The Robust Bike Sharing Rebalancing Problem: Formulations and a Branch-and-Cut Algorithm

Bruno P. Bruck[a], Walton P. Coutinho[b], Pedro Munari[c,*]

[a]*Federal University of Paraíba, Scientific Computing Department, Informatics Center, Rua dos Escoteiros, s/n, Mangabeira, 58055-000, João Pessoa-PB, Brazil*
[b]*Federal University of Pernambuco, Department of Technology, Av. Marielle Franco, s/n, km 59, Caruaru-PE, Brazil*
[c]*Federal University of São Carlos, Production Engineering Department, Rod. Washington Luís Km 235,13565-905, São Carlos-SP, Brazil*

Abstract

Bike Sharing Systems (BSSs) offer a sustainable and efficient urban transportation solution, bringing flexible and eco-friendly alternatives to city logistics. During their operation, BSSs may suffer from unbalanced bike distribution among stations, requiring rebalancing operations throughout the system. The inherent uncertain demand at the stations further complicates these rebalancing operations, even when performed during downtime. This paper addresses this challenge by introducing the Robust Bike Sharing Rebalancing Problem (RBRP), which relies on robust optimization techniques to improve rebalancing operations in BSSs. Very few studies have considered uncertainty in this context, despite it being a common characteristic with a significant impact on the performance of the system. We present two new formulations and a tailored branch-and-cut algorithm for the RBRP. The first formulation is compact and based on the linearization of recursive equations, while the second is based on robust rounded capacity inequalities and feasibility cuts. Computational results based on benchmark instances indicate the effectiveness of our approaches and highlight the benefits of using robust solutions to support decision-making in BSSs.

Keywords: Bike Sharing Systems, Vehicle Routing, Combinatorial Optimization, Robust Optimization

1. Introduction

Bike Sharing Systems (BSSs) are an excellent solution to improve urban mobility, offering a mode of transportation that is both economical and environmentally friendly. Implementing BSSs in large urban centers has the potential to alleviate heavy traffic and reduce pollution, yielding direct and indirect benefits to the surrounding population. Over the last two decades, these systems have spread worldwide and are becoming increasingly popular (Si et al. 2019). According to data from the Meddin Bike Sharing World Map (Meddin 2023), in August 2023, there were roughly 1940 BSSs in operation around the world. To maintain competitiveness

*Corresponding author
Email addresses: bruno.bruck@ci.ufpb.br (Bruno P. Bruck), walton.coutinho@ufpe.br (Walton P. Coutinho), munari@dep.ufscar.br (Pedro Munari)

and enhance customer service, some systems have integrated advanced technologies, including geolocation, internet of things, wireless payment and electric bikes.

In most BSSs, stations are spread all over the city, enabling users to rent and return bikes. Throughout the day, certain stations may experience fluctuations in bike availability, leading to surpluses or shortages, and occasionally, stations may become full or empty. As a consequence, rebalancing operations must be regularly performed to restore the desired inventory levels at each station. The Bike Sharing Rebalancing Problem (BRP) addresses the optimization of these rebalancing operations (Bruck and Subramanian 2023). In this problem, a fleet of capacitated vehicles is used to perform a series of pickups and deliveries to bring the system to a balanced state while minimizing routing costs. Each station must be visited exactly once, and the depot has the flexibility to supply or receive any quantity of bikes to and from the system (Dell'Amico et al. 2014, 2016). Notably, the BRP is a variant of the Vehicle Routing Problem (VRP), specifically belonging to the class of Pickup and Delivery Problems (PDPs), and therefore is $\mathcal{NP}$-hard.

While the literature on deterministic problems for rebalancing operations in BSSs is vast, surveys carried out by Laporte, Meunier, and Calvo (2018) and Si et al. (2019) reveal a limited number of research articles addressing this class of problems under uncertainty, despite its practical relevance. Indeed, uncertainty is an intrinsic factor in most real-world BSSs, primarily due to demand fluctuations. If not appropriately accounted for, this uncertainty can lead to ineffective or even infeasible rebalancing operations. However, Dell'Amico et al. (2018) are the only authors addressing demand uncertainty in the BRP thus far. They resort to Stochastic Programming (SP) and assume that the demand at each station follows a probability distribution represented by a finite set of scenarios.

In this paper, we introduce the Robust Bike sharing Rebalancing Problem (RBRP), an extension of the BRP that considers demand uncertainty via Robust Optimization (RO) (Ben-Tal, Ghaoui, and Nemirovski 2009, Bertsimas and Sim 2004). RO approaches do not require the use of probability distributions and allow us to model parameter variation using uncertainty sets that take into account the decision-maker's aversion to risk. The solutions provided by RO approaches are protected against variations within the uncertainty set, mitigating the risk of becoming infeasible or overly expensive during their execution. Hence, by relying on robust routes in rebalancing operations, we are likely to improve the effectiveness of BSSs.

To model and solve the RBRP, we propose two Mixed Integer Linear Programming (MILP) formulations and a specialized Branch-and-Cut (B&C) algorithm. Computational experiments performed on benchmark instances derived from real-world data show the advantages of applying RO in the context of bike rebalancing operations. Furthermore, we provide theoretical results regarding the proposed solution methods and highlight that the use of RO to address the BRP under uncertainty is a challenging and non-trivial task. Our main contributions can be summarized as follows:

- We formally introduce the RBRP under demand uncertainty. To the best of our knowledge, this is the first paper to introduce an RO approach for the BRP;

- We present a theoretical analysis regarding the worst-case demand realizations for the RBRP, which is a key component of our modeling strategy;
- We develop a compact formulation for the RBRP relying on decision variables that represent the worst-case loads after visiting each station. This formulation is obtained through the linearization of recursive equations, a technique successfully used to derive robust counterparts of vehicle routing problems (Munari et al. 2019). We are not aware of any other study using this linearization technique to incorporate uncertainties into PDPs, which is thus another relevant contribution of this article;
- We present a second formulation, based on robust rounded capacity inequalities and feasibility cuts. Based on this formulation, we develop a specialized B&C algorithm that further incorporates other types of valid inequalities;
- We discuss the potential benefits of relying on robust solutions to support decision-making in rebalancing operations. Through Monte Carlo simulations, we show that significant reductions in the risk of routes becoming infeasible in practice can be achieved without substantially increasing operational costs.

The remainder of this paper is organized as follows. In Section 2, we present the relevant literature related to the RBRP. In Section 3, we formally define the RBRP and describe how demand uncertainties are modeled. In Section 4, we propose two MILP formulations for the RBRP, while in Section 5 we describe the tailored B&C algorithm developed for this problem. In Section 6, we present the results of computational experiments using generated benchmark instances based on real-world BSSs' data. Finally, Section 7 presents the conclusions and future work directions.

2. Related literature

In this section, we provide an overview of the literature related to the RBRP. For more comprehensive surveys on shared mobility systems and bike sharing, we refer the reader to Laporte, Meunier, and Calvo (2018), Si et al. (2019) and Shui and Szeto (2020). In addition, Bruck and Subramanian (2023) presents an outline of the basic variants of the BRP and their fundamental characteristics. An extensive review of PDPs can be found in Battarra, Cordeau, and Iori (2014).

According to the classification scheme proposed by Berbeglia et al. (2007), rebalancing problems in BSSs fall within the category of *many-to-many* PDPs, as any station can serve as either the origin or destination for bikes within the system. The most elementary problem related to the BRP is known as the One-commodity Pickup-and-Delivery Traveling Salesman Problem (1-PDTSP), which is a generalization of the Traveling Salesman Problem (TSP) (Hernández-Pérez and Salazar-González 2004b). In this variant, a single capacitated vehicle is in charge of rebalancing the inventory of each customer to a predefined value by performing a series of pickups and deliveries along a single route of minimal cost. The vehicle is based

on a depot, which cannot provide nor receive goods from customers, only serving as a starting and ending point for the vehicle. Furthermore, each customer must be visited exactly once, even if their demand equals zero. The BRP is a generalization of the 1-PDTSP that considers multiple vehicles and for this reason, it is also known in the literature as the One-commodity Pickup-and-Delivery Vehicle Routing Problem (1-PDVRP) (Dell'Amico et al. 2014, 2016). Note that, as pointed out by Bruck and Subramanian (2020), the 1-PDVRP was first introduced by Dror, Fortin, and Roucairol (1998) and later studied by Gunes, van Hoeve, and Tayur (2010). In the latter study, the authors kept the same name and slightly changed the problem definition by including a maximum time constraint for each route. For other variants related to the BRP, we refer interested readers to Chemla, Meunier, and Calvo (2013), Salazar-González and Santos-Hernández (2015), Erdoğan, Laporte, and Calvo (2014), Erdoğan, Battarra, and Calvo (2015), Casazza et al. (2018), Bruck et al. (2019), Casazza, Ceselli, and Calvo (2021), Hernández-Pérez and Salazar-González (2022).

Rebalancing operations may be performed either during working hours or when the system is closed. Following the classification introduced by Pillac et al. (2013), the latter is typically modeled as a static deterministic problem (see, e.g., Cruz et al. 2017, Hernández-Pérez and Salazar-González 2018, Hernández-Pérez, Salazar-González, and Santos-Hernández 2018, Bulhões et al. 2018), assuming that the number of bikes to be collected or delivered is known and does not change after the route planning process. The former is mostly modeled as a dynamic deterministic problem, assuming that data is gradually revealed and routes may be redefined according to the realized values (see, e.g., Contardo, Morency, and Rousseau 2012, Regue and Recker 2014, Zhang et al. 2017). Most of the literature primarily focuses on static problems, given their wider applicability and computational tractability. As noted by Laporte, Meunier, and Calvo (2018) and Shui and Szeto (2020), performing rebalancing operations during the night is often more efficient due to several factors, including reduced traffic.

It is worth mentioning that independently of being performed during working hours or not, the number of bikes to be collected or delivered in each station may be nondeterministic. For example, routes may be defined in a tactical decision level based on demand estimates, as an information system that computes routes may not be available to find the best route for each specific day. In this case, even after the system is closed, drivers follow the same predefined routes every day. These routes may be preferable in practice, as they commonly bring several benefits to drivers regarding consistency and safety, and simplify the system's management. Therefore, it is relevant to consider possible demand deviations through different days, in a way to design routes that are likely to be feasible when executed. Notably, we may compute different routes for days with different demand patterns. Moreover, even when an information system is used to help compute routes for each specific day, the available data may still not be accurate, as the sensors and other technologies used in these stations may be defective, affected by errors, or misused.

The nondeterminism in the number of bikes in each station when the system is open is more evident, as users are picking up or returning bikes during the execution of the routes. Even if we assume that an information system is available to compute the best routes for

that day/shift and that the acquired data is completely accurate (which are both strong assumptions in practice), the number of bikes in each station is likely to change after the vehicle starts its route. Hence, considering these changes when designing the routes can be useful also in the operational decision level, to accommodate the changes that are natural in the system, while in operation. Approaches based on stochastic programming or robust optimization have been widely used in the VRP literature to address these types of situations, and are typically preferred because of their computational tractability (Gendreau, Jabali, and Rei 2014, Oyola, Arntzen, and Woodruff 2018, 2017). Nevertheless, very few authors have considered these approaches in the context of the BRP.

The lack of research in BRP variants under uncertainty has been pointed out by several authors over the last years (Laporte, Meunier, and Calvo 2018, Si et al. 2019, Shui and Szeto 2020, Bruck and Subramanian 2023). As mentioned above, BSSs often face uncertainty due to demand fluctuations, which makes it a relevant feature in the planning of rebalancing operations. To our knowledge, there is only one study addressing the BRP under uncertainty, and it is based on SP approaches. Namely, Dell'Amico et al. (2018) introduces the BRP with stochastic demands (BRPSD), in which the demand at each station follows a probability distribution that is represented by a finite set of scenarios. The problem is modeled as a two-stage SP problem, with a recourse function that penalizes the partially fulfilled demands in the scenarios. The authors propose new models, heuristics and several exact approaches including L-shaped and branch-and-cut methods. Computational experiments using real-world and randomly generated instances indicate that the solutions obtained with the proposed approaches reduce the chances of unmet demands in rebalancing operations.

Cavagnini et al. (2018) address a related problem using two-stage SP approaches as well, but concerning only the assignment of bikes to stations under demand uncertainty (i.e., no routing decisions). The authors use recourse actions consisting of rebalancing plans that result in different penalties added to the objective function, including congestion, starvation, fleet size and other performance measures. They propose a formulation and heuristic approaches to reduce the computational times of solving this formulation without compromising the quality of the solution. Maggioni et al. (2019) address a similar problem and propose two-stage and multistage SP models to determine the optimal number of bikes at each station.

All the aforementioned studies rely on the SP paradigm and, hence, assume that the uncertain demand can be effectively represented using probability distributions (**??**). Additionally, they represent these distributions using a limited number of scenarios that are randomly sampled from these distributions. Even though these scenarios may be created using real-world data, Yuan et al. (2019) observe that historical data from BSSs are often inaccurate or insufficient for proper demand forecasting. These observations indicate that other approaches may be more appropriate for handling demand uncertainty in BSSs.

Different from SP, RO approaches do not require the use of probability distributions to model uncertainty (Ben-Tal, Ghaoui, and Nemirovski 2009). The purpose is to generate solutions that remain feasible for any realization of the uncertain parameter belonging to an uncertainty set. There are different strategies to model this set, and the so-called cardinality-constrained set proposed by Bertsimas and Sim (2004) has become very popular

over the last decades. This set has the advantage of adjusting the desired level of robustness in the generated solutions according to the decision-maker's aversion to risk, preventing over-conservative solutions caused by unlikely realizations. Additionally, it is defined as a polyhedral set, thus the computational difficulty related to RO approaches based on this set does not increase substantially compared to their deterministic counterparts. Finally, this set has been successfully used in the literature to model other VRP variants (Agra et al. 2013, Gounaris, Wiesemann, and Floudas 2013, Munari et al. 2019, Lu and Gzara 2019, De La Vega, Munari, and Morabito 2020). These reasons motivate the use of RO to tackle the BRP under uncertainty.

RO has been used to address uncertainty in other important operations of BSSs. Lu (2016) uses RO to model the allocation of bikes to stations under demand uncertainty. The author proposes models based on time-space networks to determine the optimal flow of bikes. Fu et al. (2022) address demand uncertainty in integrated location and allocation decisions. They propose a robust two-stage optimization approach in which the first stage considers the positioning of bicycle stations and service areas, while the second stage solves a static vehicle allocation problem. None of these studies consider routing decisions, and thus, the problems addressed by them differ in essence from the BRP.

The presented literature reveals a lack of models and solution approaches for the BRP under uncertainty, especially considering RO. This is noteworthy because the importance of relying on robust solutions has often and successfully been demonstrated in the VRP literature (Campos, Munari, and Coelho 2022, Subramanyam 2023). It is clear that investigating solution approaches for robust variants of the BRP is an important and promising research avenue.

3. The Robust Bike Sharing Rebalancing Problem (RBRP)

The RBRP consists of defining a minimum-cost set of routes that ensure the rebalancing of the bike inventory at each station, through pickup and delivery operations, using the vehicle fleet available at a single depot. We assume a homogeneous fleet of m vehicles of capacity Q. Each station must be visited exactly once, for either a pickup or a delivery operation. Each route must start and end at the depot and respect vehicle capacity for any possible realization of the uncertain demand. By definition, the depot is assumed to have an unlimited storage capacity and inventory level, thus being able to provide or receive any number of bikes as long as vehicle capacity is respected. This is a reasonable assumption as the depot is expected to have a larger storage capacity than the combined availability of bikes at stations.

The RBRP can be represented by a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V} = \mathcal{I} \cup \{0\}$ is the set of vertices, $\mathcal{I} = \{1, ..., n\}$ is the set of stations and 0 represents the depot. The arc set $\mathcal{A} = \{(i, j) : i, j \in \mathcal{V}, i \neq j\}$ represents the connections between vertices in $\mathcal{V}$. We define the subset $\mathcal{A}(\mathcal{I}) \subset \mathcal{A}$ containing only arcs that connect two stations. Each arc $(i, j) \in \mathcal{A}$ is associated with an asymmetric travel cost c_{ij} and each station $i \in \mathcal{I}$ has an uncertain demand $\tilde{d}_i$, which represents its excess or shortage of bikes. We model demands as random variables whose specific values (realizations) become known only after a vehicle arrives at

each station. If the realization of $\tilde{d}_i$ is positive, there is an excess of bikes at station i with respect to its desired inventory level, and thus, this station is a pickup vertex. Conversely, if this realization is negative, there is a shortage of bikes at station i, making it a delivery vertex. Stations with demand realizations equal to zero are balanced. These stations must still be visited, as their actual demand is unknown at the planning stage.

Following the RO paradigm, we consider that each uncertain demand $\tilde{d}_i$ is characterized by a nominal value $d_i \in \mathbb{R}$ and a maximum deviation $\hat{d}_i \in \mathbb{R}^+$, such that its realization belongs to the range $[d_i - \hat{d}_i, d_i + \hat{d}_i]$, for each $i \in \mathcal{I}$. We then conveniently represent $\tilde{d}_i = d_i + \hat{d}_i \xi_i, \forall i \in \mathcal{I}$, in which ξ_i is a random variable defined in the range $[-1, 1]$. To avoid overconservatism, we model the demand uncertainty using the cardinality-constrained uncertainty set (Bertsimas and Sim 2004). As already mentioned, these sets have been successfully used in the literature to model VRP variants under uncertainty (Agra et al. 2013, Gounaris, Wiesemann, and Floudas 2013, Munari et al. 2019, De La Vega, Munari, and Morabito 2020, Campos, Munari, and Coelho 2022). For a given budget of uncertainty $\Gamma > 0$, chosen by the decision maker according to their risk aversion, we define the cardinality-constrained set for demand uncertainty as follows:

$$\mathcal{U}(\Gamma) = \{\tilde{\mathbf{d}} \in \mathbb{R}^{|\mathcal{I}|} \mid \tilde{d}_i = d_i + \hat{d}_i \xi_i, i \in \mathcal{I};\ \sum_{i \in \mathcal{I}} |\xi_i| \leq \Gamma;\ \xi_i \in [-1, 1], i \in \mathcal{I}\}. \tag{1}$$

We assume hereafter that $\Gamma \in \mathbb{Z}^+$. Thus, the budget of uncertainty can be interpreted as the maximum number of stations in which the demand realizations attain their worst case. A solution is said to be *robust-feasible* if its routes satisfy the vehicle capacity Q for every demand realization in $\mathcal{U}(\Gamma)$.

Different techniques can be used to incorporate the uncertainty set $\mathcal{U}(\Gamma)$ into a MILP model (Bertsimas and Sim 2004, Zeng and Zhao 2013, Bertsimas, Dunning, and Lubin 2016, Büsing, Gersing, and Koster 2023). Due to its generality and practicality, the dualization technique proposed by Bertsimas and Sim (2004) remains the most popular for formulating robust counterparts for deterministic MILPs by using cardinality-constrained sets. However, in VRP-like problems, this technique can significantly increase the number of variables and constraints in the robust counterpart, often resulting in elevated computation times (Agra et al. 2013, Gounaris, Wiesemann, and Floudas 2013, Munari et al. 2019). As an alternative, we may resort to a technique based on the linearization of recursive equations that model the worst-case behavior of the uncertain parameters. This technique was shown to be more intuitive and efficient for VRP variants (Yu, Cheng, and Zhu 2022, Campos, Munari, and Coelho 2022, Munari et al. 2019). Therefore, the same technique is adopted in this study for modeling the RBRP, but with the required non-trivial adaptations described as follows.

Munari et al. (2019) described the linearization technique for the VRP with time windows (VRPTW), a variant that considers either pickup or delivery operations, but not both. The application of this technique to the RBRP is not straightforward though, due to certain specificities of this problem, described as follows:

- Since there are pickup and delivery operations in the RBRP, the worst-case load in a

vehicle is no longer monotonic along its route as it is in the VRPTW;

- We have to consider both positive and negative deviations of the nominal demand value in the computation of the worst-case load, while in the VRPTW only positive deviations are relevant. Additionally, note that a station with positive nominal demand (pickup vertex) may become one with a negative demand realization (delivery vertex) depending on its deviation value. Hence, the same station may become either a pickup or delivery vertex depending on its demand realization;

- Another specificity of the RBRP is that a vehicle may depart from the depot with a positive load, while in the VRPTW all vehicles depart empty (or full if we take the delivery perspective). Hence, in the RBRP, we need to explicitly define the initial load of each vehicle during the route planning, as it affects the feasibility of the routes. Each route in the solution has to be feasible for this initial value for all demand realizations belonging to the uncertainty set.

All the mentioned characteristics bring additional challenges to writing recursive equations for calculating the worst-case load in the RBRP using dynamic programming. Fortunately, a relevant observation allows us to simplify this calculation: it suffices to consider only all positive or all negative demand deviations when calculating a vehicle's worst-case load. In fact, if demand deviations are not all positive in a given route, it is always possible to obtain a maximum vehicle load that is greater than or equal to the current by forcing all deviations to be positive. A similar reasoning applies to the minimum vehicle load.

Proposition 1. *Consider the uncertainty set $\mathcal{U}(\Gamma)$ for $\Gamma \in \mathbb{Z}^+$. Given a route r, the minimum and maximum vehicle load in this route for any realization in $\mathcal{U}(\Gamma)$ can be calculated considering only the realizations $\bar{\mathbf{d}} \in \mathcal{U}(\Gamma)$ such that $\bar{d}_i = d_i + \hat{d}_i\xi_i$ with:*

(i) $\xi_i = -1, \forall i \in \mathcal{I}$ *(minimum vehicle load); and*

(ii) $\xi_i = +1, \forall i \in \mathcal{I}$ *(maximum vehicle load).*

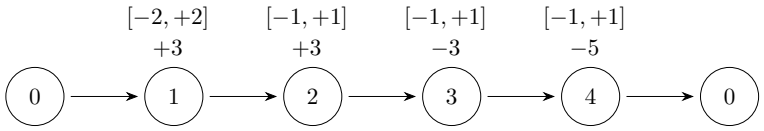
Proof. By contradiction. Let θ^- denote the minimum vehicle load in this route, considering any realization in $\mathcal{U}(\Gamma)$. Consider a realization $\mathbf{d}^1 \in \mathcal{U}(\Gamma)$ such that $d_i^1 = d_i + \hat{d}_i\xi_i$ with $\xi_k = +1$ for at least one k in route r. Let θ^1 be the minimum vehicle load in route r for this realization and consider this load is observed after k is served in the route. Assume that for this realization we have $\theta^1 = \theta^-$. Now, let $\mathbf{d}^2 \in \mathcal{U}(\Gamma)$ be the same realization as $\mathbf{d}^1$ except for having $\xi_k = -1$. Hence, $d_k^2 < d_k^1$. We denote as θ^2 the minimum vehicle load for $\mathbf{d}^2$. Since all the other components of $\mathbf{d}^2$ are the same as in $\mathbf{d}^1$, we must have $\theta^2 < \theta^1 = \theta^-$, which is a contradiction, as we have assumed θ^- is the minimum vehicle load in r for any realization in $\mathcal{U}(\Gamma)$. The proof of the maximum vehicle load follows similarly. □

Consider a budget of uncertainty $\Gamma > 0$ and a route $r = (r_0, r_1, \ldots, r_h)$, with $h > 1$, in which both r_0 and r_h represent the depot. Then, let $\theta^+_{r_j\gamma}$, for $j = 0, \ldots, h$ and $\gamma \in \{0, 1, \ldots, \Gamma\}$, specify the worst-case load of a vehicle after serving vertex r_j considering that the demand realizations of any γ stations in the route up to station r_j attain their worst-case

values with all-positive deviations. Similarly, we define $\theta^-_{r_j\gamma}$ for all-negative deviations. Let us also denote by θ^d the initial load of a vehicle leaving the depot to perform route r.

