Titre: Title:	Dynamic Rebalancing Problems in Bike-sharing Systems
Auteur: Author:	Jiaqi Liang
Date:	2023
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Liang, J. (2023). Dynamic Rebalancing Problems in Bike-sharing Systems [Thèse de doctorat, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/56998/

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Dynamic Rebalancing Problems in Bike-sharing Systems

JIAQI LIANG
Département de mathématiques et de génie industriel

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

Décembre 2023

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Cette thèse intitulée :

Dynamic Rebalancing Problems in Bike-sharing Systems

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

Guy DESAULNIERS, président
Nadia LAHRICHI, membre et directrice de recherche
Andrea LODI, membre et codirecteur de recherche
Sanjay Dominik JENA, membre et codirecteur de recherche
Carolina OSORIO, membre
Manuel IORI, membre externe

ACKNOWLEDGEMENTS

Embarking upon this intensive Ph.D. journey in the esteemed halls of the Canada Excellence Research Chair has been transformative. The experience, however formidable, was made smoother and more insightful because of the individuals who joined me, either in spirit or in person, along the way. These years have been filled with countless cherished memories and touching moments.

At the academic forefront, I express my profound gratitude to my esteemed supervisors, Prof. Andrea Lodi and Prof. Sanjay Dominik Jena. Their dedication to academic excellence, combined with their ability to impart knowledge and stimulate critical thinking, have been pillars of my academic journey. Whenever I encountered obstacles in my research, they patiently steered me in the right direction, offering invaluable insights during the writing process, from which I've greatly benefited.

Being away from home has its challenges, but the unconditional love and support of my family back in China have been my anchor. To them, I owe a debt of gratitude that words can scarcely describe. Their encouragement and faith in me, even from thousands of miles away, have been the wind beneath my wings.

The tapestry of my Ph.D. journey has been colored by my friends, both near and far. To my friends in Canada – Khalid Lazziri, Federico Bobbio, Matteo Cacciola, Didier Chetelat, Alexandre Forel, Thibaut Vidal, Defeng Liu, Gabriele Dragotto, Leandro Rochink, Jie Gao, Claudia Bongiovanni, Tommaso Schettini, Jessica Budgell, Carlos Portillo, Linyan Wang, and Sophie Boddaert – and those back in China, including Xiaomei Kang, Haoran Li, Yaping Zhang, Ziyan Zhao, Junnan Zhu, Chunlong Yu, Dalei Guo, Yan Yan, Chenmeijing Liang, and Bowen Zhang, I extend heartfelt thanks. Their camaraderie, understanding, and countless moments of joy have made the rigorous Ph.D. journey lighter and more bearable. The distance did not diminish our bond but rather deepened it. Each one of you has played a unique role, turning both challenges into shared anecdotes and achievements into collective celebrations. Throughout my journey, they have been by my side. Whether I faced challenges or doubted myself, they were always there, offering help and encouragement.

I also want to express my appreciation to the faculty members in our chair and my fellows in the department, including Clara Martins, Maria Bazotte, Margarida Carvalho, Youssouf Emine, Can Li, Antoine Prouvost, Léa Ricard, Marius Roland, Mehdi Taobane, Chun Cheng, Xiangyi Zhang, Jingyi Zhao, Nicolau Andrea, and Claudio Sole. In the world of academia, it is rare to find peers who are not just co-researchers but also friends. Our brainstorming

sessions, late-night discussions, and shared cups of coffee have not only propelled my research forward but also made the journey enjoyable. Together, we created a warm, enjoyable, and productive research environment, which enabled me to complete my projects.

Heartfelt appreciation is extended to my lovely cat Bella, whose feline presence brought a sense of calm and comfort to my study sessions, and to my boyfriend Andrew Kyres, whose encouragement and unwavering support helped light the way forward.

In this grand tapestry of experiences and memories, each one of you has woven a unique thread, making my Ph.D. journey vibrant and meaningful. Thank you.

RÉSUMