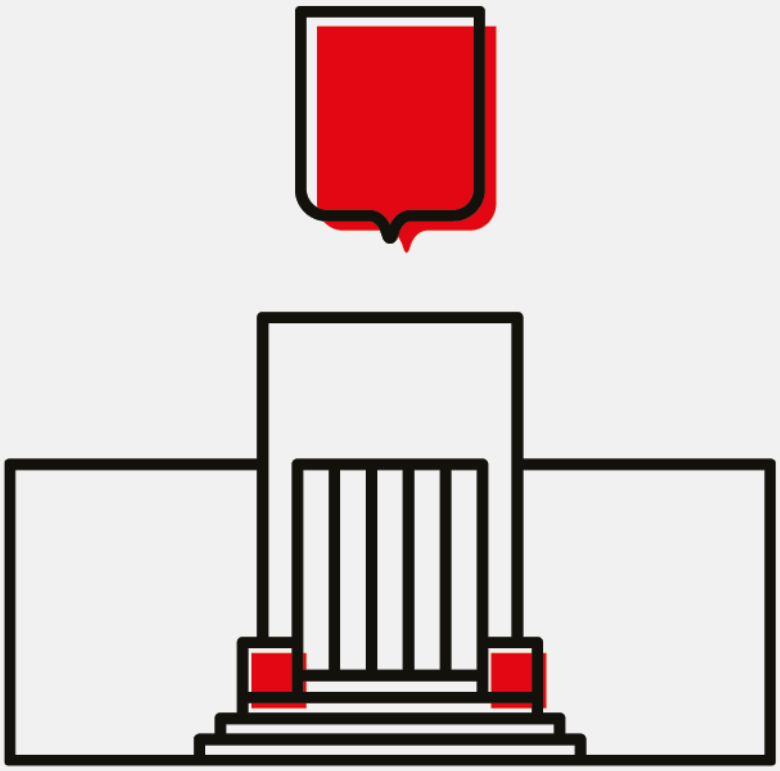
Document de travail 2025-001

SERVICE LEVEL REQUIREMENTS FOR REAL-LIFE-SIZED BICYCLE SHARING SYSTEMS



Lucas PARADA
Jean-François CÔTÉ
Michel GENDREAU

Janvier 2025

Publié par :

Faculté des sciences de l'administration
2325, rue de la Terrasse
Pavillon Palasis-Prince, Université Laval
Québec (Québec) Canada G1V 0A6

Voir tous les documents de travail

Document de travail également publié par le Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, sous le numéro CIRRELT-2025-02

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2025
Bibliothèque et Archives Canada, 2025
ISBN 978-2-89524-551-3 (PDF)

Faculté des sciences de l'administration

UNIVERSITÉ LAVAL

SERVICE LEVEL REQUIREMENTS FOR REAL-LIFE-SIZED BICYCLE SHARING SYSTEMS

Lucas Parada[1,2], **Jean-François Côté**[1,3,*], **Michel Gendreau**[1,2]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Mathematical and Industrial Engineering, Polytechnique Montréal

[3] Department of Operations and Decision Systems, FSA, Université Laval

*Corresponding author: jean-francois.cote@fsa.ulaval.ca

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 trips. We gather data from real-life BSS from Boston, Chicago, Madrid, Mexico City, Montreal, 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.

Keywords: Last-mile delivery, park-and-loop, hybrid metaheuristics.

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

1 Introduction

Bicycle sharing systems (BSS) are integral to urban transportation, intertwining with broader transit networks to enhance overall effectiveness in urban mobility (Shu et al., 2013). Similar to larger urban transit frameworks, BSS present a fertile ground for various combinatorial optimization problems (Bruck et al., 2019; Neumann-Saavedra et al., 2020), 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) (Datner et al., 2019), 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. 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 (Shapiro, 2003). 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 (Benders, 1962). 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. Finally, we assess the quality of our TLB solution by evaluating it on larger sample sizes using standard SAA methodologies (Santoso et al., 2005).

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 Dell'Amico et al. (2018). 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 Salavati-Khoshghalb et al., 2019).

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 literature: the General Bikeshare Feed Specification (GBFS, MobilityData, 2024). 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 Schuijbroek et al. (2017), Dell'Amico et al. (2018), and Freund et al. (2022). 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 found in BSS literature. This comprehensive set facilitates the 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 contractual bicycles in inventory, i.e., those the system operator has, in theory, (Beroud et al., 2024). In specific instances, the BSS required as little as 50% of the 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 contributions to the field. Firstly, we introduce a novel 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 proposes a set of real-life-sized instances.

The remainder of this paper is organized as follows. Section 2 presents the literature on BSS. Sections 3, 4 and 5 present the problem definitions, formulations and solution methods for SLR, TLB and BRP respectively. Section 6 outlines the processes of generating data. Section 7 presents the numerical results for our implementations, and lastly, Section 8 presents the conclusions.

2 Literature Review

Since bike-sharing systems (BSS) were introduced in the 1960s (Laporte et al., 2018), their significance and scale have grown immensely in contemporary urban environments. They 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 DeMaio, 2009 and Eren and Uz, 2020 for historical and contemporary evidence of these benefits of BSS). The fascination for BSS within scientific literature is profound, evidenced by studies that draw parallels between the systems' requirements and human needs, akin to Maslow's renowned hierarchy of needs (Reggiani et al., 2022). Furthermore, evidence that some BSS can fail quite spectacularly in some cities (BBC, 2021) demonstrates an ongoing need for studies detailing system requirements.

A pivotal aspect of ensuring BSS viability consists of effectively managing SLR. In this regard, the operations research literature has significantly contributed by identifying a plethora of SLR-related goals and problems over the past two decades, dating back to the work of Benchimol et al. (2011), thought to be the first article to formulate a mathematical program for the bicycle repositioning problem. These intrinsic challenges span various known combinatorial optimization problems, including service network design (Neumann-Saavedra et al., 2020), vehicle routing (Bruck et al., 2019), inventory management (Datner et al., 2019), and numerous others. Furthermore, the

mathematical programming community continually strives to develop and refine models and solution methodologies to address and improve upon these recognized problems.

Despite this interest in SLR, state-of-the-art methods struggle to cope with the dimensions posed by contemporary BSS. For example, major North American urban centers boast expansive BSS landscapes, incorporating thousands of stations and bicycles, and nearly 100,000 daily trips. Managing these systems involves intricate decision-making that encompasses bicycle repositioning strategies, pickup and delivery specifics, and strategic allocation of rebalancing vehicles.