One important observation is that even though the worst-case demand realization is achieved by considering deviations to be either all positive or all negative, we cannot assume that these cases are independent of one another. The following example illustrates how the independent computation of demand variations in the all-negative and all-positive cases may lead to a miscalculation of the required initial vehicle load. Consider the route $r = (0, 1, 2, 3, 4, 0)$ depicted in Figure 1, in which demand values and deviations in the format $[-\hat{d}_i, +\hat{d}_i]$ are shown above each station. The vehicle capacity is $Q = 10$. The figure presents the load along each arc under three different scenarios: the deterministic case and, for $\Gamma = 1$, the cases with all-positive and all-negative deviations.

Figure 1: Example of a route in which considering worst-case demand values with all-positive deviations and all-negative deviations independently leads to a miscalculation of the initial vehicle load



deterministic	2	5	8	5	0
all-positive	2	7	10	7	2
all-negative	4	5	8	5	0
all-positive ($\theta^d = 4$)	4	9	**12**	9	4

In the deterministic case, if the vehicle starts the route empty, its cumulative load at each arc in the route can be represented as $(0, 3, 6, 3, -2)$. This means that the vehicle must depart from the depot with at least $\theta^d = 2$ bikes to fulfill the demand of each station. As shown in Figure 1, the resulting flow of bikes along the route for the deterministic case is then $(2, 5, 8, 5, 0)$. Next, assuming $\Gamma = 1$, the worst-case vehicle load considering only positive deviations happens when station 1 attains its worst-case realization at $\bar{d}_1 = 3 + 2$. If the vehicle departs from the depot with 2 bikes, the flow of bikes considering only positive deviations is $(2, 7, 10, 7, 2)$.

Now, by considering only negative deviations, the worst-case is once again achieved when the demand of station 1 attains its worst-case realization, i.e., $\bar{d}_1 = 3 - 2$. In this case, assuming that the vehicle leaves the depot with only 2 bikes, the cumulative load at each arc is given by the sequence $(2, 3, 6, 3, -2)$. This indicates that the vehicle must leave the depot with at least $\theta^d = 4$ bikes to fulfill the demand of each station, with a resulting cumulative vehicle load denoted by $(4, 5, 8, 5, 0)$, as depicted in Figure 1. Considering that, apparently, in both worst cases the vehicle capacity is not violated for their specific choices of initial load, this route could be mistakenly deemed feasible. However, for this solution to be robust-feasible, the actual initial load of the vehicle must be $\theta^d = 4$, which would render the solution infeasible when considering all-positive variations, as the vehicle capacity would be violated on arc $(2, 3)$. This is highlighted at the bottom of Figure 1. Therefore, in the RBRP, we

need to specifically define the initial load of each vehicle route, and then guarantee that all these routes remain feasible for their respective initial loads, considering all realizations of the demand. The computation of the worst-case load in each route must be performed using the same initial load for both all negative and all positive deviations.

In what follows, we show how to compute worst-case vehicle loads by simultaneously considering all negative and all positive deviations. In addition, we also show the correct calculation of the required initial load values. First, we calculate $\theta^-_{r_j\gamma}$ assuming that the vehicle departs empty from the depot, using the following recursive equation:

$$\theta^-_{r_j\gamma} = \begin{cases} 0, & \text{if } j = 0, \\ \theta^-_{r_{j-1}\gamma} + d_{r_j}, & \text{if } \gamma = 0, \\ \min\{\theta^-_{r_{j-1}\gamma} + d_{r_j}, \theta^-_{r_{j-1}(\gamma-1)} + d_{r_j} - \hat{d}_{r_j}\}, & \text{otherwise.} \end{cases} \tag{2}$$

The first line in this equation corresponds to the load of a vehicle after departing from the depot. The second line represents the case in which none of the demands attains their worst-case up to vertex r_j, thus the vehicle collects (if $d_{r_j} > 0$) or delivers (if $d_{r_j} < 0$) the nominal demand of r_j only. Finally, the last line calculates the worst-case load after r_j, considering two different cases in which the vehicle collects or delivers either the nominal demand or the nominal demand minus the deviation value.

After calculating $\theta^-_{r_j\gamma}$, we define $\theta^d = \max\{0, -\min\{\theta^-_{r_j\gamma} : j = 0, \ldots, h; \gamma = 0, 1, \ldots, \Gamma\}\}$, which represents the required initial load of a vehicle departing from the depot, to ensure that there are enough bikes to meet all demands in route r, accounting for any demand realization. If the minimization term in the calculation of θ^d results in a negative value, then it represents the worst-case shortage of bikes in the vehicle at a given station, for a given demand realization. Recall that a route r is robust-feasible only if $0 \leq \theta^-_{r_j\gamma} + \theta^d \leq Q$, for all $j = 0, \ldots, h$, and $\gamma \in \{0, 1, \ldots, \Gamma\}$. This is a necessary but not sufficient condition, as we still need to check the feasibility regarding $\theta^+_{r_j\gamma}$. Hence, if none of these inequalities is violated, we calculate the worst-case load values regarding all-positive deviations using the following recursive equation:

$$\theta^+_{r_j\gamma} = \begin{cases} \theta^d, & \text{if } j = 0, \\ \theta^+_{r_{j-1}\gamma} + d_{r_j}, & \text{if } \gamma = 0, \\ \max\{\theta^+_{r_{j-1}\gamma} + d_{r_j}, \theta^+_{r_{j-1}(\gamma-1)} + d_{r_j} + \hat{d}_{r_j}\}, & \text{otherwise.} \end{cases} \tag{3}$$

Note that the initial vehicle load θ^d is used in the first line of this equation. This ensures that the same value is considered in both computations of the worst-case load. Finally, route r is robust-feasible only if the calculated values satisfy $0 \leq \theta^+_{r_j\gamma} \leq Q$, for all $j = 0, \ldots, h$, and $\gamma \in \{0, 1, \ldots, \Gamma\}$.

Recursive equations (2) and (3) can be implemented as dynamic programming algorithms to verify the robust-feasibility of a given route. We rely on these algorithms in the proposed B&C. Additionally, we linearize these equations to produce sets of constraints that guarantee the robustness of solutions in a compact formulation for the RBRP, as presented in the following section.

4. MIP Formulations

We introduce two new formulations for the RBRP under demand uncertainty. The first one is a compact model in which robust feasibility is guaranteed via a polynomial number of constraints that originate from the linearization of the recursive equations introduced in Section 3. The second formulation is non-compact, as it relies on an extensive number of constraints to ensure robust feasibility.

4.1. Compact Formulation

Our compact formulation results from linearizing and incorporating the recursive equations (2) and (3) into an existing deterministic BRP formulation. We define the decision variables of this formulation as follows.

$x_{ij} \in \{0,1\}$ assumes the value of 1 if, and only if, vertex j is visited immediately after vertex i in the same route, $\forall (i,j) \in \mathcal{A}$;

$u_j \in \mathbb{R}^+$ number of vertices visited in a route up to vertex j, $\forall j \in \mathcal{V}$;

$\theta^+_{j\gamma} \in \mathbb{R}^+$ worst-case vehicle load in a route after serving station j, when the demand realizations of any γ stations attain their worst-case value by positive deviations only, $\forall j \in \mathcal{I}$ and $\gamma \in \{0,1,\ldots,\Gamma\}$;

$\theta^-_{j\gamma} \in \mathbb{R}^+$ worst-case vehicle load in a route after serving station j, when the demand realizations of any γ stations attain their worst-case value by negative deviations only, $\forall j \in \mathcal{I}$ and $\gamma \in \{0,1,\ldots,\Gamma\}$

$\theta^d_j \in \mathbb{R}^+$ initial vehicle load in the route that has vertex j as the first visited station, $\forall j \in \mathcal{I}$.

Variables x_{ij} are common in the VRP literature and represent the flow of vehicles across the arcs of graph G. Auxiliary variables u_i are used to define the well-known Miller-Tucker-Zemlin (MTZ) subtour elimination constraints (Miller, Tucker, and Zemlin 1960). Variables $\theta^+_{i\gamma}$, $\theta^-_{i\gamma}$ and θ^d_j have the same meaning as in Section 3. Note that the definition of θ^d_j includes an index to specify the first visited station in the route. This is necessary to identify different routes.

Using the defined parameters and decision variables, we cast our compact formulation for the RBRP under demand uncertainty as follows.

$$\min \quad \sum_{(i,j)\in\mathcal{A}} c_{ij} x_{ij} \tag{4}$$

$$\text{s.t.} \quad \sum_{(i,j)\in\mathcal{A}} x_{ij} = 1, \qquad \forall j \in \mathcal{I}, \tag{5}$$

$$\sum_{(i,j)\in\mathcal{A}} x_{ij} = 1, \qquad \forall i \in \mathcal{I}, \tag{6}$$

$$\sum_{(0,j)\in\mathcal{A}} x_{0j} \geq 1, \tag{7}$$

$$\sum_{(0,j)\in\mathcal{A}} x_{0j} - \sum_{(j,0)\in\mathcal{A}} x_{j0} = 0, \qquad \forall j \in \mathcal{I}, \tag{8}$$

$$u_j \geq u_i + 1 - n(1 - x_{ij}), \qquad \forall (i,j) \in \mathcal{A} : j > 0, \tag{9}$$

$$\theta^-_{j\gamma} \leq \theta^-_{i\gamma} + d_j x_{ij} + Q(1 - x_{ij}), \qquad \forall (i,j) \in \mathcal{A}(\mathcal{I}),\ \gamma \in \{0,\dots,\Gamma\}, \tag{10}$$

$$\theta^-_{j\gamma} \leq \theta^-_{i\gamma-1} + (d_j - \hat{d}_j) x_{ij} + Q(1 - x_{ij}), \qquad \forall (i,j) \in \mathcal{A}(\mathcal{I}),\ \gamma \in \{1,\dots,\Gamma\}, \tag{11}$$

$$\theta^-_{j0} \leq \theta^d_j + d_j x_{0j} + Q(1 - x_{0j}), \qquad \forall j \in \mathcal{I}, \tag{12}$$

$$\theta^-_{j\gamma} \leq \theta^d_j + (d_j - \hat{d}_j) x_{0j} + Q(1 - x_{0j}), \qquad \forall j \in \mathcal{I},\ \gamma \in \{1,\dots,\Gamma\}, \tag{13}$$

$$\theta^+_{j\gamma} \geq \theta^+_{i\gamma} + d_j x_{ij} - Q(1 - x_{ij}), \qquad \forall (i,j) \in \mathcal{A}(\mathcal{I}),\ \gamma \in \{0,\dots,\Gamma\}, \tag{14}$$

$$\theta^+_{j\gamma} \geq \theta^+_{i\gamma-1} + (d_j + \hat{d}_j) x_{ij} - Q(1 - x_{ij}), \qquad \forall (i,j) \in \mathcal{A}(\mathcal{I}),\ \gamma \in \{1,\dots,\Gamma\}, \tag{15}$$

$$\theta^+_{j0} \geq \theta^d_j + d_j x_{0j} - Q(1 - x_{0j}), \qquad \forall j \in \mathcal{I}, \tag{16}$$

$$\theta^+_{j\gamma} \geq \theta^d_j + (d_j + \hat{d}_j) x_{0j} - Q(1 - x_{0j}), \qquad \forall j \in \mathcal{I},\ \gamma \in \{1,\dots,\Gamma\}, \tag{17}$$

$$\theta^+_{j\gamma}, \theta^-_{j\gamma} \in [0, Q], \qquad \forall j \in \mathcal{I},\ \gamma \in \{0,\dots,\Gamma\}, \tag{18}$$

$$\theta^d_j \in [0, Q], \qquad \forall j \in \mathcal{I}, \tag{19}$$

$$u_j \in [0, n], \qquad \forall j \in \mathcal{V}, \tag{20}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall (i,j) \in \mathcal{A}. \tag{21}$$

The objective function (4) consists of minimizing the total routing cost. Constraints (5) and (6) are vertex degree constraints. Constraint (7) enforces that at least one vehicle leaves the depot, while constraints (8) ensure that the number of vehicles departing from the depot is the same as the number of vehicles returning to the depot. Constraints (9) are the MTZ constraints that prevent subtours. Although other types of subtour elimination constraints can be employed, the MTZ constraints allow us to define a compact formulation. Moreover, it is worth mentioning that the remaining MTZ-like constraints cannot prevent subtours given that the vehicle load is non-monotonic in the RBRP.

