	Opt	%CplexGap	Time[s]	Cuts	Opt	%CplexGap	Time[s]	Cuts	Opt	%CplexGap	Time[s]	Cuts
Boston	0	8.5	1,800.0	4,898	1	0.0	1.5	0	0	8.2	1,800.0	4,945
Chicago	0	9.4	1,800.0	3,026	1	0.0	4.6	0	0	8.4	1,800.0	2,912
Madrid	0	4.5	1,800.0	9,607	1	0.0	2.0	0	0	5.6	1,800.0	9,197
Mexico City	0	6.0	1,800.0	1,095	1	0.0	5.6	0	0	6.1	1,800.0	1,095
Montréal	0	7.0	1,800.0	466	1	0.0	9.3	0	0	6.0	1,800.0	460
New York	0	9.0	1,800.0	118	1	0.0	35.3	0	0	8.6	1,800.0	123
San Francisco	0	12.2	1,800.0	9,983	1	0.0	1.8	0	0	11.8	1,800.0	10,285
Toronto	0	7.7	1,800.0	2,411	1	0.0	3.3	0	0	8.6	1,800.0	2,387
Washington DC	0	9.9	1,800.0	4,441	1	0.0	2.2	0	0	11.1	1,800.0	4,404

or optimal solution at the end of a 1,800-second run of the Benders decomposition algorithm and 100 scenarios. The gaps for M4 are computed using the solution of the DP algorithm in equation (6.30). Lastly, the columns ‘M2’ and ‘M3’ under the header ‘Bounds Time[s]’ denote the time in seconds required to compute the station and scenario bounds. Model M4 computation time is negligible.

The results demonstrate that model M2 is superior to all other models despite the increased time needed to compute the station scenario bounds. However, this additional computation time is significant only for New York. The time is less than an hour for Montréal, and less than 20 minutes for other cities. On the other hand, model M4 offers a valuable and highly efficient alternative to the decomposition method, for all cities except Chicago. When Chicago is excluded, the average gap for M4 decreases from 12.4 to 4.2, making it the second best model after M2. The poor performance of M4 in Chicago is likely due to the underutilization of bicycles in this city, which is further analyzed in Section 6.6.1.

Results with Fewer Available Bicycles.

This section presents the result of varying the available bicycles Q_{tot} . Initial exploratory experiments showed that for many cities, Q_{tot} was significantly greater than the sum of the target levels; hence, computing the minimal Q_{tot} that maximizes trip satisfaction provides a relevant managerial insight.

We modify constraint (6.2) to the following form:

Table 6.4 %Gaps with UB_{400} of Models M1–M4 Using 100 Scenarios

City	UB_{400}	%Gap				Bounds Time[s]	
		M1	M2	M3	M4	M2	M3
Boston	9,256.2	8.5	2.1	6.7	4.7	92.03	1.1
Chicago	15,026.3	9.4	2.3	7.2	78.3	290.7	1.9
Madrid	7,558.5	4.5	0.3	7.0	0.9	78.4	1.0
Mexico City	20,480.4	6.0	1.4	3.7	4.2	1,072.1	12.8
Montréal	40,475.9	7.0	1.3	3.4	4.6	2,928.7	33.0
New York	95,536.0	9.1	1.0	4.6	6.2	27,591.1	95.7
San Francisco	4,582.4	12.3	1.1	14.5	2.8	24.1	0.2
Toronto	18,077.8	7.7	1.8	5.9	5.6	464.1	4.2
Washington DC	10,913.3	9.9	2.1	9.1	4.6	165.0	1.2
	Avg.	8.3	1.5	6.9	12.4		

$$\sum_{i \in N} y_i \leq \phi Q_{tot}, \quad (6.44)$$

where $\phi \in [0, 1]$ is an input parameter. Then, we solve the TLB model with a given ϕ to obtain a solution that uses fewer bicycles.

Table 6.5 explores how changing Q_{tot} affects the average satisfied trips. The columns under ‘% of Available Bicycles’ show the average satisfied trips for each bicycle percentage, corresponding to the ϕ parameter in equation (6.44). Then, for each ϕ value of 5%, 10%, ..., 90%, 95%, M2 was solved with 100 scenarios, and its solution was utilized to determine the average number of satisfied trips in the set Ω_{400} , expressed as a percentage of UB_{400} . We use M2 to provide a solution because this model consistently yielded the best values among all models in Table 6.4.

The table shows that all cities have more bicycles than necessary to fulfill the current number of trips, on average. With just 50% of the available bicycles, 97.0% of average trips can be accommodated, compared to 98.4% of average satisfied trips with Q_{tot} . The satisfaction level remains consistent at 98.2%, even with only 70% of the available bicycles being used. Chicago is still more noteworthy, with just 10%

Table 6.5 Average Percentage of Satisfied Trips with Fewer Available Bicycles

City	% of Available Bicycles (ϕ)									
	10	20	30	40	50	60	70	80	90	100
Boston	58.7	81.5	89.2	93.1	95.5	96.9	97.6	97.9	97.9	97.9
Chicago	87.1	95.5	97.6	97.7	97.7	97.7	97.7	97.7	97.7	97.7
Madrid	83.9	93.2	97.1	98.8	99.5	99.7	99.7	99.7	99.7	99.7
Mexico City	55.7	80.7	88.8	92.9	95.1	96.7	97.8	98.3	98.6	98.6
Montréal	66.1	84.0	90.7	93.9	96.0	97.5	98.4	98.7	98.7	98.7
New York	74.1	88.2	92.7	95.1	96.8	97.9	98.5	98.9	99.0	99.0
San Francisco	74.8	89.8	95.4	97.9	98.7	98.9	98.9	98.9	98.9	98.9
Toronto	69.8	87.1	92.5	95.7	97.3	98.0	98.2	98.2	98.2	98.2
Washington DC	58.7	80.9	88.3	92.5	95.2	96.6	97.5	97.8	97.9	97.9
Average	69.8	86.9	92.7	95.5	97.0	97.8	98.2	98.4	98.4	98.4

of available bicycles resulting in an 87.1% average trip satisfaction rate, the highest among the nine cities.

Even the three largest BSS in New York, Montréal, and Mexico City can accommodate about 96% of average trips with just 50% of the available bicycles, as shown in Figure 6.4. The figure includes more data points than Table 6.5, providing a better understanding of the relationship between satisfied trips and available bicycles. For example, in 2023, New York had an average of over 30,000 bicycles in a given month, according to the Department of Transportation and Citi Bike data. The figure suggests that BSS operators could reduce costs by using as few as 15,000 bicycles.

Results for the SCP

This section presents the results for SCP models M5–M7 in Tables 6.6 and 6.7.

Table 6.6 shows the changes in average satisfied trips for models M5–M7 compared to model M2, for all nine cities and one hundred scenarios using budget values ranging from 0% to 20% of $\sum_{i \in N} h_i$.

The data clearly indicates that M6 represents a significant improvement over M5

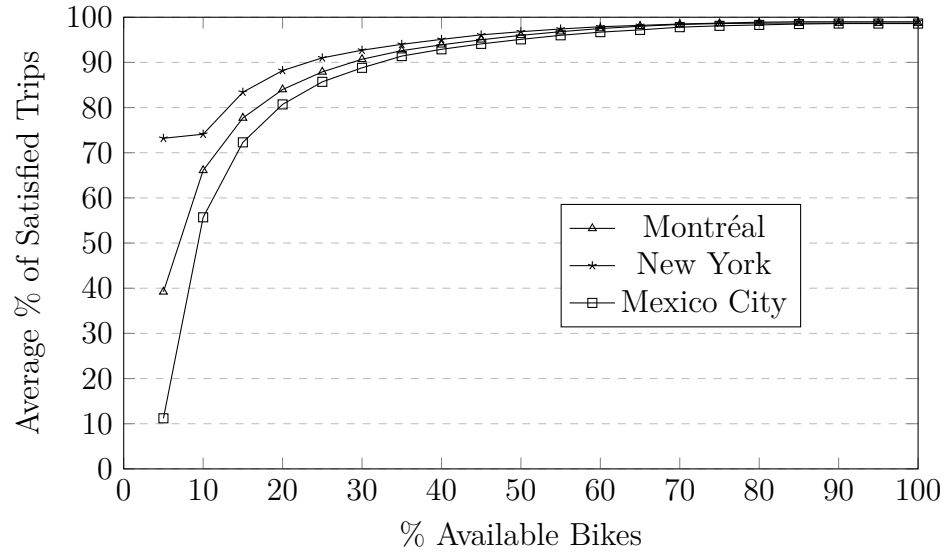


Figure 6.4 Montréal, New York, and Mexico City average percentage of satisfied trips with fewer available bicycles

Table 6.6 Increase or Decrease in Average Satisfied Trips with Respect to M2 for Models M5, M6, and M7.

	Budget as a % of $\sum_{i \in N} h_i(B)$								
	0	1	2	3	4	5	10	15	20
M5	-6.9	-6.2	-6.3	-6.1	-6.1	-6.5	-6.6	-6.8	-8.0
M6	0.0	0.4	0.5	0.7	0.7	0.8	1.1	1.3	1.6
M7	-5.5	-6.1	-6.5	-6.0	-6.1	-6.5	-6.2	-6.4	-6.9

and M7. However, it also shows that there is no substantial increase in average satisfied trips as the budget increases. M6 consistently shows an increase in average trip satisfaction, by as much as 1.6%. It is important to note that a 20% increase in the installed capacity of the BSS comes at a considerable, possibly unrealistic, cost to the system operators.

The superiority of M6 is attributed to the station scenario bounds, which have a similar effect in M2. In contrast, M5 and M7, which lack these bounds, significantly underperform compared to M6. As the budget increases, the number of potential solutions for SCP becomes larger because the variables h_i^+ can have higher values. This makes it more challenging for M5 and M7 to converge, as a larger model must be solved without additional information from the second stage. Our detailed results show that as the budget increases, the average number of satisfied trips in M5 and M7 worsens after a brief improvement with budget values of 1–5%. When we ran models M5 and M7 with up to a 50% budget, their performance was even worse for the same reason.

From Table 6.6, we have selected a budget equal to 5% of $\sum_{i \in N} h_i$ to further analyze the performance of models M5–M7 in Table 6.7. We chose this budget based on M6’s highest increase in average satisfied trips compared to M2. A 5% increase in installed capacity is also more realistic than a 20% increase, while still retaining more satisfied trips than M2. For each city, column ‘UB₄₀₀’ denotes the upper bound from Table 6.4. We include this column to compare the potential increase in satisfied trips allowed by models M5–M7 compared to models M1–M4. Column ‘UB₄₀₀^{SCP}’ denotes the upper bound computed in the set Ω_{400} according to Section 6.3.7. Next, columns ‘M5’ to ‘M7’, under the header ‘%Gap’, denote the gap with relation to ‘UB₄₀₀^{SCP}’ of the best or optimal solution found by the Benders decomposition algorithm at the end of an 1,800-second run. Lastly columns ‘M6’, ‘M7’, under the header ‘Bounds Time[s]’ denote the time in seconds to compute the station and scenario bounds for models M6, M7.

The results in Table 6.7 indicate that model M6 has significantly better gaps compared to models M5 and M7, consistent with Table 6.6. Second, the time required

Table 6.7 Results for Models M5–M7 Using a Budget of 5% of $\sum_{i \in N} h_i$, and 100 Scenarios

City	UB ₄₀₀	UB ₄₀₀ ^{SCP}	%Gap			Bounds Time[s]	
			M5	M6	M7	M6	M7
Boston	9,256.2	9,704.8	12.3	5.2	11.7	114.6	1.7
Chicago	15,026.3	15,260.2	11.0	3.3	11.2	587.9	7.2
Madrid	7,558.5	7,558.7	5.6	0.1	6.2	83.9	1.2
Mexico City	26,502.56	27,795.98	10.3	4.7	10.5	1,184.6	14.9
Montréal	40,475.9	41,892.2	9.5	4.7	9.4	3,214.3	37.4
New York	95,536.0	96,220.6	10.3	1.7	10.2	27,795.9	234.1
San Francisco	4,582.4	4,585.7	14.3	0.8	13.6	30.1	0.5
Toronto	18,077.8	18,654.9	11.0	4.6	11.0	553.1	5.6
Washington DC	10,913.3	11,044.6	12.0	2.8	12.0	191.7	2.1
Avg.	25,325.4	25,857.5	10.7	3.1	10.6		

to compute the station and scenario bounds in model M6 is similar to that of model M2, a notable fact given that the arc capacities are considerably higher in model M6. Comparing UB₄₀₀^{SCP} with UB₄₀₀ consistently shows that UB₄₀₀^{SCP} is not significantly higher, offering an additional perspective on the increased average trip satisfaction for M6 relative to M2 shown in Table 6.6 on .

6.6.2 Results for the BRP

This section presents BRP results in Tables 6.8, 6.9, and 6.10, obtained through the ALNS procedure outlined in Section 6.4.6.

Table 6.8 compares the CN and RT policies for a δ value of one. Columns include ‘Cost’ for the cost of the solution, which is the sum of the traveled distance ‘Dist’ plus the ‘Policy’ cost CN or RT, and ‘Drv.’ for the number of drivers utilized. The CN policy determines route cost as missed bicycles multiplied by δ . Conversely, for the RT policy, the cost encompasses missed bicycles multiplied by δ plus the restocking trip distances. ‘Time[s]’ denotes the computational time in seconds.

Table 6.8 shows that the RT policy yields better solutions, evident from smaller

Table 6.8 BRP Results with $\delta = 1$.

City	Continue-to-Next Policy					Restocking Trips Policy				
	Cost	Dist.	Policy	Drv.	Time	Cost	Dist.	Policy	Drv.	Time
Boston	312.4	312.4	0	5	274.1	312.4	312.4	0	5	973.4
Chicago	903.6	903.6	0	11	970.4	903.5	903.5	0	11	4,576.2
Madrid	1,656.0	1,634.0	22	16	524.7	1,577.3	1,559.1	18.2	15	8,287.8
Mexico City	560.2	560.2	0	8	721.4	377.9	377.9	0	13	9,193.7
Montréal	1,027.9	1,027.9	0	11	668.2	1,027.5	1,027.5	0	11	11,282.0
New York	4,489.1	4,471.1	18	44	1,588.0	4,469.3	4,459.3	10.0	44	11,705.2
San Francisco	1,265.4	1,239.4	26	11	314.3	1,225.8	1,216.6	9.2	11	2,791.1
Toronto	848.0	848.0	0	9	610.0	845.5	841.2	4.3	9	11,975.1
Washington DC	789.9	789.9	0	11	563.3	789.9	789.9	0	11	2,019.0
Average	1,316.9	1,309.6	7	14	692.7	1,281.0	1,276.4	4.6	14	6,978.2

average cost values compared to the CN policy. However, the RT policy requires significantly more computation time, especially in cities like Montréal and Mexico City. The effectiveness of the CN policy comes from the efficient feasibility checks described in Section 6.4.6. These checks make it easier to quickly verify if a route is feasible and, combined with the calculation of L_{HC} , effectively reduce the amount of calculations needed. Conversely, the RT policy's complexity in the DP for $H_k^R(x_k, d_k)$ leads to longer computational times. Another factor that affects RT's computational times will be discussed in the analysis of Table 6.10, later in this section.

In reference to the distances traveled, Table 6.8 illustrates that, on average, each vehicle undertakes a route of just under one hundred kilometers. This calculation is derived by dividing the mean distance under the RT policy of 1,276.4 by the 14 vehicles.

Regarding the number of drivers, Table 6.8 shows similar requirements for both policies, with slight variations in cities like Montréal and New York, as further analyzed in Table 6.10.

Table 6.9 provides an overview of the solution costs after the ALNS runs, contrasting the CN and RT cost policies across $\delta = \{1, 5, 10, 15, 20, 25\}$. The cost of each solution

is represented as UB^{ALNS} in each cell of the table.

The table illustrates that across all δ values, the ALNS run with the RT policy consistently improves solutions over the CN policy. The most substantial improvements occur at $\delta = 20$, where all nine RT solutions outperform the CN solution, and nearly all improvements fall within the 10–15% range. Moreover, our combined approach remains robust to various δ values, maintaining stable UB^{ALNS} values around 1,500 despite the increased cost of missed bicycles.

Next, Table 6.10 delves into the number of drivers used across different δ values, revealing correlations with total demand $\sum q_i$ and $|V|$, as detailed in Table 6.2. Table 6.10 also includes a column for the total absolute differences $|\bar{y}_i - h_{i0}|$, representing the number of bicycles rebalanced in the station network, and the total pickups and deliveries, representing the sum of stations with positive and negative requests.

Cities with higher total requests or absolute differences, as well as more nodes, require more drivers. For instance, while Boston, Montréal, Mexico City, and Toronto share similar total demand figures, Montréal stands out due to its more extensive node network that requires more drivers. As previously discussed, an interesting pattern emerges in San Francisco, where a significant number of drivers are required for relatively few nodes. On average, 14 drivers are needed, likely due to the substantial distances within the city. Moreover, as δ increases in all cities, indicating a higher penalty for missed bicycle requests, there is no noticeable increase in driver counts,

Table 6.9 BRP Results for All δ Values

UB^{ALNS}	δ											
	1		5		10		15		20		25	
City	CN	RT	CN	RT	CN	RT	CN	RT	CN	RT	CN	RT
Boston	312.4	312.4	321.1	321.1	334.3	334.3	324.8	324.8	328.0	322.9	333.8	324.0
Chicago	903.6	903.5	984.6	971.8	1,001.4	937.1	995.8	995.8	990.8	918.5	979.2	918.7
Madrid	1,656.0	1,577.3	1,860.5	1,535.5	1,880.9	1,762.7	2,030.6	1,685.8	2,076.7	1,793.4	1,820.5	1,623.7
Mexico City	560.2	377.9	727.3	594.6	764.8	661.9	751.4	633.7	731.5	669.0	726.6	689.3
Montréal	1,027.9	1,027.5	1,128.7	984.6	1,141.5	1,139.3	1,241.4	1,007.4	1,180.3	991.9	1,165.6	918.4
New York	4,489.1	4,469.3	5,488.4	4,648.6	5,762.2	4,531.7	5,810.8	4,536.8	5,763.4	4,688.2	5,456.3	5,451.0
San Francisco	1,265.4	1,225.8	1,378.6	1,032.6	1,490.7	1,182.5	1,418.5	1,181.0	1,421.1	1,197.9	1,506.5	1,035.5
Toronto	848.0	845.5	917.6	914.1	947.7	822.4	945.2	945.2	917.5	823.7	984.8	931.1
Washington DC	789.9	789.9	844.0	844.0	802.4	796.7	864.7	864.7	860.4	841.3	833.7	833.7
Average	1,316.9	1,281.0	1,516.8	1,316.3	1,569.6	1,352.1	1,598.1	1,352.8	1,585.5	1,360.8	1,534.1	1,413.9

meaning restocking trips efficiently avoid costly penalties.

The connection between the driver count values and the presence of non-zero demand nodes sheds light on the computational times observed in the RT policy of Table 6.8. Montréal, for example, has 762 non-zero demand nodes as shown in Table 6.2, and an average of 11 drivers shown in Table 6.10, suggesting approximately 70 stations per route. Such long routes significantly impact the RT policy’s computational times, as the DP’s complexity increases. Furthermore, a closer examination of our results reveals that many of the routes in our solutions have hundreds of stations. Routes with over 100 nodes remain a significant challenge within vehicle routing literature, with few applications encountering such extensive routes in practice. Remarkably, our findings demonstrate their prevalence within the context of SLR across nearly all our instances.

The inverse relationship between driver counts and the number of non-zero demand nodes underscores the necessity for refined station network design. Considering the costliness of large fleets, this insight has significant implications for system operators. Future research avenues could explore techniques such as the Poisson process outlined in Section 6.5 to model trip demands, analyze inflow and outflow patterns, and optimize station use to reduce total request sums and vehicle requirements. Chicago, for instance, presents numerous underutilized stations, suggesting that substantial cost savings would be possible through relocation or capacity adjustments.

6.7 Conclusion

We have proposed a two-step approach to efficiently address the strategic redistribution of bicycles within Service Level Requirements (SLR) maximization for bicycle sharing systems (BSS) by breaking down the SLR into the Target-Level Problem (TLB) and the Bicycle Rebalancing Problem (BRP). Through numerical computations using real-world data, we provide valuable insights into the efficiency and practical applicability of the TLB and BRP formulations within BSS environments, introducing novel optimization problems and solution methodologies to enhance sys-

Table 6.10 BRP Driver Counts for All δ Values.

City	Pickups	Deliveries	$\sum \bar{y}_i - h_{i0} $	δ					
				1	5	10	15	20	25
Boston	1,129	-1,029	2,158	5	5	5	6	5	5
Chicago	2,123	-2,074	4,197	11	10	10	12	9	10
Madrid	3,824	-3,643	7,467	15	17	19	25	17	19
Mexico City	2,706	-2,606	5,312	13	9	7	10	6	7
Montréal	3,181	-3,081	6,262	11	11	11	11	11	12
New York	11,709	-11,590	23,299	44	48	47	51	49	46
San Francisco	1,988	-1,939	3,927	11	13	14	15	15	15
Toronto	2,830	-2,730	5,560	9	9	9	11	12	9
Washington DC	2,005	-1,856	3,861	11	12	10	11	11	12

tem performance and operational efficiency.

The results of our computational experiments demonstrate the superiority of Model 2 to other TLB models, consistently achieving optimal solutions across various scenarios and cities within the given time constraints. Additionally, our analysis indicates that significant reductions in the total number of bicycles (Q_{tot}) can be achieved without substantial impacts on the average number of satisfied trips, highlighting potential cost-saving opportunities. Moreover, for the optimal station capacity models, Model 6, analogous to Model 2 in the TLB analysis, is the most effective, particularly with a budget of 5% of system capacity for modifying the station. This suggests that targeted investments in capacity modifications can optimize system performance. Overall, the integration of station and scenario bounds significantly enhances the convergence of Benders decomposition, affirming the strategic advantage of these models in large-scale urban transportation planning.

Regarding the BRP, results show that the restocking trips (RT) policy is more effective in optimizing solutions than the continue-to-next (CN) policy, as demonstrated across a range of values for missed bicycle cost. Although the RT policy requires more computational time, its ability to generate superior solutions, particularly in cities with complex station networks, signifies its practical utility. Moreover, analysis re-

veals that driver counts are correlated to total demand, highlighting the importance of optimizing station networks to mitigate the need for larger fleets. These findings offer valuable insights for system operators seeking cost-effective strategies in managing BSS, and emphasize the potential of further research to refine network design and demand modeling to enhance operational efficiency and reduce resource requirements.

Integrating the TLB and BRP approaches offers a unified framework that combines capacity modification strategies from the TLB with tailored rebalancing policies from the BRP to achieve efficient resource allocation and improved service quality. This integrated approach has the potential to empower operators to make informed decisions aligned with the broader goals of SLR, driving their success and sustainability in urban environments.

6.8 Acknowledgments

Financial support for this work was provided by the Canadian Natural Sciences and Engineering Research Council (NSERC) under grant 2021-04037. We thank Digital Research Alliance of Canada for providing high-performance computing facilities.

CHAPTER 7 CONCLUSION

This chapter provides a comprehensive summary of our research’s key findings, outlines the proposed solutions’ limitations, and suggests directions for future research.

7.1 Synthèse des travaux / Summary of Works

Our work presents innovative approaches to solving various stochastic optimization problems, focusing on three main themes within the larger context of algorithms for the stochastic bicycle repositioning problem.

In our initial contribution, we presented a new technique for resolving a category of stochastic integer programs with first-stage solutions that can be broken down into separate components. This approach addressed the vehicle routing problem with uncertain demands, resulting in state-of-the-art results. The effectiveness of this method is credited to the introduction of new optimality cuts and improved lower bounding functionals, which surpass existing limitations.

We developed a novel formulation, valid cuts, and lower bounds for the stochastic bicycle repositioning problem in the second contribution. Using the disaggregated integer L-shaped method proposed in the first contribution, we incorporated a mixture of optimality cuts, including so-called hybrid cuts and lower bounding functionals. Our method achieved state-of-the-art results on benchmark sets and proposed challenging new instances.

In the third contribution, we proposed a two-step approach to address the goal of strategic redistribution of bicycles within service level requirements (SLR) for bicycle sharing systems (BSS). By breaking down the SLR goal into a target-level problem (TLB) and a bicycle rebalancing problem (BRP), we provided insights into our models’ efficiency and practical applicability for real-life BSS. Our computational experiments demonstrated the superiority of model 2 in TLB analysis and model 6 in station capacity optimization. Integrating TLB and BRP approaches offers a unified

framework for efficient resource allocation and improved service quality.

7.2 Limitations de la solution proposée / Limitations

While our approaches have demonstrated significant improvements in solving stochastic optimization problems, some limitations remain.

For the vehicle routing problem with stochastic demands, the efficiency of our method relies heavily on the availability of tight bounds on disjoint components of the recourse. In cases where such bounds are not available or are weak, the method’s performance may degrade. Secondly, our approach achieves state-of-the-art results for problem families that require routes with a large number of customers, but for other families where shorter routes are required (less than ten customers per route, for example), the branch-and-price approach is more efficient than our method.

In the stochastic bicycle repositioning problem, our method’s success relies significantly on the availability of tight fleet size lower bounds, which could not be computed for the existing instance benchmark set. For some of those instances, our fleet size lower bounds represent only a fraction of the vehicles used by the best solutions found, highlighting a knowledge gap.

Regarding the two-step approach for bicycle sharing systems, the BRP’s reliance on the restocking trips (RT) policy results in higher computational times, making it potentially infeasible for real-time applications. Specifically, the RT policy requires low-complexity procedures for feasibility checks and lower bounds for the route cost because, in this problem, the vehicle routes encompass several hundred stations. While such procedures were developed and implemented for the continue-to-next policy, proving very efficient in practice, they could not be created for the RT policy. Additionally, the effectiveness of our models is highly dependent on the accuracy of demand forecasts and the granularity of real-world data, which are often not readily available for many real-life BSS scenarios.