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 variations of the inventory routing problem (Schuijbroek et al., 2017; Ghosh et al., 2017; Bruck et al., 2019) and vehicle routing problems (Dell'Amico et al., 2018). The former differs from traditional vehicle routing problems in that it imposes only one rebalancing visit per station. It contrasts with inventory management, which overlooks vehicle route design. Many of the above references also integrate additional BSS features such as operational timelines, which entail fine-graining the intervals at which rebalancing operations are necessary into minutes or hours. Researchers tackle these challenges with methods like the L-shaped method, Lagrangian dual decomposition, and heuristics. However, even for 50–150 stations, the resulting problems are so complex that only a few optimal solutions are reported using these methods.

An exception is the work by Freund et al. (2022), which addresses real-life-sized US-based BSS but does not consider the rebalancing of bicycles by a capacitated vehicle fleet. The authors' problem is to determine the initial quantity of bikes and empty docks at each station that will minimize events where users cannot find a bicycle or an empty dock at a given station (referred to as out-of-stock events). Following their recommendations, a pilot program was implemented in New York's BSS, which involved relocating 34 docks between 6 stations. Using real data from April 2018, the authors estimated that the 34 relocations reduced monthly out-of-stock occurrences by 831 to 1,121.

The latest heuristic methods also struggle with real-life scenarios. For example, in their research, Cavagnini et al. (2024) utilized a mathematical heuristic to solve instances based on San Francisco's Bay Wheels BSS, which had 340 stations. However, in 2023, Bay Wheels had over 500 stations according to peak-usage summer data taken from GBFS.

The size disparity between real-life-sized BSS and the solved instance sizes in the literature illustrates the limitations of state-of-the-art methods in handling real-sized BSS, restricting their practical application and managerial insights. We detail the sizes of the BSS used in our study in Table 1. For each BSS, we took real-life data from its curated data and GBFS endpoints for July 2023, considered the busiest summer month for BSS in northern hemisphere cities. The following columns are shown: 'Q_{tot}' denotes the available inventory of bicycles, 'N' denotes the number of stations in the BSS, '$\sum_{i \in N} h_i$' denotes the sum of station bicycle capacities h_i, and 'Trips[8,22h]' denotes the number of trips in the interval from 8:00 AM to 10:00 PM. Additionally, two Key Performance Indices (KPI) are included as metrics: 'Trips/Day' denotes the average system usage, and 'Trips/Station&Day' represents the average station usage. These metrics are calculated as trips divided by stations and days in July 2023 (31 days).

Table 1 shows that Mexico City, Montreal, and New York are the largest BSS based on the 'Trips/Station&Day' KPI. This metric helps identify BSS with high trip demand without overburdening the system. For instance, Chicago has a high 'Trips/Day' but a low 'Trips/Station&Day,' suggesting underutilization. Our findings in Section 7 support this observation. From this table, especially from the number of stations, we can observe that the instances constructed and solved in the literature that is presented in this section are small compared to these real-life sizes.

One reason behind the literature's struggle to handle real-life sizes is the limited emphasis on

City	BSS	Q_{tot}	N	$\sum_{i \in N} h_i$	Trips[8,22h]	Trips / Day	Trips / Station & Day
Boston	Blue Bikes	4,000	423	7,316	299,131	9,649.4	22.8
Chicago	Divvy Bikes	16,500	1,677	15,302	473,233	15,265.6	9.1
Madrid	BiciMAD	7,500	610	26,444	235,618	7,600.6	12.5
Mexico City	Ecobici	6,800	661	17,447	856,778	27,638.0	41.8
Montreal	Bixi	10,000	800	17,580	1,293,849	41,737.1	52.2
New York	Citi Bike	32,000	2,175	67,365	2,983,001	96,225.8	44.2
San Francisco	Lyft - Bay Wheels	70,000	554	12,196	145,195	4,683.7	8.5
Toronto	Bike Share Toronto	9,000	791	15,063	585,504	18,887.2	23.9
Washington DC	Capital Bike Share	6,000	738	12,681	343,647	11,085.4	15.0

Table 1: Summary of Real-World Data for Major BSS, July 2023.

developing efficient models for the sub-problems inherent in bicycle repositioning problems. While Warrington and Ruchti (2019) identified a min-cost flow sub-problem within a broader bicycle repositioning framework, their approach did not involve solving via exact methods. Typically, the literature does not delve into such detailed sub-problem resolutions. Consequently, when aiming for exact methods—such as employing decomposition approaches—the sub-problem resolution necessitates solving an NP-hard integer program. Moreover, when the master problem embedded in a decomposition approach is an NP-hard problem, such as designing minimum-cost rebalancing vehicle routes, the manageable size of solvable instances diminishes considerably.

Additional literature pertinent to our work focuses on the optimal restocking policy for vehicle routing with stochastic demands (VRPSD) (Salavati-Khoshghalb et al., 2019). Surprisingly, this policy remains unimplemented in BSS literature despite its simplicity and potential to reduce vehicle fleet travel distance while minimizing unsatisfied station requirements. In contrast, both theoretical and practical aspects of this policy and its variants have received extensive attention in VRPSD studies (Gendreau et al., 2016; Florio et al., 2021).

This paper proposes a two-step approach to efficiently manage SLR in real-life-sized BSS. Our approach is based on the two-step method introduced by (Schuijbroek et al., 2017). It involves calculating desired bicycle quantity intervals at each station (step one) and formulating and solving a mixed integer program to route rebalancing vehicles that minimizes deviations of bicycle quantities from the intervals within a specified time horizon (step two). We start by introducing a novel problem denoted as 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 (Benders, 1962). The sub-problem is modeled as a min-cost flow problem, enabling us to utilize specialized solvers that implement polynomial algorithms. In the second step, we design vehicle routes to rebalance the bicycle quantities, from the quantities initially given up to 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 station rebalancing, thus enhancing our ability to tackle larger BSS instances. We direct the reader to Table 1 in the study by Cavagnini et al. (2024) for a comprehensive list of other examples of two-step approaches in the literature.

3 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 an independent Poisson process.
2. The trips 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 demand for trips.

While these assumptions may not fully reflect the complexities of BSS operations, they do not significantly impact our SLR maximization framework. The first two assumptions, based on earlier work by Schuijbroek et al. (2017), states that a trip is the fundamental demand unit and that multiple trips are allowed. A trip occurs when a user picks up and returns a bicycle, requiring both a bicycle and a dock to be available.

The third assumption is that trips happen continuously at varying rates. Based on real-life data in Table 1, we propose determining bicycle quantities at each station by 10:00 PM, before the demand surge at 8:00 AM the next day. We define the rebalancing process as the interval between setting these quantities and the next day's demand, assuming negligible demand between 10:00 PM and 8:00 AM.

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. Alternatively, demand at a station may be low if users know it is often empty, but their reaction to the sudden availability of bicycles is uncertain. 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 reader to Goto et al. (2004) for examples of the complexities 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. For instance, in the case where a station is often empty, we assume that the demand will remain constant even if there is a sudden increase in the availability of bicycles.

4 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 4.1, and the complete problem formulation is presented in Section 4.2. Section 4.3 presents the exact decomposition method to solve the problem. Five different models are presented in Sections 4.4–4.6, and a generalized problem to optimize station capacities is presented in Section 4.7.

4.1 A Scenario Formulation for the TLB

For each scenario ω of the TLB, the number of trips can be approximated by solving 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

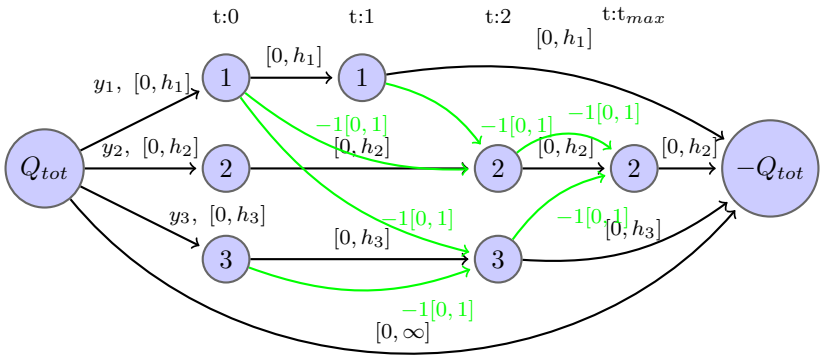


Figure 1: Min-Cost Flow Formulation $G^\omega = (V^\omega, A^\omega)$ for one scenario of the TLB with $n = 3$ trips and $t_{\max} = 4$.

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, 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 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 $y_i,\ i = \{1, 2, 3\}$.

4.2 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 4.1. Let $e = (pick_e, del_e, start_e, end_e)$ denote a trip, where $pick_e,\ del_e$ 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:

$$(TLB)\ \max \sum_{\omega \in \Omega} \mathbb{P}(\omega) \sum_{e \in E_\omega} v_e \tag{1}$$

$$\text{s.t.} \quad \sum_{i \in N} y_i \leq Q_{\text{tot}}, \tag{2}$$

$$f_{i0\omega} = y_i, \quad i \in N,\ \omega \in \Omega, \tag{3}$$

$$f_{it\omega} = f_{it-1\omega} + \sum_{r \in U^\omega(i,t)} v_e - \sum_{e \in O^\omega(i,t)} v_e, \quad i \in N,\ t \in T \setminus \{0\},\ \omega \in \Omega, \tag{4}$$

$$0 \leq y_i \leq h_i, \text{ and integer}, \quad i \in N, \tag{5}$$

$$0 \leq f_{it\omega} \leq h_i, \quad i \in N,\ t \in T,\ \omega \in \Omega, \tag{6}$$

$$0 \leq v_e \leq 1, \quad e \in E^\omega,\ \omega \in \Omega. \tag{7}$$

The objective in (1) maximizes the expected number of satisfied trips in the sample. Constraint (2) ensures that the quantity of bicycles used by the targets does not exceed the available number of bicycles. Constraints (3)–(4) are flow conservation constraints. Lastly, constraints (5)–(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:

$$UB = \sum_{\omega \in \Omega} \mathbb{P}(\omega) c^\omega u^{*\omega}. \tag{8}$$

Lastly, the problem formulated in (1)–(7) can be referred to as a sample average approximation (SAA) problem (Shapiro, 2003). 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 (Kleywegt et al., 2002). We evaluate the quality of our solutions using large samples in Section 7.

4.3 The Benders Decomposition Algorithm

To solve the TLB using Benders decomposition, we introduce a continuous variable $\theta > 0$ as an underestimator of the expected recourse cost. The master problem includes the first-stage decisions, the recourse underestimator, and optimality cuts generated on the fly whenever a feasible first-stage solution is found, following the branch-and-check method by Thorsteinsson (2001).

We begin by formulating the TLB as a two-stage mathematical program whose objective is to maximize the expected recourse function $\mathcal{Q}(y)$ subject to constraints (2) and (5). To calculate $\mathcal{Q}(y)$, 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, we adjust the demand at the source to $Q_{tot} - \sum_{i \in N} y_i^\nu$. Second, we 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^ω, $u_1^{*\omega}$ 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:

$$\mathcal{Q}(y^\nu) = \sum_{\omega \in \Omega} \mathbb{P}(\omega) c_1^\omega u_1^{*\omega}. \tag{9}$$

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

$$\begin{aligned}
&\text{(M1)} \ \max \theta && (10)\\
&\quad \text{s.t.} \quad (2), (5),\\
&\qquad \theta \leq UB, && (11)\\
&\qquad \text{Optimality cuts},\\
&\qquad \theta \geq 0. && (12)
\end{aligned}$$

We refer to the model in equations (2),(5),(10)–(12), as TLB Model 1 or simply M1.

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 $(i,j) \in A_1^\omega \backslash A_1^\omega(N)$, let $\gamma_{ij}^{*\omega}$ be the optimal values of the dual variables associated with the upper capacity constraints of all arcs in $A_1^\omega \backslash A_1^\omega(N)$, 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:

$$\begin{aligned}
\theta \leq \sum_{\omega \in \Omega} \mathbb{P}(\omega) \Bigg(& Q_{\text{tot}} \big(\pi(\text{source}) - \pi(\text{sink})\big) + \sum_{i \in N} y_i \big(\gamma_i^{*\omega} - \epsilon_i^{*\omega}\big) \\
& + \sum_{(g,j) \in A_{1E^\omega}^\omega} \gamma_{gj}^{*\omega} + \sum_{(g,j) \in A_1^\omega \backslash \left(A_{1E^\omega}^\omega \cup A_1^\omega(N)\right)} h_i \gamma_{gj}^{*\omega} \Bigg)
\end{aligned} \tag{13}$$

To improve the quality of the generated cut and take advantage of the min-cost flow structure of our subproblem, we attempted the Pareto-optimal cut separation method introduced by Magnanti et al. (1986). Their approach transforms the min-cost flow problem into a graph with much higher capacities and demands, which slows down the resolution. Unfortunately, results obtained with this technique did not improve over those obtained with the traditional Benders cut. For this reason, we opt out of using the approach.

4.4 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 (Rahmaniani et al., 2020). To address this, we propose a second model, called M2. The idea for this model comes from the observation that having too few or too many bikes at a station can reduce the number of satisfied trips. In a preprocessing phase, we compute the minimal and the maximal amount of bikes at each station and for each scenario ω that do not reduce the best number of trips that can be satisfied that scenario. Variables are then introduced to penalize the deviation from those quantities. Additionally, we can compute the global minimal and the maximal amount of bikes in the network so that the best number of trips that can be satisfied a scenario is not reduced. These variables convey additional information from the second-stage model to the first-stage model to help the convergence.

Let lb_i^ω and ub_i^ω represent the lower and upper bounds of the bicycle quantity at station i in scenario ω, and lb^ω and ub^ω denote the lower and upper bounds on the total number of bicycles required in scenario ω. We perform the following calculations to get those bounds. For scenario ω, we modify G^ω to get lb^ω and ub^ω as follows. The cost of the trip arcs is set to $-M$, with M being a high value, and the cost of source-to-sink arc is set to 1. The incentive for the min-cost flow problem is to serve as many trips as possible, while sending a minimal amount of bikes at the stations. Next, the min-cost flow problem is solved, and the flow from the source to stations gives the lower bound lb^ω. To compute ub^ω, we repeat the procedure with the source-to-sink arc cost of -1 instead of 1.

We perform similar calculations for lb_i^ω and ub_i^ω of station i and scenario ω. The cost of the arc from source to station i is set to 1, the source-to-sink arc cost is set to 0, and the trip arc cost is set to $-M$. We solve the min-cost flow problem and the flow from the source to the station i is lb_i^ω. For ub_i^ω, we repeat the procedure with the a cost from source to station i of -1.

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:

$$\text{(M2)} \quad \max \ \theta \tag{14}$$

$$\text{s.t.} \quad (2), (5), (12), (13),$$

$$\sum_{i \in N} y_i + l^\omega \geq lb^\omega, \quad \omega \in \Omega, \tag{15}$$

$$\sum_{i \in N} y_i - m^\omega \leq ub^\omega, \quad \omega \in \Omega, \tag{16}$$

$$\theta \leq UB - \sum_{\omega \in \Omega} \mathbb{P}(\omega)(l^\omega + m^\omega), \tag{17}$$

$$y_i + l_i^\omega \geq lb_i^\omega, \quad \omega \in \Omega,\ i \in N, \tag{18}$$

$$y_i - m_i^\omega \leq ub_i^\omega, \quad \omega \in \Omega,\ i \in N, \tag{19}$$

$$\theta \leq UB - \sum_{\omega \in \Omega} \mathbb{P}(\omega) \sum_{i \in N} (l_i^\omega + m_i^\omega), \tag{20}$$

$$l^\omega,\ m^\omega \geq 0, \quad \omega \in \Omega, \tag{21}$$

$$l_i^\omega,\ m_i^\omega \geq 0, \quad \omega \in \Omega,\ i \in N. \tag{22}$$

Constraints (15)–(16) bind the total bicycles used in a scenario, and constraints (18)–(19) bind the bicycles used by the individual stations in different scenarios. Constraints (17) and (20) bind the expected recourse cost to the violation variables l^ω, m^ω, l_i^ω and m_i^ω. Constraints (21)–(22) are definitions of variables.

Finally, the computations of those bounds requires solving $2|\Omega|(N+1)$ min-cost flow problems. This can take a significant amount of time for large problems. Also, it is 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.

4.5 A Second Variant Formulation of the TLB

Solving M2 can be computationally expensive due to the requirement to compute all station and scenario bounds. 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:

$$
\begin{aligned}
&\text{(M3)}\ \max\ \theta && && (23)\\
&\quad \text{s.t.}\quad (2), (5), (12), (13), && &&\\
&\qquad y_i + l_i \geq \max_{\omega\in\Omega}\ lb_i^{\omega}, && i \in N, && (24)\\
&\qquad y_i - m_i \leq \min_{\omega\in\Omega}\ ub_i^{\omega}, && i \in N, && (25)\\
&\qquad \theta \leq UB - \sum_{i\in N} (l_i + m_i), && && (26)\\
&\qquad l_i,\ m_i \geq 0. && && (27)
\end{aligned}
$$

Constraints (24)–(25) bind the total bicycles used to a single upper and lower bound for each station. Constraint (26) binds the expected recourse cost to the violation variables l_i and m_i. Constraints (27) 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 $G^{\omega},\ \forall\omega\in\Omega$. 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.

4.6 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:

$$
\begin{aligned}
&\min_{y} \sum_{i\in N} (y_i - 0.5h_i)^2 && (28)\\
&\qquad \text{s. to } (2), (5).
\end{aligned}
$$

The program in (28), (2), (5) is quadratic and can be solved by 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 for $i = 1, \ldots, n$, follows:

$$
H_i^{M4}(z) = \min_{0\leq y_i \leq \min\{z, 0.5h_i\}} \Big\{ (y_i - 0.5h_i)^2 \quad + H_{i+1}^{M4}(z - y_i) \Big\}, \tag{29}
$$

with the boundary condition of $H_i^{M4}(z) = 0$, for $i = n + 1$. The minimum cost is obtained by calculating $H_1^{M4}(Q_{tot})$ and the time complexity is $O(n \sum_{i=1}^{n} h_i)$.

4.7 Optimizing the Station Capacity

This section presents an approach for optimizing 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:

$$
\begin{aligned}
\text{(M5)}\ \max\ & \theta && (30)\\
\text{s.t.}\ & (2), (12), \\
& \sum_{i \in N} h_i^+ \leq B, && (31)\\
& 0 \leq y_i - h_i^+ \leq h_i, && (32)\\
& h_i^+,\ y_i\ \geq 0,\ \text{and integer}, && (33)\\
& \text{Optimality Cuts.}
\end{aligned}
$$

Equation (30) aims to maximize the expected number of satisfied trips. Inequality (31) represents the budget constraint, while inequality (32) is a generalization of inequality (5). Inequalities (33) 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 begin by constructing 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. Next, the separation of the cut is performed with a procedure similar to that of model M1.

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 $h_i + B$. The rest of the graph remains the same, and UB^{SCP} is given by averaging the scenario optimal solutions.

We propose two variants for M5, similar to the variants of M1 concerning station and scenario bounds. The first variant, Model 6 (M6), adds to the constraints in (31)–(33) the station scenario bound constraints from M2. The second variant, Model 7 (M7), applies a similar logic by adding the constraints in M3.

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.

5 The Bicycle Rebalancing Problem

This section presents the Bicycle Rebalancing problem (BRP). Section 5.1 presents the problem formulation, while the policies to compute the cost of a route are given in sections 5.3 and 5.4. Lastly, Section 5.6 presents an heuristic solution method. Throughout this section, we follow the notation of the TLB given in Section 4, while introducing new notation as needed.

5.1 Problem Definition

The BRP is defined as the problem of designing vehicle routes that start and end at the depot, to 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 4, 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 satisfy the conditions $|q_i| \leq h_i$ and $|q_i| \leq Q$ for all $i \in V$.

5.2 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 as the continue-to-next (CN) policy and the restocking trips (RT) policy, and they are presented in Sections 5.3 and 5.4.

5.3 The Continue-to-Next Policy

The cost computation for a route $r = \{0, i_1, \ldots, i_t, 0\}$ 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 $H_k^C(x_k)$ for $k = 1, \ldots, t$, as follows:

$$H_k^C(x_k) = \min_{\substack{-\bar{w}_k^- \leq w_k \leq \bar{w}_k^+ \\ 0 \leq x_k + w_k \leq Q - q_k}} \Big\{ \delta |w_k| \quad + H_{k+1}^C(q_k + x_k + w_k) \Big\}, \tag{34}$$

with the boundary condition $H_k^C(x_k) = 0$ for $k = t + 1$. 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 $w_{max} = \max_{1 \leq k \leq t} \{\bar{w}_k^-, \bar{w}_k^+\}$.

5.4 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 $k = 1, \ldots, t$, given by the following DP algorithm:

$$H_k^R(x_k, d_k) = \min \Bigg\{ r_k + \min_{\substack{-\bar{w}_k^- \le w_k \le \bar{w}_k^+ \\ 0 \le x_k + w_k \le Q - q_k}} \{\delta |w_k|\} \quad + \min_{0 \le y \le Q} \{H_{k+1}^R(y, d_k - r_k)\}, \\ \min_{\substack{-\bar{w}_k^- \le w_k \le \bar{w}_k^+ \\ 0 \le x_k + w_k \le Q - q_k}} \Big(\delta |w_k| + H_{k+1}^R(q_k + x_k + w_k, d_k) \Big) \Bigg\}. \tag{35}$$

with the boundary condition $H_k^R(x_k, d_k) = 0$ for $k = t+1$.

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 $\sum_{k=0}^{t} d_{i_k i_{k+1}} + d_{i_t 0} + \min_{0 \le x_0 \le Q} H_1^R(x_0, D)$.

5.5 Feasibility of a Route

Under both cost policies described in Section 5.2, a route may be infeasible depending on the sequencing of the stations and the vehicle's remaining autonomy. In this section, we specifically focus on the CN policy to propose conditions that determine whether a given route of stations is feasible.

Let $r = \{0, i_1, \ldots, i_t, 0\}$ 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:

$$\sum_{\substack{i \in S \\ q > 0}} q_i \le Q + \sum_{i \in S} \bar{w}_i^+ + \sum_{\substack{i \in S \\ q_i < 0}} |q_i|, \tag{36}$$

$$\sum_{\substack{i \in S \\ q_i < 0}} |q_i| \le Q + \sum_{i \in S} \bar{w}_i^- + \sum_{\substack{i \in S \\ q_i > 0}} q_i, \tag{37}$$

Equations (36), (37) 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 Parada et al. (2024).

5.6 Solution Method

To solve the BRP, we implement the Adaptive Large Neighborhood Search (ALNS) metaheuristic as proposed by Ropke and Pisinger (2006). 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_{H^C} 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:

$$L_{H^C} = \delta \max\{0, |\sum_{i \in S} q_i| - Q\}. \tag{38}$$

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 (38) 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 $k+1, \ldots, t$, 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 7.2.

6 Data and Instance Characterization

This section outlines the sources of real-life BSS data and our benchmark instances for the TLB and BRP. Section 6.1 details the data, and Section 6.2 describes the generated instances.

6.1 Real-Life-Sized Data

BSS data comes from two main sources: JSON schemas and historical trip data. The General Bikeshare Feed Specification (GBFS) standardizes real-time information about BSS, including station locations and bicycle availability. Operators of systems or city administrators also provide historical trip data.

We collected data from nine BSS: Blue Bikes (Boston, MA), Divvy Bikes (Chicago, IL), BiciMAD (Madrid, Spain), Ecobici (Mexico City, Mexico), BIXI (Montreal, Canada), Citi Bike (New York, NY), Bay Wheels (San Francisco, CA), Bike Share Toronto (Toronto, Canada), and Capital Bike Share (Washington, D.C.). We note that for Madrid two thirds of the trips are missing some information and had to be discarded. Thus, the results for Madrid should be taken with a grain of salt.

An exploratory analysis of JSON data and historical trips revealed significant differences in size and usage among nine BSS. From April to July 2023, Citi Bike, Bixi, and Ecobici had the highest number of trips, each exceeding three million. In contrast, the BSS in Washington, D.C., and Toronto recorded fewer than one million trips each. Notably, Citi Bike (New York) had more than ten million trips during this period.

We analyzed data from July 2023, the busiest month for the BSS in our study, to ensure comparability. Figure 2 shows the average number of trips for the three largest BSS and Boston. The average trips in Figure 2 are based on hourly counts between 8:00 AM and 10:00 PM, averaged over 31 days.

The data depicted in Figure 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 Boston experience peak usage between 4:00 PM and 7:00 PM, 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 7:00 PM. In terms of size, New York

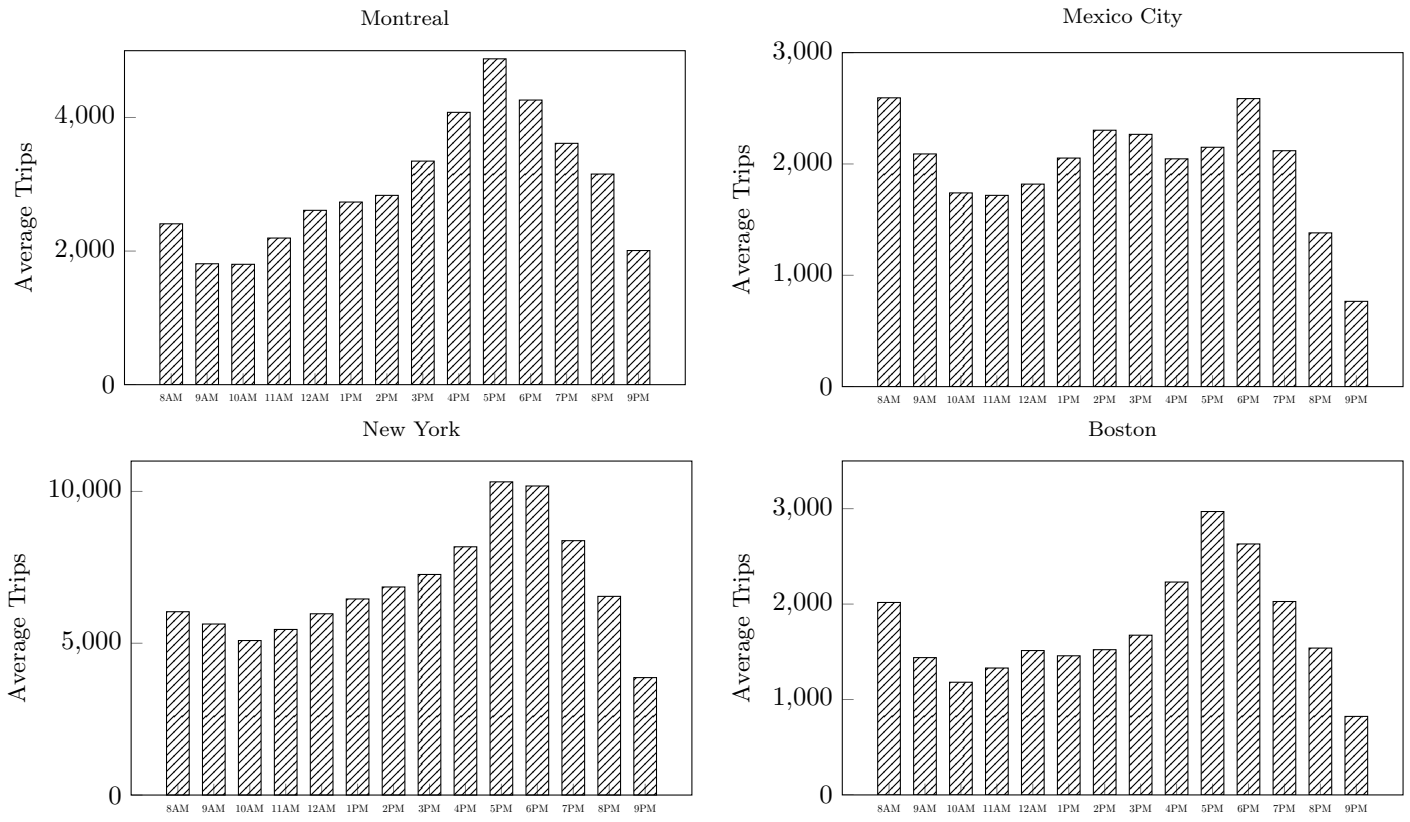


Figure 2: Histograms of Average Daily Trips from 8AM to 10PM for some BSS in July 2023.

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
Montreal	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

Table 2: Summary of Instances with $|\Omega| = 100$ for the TLB.

shows significant demand during its low-demand hours from 8:00 PM to 10:00 PM, exceeding the peak demand in Mexico City at any time of the day and matching many peak hours in Montreal. Specifically, New York experiences demand for more than ten thousand trips between 5:00 PM and 6:00 PM. Montreal's peak occurs at nearly five thousand trip demands at 5:00 PM. In comparison, Mexico City peaks at less than three thousand trips at various hours throughout the day.

6.2 Characterization of the Instances

In this section, we outline the instances generated for the TLB and the BRP.

For the TLB, we generate five instances, with 15, 25, 50, 75, and 100 scenarios, for each of the nine cities, totaling 45 instances. To generate each instance we sampled random variables for each scenario, allowing us to simulate independent Poisson processes for each trip.

Table 2 provides an overview of the largest instances (with 100 scenarios), detailing metrics such as the number of real-life stations (N), contractual total available bicycles (Q_{tot}), and the sizes of scenario graphs G^{ω}, represented by the average number of MCF arcs and nodes. Regarding the contractual number of bicycles is the theoretical quantity publicly defined by either the BSS or city operators; however, in practice, many BSS use a fraction of this daily. Such a quantity is not publicly available.

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 2, the instances of New York, Montreal, and Mexico City are the largest.

Additionally, we generated a 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. For this problem, we solved the TLB to get the initial quantities with a Q_{tot} equal to the total sum of bicycle quantities that are currently in the network to avoid high disparities between what is currently in use and the contractual quantities.

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 (MontrealGazette, 2015). 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. 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. Additionally, we remove nodes with $q_i = 0$ from each instance, as they were not significantly used as transshipment nodes.

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
Montreal	800	762	100
New York	2,175	2,111	119
San Francisco	554	528	49
Toronto	791	724	100
Washington DC	738	693	149

Table 3: Summary of Instances for the BRP.

Table 3 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.

7 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 7.1 and 7.2 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:

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

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

7.1 Results for the TLB

We present the results for the TLB models M1–M4 in Section 7.1.1. Additionally, we present the results of changing the number of available bicycles in Section 7.1.2, and the results for the OSC models M5–M7 in Section 7.1.3.

7.1.1 Results for Models M1-M4

The results of solving models M1–M3 are shown in Figure 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, 25, 50, 75, and 100). The unbiased estimators were calculated as the average cost of a TLB model solution in $G^\omega,\ \forall \omega \in \Omega_{400}$, using the Benders decomposition algorithm with a time limit of 1,800 seconds. UB_{400} is the average optimal cost of $G^\omega,\ \forall \omega \in \Omega$. 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.

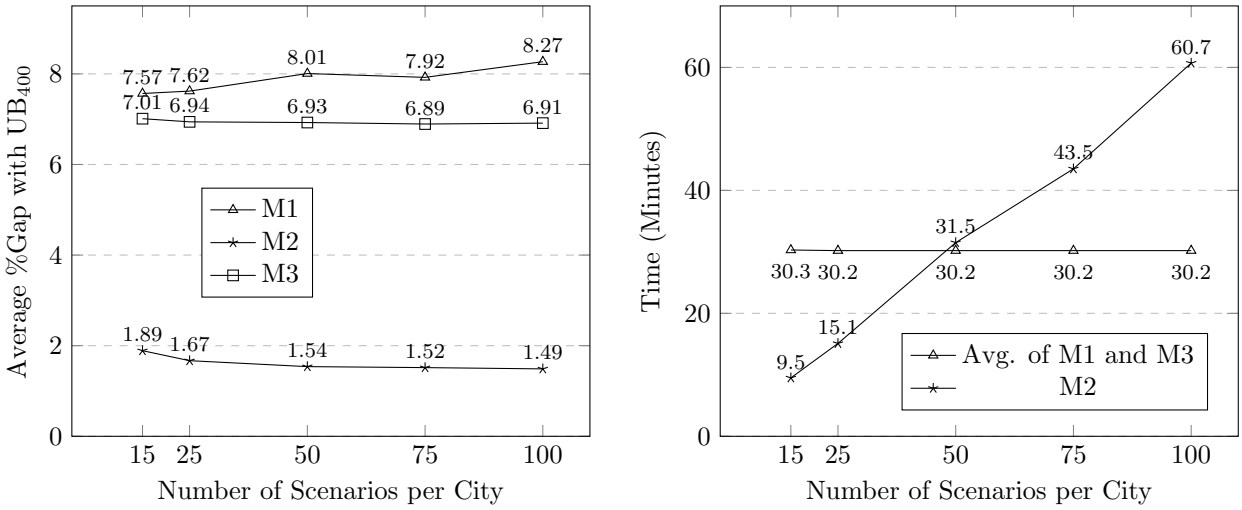


Figure 3: Comparison of results for Models M1-M3 with different scenarios: (a) Average % Gap with UB_{400} and (b) Average time taken

Figure 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 5.

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 4.

Table 4 presents a detailed comparison of the Benders decomposition algorithm for models M1–M3 for 100 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 4 confirms M2's superiority over M1 and M3, as it is the only one of the 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 5 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 or optimal solution at the end of

	M1				M2				M3			
City	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
Montreal	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

Table 4: Performance of Models M1-M3 in Cities with 100 Scenarios.

		%Gap				Bounds Time[s]	
City	UB_{400}	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
Montreal	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		

Table 5: %Gaps with UB_{400} of Models M1-M4 Using 100 Scenarios

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 (29). 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 Montreal, 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 7.1.2.

7.1.2 Results with Fewer Available Bicycles.

This section presents the result of varying the contractual number of 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 (2) to the following form:

$$\sum_{i \in N} y_i \leq \phi Q_{tot}, \tag{39}$$

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.

	% of Available Bicycles (ϕ)									
City	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
Montéal	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

Table 6: Average Percentage of Satisfied Trips with Fewer Available Bicycles

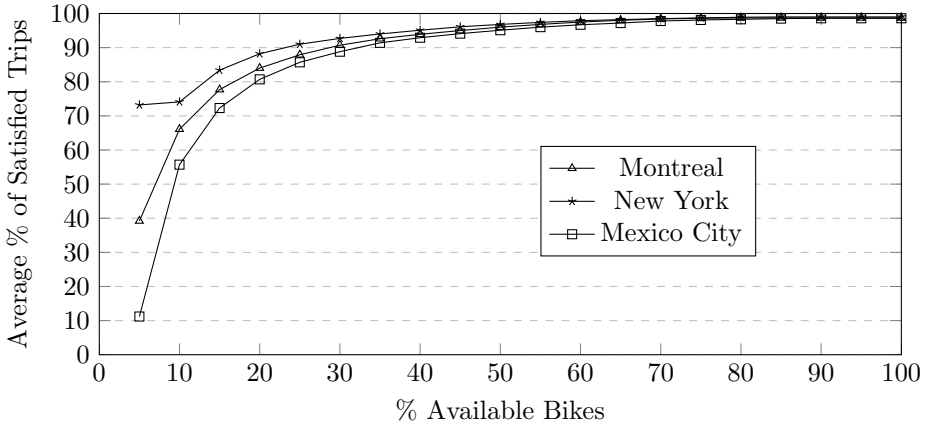


Figure 4: Montreal, New York, and Mexico City average percentage of satisfied trips with fewer available bicycles

Table 6 explores how changing Q_{tot} affects the average satisfied trips. The columns under '% of Available Bicycles' show the the number of satisfied trips for each provided bicycle percentage, corresponding to the ϕ parameter in equation (39). Then, for each of the following ϕ values 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}. This average is 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 Figure 5.

The table shows that the contractual number of bicycles is significantly higher than the required quantity. With just 50% of contractual 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 bicycles being used. Chicago is still more noteworthy, with just 10% of 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 contractual bicycles, as shown in Figure 4. The figure includes more data points than Table 6, 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 contractual bicycles per 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.

	Budget as a % of $\sum_{i \in N} h_i$ (B)									
City	$\sum_{i \in N} h_i$	0	1	2	3	4	5	10	15	20
Boston	7,316	0.0	1.3	1.7	1.8	2.0	2.1	2.7	2.8	3.5
Chicago	15,302	0.0	0.3	0.3	0.6	0.8	0.7	0.8	0.9	1.2
Madrid	26,444	0.0	0.1	0.1	0.1	0.2	0.2	0.2	0.2	0.2
Mexico City	17,447	0.0	0.8	1.5	1.6	1.9	2.1	2.5	2.9	3.0
Montreal	17,580	0.0	0.0	0.0	0.3	0.0	0.3	0.6	0.9	1.1
New York	67,365	0.0	0.3	0.1	0.0	0.0	0.1	0.5	0.7	1.0
San Francisco	12,196	0.0	0.2	0.4	0.5	0.5	0.5	0.3	0.4	0.6
Toronto	15,063	0.0	0.2	0.3	0.5	0.6	0.8	1.3	1.7	1.9
Washington DC	12,681	0.0	0.2	0.3	0.5	0.7	0.8	1.3	1.5	1.6
Avg		0.0	0.4	0.5	0.7	0.7	0.8	1.1	1.3	1.6

Table 7: Increase in Average Satisfied Trips for a Given Budget

.

7.1.3 Results for the SCP

This section presents the results for the SCP in Table 7. The table shows the increases in average satisfied trips for the solution of model M6 compared to the solution of 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 table also shows the sum of current station capacities $\sum_{i \in N} h_i$.

The data indicates that, in general, M6 improves upon M2 in nearly all cities and budgets. A closer inspection of the table suggests that there are BSS where targeted investments in increased capacity could lead to better service than others. Such is the case in Mexico City and Boston, where, with relatively low budgets, the system can serve more trips. On the contrary, the table shows that neither Montréal nor New York, for example, attain such gains, indicating that capacities have been perhaps more meticulously decided. It's also important to note that increasing station capacity does not affect the station's incoming and outgoing trip demands, as outlined in the fourth assumption of the SLR problem definition.

Lastly, results for M5 and M7 are not presented, as they exhibit similar behavior to those of M1 and M3. Furthermore, additional research is necessary to understand the implications of assumption 4 outlined in Section 3 and modifed station capacities, namely how variations in the initial station capacities can influence trip demands.

7.2 Results for the BRP

This section presents BRP results in Tables 8, and 9, obtained through the ALNS procedure outlined in Section 5.6.

Table 8 compares the CN and RT policies for a $\delta = 1$. 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 8 shows that the RT policy yields better solutions, evident from smaller average cost values compared to the CN policy. However, the RT policy requires significantly more computation time, especially in cities like Montreal and Mexico City. The effectiveness of the CN policy comes from the efficient feasibility checks described in Section 5.6. These checks make it easier to quickly verify if a route is feasible and, combined with the calculation of L_{H^C}, 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, especially when the instance allows long routes.

Table 9 provides an overview of the solution costs after the ALNS runs, contrasting the CN

	Continue-to-Next Policy					Restocking Trips Policy				
City	Cost	Dist.	Policy	Drv.	Time[s]	Cost	Dist.	Policy	Drv.	Time[s]
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
Montreal	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

Table 8: BRP Results with $\delta = 1$.

	δ											
UB^{ALNS}	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
Montreal	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

Table 9: BRP Results for All δ Values

and RT cost policies across various δ values, specifically $\delta = \{1, 5, 10, 15, 20, 25\}$. 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.

8 Conclusion

We propose a two-step approach to optimizing the redistribution of bicycles in bicycle-sharing systems by breaking Service-Level Requirements into the Target-Level Problem (TLB) and the Bicycle Rebalancing Problem (BRP). Our computational experiments show that Model 2 outperforms other TLB models, achieving optimal solutions and enabling substantial reductions in total bicycles without significantly impacting satisfied trips. Furthermore, Model 6 is the most effective for determining increases in station capacities that lead to a higher average number of satisfied trips.

In the BRP context, the restocking trips (RT) policy outperforms the continue-to-next (CN) policy by providing superior solutions in cities with complex networks despite requiring more computational time.

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.

References

BBC (2021). Why some bike shares work and others don't. `https://www.bbc.com/future/article/20210112-the-vast-bicycle-graveyards-of-china`. Accessed: April 17, 2024.

Benchimol, M., Benchimol, P., Chappert, B., De La Taille, A., Laroche, F., Meunier, F., and Robinet, L. (2011). Balancing the stations of a self service "bike hire" system. *RAIRO-Operations Research*, 45(1):37–61.

Benders, J. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik*, 4(1):238–252.

Beroud, B., Van Zeebroeck, B., and Peduzzi, E. (2024). Quel avenir pour le service public bruxellois de vélo en libre-service ? synthèse de l'étude préparatoire pour le vls de la région de bruxelles-capitale en 2026 : Benchmark et recommandations. page 30.

Bruck, B. P., Cruz, F., Iori, M., and Subramanian, A. (2019). The static bike sharing rebalancing problem with forbidden temporary operations. *Transportation Science*, 53(3):882–896.

Cavagnini, R., Maggioni, F., Bertazzi, L., and Hewitt, M. (2024). A two-stage stochastic programming model for bike-sharing systems with rebalancing. *EURO Journal on Transportation and Logistics*, 13:100140.

Datner, S., Raviv, T., Tzur, M., and Chemla, D. (2019). Setting inventory levels in a bike sharing network. *Transportation Science*, 53(1):62–76.

Dell'Amico, M., Iori, M., Novellani, S., and Subramanian, A. (2018). The bike sharing rebalancing problem with stochastic demands. *Transportation research part B: methodological*, 118:362–380.

DeMaio, P. (2009). Bike-sharing: History, impacts, models of provision, and future. *Journal of public transportation*, 12(4):3.

Eren, E. and Uz, V. E. (2020). A review on bike-sharing: The factors affecting bike-sharing demand. *Sustainable Cities and Society*, 54:101882.

Florio, A. M., Hartl, R. F., Minner, S., and Salazar-González, J.-J. (2021). A branch-and-price algorithm for the vehicle routing problem with stochastic demands and probabilistic duration constraints. *Transportation Science*, 55(1):122–138.

Freund, D., Henderson, S. G., and Shmoys, D. B. (2022). Minimizing multimodular functions and allocating capacity in bike-sharing systems. *Operations Research*, 70(5):2715–2731.

Gendreau, M., Jabali, O., and Rei, W. (2016). 50th anniversary invited article—future research directions in stochastic vehicle routing. *Transportation Science*, 50(4):1163–1173.

Ghosh, S., Varakantham, P., Adulyasak, Y., and Jaillet, P. (2017). Dynamic repositioning to reduce lost demand in bike sharing systems. *Journal of Artificial Intelligence Research*, 58:387–430.

Goto, J. H., Lewis, M. E., and Puterman, M. L. (2004). Coffee, tea, or ...?: A markov decision process model for airline meal provisioning. *Transportation Science*, 38(1):107–118.

Kleywegt, A. J., Shapiro, A., and Homem-de Mello, T. (2002). The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502.

Laporte, G., Meunier, F., and Calvo, R. W. (2018). Shared mobility systems: an updated survey. *Annals of Operations Research*, 271(1):105–126.

Magnanti, T. L., Mireault, P., and Wong, R. T. (1986). *Tailoring Benders decomposition for uncapacitated network design*, pages 112–154. Springer Berlin Heidelberg, Berlin, Heidelberg.

MobilityData (2024). General bikeshare feed specification (gbfs). Accessed: 2024-06-08.

MontrealGazette (2015). 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`. Accessed: April 17, 2024.

Neumann-Saavedra, B. A., Crainic, T. G., Gendron, B., Mattfeld, D. C., and Römer, M. (2020). Integrating resource management in service network design for bike-sharing systems. *Transportation Science*, 54(5):1251–1271.

Parada, L., Côté, J.-F., and Gendreau, M. (2024). An integer l-shaped method for the static stochastic bicycle repositioning problem. Technical Report CIRRELT-2024-26, CIRRELT.

Rahmaniani, R., Ahmed, S., Crainic, T. G., Gendreau, M., and Rei, W. (2020). The benders dual decomposition method. *Operations Research*, 68(3):878–895.

Reggiani, G., Salomons, A. M., Sterk, M., Yuan, Y., O'Hern, S., Daamen, W., and Hoogendoorn, S. (2022). Bicycle network needs, solutions, and data collection systems: A theoretical framework and case studies. *Case Studies on Transport Policy*, 10(2):927–939.

Ropke, S. and Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472.

Salavati-Khoshghalb, M., Gendreau, M., Jabali, O., and Rei, W. (2019). An exact algorithm to solve the vehicle routing problem with stochastic demands under an optimal restocking policy. *European Journal of Operational Research*, 273(1):175–189.

Santoso, T., Ahmed, S., Goetschalckx, M., and Shapiro, A. (2005). A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167(1):96–115.

Schuijbroek, J., Hampshire, R., and van Hoeve, W.-J. (2017). Inventory rebalancing and vehicle routing in bike sharing systems. *European Journal of Operational Research*, 257(3):992–1004.

Shapiro, A. (2003). Monte carlo sampling methods. In *Stochastic Programming*, volume 10 of *Handbooks in Operations Research and Management Science*, pages 353–425. Elsevier.

Shu, J., Chou, M. C., Liu, Q., Teo, C.-P., and Wang, I.-L. (2013). Models for effective deployment and redistribution of bicycles within public bicycle-sharing systems. *Operations Research*, 61(6):1346–1359.

Thorsteinsson, E. S. (2001). Branch-and-check: A hybrid framework integrating mixed integer programming and constraint logic programming. In Walsh, T., editor, *Principles and Practice of Constraint Programming — CP 2001*, pages 16–30, Berlin, Heidelberg. Springer Berlin Heidelberg.

Warrington, J. and Ruchti, D. (2019). Two-stage stochastic approximation for dynamic rebalancing of shared mobility systems. *Transportation Research Part C: Emerging Technologies*, 104:110–134.