Constraints (10)–(17) are linearizations of the recursive equations (2) and (3) that ensure the robustness of solutions. Constraints (10) correspond to the deterministic cases in equation (2), i.e. when demands take their nominal value. Constraints (11) refer to the case considering the worst-case deviations. Constraints (12) and (13) enforce the initial vehicle load for nominal and worst-case values of the demand. Hence, these constraints work together to ensure that if stations i and j are visited consecutively, the vehicle load after serving j is given by the minimum of the following values: (*i*) the vehicle load just after serving i plus the nominal demand of j, assuming that the demand realizations of any γ stations up to j already reached their worst-case negative variations; (*ii*) the vehicle load right after serving i plus the worst-case demand realization at j, considering that the demand in any $\gamma-1$ stations reached their worst-case values up to station i; (*iii*) the vehicle load right after departing the depot plus the nominal demand of j; and (*iv*) the vehicle load right after departing the depot plus the worst-case demand of j. Constraints (14)–(17) act similarly for positive worst-case deviations. Finally, constraints (18)–(21) impose the domain of the decision variables.

From an application standpoint, formulation (4)–(21) has the advantage of being compact and relatively easy to understand and implement using off-the-shelf optimization software. This may be beneficial for practitioners who can take advantage of this model's compactness to solve real-world problems of reasonable size. Every feasible solution of this formulation corresponds to a set of robust-feasible routes, which means that they remain feasible for any demand realization within the uncertainty set $\mathcal{U}(\Gamma)$. Note that the deterministic counterpart related to this model is easily obtained by discarding index γ from variables $\theta^+_{i\gamma}$ and $\theta^-_{i\gamma}$, and dropping constraints (11) and (15).

4.2. Cut-based Formulation

Our second formulation is non-compact and relies on extensive cutting planes to ensure robust feasibility and improved LP relaxation bounds. This formulation is the basis of the B&C algorithm presented in Section 5 and, differently from formulation (4)–(21), it can be used to tackle medium- to large-sized instances.

Consider the decision variable x_{ij} exactly as defined in the previous subsection. Additionally, let $\mathcal{R}$ be the set of all infeasible routes, and $A(R)$ be the set of arcs in route $R \in \mathcal{R}$. We also define $x(\bar{S}, S) = \sum_{i \in \bar{S}} \sum_{j \in S} x_{ij}$. Using this notation, we state the cut-based formulation for the RBRP under demand uncertainty as follows:

$$\min \sum_{(i,j) \in \mathcal{A}} c_{ij} x_{ij} \tag{22}$$

s.t. Constraints (5) − (8) and (21),

$$x(\bar{S}, S) \geq \max \left\{ 1, \left\lceil \left| \frac{1}{Q} \left(\sum_{i \in S} d_i + \max_{\substack{\mathcal{J} \subseteq S \\ |\mathcal{J}| \leq \Gamma}} \sum_{i \in \mathcal{J}} \hat{d}_i \right) \right| \right\rceil, \left\lceil \left| \frac{1}{Q} \left(\sum_{i \in S} d_i - \max_{\substack{\mathcal{J} \subseteq S \\ |\mathcal{J}| \leq \Gamma}} \sum_{i \in \mathcal{J}} \hat{d}_i \right) \right| \right\rceil \right\}, \quad \forall\ S \subseteq \mathcal{I}, \bar{S} = \mathcal{V} \setminus S, \tag{23}$$

$$\sum_{(i,j) \in A(R)} x_{ij} \leq |A(R)| - 1, \quad \forall\ R \in \mathcal{R}. \tag{24}$$

As in the compact formulation, the objective function (22) aims at minimizing the total routing cost. Constraints (23) are robust rounded capacity inequalities that prevent subtours and ensure that vehicle capacity is not exceeded at any point in the routes. These constraints are not sufficient to ensure robust-feasibility though. As explained in Section 4.1, it is possible that the initial load of the vehicle when assuming only negative variations is larger than when considering only positive ones. In these cases, the actual load of the vehicle is defined by the former and may render the solution infeasible for positive variations. Consequently, constraints (24) are feasibility cuts required to eliminate certain solutions that are not robust-feasible and do not violate constraints (23).

5. Branch-and-Cut Algorithm

We describe a B&C algorithm based on the combinatorial relaxation of formulation (22)–(24) regarding constraints (23) and (24). In the B&C, these constraints are separated for

each incumbent candidate integer solution as well as for non-integer optimal solutions of the LP relaxations. Additionally, we generate valid inequalities to strengthen the LP relaxations and improve the overall performance of the algorithm.

5.1. Valid Inequalities

We consider three sets of valid inequalities originally proposed by Dell'Amico et al. (2014) for the deterministic case. The first two sets are clique inequalities involving three vertices. Given a pair of stations $i, j \in \mathcal{I}$, let $S(i,j) = \{k \in \mathcal{I} \setminus \{i,j\} : |d_i + d_j + d_k| > Q\}$ be the set of stations such that the combined nominal demand of i, j and any station $k \in S(i,j)$ is greater than Q. This means that these vertices cannot be visited consecutively. The resulting clique inequalities are as follows:

$$x_{ij} + \sum_{h \in S(i,j)} x_{jh} \leq 1, \qquad \forall\ i,j \in \mathcal{I}, \tag{25}$$

$$\sum_{h \in S(i,j)} x_{hi} + x_{ij} \leq 1 \qquad \forall\ i,j \in \mathcal{I}. \tag{26}$$

The third set of valid inequalities is a generalization of the classical tournament constraints for the TSP (Ascheuer, Fischetti, and Grötschel 2000). This set replaces constraints (24) given that they are stronger than the usual *no-good* cuts. For each infeasible route $r = (r_0, r_1, \ldots, r_h)$ in $\mathcal{R}$, where r_0 and r_h represent the depot, let $k \in \{2, \ldots, h\}$ be the smallest index such that the subpath $(r_0, r_1, \ldots, r_k)$ is infeasible. Then, we define the following inequality for this route:

$$x_{0r_1} + \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} x_{r_i r_j} \leq k - 1. \tag{27}$$

5.2. Separation Procedures

In this section, we present the separation procedures implemented to separate cuts and valid inequalities for both candidate incumbent (integer) and fractional solutions in the B&C algorithm. It is worth mentioning that, when using the clique inequalities (25) and (26), we enumerate all these constraints and add them directly to the model in a preprocessing step before the main algorithm. The same is done for constraints (23) in the deterministic case and for all sets S containing two stations.

5.2.1. Separation for Candidate Incumbent Solutions.

The separation procedure for constraints (23) considering candidate incumbent solutions starts by checking for subtours in each route of this solution. For each subtour that is found, it generates a cut using constraints (23) and selects the most violated term on the right-hand side of this cut. During this inspection, we store all the routes that compose the solution and, in case no subtours are found, we call Algorithm 1 for each route R, which verifies whether this route is robust-feasible.

According to Proposition 1, the worst-case load in the vehicle can be calculated considering that the demand deviations of the customers in the corresponding route are either all positive

Algorithm 1: Separation of capacity constraints for incumbent solutions

function SeparationCapacityIncumbent(R)
  $\Theta^-, \Theta^+, \Theta^-_{min}, \Theta^+_{max} \leftarrow 0$
  $L \leftarrow \emptyset$
  first_infeasible $\leftarrow -1$
  added_cuts $\leftarrow false$
  for $k \leftarrow 2$ ***to*** $|R| - 1$ **do** `// $R_1$ and $R_{|R|}$ are the depot`
    $i \leftarrow R_k$
    $\Theta^- \leftarrow \Theta^- + d_i$
    $\Theta^+ \leftarrow \Theta^+ + d_i$
    if $\Gamma > 0$ **then**
      $\Theta^- \leftarrow \Theta^- - \hat{d}_i$
      $\Theta^+ \leftarrow \Theta^+ + \hat{d}_i$
      $L \leftarrow L \cup \{i\}$
    else if $\hat{d}_i > \min_{j \in L}\{\hat{d}_j\}$ **then**
      $j \leftarrow \arg\min_{j \in L}\{\hat{d}_j\}$
      $\Theta^- \leftarrow \Theta^- + \hat{d}_j - \hat{d}_i$
      $\Theta^+ \leftarrow \Theta^+ - \hat{d}_j + \hat{d}_i$
      $L \leftarrow \{L \setminus \{j\}\} \cup \{i\}$
    $\Theta^-_{min} \leftarrow \min(\Theta^-_{min}, \Theta^-)$
    $\Theta^+_{max} \leftarrow \max(\Theta^+_{max}, \Theta^+)$

    if $\Theta^+_{max} - \Theta^-_{min} > Q$ ***and*** *first_infeasible* $= -1$ **then**
      first_infeasible $\leftarrow i$

    if $|\Theta^-_{min}| > Q$ ***or*** $\Theta^+_{max} > Q$ **then**
      addCapacityCut($S = \{R_2, \ldots, R_i\}, L$) `// $R_1$ is the depot`
      added_cuts $\leftarrow true$

  if ***not*** *added_cuts* ***and*** *first_infeasible* > 0 **then**
    addTournamentCut($S = \{R_1, \ldots, R_i\}$)

or all negative. Based on this observation, Algorithm 1 starts by initializing variables Θ^+ and Θ^-, which specify the current load of the vehicle when assuming that deviations are all positive or all negative, respectively. Variables $\Theta^+_{max} \geq 0$ and $\Theta^-_{min} \leq 0$ are used to store, respectively, the maximum non-negative value of Θ^+ and the minimum non-positive value of Θ^- along route R. While route R is traversed, set L is used to keep track of which stations are currently assumed to have their demand realizations attaining the worst-case. The setup of the algorithm is completed by defining auxiliary variables *added_cuts* and *first_infeasible*, which help to detect whether constraints (23) are not sufficient to cut an infeasible solution.

The main loop of Algorithm 1 checks the worst-case load at each node visited by route R. For each station $i \in R$, the vehicle load is updated according to the nominal demand d_i (lines 8 and 9) and it is decided whether or not to include station i into the set L, which contains stations attaining their worst case in terms of demand deviation (lines $10-18$). At each visit, we may update the minimum and maximum load of the vehicle when assuming the worst-case deviations in L (lines 19 and 20). Naturally, Θ^-_{min} and Θ^+_{max} specify, respectively, the lowest and highest load values in a possibly robust-feasible solution. If Θ^-_{min} is negative, the vehicle needs additional bikes to perform the delivery demands in the route, and these bikes should be provided by the depot. Thus, $|\Theta^-_{min}|$ determines the initial load of the vehicle when departing from the depot to perform route R. Therefore, if at visit k on the route, $\Theta^+_{max} - \Theta^-_{min} > Q$, then the partial path up to this visit is infeasible. Note that, in this scenario, we only record position k and do not immediately add a cut (lines 22 and 23). In fact, we only add the tournament constraints (27) when strictly necessary to ensure feasibility (lines 29 and 30). The last check performed in the main loop verifies whether the vehicle capacity is violated by either Θ^-_{min} or Θ^+_{max} and, if so, a cut in the form of constraints (23) is added for $S = \{R_2, \ldots, R_i\}$ assuming that the demand of all stations in L attain their worst-case deviation (lines 25-26).

5.2.2. Separation for Fractional Solutions.

The first procedure, namely FracSepST, is used to separate subtour cuts and uses the standard max-flow-based separation algorithm for VRPs. For the sake of completeness, we describe the algorithm as follows. Given a fractional solution $\bar{x}$, a support graph $\bar{\mathcal{G}} = (\mathcal{V}, \bar{\mathcal{A}})$ is built such that $\bar{\mathcal{A}} = \{(i,j) \in \mathcal{A} : \bar{x}_{ij} > 0\}$. Each arc $(i,j) \in \bar{\mathcal{A}}$ is assigned a capacity equal to $\bar{x}_{ij}$. Then, for each station $i \in \mathcal{I}$, we evaluate the max-flow/min-cut on $\bar{\mathcal{G}}$ from the depot to i. If the resulting min-cut value is less than one, the subset S induced by the min-cut containing vertex i violates constraints (23). Whenever a violation is detected, a cut of the same form is generated using the most violated term on the right-hand side with $\Gamma = 0$.

A second procedure called FracSepCap, is used to separate the following capacity constraints, which are a subset of constraints (23) for the deterministic case:

$$x(\bar{S}, S) \geq \left\lceil -\frac{\sum_{i \in S} d_i}{Q} \right\rceil, \qquad \forall\ S \subseteq \mathcal{I}, \bar{S} = \mathcal{V} \setminus S, \tag{28}$$

$$x(\bar{S}, S) \geq \left\lceil \frac{\sum_{i \in S} d_i}{Q} \right\rceil, \qquad \forall\ S \subseteq \mathcal{I}, \bar{S} = \mathcal{V} \setminus S. \tag{29}$$

We begin by building a support graph $\bar{\mathcal{G}}' = (\bar{\mathcal{V}}', \bar{\mathcal{A}}')$ such that $\bar{\mathcal{V}}' = \mathcal{V} \cup \{n+1, n+2\}$, where $n+1$ and $n+2$ are dummy vertices. The set $\bar{\mathcal{A}}'$ is composed by the following three types of arcs: $(i,j) \in \mathcal{A}$ such that $\bar{x}_{ij} > 0$, with an assigned capacity equal to $\bar{x}_{ij}$; $(n+1, j)$ for all $j \in \mathcal{I} : d_j > 0$, with capacity d_j/Q; and $(i, n+2)$ for all $i \in \mathcal{I} : d_i < 0$, with capacity $-d_i/Q$. We then compute the maximum flow from $n+1$ to $n+2$ in $\bar{\mathcal{G}}'$. Let S be the subset induced by the associated min-cut that contains vertex $n+2$, let $\bar{S} = \bar{\mathcal{V}}' \setminus S$, and consider f as the resulting min-cut value. If $f < \sum_{i \in \mathcal{I}: d_i < 0} -d_i/Q$, then subset S violates constraints (28). This separation procedure was used by Dell'Amico et al. (2014) for the BRP and is basically the same as the one proposed by Hernández-Pérez and Salazar-González (2004a) for the 1-PDTSP, with the exception that, for the BRP, the depot must be included in the separation and the support graph contains cycles. Depending on the problem, having cycles on this type of support graph may lead to issues, as bikes that return to the depot at the end of a route might be used to fulfill the demand of customers in other routes. However, this is not an issue for either the BRP or the RBPR because, in these problems, the depot can provide any number of additional bikes to the system and absorb any surplus. Considering such observations and the fact that the proof of correctness has not been shown by Dell'Amico et al. (2014), we present it in the following proposition.

Proposition 2. *The relaxed version of constraints* (28) *can be separated in polynomial time.*

Proof. We are given a possibly fractional solution $\bar{x}$. Let $d(S) = \sum_{i \in S} d_i$ and let us partition $d(S) = d^+(S) + d^-(S)$, where $d^+(S) = \sum_{i \in S: d_i > 0} d_i$ and $d^-(S) = \sum_{i \in S: d_i < 0} d_i$. Then, let us rewrite a relaxed version of constraints (28) as

$$x(\bar{S} : S) \geq -\left(\frac{d^+(S) + d^-(S)}{Q} \right). \tag{30}$$

By extension, we can also partition $d(\mathcal{V}) = d^+(\mathcal{V}) + d^-(\mathcal{V})$, where $d^-(\mathcal{V}) = d^-(S) + d^-(\bar{S})$. This enables us to rewrite (30) as

$$x(\bar{S} : S) + \frac{d^+(S)}{Q} \geq \frac{d^-(\bar{S}) - d^-(\mathcal{V})}{Q}, \tag{31}$$

and then

$$x(\bar{S} : S) + \frac{d^+(S)}{Q} - \frac{d^-(\bar{S})}{Q} \geq -\frac{d^-(\mathcal{V})}{Q}. \tag{32}$$

Now, consider the support graph $\bar{\mathcal{G}}'$ built by FracSepCap. The procedure solves a max-flow problem considering vertices $n+1$ and $n+2$ as the source and sink, respectively. Figure 2 presents an example of part of the graph $\bar{\mathcal{G}}'$ focusing on a station $j \in \mathcal{I}$ and its relationship with the min-cut, represented by the dashed line in the figure.

Note that, for each station $j \in \mathcal{I}$, there are two possible cases:

1. if $j \in S$, by construction, its contribution to the min-cut value f is equal to $\sum_{i \in \bar{S}} \bar{x}_{ij}$. In case $d_j > 0$, we must also add d_j/Q.

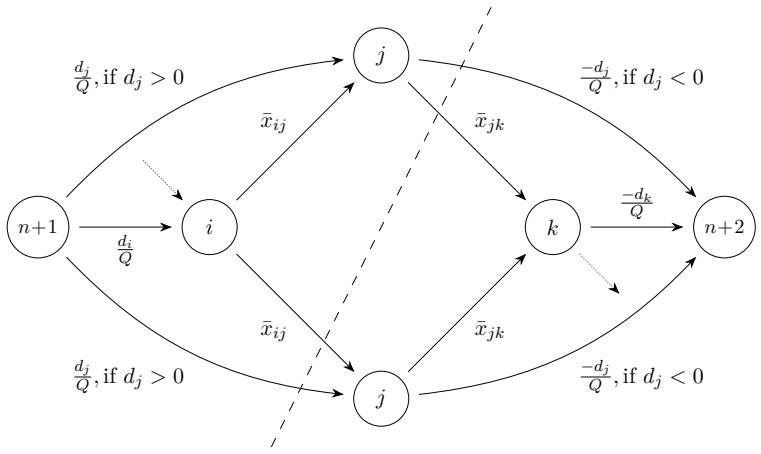


Figure 2: Example of the support graph $\bar{\mathcal{G}}'$ and the associated min-cut represented by the dashed line.

2. otherwise, if $j \in \bar{S}$, its contribution to f is given by $\sum_{k \in S} \bar{x}_{jk}$, plus $-d_j/Q$ depending on whether $d_j < 0$.

The value of f can be then expressed as the sum of the contributions from all stations such that:

$$f \;=\; \sum_{i \in \bar{S}} \sum_{j \in S} \bar{x}_{ij} + \sum_{j \in S : d_j > 0} \frac{d_j}{Q} + \sum_{j \in \bar{S} : d_j < 0} \frac{-d_j}{Q} \;=\; x(\bar{S} : S) + \frac{d^+(S)}{Q} - \frac{d^-(\bar{S})}{Q}. \tag{33}$$

Therefore, the procedure is correct considering the fact that the left-hand side of (32) is equivalent to the value of the min-cut. □

To separate constraints (29), procedure FracSepCap builds a second support graph $\bar{\mathcal{G}}''$. The only differences are that, in $\bar{\mathcal{G}}''$, the dummy vertex $n+1$ is connected to each delivery station $i \in \mathcal{I} : d_i < 0$ by an arc with capacity $-d_i/Q$, and each pickup station $i \in \mathcal{I} : d_i > 0$ is connected to vertex $n+2$ by an arc with capacity d_i/Q. When the resulting min-cut value is less than $d^+(\mathcal{V})/Q$, constraints (29) are violated by the induced set S. Note that the proof of correctness of this second procedure follows the same steps as for Proposition 2.

Additionally, we run separation procedures considering two specific demand realizations, as follows. Let $\tilde{d}^+_{max} \in \mathcal{U}(\Gamma)$ be the demand realization in which $\xi_i = 1$ for all $i \in S_{max}$, where $S_{max} \subseteq \mathcal{I}$, $|S_{max}| = \Gamma$ and $\sum_{i \in S_{max}} \hat{d}_i$ is the largest. This is a particular realization in which the demands of the Γ stations with the largest demand deviations attain their worst case. Since the load in the vehicle is non-monotonic in the RBRP, this realization may not be the one yielding the worst-case vehicle load (as it would be in other VRP variants, such as the VRPTW). However, ensuring robust-feasibility regarding this realization is still a necessary condition. A similar observation is valid for realization $\tilde{d}^-_{max} \in \mathcal{U}(\Gamma)$ when assuming $\xi_i = -1$ for all $i \in S_{max}$. We use these observations to derive a third separation procedure, called FracSepCapPlus. In this procedure, we initially run FracSepCap with the demand of each station $i \in S_{max}$ set to $d_i + \hat{d}_i$ and adjusting the value of d_0 accordingly. If a violation is found, a cut in the form of Constraints (23) is added considering $S = S_{max}$. After that,

the procedure runs FracSepCap for a second time by adjusting the demand of each station $i \in S_{max}$ to $d_i - \hat{d}_i$.

Finally, as in Dell'Amico et al. (2014), the separation of the tournament constraints (27) is performed by a path extension algorithm. This procedure, called FracSepTournament, starts by creating a path R that, initially, only contains the depot. Then, by using a depth-first strategy, the path is iteratively extended to the next vertex $i \in \mathcal{I}$ such that arc $(R_{|R|}, i)$ has a positive value in $\bar{x}$. Because the initial load of the vehicle is only known when the path is complete, the procedure always keeps track of the current initial load required and updates it whenever a new vertex is added. When the path becomes infeasible with respect to the vehicle capacity, a tournament cut is added, and the procedure backtracks. Furthermore, whenever the current path has no chance of violating constraints (27) (i.e., $\sum_{(i,j)\in A(R)} \bar{x}_{ij} < |A(R)|-1$), the algorithm backtracks to the previous vertex.

6. Computational Results

In this section, we show the results of computational experiments that verify the performance of the proposed approaches and their relevance to decision-making under uncertainty in BSSs. Section 6.1 shows the results of experiments with different configurations of our B&C algorithm, whereas Section 6.2 compares the performance of the proposed compact model against the performance of the best configuration of the B&C algorithm. Finally, Section 6.3 presents a robustness analysis of the solutions provided by the RBRP under demand uncertainty. For the sake of clarity, the tables in this section present summarized results. Detailed results can be found at the website `http://www.dep.ufscar.br/munari/rbrp`.

All approaches were coded in C++ and executed on an Intel Xeon(R) E5-2650 2.20GHz computer, with 128 GB of RAM and the Ubuntu 16.04.6 LTS operating system. The mathematical formulations and the B&C algorithm were implemented using IBM CPLEX Optimization Studio v.22.1. A time limit of one hour was set for each experiment and CPLEX was set to default options on a single thread.

We derived a set of benchmark instances based on the set proposed by Dell'Amico et al. (2014). The original set is composed of 65 instances generated based on real data collected from 22 BSS operators located all around the world. Similarly to Dell'Amico et al. (2014), we consider an unlimited fleet. To model uncertainty, the demand deviations are defined as $\hat{d}_i = \lceil |d_i| \times \alpha \rceil$ for each station $i \in \mathcal{I}$ and $\alpha \in \{0.1, 0.25, 0.5\}$, where d_i is the nominal demand of station i provided in the instance. Hence, we consider deviations of 10%, 25% and 50% from the nominal demands. Considering that split demands are not allowed, if $|d_i| + \hat{d}_i > Q$ for any station $i \in \mathcal{I}$, that instance is considered infeasible and is discarded. We consider the budget values $\Gamma \in \{0, 1, 5, 10\}$, where $\Gamma = 0$ corresponds to the deterministic case. The resulting benchmark set comprises a total of 515 instances. For the sake of clarity, instances are grouped into subsets called G1 ($n = 13, \ldots, 21$), G2 ($n = 23, \ldots, 45$) and G3 ($n = 51, \ldots, 80$) according to their number of stations.

6.1. Computational Performance of the B&C Algorithm

The first round of experiments assesses the effectiveness of the separation procedures presented in Section 5, highlighting the best configuration for the proposed B&C algorithm. Recall that the separation procedures considering candidate incumbent solutions are mandatory to ensure feasibility, whereas the procedures for fractional solutions are optional. Moreover, the order in which these procedures are called is important because, once a procedure finds at least one valid cut, the separation process is finished and no other procedure is called in this round of cut generation. Table 1 presents the configurations tested. The procedures are shown in the same order as they are called.

Table 1: Configurations for the proposed B&C algorithm

ID	Separation procedures for fractional solutions
B&C Incumbent	No separation for fractional solutions
B&C FracSepST	FracSepST
B&C FracSepCap	FracSepST, FracSepCap
B&C FracSepCapPlus	FracSepST, FracSepCapPlus
B&C FracSepTour	FracSepST, FracSepCapPlus, FracSepTour

Table 2 summarizes the results of the experiments with the different configurations of the B&C algorithm. For these experiments, the separation of cuts considering fractional solutions was performed at the root node only. Column *Group* identifies the instance group, while columns Γ and *Dev* present the values of the budget of uncertainty and deviations, respectively. Column *#Feas* specifies the number of feasible instances associated with the respective group, and values of Γ and *Dev*. Then, for each configuration of the algorithm, column *Gap* presents the average percentage gap (calculated as $(UB - LB)/LB \times 100$, where UB and LB represent the primal and dual bounds, respectively); column *Time* shows the average computing time (in seconds); and column *#Opt* gives the number of instances solved to optimality. The best results are highlighted in bold. Additionally, the expression *t.lim.* specifies that the time limit was reached.

The results in Table 2 show that separating cuts for fractional solutions is worthwhile and even the simplest configuration can yield improvements. Notably, on average, the best configuration over all aspects is B&C FracSepCapPlus, which proved optimality for approximately 5% more instances than B&C Incumbent. The largest improvements can be seen in the gap values of B&C FracSepCapPlus, which are overall about 21% better than B&C Incumbent. Furthermore, there is a consistent improvement from one configuration to the next as more separation procedures are considered. The only exception is for B&C FracSepTour, in which the separation of the tournament constraints (27) is included. The results imply that separating these constraints can be detrimental to the overall performance, which might be explained by the fact that they are relatively weak.

An interesting observation from the results in Table 2 is that the gain in performance from separating cuts for fractional solutions is considerably larger in the deterministic case ($\Gamma = 0$). On average, the overall gap for B&C Incumbent when $\Gamma = 0$ across all instances is about 22%. In turn, for B&C FracSepST it drops to about 13%, which corresponds to

Table 2: Comparison of different configurations of the B&C algorithm.

Group	Γ	Dev	#Feas	B&C Incumbent			B&C FracSepST			B&C FracSepCap			B&C FracSepCapPlus			B&C FracSepTour		
				Gap (%)	Time (s)	#Opt	Gap (%)	Time (s)	#Opt	Gap (%)	Time (s)	#Opt	Gap (%)	Time (s)	#Opt	Gap (%)	Time (s)	#Opt
G1	0	0	23	**0.0**	120.3	**23**	**0.0**	**4.7**	**23**	**0.0**	**4.7**	**23**	**0.0**	6.2	**23**	**0.0**	**4.7**	**23**
	1	10	20	**0.2**	192.6	**19**	0.2	197.8	**19**	0.2	**187.5**	**19**	0.2	188.9	**19**	0.2	197.8	**19**
	1	25	20	**0.1**	189.2	**19**	0.1	196.8	**19**	0.1	**184.7**	**19**	0.1	187.0	**19**	0.1	196.8	**19**
	1	50	20	3.4	547.0	**17**	**3.3**	544.2	**17**	3.4	**540.9**	**17**	3.5	541.0	**17**	**3.3**	544.2	**17**
	5	10	20	6.6	913.1	15	7.2	911.2	15	**6.2**	845.7	**16**	6.5	**734.4**	**16**	7.2	911.2	15
	5	25	20	8.1	1010.2	**15**	8.1	1041.6	**15**	**7.2**	980.4	**15**	7.3	**920.2**	**15**	8.1	1041.6	**15**
	5	50	20	10.4	**1343.0**	**13**	10.7	1396.2	**13**	9.8	1373.5	**13**	**8.8**	1379.3	**13**	10.7	1396.2	**13**
	10	10	20	10.6	1444.6	12	10.1	1395.5	**13**	9.0	1306.8	**13**	**8.2**	**1267.0**	**13**	10.1	1395.5	**13**
	10	25	20	11.5	1354.5	13	11.4	1342.0	13	10.1	1301.7	13	**8.9**	**1096.6**	**14**	11.4	1342.0	13
	10	50	20	12.1	1843.8	11	11.7	1832.3	11	11.7	1804.2	11	**9.2**	**1705.9**	**12**	11.7	1832.3	11
Avg./sum	-	-	203	6.2	884.4	157	6.2	873.2	158	5.7	840.5	159	**5.2**	**790.9**	**161**	6.2	873.2	158
G2	0	0	18	8.7	1431.7	11	2.0	**825.0**	**14**	2.0	**825.0**	**14**	**1.9**	828.8	**14**	2.0	**825.0**	**14**
	1	10	14	10.2	1702.2	8	10.2	1765.0	8	3.9	1561.2	**9**	**3.3**	**1363.9**	**9**	10.2	1765.0	8
	1	25	13	13.5	1798.6	7	13.0	1758.0	7	7.3	1677.7	7	**6.8**	**1489.8**	**8**	13.0	1758.0	7
	1	50	11	12.9	**1952.6**	**6**	14.3	1998.2	**6**	8.1	2001.6	5	**7.5**	1998.6	5	14.3	1998.2	**6**
	5	10	14	21.0	2834.4	3	20.4	2836.6	3	**15.7**	2594.9	**5**	16.6	**2575.3**	**5**	20.4	2836.6	3
	5	25	13	25.3	**3027.5**	**3**	26.5	3046.4	2	20.7	3046.3	2	**20.6**	3046.4	2	26.5	3046.4	2
	5	50	11	34.4	3315.5	**1**	34.4	3295.0	**1**	33.8	**3287.8**	**1**	**31.2**	3292.5	**1**	34.4	3295.0	**1**
	10	10	14	30.0	**3118.9**	**2**	31.8	3187.3	**2**	25.8	3186.9	**2**	**24.2**	3225.7	**2**	31.8	3187.3	**2**
	10	25	13	36.8	3323.9	**1**	33.7	3326.6	**1**	31.4	**3323.5**	**1**	**27.0**	3323.6	**1**	33.7	3326.6	**1**
	10	50	11	44.0	t.lim.	**0**	43.5	t.lim.	**0**	41.2	t.lim.	**0**	**38.2**	t.lim.	**0**	43.5	t.lim.	**0**
Avg./sum	-	-	132	22.7	2548.9	42	21.8	2480.5	44	17.9	2424.6	46	**16.7**	**2387.9**	**47**	21.8	2480.5	44
G3	0	0	24	54.0	3020.5	4	35.5	2935.0	**5**	35.5	2935.0	**5**	**34.2**	**2918.8**	**5**	35.5	2935.0	**5**
	1	10	21	56.6	3245.6	**4**	58.8	3303.9	2	**39.4**	**3132.6**	3	41.0	3156.3	**4**	58.8	3303.9	2
	1	25	17	57.5	3388.6	1	57.3	3388.7	1	45.2	3388.6	1	**41.2**	**3305.2**	**2**	57.3	3388.7	1
	1	50	14	63.7	**3510.5**	**1**	61.7	3585.9	**1**	49.7	3574.5	**1**	**47.6**	3556.7	**1**	61.7	3585.9	**1**
	5	10	21	70.0	3433.9	**1**	71.6	3440.8	**1**	59.6	3429.3	**1**	**58.0**	**3429.3**	**1**	71.6	3440.8	**1**
	5	25	17	70.1	t.lim.	**0**	71.2	t.lim.	**0**	61.4	t.lim.	**0**	**60.5**	**3598.2**	**0**	71.2	t.lim.	**0**
	5	50	14	76.5	t.lim.	**0**	76.1	t.lim.	**0**	**68.9**	t.lim.	**0**	69.2	t.lim.	**0**	76.1	t.lim.	**0**
	10	10	21	73.9	t.lim.	**0**	74.7	t.lim.	**0**	63.3	t.lim.	**0**	**61.6**	**3569.4**	**0**	74.7	t.lim.	**0**
	10	25	17	73.2	t.lim.	**0**	75.6	t.lim.	**0**	68.0	t.lim.	**0**	**63.1**	t.lim.	**0**	75.6	t.lim.	**0**
	10	50	14	79.2	t.lim.	**0**	78.3	t.lim.	**0**	71.5	t.lim.	**0**	**67.6**	t.lim.	**0**	78.3	t.lim.	**0**
Avg./sum	-	-	180	66.6	3435.2	11	64.7	3437.3	10	54.9	3415.0	11	**53.2**	**3402.6**	**13**	64.7	3437.3	10

approximately 39% of improvement. On the other hand, when $\Gamma > 0$, the improvements are less significant. This difference in performance is partially due to the inherited complexity associated with the robust case, but it might also be explained by the fact that most of the separation procedures for fractional solutions are tailored to deterministic solutions.

Our next set of experiments aims at empirically determining the maximum depth in the branch-and-bound tree, to which fractional solutions should be considered in the separation procedures. In these experiments, the separation of fractional solutions follows the best configuration indicated by the results in Table 2, namely B&C FracSepCapPlus. The results are summarized in Table 3 considering all instances in the groups G1, G2 and G3. The meaning of columns *Γ*, *Dev* and *#Feas* is the same as in Table 2. The next columns refer to the B&C algorithm with different choices of the maximum depth for cut separation using fractional solutions. In *Root only*, fractional solutions were separated only at the root node, whereas in *Max depth = 5* and *Max depth = 10*, they were separated up to the fifth and tenth levels of the B&C tree, respectively. Finally, *Unlimited* means that the separation considering fractional solutions was called in all levels of the tree. The columns *Gap*, *Time* and *#Opt* have the same meaning as in Table 2.

Table 3: Results of the B&C algorithm with different choices of maximum depth for separating cuts using fractional solutions.

Γ	Dev	#Feas	Root only			Max depth = 5			Max depth = 10			Unlimited		
			Gap (%)	Time (s)	#Opt	Gap (%)	Time (s)	#Opt	Gap (%)	Time (s)	#Opt	Gap (%)	Time (s)	#Opt
0	0	65	13.2	1309.4	42	12.9	1124.2	45	10.9	1080.2	46	**8.7**	**925.4**	**51**
1	10	55	16.6	1621.0	32	**14.4**	1439.1	34	15.1	1442.9	34	14.9	**1383.7**	**35**
1	25	50	15.8	1585.9	29	15.9	1505.5	**30**	**14.7**	1500.7	**30**	14.8	**1489.5**	**30**
1	50	45	18.2	1835.5	23	18.1	**1571.0**	**26**	**16.3**	1591.7	**26**	17.9	1600.9	25
5	10	55	28.7	2232.0	22	**27.1**	**2139.4**	**23**	28.4	2160.4	22	29.1	2159.9	**23**
5	25	50	28.8	2383.5	17	**27.4**	2220.2	19	28.1	2192.0	**20**	28.4	**2179.6**	**20**
5	50	45	33.1	2537.9	**14**	32.2	**2490.4**	**14**	**31.9**	2498.8	**14**	33.3	2500.9	**14**
10	10	55	32.7	2644.7	15	**31.5**	2592.0	**16**	31.9	**2576.5**	**16**	32.8	2604.7	**16**
10	25	50	32.1	2526.9	15	31.1	2456.1	**16**	**30.5**	2453.7	**16**	32.1	**2453.6**	**16**
10	50	45	34.5	**2758.3**	**12**	32.8	2783.8	11	**32.6**	2780.8	11	33.2	2790.9	11
Avg./sum		515	24.9	2113.1	221	23.9	1999.2	234	**23.6**	1993.5	235	24.0	**1970.1**	**241**

The results in Table 3 suggest that, on average, the *unlimited* configuration has a better performance in terms of computing times and the number of instances solved to optimality. By following this strategy, the proposed B&C is around 7% faster and finds 8% more optimal solutions than its version separating fractional solutions at the root node only. Notably, setting the maximum depth to 10 leads to better average gaps overall.

A closer look at Table 3 reveals interesting observations. For example, the configuration with maximum depth equal to 5 seems to present slightly better results for $\Gamma = 1$ and $Dev = 50$, and $\Gamma = 5$ with deviations of $10\%, 25\%$ and 50%. On the other hand, setting the maximum depth to 10 seems to yield marginally better results for $\Gamma = 10$ when compared to the configurations *unlimited* and *root only*. In the deterministic case, when $\Gamma = 1$, the *unlimited* configuration results in improved performance in terms of computing time and the number of optimal solutions, without significantly degrading gap values.

6.2. Compact Formulation and the Best B&C Configuration

Table 4 presents a comparison between the compact formulation and our best B&C algorithm, given by B&C FracSepCapPlus with no restrictions on the maximum depth for the separation of cuts using fractional solutions. Columns have the same meaning as in the previous tables.

Table 4: Comparison of the results obtained with the compact formulation and the best configuration of the B&C algorithm.

Group	Γ	Dev	#Feas	Compact formulation			B&C		
				Gap (%)	Time (s)	#Opt	Gap (%)	Time (s)	#Opt
G1	0	0	23	0.69	319.47	21	**0.00**	**0.08**	**23**
	1	10	20	0.64	473.63	18	**0.18**	**206.90**	**19**
	1	25	20	1.16	753.14	17	**0.10**	**186.14**	**19**
	1	50	20	4.09	931.03	15	**3.51**	**540.50**	**17**
	5	10	20	8.65	1319.00	13	**6.62**	**738.03**	**16**
	5	25	20	8.39	1387.40	14	**7.81**	**929.64**	**15**
	5	50	20	12.05	1870.68	11	**10.42**	**1305.97**	**13**
	10	10	20	12.77	1801.25	11	**7.84**	**1266.51**	**13**
	10	25	20	12.37	1921.55	11	**8.53**	**1088.51**	**14**
	10	50	20	16.76	2610.01	7	**9.89**	**1779.34**	**11**
Avg./sum	-	-	203	7.65	1323.65	138	**5.41**	**792.28**	**160**
G2	0	0	18	11.14	2052.13	9	**0.00**	**28.75**	**18**
	1	10	14	12.00	2483.68	5	**0.40**	**685.97**	**12**
	1	25	13	15.41	2543.02	4	**4.18**	**1119.09**	**9**
	1	50	11	12.94	2373.57	4	**3.71**	**984.42**	**8**
	5	10	14	21.65	3199.08	2	**12.11**	**2286.42**	**6**
	5	25	13	24.44	3287.46	2	**16.37**	**2244.93**	**5**
	5	50	11	33.40	t.lim.	0	**28.19**	**3274.24**	**1**
	10	10	14	29.73	t.lim.	0	**19.02**	**3023.26**	**3**
	10	25	13	34.57	t.lim.	0	**22.09**	**3054.29**	**2**
	10	50	11	42.00	t.lim.	0	**28.10**	t.lim.	0
Avg./sum	-	-	132	22.93	2990.98	26	**12.54**	**1926.82**	**64**
G3	0	0	24	34.18	t.lim.	0	**23.62**	**2484.74**	**10**
	1	10	21	42.10	t.lim.	0	**38.61**	**2969.73**	**4**
	1	25	17	**39.65**	t.lim.	0	40.16	**3306.26**	**2**
	1	50	14	**44.00**	t.lim.	0	49.76	t.lim.	0
	5	10	21	**59.24**	t.lim.	0	61.90	**3429.67**	**1**
	5	25	17	**61.36**	t.lim.	0	61.81	t.lim.	0
	5	50	14	**69.79**	t.lim.	0	69.89	t.lim.	0
	10	10	21	73.18	t.lim.	0	**65.86**	t.lim.	0
	10	25	17	71.62	t.lim.	0	**67.53**	t.lim.	0
	10	50	14	77.79	t.lim.	0	**70.51**	t.lim.	0
Avg./sum	-	-	180	56.12	t.lim.	0	**53.36**	**2497.71**	**17**

The results indicate that the performance of both approaches is affected by the level of uncertainty in the instance. The larger the budget of uncertainty, the more challenging the instances become, on average. We observe a similar trend for the deviation values. Even for $\Gamma = 1$ and a demand deviation of 10%, we see a significant impact on the number of instances solved to optimality with respect to the deterministic case. For the B&C, we observe a considerable increase in the running time in the same comparison, indicating that the RBRP under demand uncertainty is more challenging than its deterministic counterpart.

The compact formulation can be efficiently used to solve small-sized instances such as the ones of group G1, for which it was able to prove optimality for 138 instances. This is a good result as instances in group G1 model real-world BSSs, showing that this formulation is a

simple, yet viable option for similar cases. As expected, the efficiency of the MIP solver with this formulation was severely impaired when considering larger instances though, proving only 26 optimal solutions for group G2 and none for group G3. If we consider the optimal primal bounds proved by the B&C, it can be observed that, although it was not able to close the gap, the compact formulation found 41 additional optimal solutions.

Notably, the B&C algorithm shows a superior performance in all groups of instances and it proved optimality for almost 50% more instances than the compact formulation. In terms of computational time, there is also a significant difference between both approaches, especially in groups G2 and G3. Nevertheless, it is important to emphasize that both approaches have their value in practice. Despite being outperformed, the compact formulation can still be relevant for OR practitioners as it is efficient for small-sized instances, does not require advanced coding techniques, and can be solved by any general-purpose integer programming software.

6.3. Robustness analysis

We now analyze the relevance of incorporating uncertainty into the design of routes for rebalancing operations in BSSs. To do so, we run Monte Carlo simulations to verify the robustness of the solutions provided by the RBRP under demand uncertainty. By robustness of a solution, we mean how protected this solution is against the realization of random variables. For the simulations, we generated 10,000 uniformly random realizations (scenarios) of the demand in the interval $[d_i - \hat{d}_i, d_i + \hat{d}_i], \forall i \in \mathcal{I}$. Then, the feasibility of each solution is tested against each demand realization. The percentage of realizations in which this solution is infeasible gives the empirical probability of constraint violation, or *risk*.

The solutions of the RBRP are expected to be feasible with a greater probability in comparison to the solutions obtained from its deterministic counterpart (BRP), especially for large deviations. In addition, we expect that gains in terms of feasibility can counterbalance possibly increasing routing costs caused by the incorporation of robustness. With this in mind, the following metrics were used to evaluate each solution of the RBRP: (*i*) the price of robustness (PoR), computed as the relative change in the objective function value of an RBRP solution with respect to its deterministic counterpart, given a deviation α and budget Γ; (*ii*) the risk, calculated as the relative frequency of infeasible solutions among the total number of demand realizations given by the Monte Carlo simulations.

Table 5 presents the results of the Monte Carlo simulations for the solutions obtained with the best configuration of the B&C algorithm (see Section 6.1). For each combination of $\Gamma \in \{0, 1, 5, 10\}$ and demand deviation (10, 25 and 50%), this table shows average values of PoR and risk. It also presents the overall results for each Γ, considering all deviations. To ensure a fair comparison, especially for measuring the PoR, we only consider instances solved to optimality for both the robust and deterministic cases. The results presented in the table confirm that the deterministic approach (i.e., $\Gamma = 0$), is ineffective in safeguarding solutions against uncertainties. On average, there is approximately a 73% chance that solutions become infeasible, even in the presence of minor demand deviations. If we exclude small-sized instances (group G1), this risk increases to about 87%.

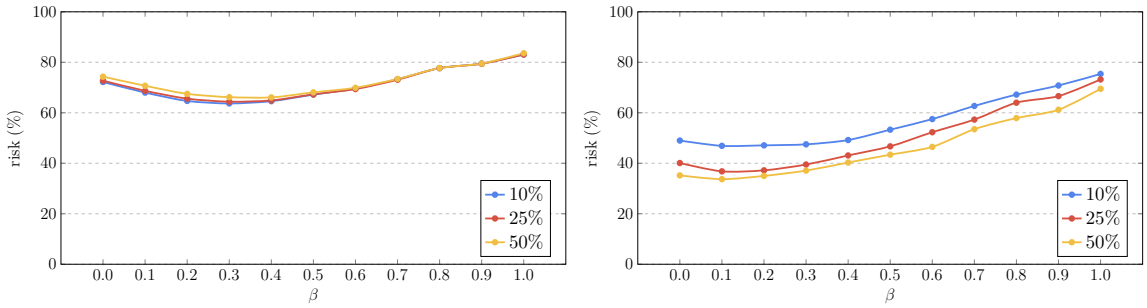
Table 5: Average price of robustness (PoR) and probability of constraint violation (risk) considering different values of budget of uncertainty and deviation

Γ	10%		25%		50%		Overall	
	PoR (%)	Risk (%)	PoR (%)	Risk (%)	PoR (%)	Risk (%)	PoR (%)	Risk (%)
0	—	72.18	—	72.72	—	74.33	—	73.03
1	2.32	47.59	3.40	39.22	5.48	34.20	3.56	41.03
5	6.60	3.96	10.75	2.24	17.40	0.48	10.71	2.50
10	10.38	0.06	14.09	0.02	25.33	0.00	15.58	0.03

Conversely, the solutions of the RBRP promote significant risk reductions. Even in cases where the number of stations deviating from their nominal demand is small (i.e., $\Gamma = 1$), the overall risk decreases to approx. 41% with an increase in solution costs of around ∼3%, on average. Notably, the risk does not seem to be as sensitive to changes in the demand deviations for a fixed Γ value. A larger budget ($\Gamma = 5$) causes a substantial decrease in risk but at the expense of a larger solution cost, which rises to $\sim$ 11%. Although it is possible to reach close-to-zero risk by incrementally increasing the value of Γ, in real-world applications, there comes a point where decreasing risk will not compensate for the growing PoR. In practice, it is up to the decision-maker to assess the trade-off between mitigating risk at the expense of routing costs in order to find the best compromise for their business, and the proposed approaches are valuable tools in this context.

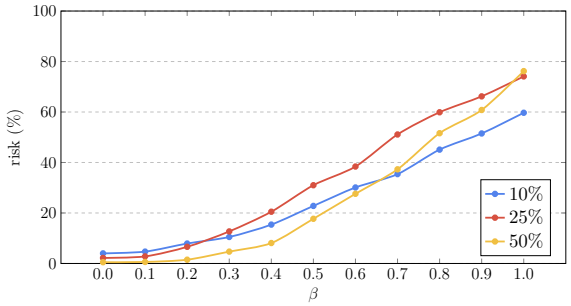
In what follows, we broaden our discussion by delving into an interesting insight gained from a careful analysis of our solutions. In the RBRP, each vehicle usually departs from the depot with the minimal amount of bikes needed to fulfill the demand of each station in its route, while accounting for a possible budget of uncertainty. This is due to the calculation of the initial load which simultaneously considers positive and negative deviations as shown in Section 3. At the same time, it is likely that vehicles will not depart with a full load from the depot. In this case, any increment in the initial load would not impact a solution's feasibility or its cost. Clearly, in practice, sending a vehicle with additional bikes would result in additional effort for the working crew. However, considering that the load of vehicles is non-monotonic in the RBRP, there might be benefits in doing so regarding the risk measure. To investigate that, we performed a second round of simulations in which the initial load of the vehicle (θ^d) for each route is modified to $\theta^d = \theta^d + (Q - \theta^d) \times \beta$, where $\beta \in [0, 1]$. The results of these simulations are presented in Figure 3, in which we compare the associated risk for different values of β and $\alpha = \{10, 25, 50\}$.

Figure 3a shows that it is possible to reduce the overall average risk in the deterministic case from ∼72% to ∼64% by simply modifying the initial load of vehicles by a factor of $\beta = 0.3$. This result is not entirely surprising considering that, in the deterministic case, there is often little margin for demand variations, and any preemptive measure to add more flexibility to initial vehicle loads can potentially result in positive outcomes. On the other hand, the benefits of changing initial loads fade as we increase the budget of uncertainty. Indeed, for $\Gamma = 5$, there is almost no benefit at all at carrying more bikes from the depot, and any value of $\beta \geq 0.4$ results in significantly larger risk values. This suggests that robust-

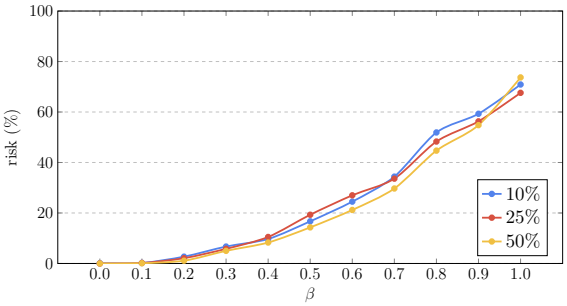


(a) Deterministic case

(b) $\Gamma = 1$



(c) $\Gamma = 5$



(d) $\Gamma = 10$

Figure 3: Results of Monte Carlo simulations to investigate the impact of changing the initial load of each vehicle.

feasible solutions are usually protected enough so that the decision-maker is not required to perform additional changes to initial loads.

7. Conclusions

We introduced the Robust Bike Sharing Rebalancing Problem (RBRP), an extension of the BRP that takes into account uncertainties on demand. One of the characteristics that make the RBRP a challenging problem is that demand realizations at each station may completely differ from their forecast (nominal) values. Therefore, delivery stations may become pickup stations due to such variations, making a route planned according to the demand forecast infeasible in practice.

We proposed two different formulations for the RBRP. The first one is a compact formulation, derived from the linearization of recursive equations that model the vehicles' load along the routes. This linearization technique allowed us to tackle the complex RBRP by using MTZ-like constraints to represent robustness. The second formulation is based on cutting planes that ensure robust feasibility and better LP relaxation lower bounds. A B&C algorithm tailored for the RBRP was proposed to solve this second formulation.

A number of computational experiments were performed on a set of instances in order to assess the efficiency of the proposed solution approaches. Results from the first set of experiments revealed the best configuration for the B&C algorithm. Next, we compared the compact formulation with the best B&C configuration. Results from this second set of experiments demonstrated that the compact formulation can be used to efficiently solve

small-sized instances with up to 21 stations. This indicates the importance of this formulation as it represents a simple yet effective solution method for practitioners. On the other hand, the B&C algorithm showed superior performance, as expected, in all groups of instances.

From an application standpoint, results from Monte Carlo simulations attested to the relevance of applying robustness in the context of the RBRP. In summary, we observed that it is possible to protect solutions against uncertainty up to a certain level without significantly increasing routing costs. Moreover, in a second round of Monte Carlo simulations, we showed that solutions in the deterministic case can be better protected from uncertainties by simply varying the initial load of the vehicles. However, the benefit of such a strategy decreases as the budget of uncertainty increases, showing the importance of considering uncertainties during the route planning phase.

Future research may focus on different sources of uncertainties in the RBRP and related problems, bringing more insights regarding the importance of robust solutions in decision-making for BSSs. We believe that a greater focus on the development of robust-feasibility cuts could improve the computational performance of the proposed B&C. In addition, this research motivates the development of other types of exact and heuristic algorithms, based on, e.g., branch-and-price and metaheuristics, respectively.

Acknowledgments

This research was partially supported by: National Council for Scientific and Technological Development (CNPq) [grant numbers 406245/2021-5, 405702/2021-3, 313220/2020-4]; Paraíba State Research Foundation (FAPESQ) [grant number 261/2020]; São Paulo Research Foundation (FAPESP) [grant numbers 13/07375-0, 19/23596-2, 22/05803-3]; and Universidade Federal da Paraíba through the Public Call n. 03/2020 "Produtividade em Pesquisa PROPESQ/PRPG/UFPB", proposal code PVL13394-2020.

References

Agra A, Christiansen M, Figueiredo R, Hvattum LM, Poss M, Requejo C, 2013 *The robust vehicle routing problem with time windows. Computers & Operations Research* 40(3):856–866.

Ascheuer N, Fischetti M, Grötschel M, 2000 *A polyhedral study of the asymmetric traveling salesman problem with time windows. Networks* 36(2):69–79.

Battarra M, Cordeau JF, Iori M, 2014 *Chapter 6: Pickup-and-delivery problems for goods transportation. Vehicle Routing: Problems, Methods, and Applications, Second Edition*, 161–191 (SIAM).

Ben-Tal A, Ghaoui LE, Nemirovski A, 2009 *Robust optimization* (Princeton university press).

Berbeglia G, Cordeau JF, Gribkovskaia I, Laporte G, 2007 *Static pickup and delivery problems: a classification scheme and survey. Top* 15(1):1–31.

Bertsimas D, Dunning I, Lubin M, 2016 *Reformulation versus cutting-planes for robust optimization. Computational Management Science* 13(2):195–217.

Bertsimas D, Sim M, 2004 *The price of robustness. Operations Research* 52(1):35–53.

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

Bruck BP, Subramanian A, 2020 *Bike-Sharing Rebalancing Problems*, 1–9 (Cham: Springer International Publishing).

Bruck BP, Subramanian A, 2023 *Bike-Sharing Rebalancing Problems*, 1–9 (Cham: Springer International Publishing).

Bulhões T, Subramanian A, Erdoğan G, Laporte G, 2018 *The static bike relocation problem with multiple vehicles and visits. European Journal of Operational Research* 264(2):508–523.

Büsing C, Gersing T, Koster AM, 2023 *A branch and bound algorithm for robust binary optimization with budget uncertainty. Mathematical Programming Computation* 15:269–326.

Campos R, Munari P, Coelho LC, 2022 *Compact formulations for the robust vehicle routing problem with time windows under demand and travel time uncertainty.* Technical report, CIRRELT-2022-34, Faculté des Sciences de l'Administration, Université Laval, CIRRELT, Canada.

Casazza M, Ceselli A, Calvo RW, 2021 *A route decomposition approach for the single commodity split pickup and split delivery vehicle routing problem. European Journal of Operational Research* 289(3):897–911.

Casazza M, Ceselli A, Chemla D, Meunier F, Calvo RW, 2018 *The multiple vehicle balancing problem. Networks* 72(3):337–357.

Cavagnini R, Bertazzi L, Maggioni F, Hewitt M, 2018 *A two-stage stochastic optimization model for the bike sharing allocation and rebalancing problem.* Technical report, Working paper.

Chemla D, Meunier F, Calvo RW, 2013 *Bike sharing systems: Solving the static rebalancing problem. Discrete Optimization* 10(2):120–146.

Contardo C, Morency C, Rousseau LM, 2012 *Balancing a dynamic public bike-sharing system.* Technical report, CIRRELT-2012-09, Montréal, Canada.

Cruz F, Subramanian A, Bruck BP, Iori M, 2017 *A heuristic algorithm for a single vehicle static bike sharing rebalancing problem. Computers and Operations Research* 79:19–33.

De La Vega J, Munari P, Morabito R, 2020 *Exact approaches to the robust vehicle routing problem with time windows and multiple deliverymen. Computers & Operations Research* 124:105062.

Dell'Amico M, Hadjicostantinou E, Iori M, Novellani S, 2014 *The bike sharing rebalancing problem: Mathematical formulations and benchmark instances. Omega* 45(0):7–19.

Dell'Amico M, Iori M, Novellani S, Stützle T, 2016 *A destroy and repair algorithm for the bike sharing rebalancing problem. Computers and Operations Research* 71:149–162.

Dell'Amico M, Iori M, Novellani S, Subramanian A, 2018 *The bike sharing rebalancing problem with stochastic demands. Transportation Research Part B: Methodological* 118:362–380.

Dror M, Fortin D, Roucairol C, 1998 *Redistribution of self-service electric cars: A case of pickup and delivery.* Technical report, INRIA.

Erdoğan G, Battarra M, Calvo RW, 2015 *An exact algorithm for the static rebalancing problem arising in bicycle sharing systems. European Journal of Operational Research* 245:667–679.

Erdoğan G, Laporte G, Calvo RW, 2014 *The static bicycle relocation problem with demand intervals. European Journal of Operational Research* 238:451–457.

Fu C, Zhu N, Ma S, Liu R, 2022 *A two-stage robust approach to integrated station location and rebalancing vehicle service design in bike-sharing systems. European Journal of Operational Research* 298(3):915–938.

Gendreau M, Jabali O, Rei W, 2014 *Stochastic vehicle routing problems. Vehicle routing: problems, methods, and applications, 2nd edn*, 213–239 (Society for Industrial and Applied Mathematics).

Gounaris CE, Wiesemann W, Floudas CA, 2013 *The robust capacitated vehicle routing problem under demand uncertainty. Operations Research* 61(3):677–693.

Gunes C, van Hoeve WJ, Tayur S, 2010 *Vehicle routing for food rescue programs: A comparison of different approaches. International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming*, 176–180 (Springer).

Hernández-Pérez H, Salazar-González JJ, 2004a *A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. Discrete Applied Mathematics* 145(1):126–139.

Hernández-Pérez H, Salazar-González JJ, 2004b *Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. Transportation Science* 38(2):245–255.

Hernández-Pérez H, Salazar-González JJ, 2018 *An exact algorithm for the split-demand one-commodity pickup-and-delivery travelling salesman problem. International Symposium on Combinatorial Optimization*, 241–252 (Springer).

Hernández-Pérez H, Salazar-González JJ, 2022 *A branch-and-cut algorithm for the split-demand one-commodity pickup-and-delivery travelling salesman problem. European Journal of Operational Research* 297(2):467–483.

Hernández-Pérez H, Salazar-González JJ, Santos-Hernández B, 2018 *Heuristic algorithm for the split-demand one-commodity pickup-and-delivery travelling salesman problem. Computers and Operations Research* 97:1–17.

Laporte G, Meunier F, Calvo RW, 2018 *Shared mobility systems: an updated survey. Annals of Operations Research* 271:105–126.

Lu CC, 2016 *Robust multi-period fleet allocation models for bike-sharing systems. Networks and Spatial Economics* 16:61–82.

Lu D, Gzara F, 2019 *The robust vehicle routing problem with time windows: Solution by branch and price and cut. European Journal of Operational Research* 275:925–938.

Maggioni F, Cagnolari M, Bertazzi L, Wallace SW, 2019 *Stochastic optimization models for a bike-sharing problem with transshipment. European Journal of Operational Research* 276(1):272–283.

Meddin R, 2023 *The bike-sharing world map.* `http://bikesharingworldmap.com`, accessed: August, 2023.

Miller CE, Tucker AW, Zemlin RA, 1960 *Integer programming formulation of traveling salesman problems. Journal of the ACM (JACM)* 7(4):326–329.

Munari P, Moreno A, De La Vega J, Alem D, Gondzio J, Morabito R, 2019 *The robust vehicle routing problem with time windows: compact formulation and branch-price-and-cut method. Transportation Science* 53(4):1043–1066.

Oyola J, Arntzen H, Woodruff DL, 2017 *The stochastic vehicle routing problem, a literature review, Part II: solution methods. EURO Journal on Transportation and Logistics* 6(4):349–388.

Oyola J, Arntzen H, Woodruff DL, 2018 *The stochastic vehicle routing problem, a literature review, Part I: models. EURO Journal on Transportation and Logistics* 7(3):193–221.

Pillac V, Gendreau M, Guéret C, Medaglia AL, 2013 *A review of dynamic vehicle routing problems. European Journal of Operational Research* 225(1):1–11.

Regue R, Recker W, 2014 *Proactive vehicle routing with inferred demand to solve the bikesharing rebalancing problem. Transportation Research Part E: Logistics and Transportation Review* 72:192–209.

Salazar-González JJ, Santos-Hernández B, 2015 *The split-demand one-commodity pickup-and-delivery travelling salesman problem. Transportation Research Part B: Methodological* 75:58–73.

Shui C, Szeto W, 2020 *A review of bicycle-sharing service planning problems. Transportation Research Part C: Emerging Technologies* 117:102648.

Si H, Shi J, Wu G, Chen J, Zhao X, 2019 *Mapping the bike sharing research published from 2010 to 2018: A scientometric review. Journal of Cleaner Production* 213:415–427.

Subramanyam A, 2023 *Robust vehicle routing problems. Encyclopedia of Optimization*, 1–7 (Springer).

Yu Q, Cheng C, Zhu N, 2022 *Robust team orienteering problem with decreasing profits. INFORMS Journal on Computing* 34(6):3215–3233.

Yuan M, Zhang Q, Wang B, Liang Y, Zhang H, 2019 *A mixed integer linear programming model for optimal planning of bicycle sharing systems: A case study in beijing. Sustainable Cities and Society* 47:101515.

Zeng B, Zhao L, 2013 *Solving two-stage robust optimization problems using a column-and-constraint generation method. Operations Research Letters* 41(5):457–461.

Zhang D, Yu C, Desai J, Lau H, Srivathsan S, 2017 *A time-space network flow approach to dynamic repositioning in bicycle sharing systems. Transportation Research Part B: Methodological* 103:188–207.