7.3 Améliorations futures / Future Research

Future research should focus on several avenues to enhance the proposed methods and address their limitations.

For the vehicle routing problem with stochastic demands, extending the method to other variants of stochastic vehicle routing problems and two-stage stochastic integer programs could provide valuable insights. Additionally, deriving tight bounds on disjoint components for different problems of interest and systematically studying the monotonicity of the recourse function are promising directions.

In the stochastic bicycle repositioning problem, future research should focus on creating more precise estimates for fleet size and exploring alternative formulations to reduce computational workload. Additionally, streamlining the integration of various optimality cuts could improve the methods' efficiency.

For bicycle sharing systems, refining the demand forecasting models and improving the computational efficiency of the RT policy are crucial steps. Further research could explore new combinatorial optimization problems to enhance network design and demand modeling. Additionally, investigating the potential of combining capacity modification strategies with rebalancing policies in a real-time decision-making framework could significantly improve operational efficiency and resource utilization.

REFERENCES

- [1] O. Jabali, W. Rei, M. Gendreau, and G. Laporte, “Partial-route inequalities for the multi-vehicle routing problem with stochastic demands,” *Discrete Applied Mathematics*, vol. 177, pp. 121–136, 2014.
- [2] Y. Hoogendoorn and R. Spliet, “An improved integer l-shaped method for the vehicle routing problem with stochastic demands,” *INFORMS Journal on Computing*, vol. 35, no. 2, pp. 423–439, 2023.
- [3] C. Christiansen and J. Lysgaard, “A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands,” *Operations Research Letters*, vol. 35, no. 6, pp. 773–781, 2007.
- [4] C. Gauvin, G. Desaulniers, and M. Gendreau, “A branch-cut-and-price algorithm for the vehicle routing problem with stochastic demands,” *Computers & Operations Research*, vol. 50, pp. 141–153, 2014.
- [5] A. Florio, R. Hartl, and S. Minner, “New exact algorithm for the vehicle routing problem with stochastic demands,” *Transportation Science*, vol. 54, no. 4, pp. 1073–1090, 2020.
- [6] M. Dell’Amico, M. Iori, S. Novellani, and A. Subramanian, “The bike sharing rebalancing problem with stochastic demands,” *Transportation research part B: methodological*, vol. 118, pp. 362–380, 2018.
- [7] G. Laporte and F. Louveaux, “The integer L-shaped method for stochastic integer programs with complete recourse,” *Operations research letters*, vol. 13, no. 3, pp. 133–142, 1993.
- [8] F. Tillman, “The multiple terminal delivery problem with probabilistic demands,” *Transportation Science*, vol. 3, no. 3, pp. 192–204, 1969.

- [9] M. Gendreau, O. Jabali, and W. Rei, “50th anniversary invited article—future research directions in stochastic vehicle routing,” *Transportation Science*, vol. 50, no. 4, pp. 1163–1173, 2016.
- [10] D. Bertsimas, P. Jaillet, and A. Odoni, “A priori optimization,” *Operations Research*, vol. 38, no. 6, pp. 1019–1033, 1990.
- [11] M. Dror, G. Laporte, and P. Trudeau, “Vehicle routing with stochastic demands: Properties and solution frameworks,” *Transportation science*, vol. 23, no. 3, pp. 166–176, 1989.
- [12] V. Pillac, M. Gendreau, C. Gu  ret, and A. Medaglia, “A review of dynamic vehicle routing problems,” *European Journal of Operational Research*, vol. 225, no. 1, pp. 1–11, 2013.
- [13] M. Gendreau, G. Laporte, and R. S  guin, “An exact algorithm for the vehicle routing problem with stochastic demands and customers,” *Transportation science*, vol. 29, no. 2, pp. 143–155, 1995.
- [14] G. Laporte, F. Louveaux, and L. Van Hamme, “An integer L-shaped algorithm for the capacitated vehicle routing problem with stochastic demands,” *Operations Research*, vol. 50, no. 3, pp. 415–423, 2002.
- [15] W.-H. Yang, K. Mathur, and R. Ballou, “Stochastic vehicle routing problem with restocking,” *Transportation Science*, vol. 34, no. 1, pp. 99–112, 2000.
- [16] F. Louveaux and J.-J. Salazar-Gonz  lez, “Exact approach for the vehicle routing problem with stochastic demands and preventive returns,” *Transportation Science*, vol. 52, no. 6, pp. 1463–1478, 2018.
- [17] M. Salavati-Khoshghalb, M. Gendreau, O. Jabali, and W. Rei, “An exact algorithm to solve the vehicle routing problem with stochastic demands under an optimal restocking policy,” *European Journal of Operational Research*, vol. 273, no. 1, pp. 175–189, 2019.

- [18] A. Ak and A. L. Erera, “A paired-vehicle recourse strategy for the vehicle-routing problem with stochastic demands,” *Transportation Science*, vol. 41, no. 2, pp. 222–237, 2007.
- [19] N. Secomandi and F. Margot, “Reoptimization approaches for the vehicle-routing problem with stochastic demands,” *Operations Research*, vol. 57, no. 1, pp. 214–230, 2009.
- [20] A. M. Florio, D. Feillet, M. Poggi, and T. Vidal, “Vehicle routing with stochastic demands and partial reoptimization,” *Transportation Science*, vol. 56, no. 5, pp. 1393–1408, 2022.
- [21] M. J. Ota and R. Fukasawa, “Hardness of pricing routes for two-stage stochastic vehicle routing problems with scenarios,” *Operations Research*, 2024.
- [22] R. Wets, “Programming under uncertainty: the equivalent convex program,” *SIAM Journal on Applied Mathematics*, vol. 14, no. 1, pp. 89–105, 1966.
- [23] R. Séguin, “Problèmes stochastiques de tournées de véhicules,” Ph.D. dissertation, Université de Montréal, Montréal, 1996.
- [24] J.-F. Côté, M. Gendreau, and J.-Y. Potvin, “The vehicle routing problem with stochastic two-dimensional items,” *Transportation Science*, vol. 54, no. 2, pp. 453–469, 2020.
- [25] M. Dror and M. Ball, “Inventory/routing: Reduction from an annual to a short-period problem,” *Naval Research Logistics*, vol. 34, no. 6, pp. 891–905, 1987.
- [26] J. Kreimer and M. Dror, “The monotonicity of the threshold detection probability in a stochastic accumulation process,” *Computers & Operations Research*, vol. 17, no. 1, pp. 63–71, 1990.
- [27] G. Laporte, F. Meunier, and R. W. Calvo, “Shared mobility systems: an updated survey,” *Annals of Operations Research*, vol. 271, no. 1, pp. 105–126, 2018.

- [28] P. DeMaio, “Bike-sharing: History, impacts, models of provision, and future,” *Journal of public transportation*, vol. 12, no. 4, p. 3, 2009.
- [29] E. Eren and V. E. Uz, “A review on bike-sharing: The factors affecting bike-sharing demand,” *Sustainable Cities and Society*, vol. 54, p. 101882, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2210670719312387>
- [30] C. Shui and W. Szeto, “A review of bicycle-sharing service planning problems,” *Transportation Research Part C: Emerging Technologies*, vol. 117, p. 102648, 2020.
- [31] J. Todd, O. O’Brien, and J. Cheshire, “A global comparison of bicycle sharing systems,” *Journal of Transport Geography*, vol. 94, p. 103119, 2021.
- [32] Bixi-Montreal, “BIXI Montréal system size,” <https://bixi.com/en/who-we-are>, 2024, accessed: April 8, 2024.
- [33] Yahoo-Finance, “Bike sharing market size is projected to reach usd 9.96 billion by 2030,” <https://finance.yahoo.com/news/bike-sharing-market-size-projected-144000935.html>, 2023, accessed: April 8, 2024.
- [34] G. Reggiani, A. M. Salomons, M. Sterk, Y. Yuan, S. O’Hern, W. Daamen, and S. Hoogendoorn, “Bicycle network needs, solutions, and data collection systems: A theoretical framework and case studies,” *Case Studies on Transport Policy*, vol. 10, no. 2, pp. 927–939, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2213624X2200058X>
- [35] BBC, “Why some bike shares work and others don’t,” <https://www.bbc.com/future/article/20210112-the-vast-bicycle-graveyards-of-china>, 2021, accessed: April 17, 2024.
- [36] Y.-H. Kuo, J. M. Leung, and Y. Yan, “Public transport for smart cities: Recent innovations and future challenges,” *European Journal of*

- Operational Research*, vol. 306, no. 3, pp. 1001–1026, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S037722172200546X>
- [37] M. Benchimol, P. Benchimol, B. Chappert, A. De La Taille, F. Laroche, F. Meunier, and L. Robinet, “Balancing the stations of a self service “bike hire” system,” *RAIRO-Operations Research*, vol. 45, no. 1, pp. 37–61, 2011.
 - [38] B. A. Neumann-Saavedra, T. G. Crainic, B. Gendron, D. C. Mattfeld, and M. Römer, “Integrating resource management in service network design for bike-sharing systems,” *Transportation Science*, vol. 54, no. 5, pp. 1251–1271, 2020.
 - [39] B. P. Bruck, F. Cruz, M. Iori, and A. Subramanian, “The static bike sharing rebalancing problem with forbidden temporary operations,” *Transportation Science*, vol. 53, no. 3, pp. 882–896, 2019.
 - [40] S. Datner, T. Raviv, M. Tzur, and D. Chemla, “Setting inventory levels in a bike sharing network,” *Transportation Science*, vol. 53, no. 1, pp. 62–76, 2019.
 - [41] X. Luo, L. Li, L. Zhao, and J. Lin, “Dynamic intra-cell repositioning in free-floating bike-sharing systems using approximate dynamic programming,” *Transportation Science*, vol. 56, no. 4, pp. 799–826, 2022.
 - [42] J. Brinkmann, M. Ulmer, and D. Mattfeld, “Inventory routing for bike sharing systems,” *Transportation research procedia*, vol. 19, pp. 316–327, 2016.
 - [43] G. Erdoğan, G. Laporte, and R. W. Calvo, “The static bicycle relocation problem with demand intervals,” *European Journal of Operational Research*, vol. 238, no. 2, pp. 451–457, 2014.
 - [44] T. G. Crainic, B. Gendron, and M. R. Akhavan Kazemzadeh, “A taxonomy of multilayer network design and a survey of transportation and telecommunication applications,” *European Journal of Operational Research*, vol. 303, no. 1, pp. 1–13, 2022.

- [45] O. Caspi, M. J. Smart, and R. B. Noland, “Spatial associations of dockless shared e-scooter usage,” *Transportation Research Part D: Transport and Environment*, vol. 86, p. 102396, 2020.
- [46] B. P. Bruck, F. Cruz, M. Iori, and A. Subramanian, “The static bike sharing rebalancing problem with forbidden temporary operations,” *Transportation Science*, vol. 53, no. 3, pp. 882–896, 2019.
- [47] D. Chemla, F. Meunier, and R. W. Calvo, “Bike sharing systems: Solving the static rebalancing problem,” *Discrete Optimization*, vol. 10, no. 2, pp. 120–146, 2013.
- [48] T. Raviv, M. Tzur, and I. A. Forma, “Static repositioning in a bike-sharing system: models and solution approaches,” *EURO Journal on Transportation and Logistics*, vol. 2, no. 3, pp. 187–229, 2013.
- [49] Z. Haider, A. Nikolaev, J. E. Kang, and C. Kwon, “Inventory rebalancing through pricing in public bike sharing systems,” *European Journal of Operational Research*, vol. 270, no. 1, pp. 103–117, 2018.
- [50] J. Schuijbroek, R. Hampshire, and W.-J. van Hoes, “Inventory rebalancing and vehicle routing in bike sharing systems,” *European Journal of Operational Research*, vol. 257, no. 3, pp. 992–1004, 2017. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0377221716306658>
- [51] S. Ghosh, P. Varakantham, Y. Adulyasak, and P. Jaillet, “Dynamic repositioning to reduce lost demand in bike sharing systems,” *Journal of Artificial Intelligence Research*, vol. 58, pp. 387–430, 2017.
- [52] C. Contardo, C. Morency, and L.-M. Rousseau, *Balancing a dynamic public bike-sharing system*. Cirrelet Montreal, Cansada, 2012, vol. 4.
- [53] D. Freund, S. G. Henderson, and D. B. Shmoys, “Minimizing multimodular functions and allocating capacity in bike-sharing systems,” *Operations Research*, vol. 70, no. 5, pp. 2715–2731, 2022.

- [54] CitiBike, “Citi bike nyc operating reports,” <https://citibikenyc.com/system-data/operating-reports>, 2023, accessed: January 1, 2024.
- [55] R. Cavagnini, F. Maggioni, L. Bertazzi, and M. Hewitt, “A two-stage stochastic programming model for bike-sharing systems with rebalancing,” *EURO Journal on Transportation and Logistics*, vol. 13, p. 100140, 2024.
- [56] J. Benders, “Partitioning procedures for solving mixed-variables programming problems,” *Numerische mathematik*, vol. 4, no. 1, pp. 238–252, 1962.
- [57] P. Toth and D. Vigo, *Vehicle routing: problems, methods, and applications*. SIAM, 2014.
- [58] H. Hernández-Pérez and J.-J. Salazar-González, “A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery,” *Discrete Applied Mathematics*, vol. 145, no. 1, pp. 126–139, 2004.
- [59] J. Li, Z. Luo, R. Baldacci, H. Qin, and Z. Xu, “A new exact algorithm for single-commodity vehicle routing with split pickups and deliveries,” *INFORMS Journal on Computing*, vol. 35, no. 1, pp. 31–49, 2023.
- [60] R. M. Van Slyke and R. Wets, “L-shaped linear programs with applications to optimal control and stochastic programming,” *SIAM Journal on Applied Mathematics*, vol. 17, no. 4, pp. 638–663, 1969.
- [61] J. R. Birge and R. J.-B. Wets, *Designing approximation schemes for stochastic optimization problems, in particular for stochastic programs with recourse*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1986.
- [62] J. Hooker and G. Ottosson, “Logic-based benders decomposition,” *Mathematical Programming*, vol. 96, no. 1, pp. 33–60, 2003.
- [63] R. Rahmaniani, T. G. Crainic, M. Gendreau, and W. Rei, “The benders decomposition algorithm: A literature review,” *European Journal of Operational Research*, vol. 259, no. 3, pp. 801–817, 2017.

- [64] B. Denton, A. Miller, H. Balasubramanian, and T. Huschka, “Optimal allocation of surgery blocks to operating rooms under uncertainty,” *Operations Research*, vol. 58, no. 4-part-1, pp. 802–816, 2010.
- [65] Y. Li, J.-F. Côté, L. Callegari-Coelho, and P. Wu, “Novel formulations and logic-based benders decomposition for the integrated parallel machine scheduling and location problem,” *INFORMS Journal on Computing*, vol. 34, no. 2, pp. 1048–1069, 2022.
- [66] O. Elçi and J. Hooker, “Stochastic planning and scheduling with logic-based benders decomposition,” *INFORMS Journal on Computing*, vol. 34, no. 5, pp. 2428–2442, 2022.
- [67] J. Lysgaard, “Cvrpsep: A package of separation routines for the capacitated vehicle routing problem,” Department of Management Science and Logistics, Aarhus School of Business, Tech. Rep., 2003.
- [68] S. Ropke and D. Pisinger, “An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows,” *Transportation Science*, vol. 40, no. 4, pp. 455–472, 2006.
- [69] E. Uchoa, D. Pecin, A. Pessoa, M. Poggi, T. Vidal, and A. Subramanian, “New benchmark instances for the capacitated vehicle routing problem,” *European Journal of Operational Research*, vol. 257, no. 3, pp. 845–858, 2017.
- [70] C. Hjorring and J. Holt, “New optimality cuts for a single-vehicle stochastic routing problem,” *Annals of Operations Research*, vol. 86, pp. 569–584, 1999.
- [71] L. Parada, R. Legault, J.-F. Côté, and M. Gendreau, “A disaggregated integer l-shaped method for stochastic vehicle routing problems with monotonic recourse,” *European Journal of Operational Research*, 2024.
- [72] D. Bertsimas, P. Jaillet, and A. Odoni, “A priori optimization,” *Operations Research*, vol. 38, no. 6, pp. 1019–1033, 1990.

- [73] M. R. Garey and D. S. Johnson, *Computers and intractability*. freeman San Francisco, 1979, vol. 174.
- [74] J. Shu, M. C. Chou, Q. Liu, C.-P. Teo, and I.-L. Wang, “Models for effective deployment and redistribution of bicycles within public bicycle-sharing systems,” *Operations Research*, vol. 61, no. 6, pp. 1346–1359, 2013.
- [75] J. Birge and F. Louveaux, *Introduction to stochastic programming*. Springer Science & Business Media, 2011.
- [76] A. Shapiro, “Monte carlo sampling methods,” in *Stochastic Programming*, ser. Handbooks in Operations Research and Management Science. Elsevier, 2003, vol. 10, pp. 353–425. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0927050703100060>
- [77] J. Warrington and D. Ruchti, “Two-stage stochastic approximation for dynamic rebalancing of shared mobility systems,” *Transportation Research Part C: Emerging Technologies*, vol. 104, pp. 110–134, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0968090X18314104>
- [78] T. Santoso, S. Ahmed, M. Goetschalckx, and A. Shapiro, “A stochastic programming approach for supply chain network design under uncertainty,” *European Journal of Operational Research*, vol. 167, no. 1, pp. 96–115, 2005.
- [79] M. Salavati-Khoshghalb, M. Gendreau, O. Jabali, and W. Rei, “An exact algorithm to solve the vehicle routing problem with stochastic demands under an optimal restocking policy,” *European Journal of Operational Research*, vol. 273, no. 1, pp. 175–189, 2019.
- [80] MobilityData, “General bikeshare feed specification (gbfs),” 2024, accessed: 2024-06-08. [Online]. Available: <https://github.com/MobilityData/gbfs>
- [81] J. H. Goto, M. E. Lewis, and M. L. Puterman, “Coffee, tea, or . . .?: A markov decision process model for airline meal provisioning,” *Transportation Science*, vol. 38, no. 1, pp. 107–118, 2004.

- [82] A. J. Kleywegt, A. Shapiro, and T. Homem-de Mello, “The sample average approximation method for stochastic discrete optimization,” *SIAM Journal on Optimization*, vol. 12, no. 2, pp. 479–502, 2002.
- [83] E. S. Thorsteinsson, “Branch-and-check: A hybrid framework integrating mixed integer programming and constraint logic programming,” in *Principles and Practice of Constraint Programming — CP 2001*, T. Walsh, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 16–30.
- [84] T. L. Magnanti, P. Mireault, and R. T. Wong, *Tailoring Benders decomposition for uncapacitated network design*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1986, pp. 112–154.
- [85] T. L. Magnanti and R. T. Wong, “Accelerating benders decomposition: Algorithmic enhancement and model selection criteria,” *Operations Research*, vol. 29, no. 3, pp. 464–484, 1981.
- [86] W. Rei, J.-F. Cordeau, M. Gendreau, and P. Soriano, “Accelerating benders decomposition by local branching,” *INFORMS Journal on Computing*, vol. 21, no. 2, pp. 333–345, 2009.
- [87] R. Rahmaniani, S. Ahmed, T. G. Crainic, M. Gendreau, and W. Rei, “The benders dual decomposition method,” *Operations Research*, vol. 68, no. 3, pp. 878–895, 2020.
- [88] L. Parada, J.-F. Côté, and M. Gendreau, “An integer l-shaped method for the static stochastic bicycle repositioning problem,” CIRRELT, Tech. Rep. CIRRELT-2024-26, 2024.
- [89] S. M. Ross, *Introduction to probability models*. Academic press, 2014.
- [90] S. K. Park and K. W. Miller, “Random number generators: good ones are hard to find,” *Communications of the ACM*, vol. 31, no. 10, pp. 1192–1201, 1988.

- [91] D. F. Carta, “Two fast implementations of the “minimal standard” random number generator,” *Communications of the ACM*, vol. 33, no. 1, pp. 87–88, 1990.
- [92] M. Jöhnk, “Erzeugung von betaverteilten und gammaverteilten zufallszahlen,” *Metrika*, vol. 8, pp. 5–15, 1964.
- [93] G. Marsaglia and W. W. Tsang, “A simple method for generating gamma variables,” *ACM Transactions on Mathematical Software (TOMS)*, vol. 26, no. 3, pp. 363–372, 2000.
- [94] MontrealGazette, “Managing the bixi maze: A day in the life of montreal’s bike-sharing service,” <https://montrealgazette.com/news/local-news/managing-the-bixi-maze-a-day-in-the-life-of-montreals-bike-sharing-service-part-i>, 2015, accessed: April 17, 2024.

APPENDIX A MONOTONICITY PROOFS FOR THE VEHICLE ROUTING PROBLEM WITH STOCHASTIC DEMANDS.

A.1 Proofs of Proposition 5, 6 and 7

This appendix presents the proofs that the monotonicity property is respected, under some conditions, when customer demands follow independent binomial, Erlang and negative binomial probability distributions. For the sake of completeness, we also provide the corresponding propositions.

Proposition 5. *Any set S of customers whose demands are given by independent binomial RVs $\xi_i \sim \text{Bin}(n_i, p)$ sharing a common success probability p with total expected value $\sum_{i \in S} n_i p \leq Q$ respects the monotonicity condition.*

Proof. Consider a pair $(a, b) \in S \times S$ such that $a \neq b$ and a subset $\tilde{S} \subseteq S \setminus \{a, b\}$ with total expected demand $\tilde{n}p$, where $\tilde{n} = \sum_{i \in \tilde{S}} n_i$. Using the notation of Proposition 1, the demands $\tilde{\xi}$, ξ_a and ξ_b can respectively be decomposed as the sum of $\tilde{\rho} = \tilde{n}$, $\rho_a = n_a$ and $\rho_b = n_b$ i.i.d. RVs ξ_k , $k \in \mathbb{N}$, such that $\xi_k \sim \text{Bern}(p)$. From there, the sufficient condition (4.13) can be rewritten as follows.

$$P_{lQ}(\tilde{n} + n_a + j) \geq P_{lQ}(\tilde{n} + j) \quad \forall l \in \mathbb{N}, \forall j \in \{1, \dots, n_b\} \quad (\text{A.1})$$

It follows from [26] that, for any $l \in \mathbb{N}$, the sequence $\{P_{lQ}(s)\}_{s=1}^{\lceil \frac{lQ}{p} \rceil}$ is non-decreasing. Since $\tilde{n} + n_a + n_b \leq \sum_{i \in S} n_i \leq Q/p$, this allows to conclude that (A.1) holds. $\square$

Proposition 6. *Any set S of customers whose demands are given by independent Erlang RVs $\xi_i \sim \text{Erlang}(n_i, \lambda)$ sharing a common rate parameter λ with total expected value $\sum_{i \in S} n_i / \lambda \leq Q$ respects the monotonicity condition.*

Proof. Consider a pair $(a, b) \in S \times S$ such that $a \neq b$ and a subset $\tilde{S} \subseteq S \setminus \{a, b\}$ with total expected demand $\tilde{n}/\lambda$, where $\tilde{n} = \sum_{i \in \tilde{S}} n_i$. Using the notation of Proposition 1,

the demands $\tilde{\xi}$, ξ_a and ξ_b can respectively be decomposed as the sum of $\tilde{\rho}=\tilde{n}$, $\rho_a=n_a$ and $\rho_b=n_b$ i.i.d. RVs ξ_k , $k \in \mathbb{N}$, such that $\zeta_k \sim \text{Exp}(\lambda)$. From there, the sufficient condition (4.13) can be rewritten as follows.

$$P_{lQ}(\tilde{n} + n_a + j) \geq P_{lQ}(\tilde{n} + j) \quad \forall l \in \mathbb{N}, \forall j \in \{1, \dots, n_b\} \quad (\text{A.2})$$

It follows from [26] that, for any $l \in \mathbb{N}$, the sequence $\{P_{lQ}(s)\}_{s=1}^{\lceil \lambda lQ \rceil}$ is non-decreasing. Since $\tilde{n}+n_a+n_b \leq \sum_{i \in S} n_i \leq \lambda Q$, this allows to conclude that (A.2) holds. $\square$

Proposition 7. *Any set S of customers whose demands are given by independent negative binomial RVs $\xi_i \sim \text{NB}(r_i, p)$ sharing a common success probability p with total expected value $\sum_{i \in S} r_i(1-p)/p \leq Q$ respects the monotonicity condition.*

Proof. Consider a pair $(a, b) \in S \times S$ such that $a \neq b$, a subset $\tilde{S} \subseteq S \setminus \{a, b\}$ with total expected demand $\tilde{r}(1-p)/p$, where $\tilde{r} = \sum_{i \in \tilde{S}} r_i$ and a positive integer $l \in \mathbb{N}$. We use the fact that the sum of s independent negative binomial RVs with parameters $r_1, \dots, r_s$ and the same success probability p is also a negative binomial RV with parameters $r = \sum_{k=1}^s r_k$ and p to rewrite inequality (4.12) as follows.

$$\mathbb{P}(\tilde{\xi} + \xi_a \leq lQ < \tilde{\xi} + \xi_a + \xi_b) \geq \mathbb{P}(\tilde{\xi} \leq lQ < \tilde{\xi} + \xi_b) \quad (\text{A.3})$$

$$\iff \mathbb{P}\left(\sum_{k=1}^{lQ+\tilde{r}+r_a} \zeta_k \leq lQ < \sum_{k=1}^{lQ+\tilde{r}+r_a+r_b} \zeta_k\right) \geq \mathbb{P}\left(\sum_{k=1}^{lQ+\tilde{r}} \zeta_k \leq lQ < \sum_{k=1}^{lQ+\tilde{r}+r_b} \zeta_k\right) \quad (\text{A.4})$$

To obtain inequality (A.4), we use the fact that the CDF of a negative binomial with parameters r and p evaluated at $lQ \in \mathbb{N}$ is equal to the CDF of a binomial distribution with $lQ+r$ trials and probability of success $1-p$ evaluated at lQ . In inequality (A.4), these binomial RVs are decomposed into sums of i.i.d. RVs $\zeta_k \sim \text{Bern}(1-p)$. Using this reformulation of the monotonicity condition based on the RVs ζ_k , it suffices to show that $P_{lQ}(lQ + \tilde{r} + r_a + j) \geq P_{lQ}(lQ + \tilde{r} + j) \quad \forall j \in \{1, \dots, r_b\}$ to conclude the proof, by Proposition 1. For a given $j \in \{1, \dots, r_b\}$, this inequality can be developed

as follows:

$$P_{lQ}(lQ + \tilde{r} + r_a + j) \geq P_{lQ}(lQ + \tilde{r} + j) \quad (\text{A.5})$$

$$\iff \mathbb{P} \left(\sum_{k=1}^{lQ+\tilde{r}+r_a+j-1} \zeta_k \leq lQ < \sum_{k=1}^{lQ+\tilde{r}+r_a+j} \zeta_k \right) \geq \mathbb{P} \left(\sum_{k=1}^{lQ+\tilde{r}+j-1} \zeta_k \leq lQ < \sum_{k=1}^{lQ+\tilde{r}+j} \zeta_k \right) \quad (\text{A.6})$$

$$\iff \mathbb{P}(\chi \leq lQ < \chi + \zeta_1) \geq \mathbb{P}(\bar{\chi} \leq lQ < \bar{\chi} + \zeta_1), \quad (\text{A.7})$$

where $\chi \sim \text{Bin}(lQ + \tilde{r} + r_a + j - 1, 1-p)$ and $\bar{\chi} \sim \text{Bin}(lQ + \tilde{r} + j - 1, 1-p)$. Note that the event in the left-hand side probability of inequality (A.7) occurs if and only if $\chi = lQ$ and $\zeta_1 = 1$. Since χ and ζ_1 are independent, the joint probability of these two events is equal to the product of their probabilities. Applying the same reasoning to the left-hand side, inequality (A.7) can be written as follows.

$$\mathbb{P}(\chi = lQ) \mathbb{P}(\zeta_1 = 1) \geq \mathbb{P}(\bar{\chi} = lQ) \mathbb{P}(\zeta_1 = 1) \quad (\text{A.8})$$

$$\iff \mathbb{P}(\chi = lQ) \geq \mathbb{P}(\bar{\chi} = lQ) \quad (\text{A.9})$$

$$\iff \frac{(lQ + \tilde{r} + r_a + j - 1)!}{(lQ)!(\tilde{r} + r_a + j - 1)!} (1-p)^{lQ} p^{\tilde{r}+r_a+j-1} \geq \frac{(lQ + \tilde{r} + j - 1)!}{(lQ)!(\tilde{r} + j - 1)!} (1-p)^{lQ} p^{\tilde{r}+j-1} \quad (\text{A.10})$$

$$\iff \frac{\prod_{i=1}^{r_a} (lQ + \tilde{r} + i + j - 1)}{\prod_{i=1}^{r_a} (\tilde{r} + i + j - 1)} p^{r_a} \geq 1 \quad (\text{A.11})$$

$$\iff \prod_{i=1}^{r_a} \left(\frac{p(lQ + \tilde{r} + i + j - 1)}{\tilde{r} + i + j - 1} \right) \geq 1 \quad (\text{A.12})$$

To complete the proof, we demonstrate that inequality (A.12) holds by showing that each term in the product is greater than or equal to 1. For $i \in \{1, \dots, r_a\}$, we have:

$$\frac{p(lQ + \tilde{r} + i + j - 1)}{\tilde{r} + i + j - 1} = \frac{plQ}{\tilde{r} + i + j - 1} + p \quad (\text{A.13})$$

$$\geq \frac{pQ}{\tilde{r} + r_a + r_b} + p \quad (\text{A.14})$$

$$\geq \frac{pQ}{\sum_{i \in S} r_i} + p \quad (\text{A.15})$$

$$\geq \frac{(1-p)pQ}{pQ} + p \quad (\text{A.16})$$

$$= 1 \quad (\text{A.17})$$

To obtain inequality (A.14), we notice that $l \geq 1$ and $i+j-1 \leq r_a+r_b-1 \leq r_a+r_b$. Inequality (A.16) follows from the assumption that the total expected demand of the customers of set S does not exceed the vehicle capacity. $\square$

A.2 Proof of Proposition 12

This appendix presents the proof that arranging a set of customers adhering to the monotonicity property, in non-increasing order based on their individual distances to the depot, provides a lower bound for the expected recourse cost.

Proposition 12. *Let $S \subseteq N$ be a set of customers respecting the monotonicity condition. Let $p^* = (i_1^*, i_2^*, \dots, i_{|S|}^*) \in \mathcal{P}^1(S)$ be a path such that $c_{0i_1^*} \geq c_{0i_2^*} \geq \dots \geq c_{0i_{|S|}^*}$. The expected recourse $\mathcal{Q}(p^*)$ of path p^* is a valid lower bound on the expected recourse $\mathcal{Q}(p)$ of any path $p \in \mathcal{P}^1(S)$.*

Proof. Let us consider a path $p = (i_1, \dots, i_{|S|}) \in \mathcal{P}^1(S)$ in which the customers are not sorted in non-increasing order of distances to the depot. In particular, this implies that there exists at least one index $h \in \{1, \dots, |S|-1\}$ such that $c_{0i_h} < c_{0i_{h+1}}$. We show that the expected recourse of path $p' = (i_1, \dots, i_{h+1}, i_h, \dots, i_{|S|}) \in \mathcal{P}^1(S)$, in which the order of visit of customers i_h and i_{h+1} is inverted, is less than or equal to that of path p .

We first develop the expression of $\mathcal{Q}(p)$:

$$\mathcal{Q}(p) = 2 \sum_{j=1}^{|S|} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j}$$

$$\begin{aligned}
&= 2 \sum_{j=1}^{h-1} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j} + 2 \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{h-1} \xi_{i_k} \leq lQ < \sum_{k=1}^h \xi_{i_k} \right) c_{0i_h} \\
&\quad + 2 \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^h \xi_{i_k} \leq lQ < \sum_{k=1}^{h+1} \xi_{i_k} \right) c_{0i_{h+1}} + 2 \sum_{j=h+2}^{|S|} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j} \\
&= 2 \sum_{j=1}^{h-1} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j} + 2 \sum_{l=1}^{\infty} \mathbb{P} \left(\tilde{\xi} \leq lQ < \tilde{\xi} + \xi_{i_h} \right) c_{0i_h} \\
&\quad + 2 \sum_{l=1}^{\infty} \mathbb{P} \left(\tilde{\xi} + \xi_{i_h} \leq lQ < \tilde{\xi} + \xi_{i_h} + \xi_{i_{h+1}} \right) c_{0i_{h+1}} + 2 \sum_{j=h+2}^{|S|} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j},
\end{aligned}$$

where $\tilde{\xi} = \sum_{k=1}^{h-1} \xi_{i_k}$. Doing the same for $\mathcal{Q}(p')$, we obtain:

$$\begin{aligned}
\mathcal{Q}(p') &= 2 \sum_{j=1}^{h-1} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j} + 2 \sum_{l=1}^{\infty} \mathbb{P} \left(\tilde{\xi} \leq lQ < \tilde{\xi} + \xi_{i_{h+1}} \right) c_{0i_{h+1}} \\
&\quad + 2 \sum_{l=1}^{\infty} \mathbb{P} \left(\tilde{\xi} + \xi_{i_{h+1}} \leq lQ < \tilde{\xi} + \xi_{i_{h+1}} + \xi_{i_h} \right) c_{0i_h} + 2 \sum_{j=h+2}^{|S|} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k} \leq lQ < \sum_{k=1}^j \xi_{i_k} \right) c_{0i_j}
\end{aligned}$$

Since the expected recourse cost at customers i_1 to i_{h-1} and at customers i_{h+2} to $i_{|S|}$ is identical in both paths, the inequality $\mathcal{Q}(p) \geq \mathcal{Q}(p')$ simplifies to:

$$\begin{aligned}
&\mathcal{Q}(p) \geq \mathcal{Q}(p') \\
&\iff 2 \sum_{l=1}^{\infty} \mathbb{P} \left(\tilde{\xi} \leq lQ < \tilde{\xi} + \xi_{i_h} \right) c_{0i_h} + 2 \sum_{l=1}^{\infty} \mathbb{P} \left(\tilde{\xi} + \xi_{i_h} \leq lQ < \tilde{\xi} + \xi_{i_h} + \xi_{i_{h+1}} \right) c_{0i_{h+1}} \\
&\quad \geq 2 \sum_{l=1}^{\infty} \mathbb{P} \left(\tilde{\xi} \leq lQ < \tilde{\xi} + \xi_{i_{h+1}} \right) c_{0i_{h+1}} + 2 \sum_{l=1}^{\infty} \mathbb{P} \left(\tilde{\xi} + \xi_{i_{h+1}} \leq lQ < \tilde{\xi} + \xi_{i_{h+1}} + \xi_{i_h} \right) c_{0i_h} \\
&\iff \sum_{l=1}^{\infty} \left(\mathbb{P} \left(\tilde{\xi} \leq lQ < \tilde{\xi} + \xi_{i_h} \right) - \mathbb{P} \left(\tilde{\xi} + \xi_{i_{h+1}} \leq lQ < \tilde{\xi} + \xi_{i_{h+1}} + \xi_{i_h} \right) \right) c_{0i_h} \\
&\quad \geq \sum_{l=1}^{\infty} \left(\mathbb{P} \left(\tilde{\xi} \leq lQ < \tilde{\xi} + \xi_{i_{h+1}} \right) - \mathbb{P} \left(\tilde{\xi} + \xi_{i_h} \leq lQ < \tilde{\xi} + \xi_{i_h} + \xi_{i_{h+1}} \right) \right) c_{0i_{h+1}} \\
&\iff \sum_{l=1}^{\infty} \left(\mathbb{P} \left(\tilde{\xi} + \xi_{i_h} > lQ \right) - \mathbb{P} \left(\tilde{\xi} > lQ \right) - \mathbb{P} \left(\tilde{\xi} + \xi_{i_{h+1}} + \xi_{i_h} > lQ \right) + \mathbb{P} \left(\tilde{\xi} + \xi_{i_{h+1}} > lQ \right) \right) c_{0i_h}
\end{aligned}$$

$$\begin{aligned}
&\geq \sum_{l=1}^{\infty} \left(\mathbb{P}(\tilde{\xi} + \xi_{i_{h+1}} > lQ) - \mathbb{P}(\tilde{\xi} > lQ) - \mathbb{P}(\tilde{\xi} + \xi_{i_h} + \xi_{i_{h+1}} > lQ) + \mathbb{P}(\tilde{\xi} + \xi_{i_h} > lQ) \right) c_{0i_{h+1}} \\
&\iff \sum_{l=1}^{\infty} \left(-\mathbb{P}(\tilde{\xi} + \xi_{i_h} \leq lQ) + \mathbb{P}(\tilde{\xi} \leq lQ) + \mathbb{P}(\tilde{\xi} + \xi_{i_{h+1}} + \xi_{i_h} \leq lQ) - \mathbb{P}(\tilde{\xi} + \xi_{i_{h+1}} \leq lQ) \right) c_{0i_h} \\
&\geq \sum_{l=1}^{\infty} \left(-\mathbb{P}(\tilde{\xi} + \xi_{i_{h+1}} \leq lQ) + \mathbb{P}(\tilde{\xi} \leq lQ) + \mathbb{P}(\tilde{\xi} + \xi_{i_h} + \xi_{i_{h+1}} \leq lQ) - \mathbb{P}(\tilde{\xi} + \xi_{i_h} \leq lQ) \right) c_{0i_{h+1}} \\
&\iff (c_{0i_{h+1}} - c_{0i_h}) \sum_{l=1}^{\infty} \left(\mathbb{P}(\tilde{\xi} + \xi_{i_h} \leq lQ) - \mathbb{P}(\tilde{\xi} \leq lQ) - \mathbb{P}(\tilde{\xi} + \xi_{i_{h+1}} + \xi_{i_h} \leq lQ) + \mathbb{P}(\tilde{\xi} + \xi_{i_{h+1}} \leq lQ) \right) \\
&\geq 0 \\
&\iff \mathbb{P}(\tilde{\xi} + \xi_{i_h} \leq lQ) - \mathbb{P}(\tilde{\xi} + \xi_{i_{h+1}} + \xi_{i_h} \leq lQ) \geq \mathbb{P}(\tilde{\xi} \leq lQ) - \mathbb{P}(\tilde{\xi} + \xi_{i_{h+1}} \leq lQ), \forall l \geq 1
\end{aligned}$$

Since S respects the monotonicity condition, the last inequality, which corresponds to inequality (4.12) with $\tilde{S} = \{i_1, \dots, i_{h-1}\}$ and $(a, b) = (i_h, i_{h+1})$, is verified for any $l \in \mathbb{N}$.

This implies that, starting from any path $p \in \mathcal{P}^1(S)$, one can iteratively perform inversions of consecutive customers $(a, b) \in S \times S$ such that $c_a < c_b$ until all the customers are sorted in non-increasing order of their distance to the depot without increasing the expected recourse cost. Therefore, the set $\arg \min_{p \in \mathcal{P}^1(S)} \mathcal{Q}(p)$ contains at least one path $\tilde{p} = (\tilde{i}_1, \dots, \tilde{i}_{|S|})$ such that $c_{0\tilde{i}_1} \geq c_{0\tilde{i}_2} \geq \dots \geq c_{0\tilde{i}_{|S|}}$.

To conclude that $p^* \in \arg \min_{p \in \mathcal{P}^1(S)} \mathcal{Q}(p)$, and thus that $\mathcal{Q}(p^*)$ is a valid lower bound on the expected recourse of any path $p \in \mathcal{P}^1(S)$, it remains to show that $\mathcal{Q}(\tilde{p}) = \mathcal{Q}(p^*)$. To do so, we sort the distances separating the customers of S from the depot and remove the duplicates to obtain an ordered list $c_{0(1)} > \dots > c_{0(t)}$ of length $t \leq |S|$. For each $s \in \{1, \dots, t\}$, let us denote by $C_s = \{i \in S : c_{0i} = c_{0(s)}\}$ the set of customers that share the same distance $c_{0(s)}$ from the depot and by $\xi^s = \sum_{i \in C_s} \xi_i$ the sum of their demands. By construction, in both paths $\tilde{p}$ and p^* , a permutation of C_1 will first be visited, followed by a permutation of C_2 and so on until C_t . Denoting the number of customers in group C_s as $v_s = |C_s|$ and the total number of customers visited before moving to group C_{s+1} as $V_s = \sum_{q=1}^s v_q$, we can demonstrate

the expected equality.

$$\mathcal{Q}(\tilde{p}) = 2 \sum_{j=1}^{|S|} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{\tilde{i}_k} \leq lQ < \sum_{k=1}^j \xi_{\tilde{i}_k} \right) c_{0\tilde{i}_j} \quad (\text{A.18})$$

$$= 2 \sum_{s=1}^t \sum_{j=V_{s-1}+1}^{V_s} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{\tilde{i}_k} \leq lQ < \sum_{k=1}^j \xi_{\tilde{i}_k} \right) c_{0(s)} \quad (\text{A.19})$$

$$= 2 \sum_{s=1}^t c_{0(s)} \sum_{l=1}^{\infty} \left(\sum_{j=V_{s-1}+1}^{V_s} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{\tilde{i}_k} \leq lQ < \sum_{k=1}^j \xi_{\tilde{i}_k} \right) \right) \quad (\text{A.20})$$

$$= 2 \sum_{s=1}^t c_{0(s)} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{q=1}^{s-1} \xi^q \leq lQ < \sum_{q=1}^s \xi^q \right) \quad (\text{A.21})$$

$$= 2 \sum_{s=1}^t c_{0(s)} \sum_{l=1}^{\infty} \left(\sum_{j=V_{s-1}+1}^{V_s} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k^*} \leq lQ < \sum_{k=1}^j \xi_{i_k^*} \right) \right) \quad (\text{A.22})$$

$$= 2 \sum_{s=1}^t \sum_{j=V_{s-1}+1}^{V_s} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k^*} \leq lQ < \sum_{k=1}^j \xi_{i_k^*} \right) c_{0(s)} \quad (\text{A.23})$$

$$= 2 \sum_{j=1}^{|S|} \sum_{l=1}^{\infty} \mathbb{P} \left(\sum_{k=1}^{j-1} \xi_{i_k^*} \leq lQ < \sum_{k=1}^j \xi_{i_k^*} \right) c_{0i_j^*} \quad (\text{A.24})$$

$$= \mathcal{Q}(p^*) \quad (\text{A.25})$$

The equivalence of equalities (A.20) and (A.21) and, analogously, of equalities (A.21) and (A.22), comes from the fact that the l -th restocking trip will occur at a customer of set C_s if and only if the total demand of the customers of sets C_1 to C_{s-1} is less than or equal to lQ and the total demand of the customers of sets C_1 to C_s exceeds lQ . The sum of the probabilities of the mutually exclusive events that are considered for each $s \in \{1, \dots, t\}$ and for each $l \in \mathbb{N}$ in both equations (A.20) and (A.22) is thus equal to the probability of their disjunction, which is given in (A.21). $\square$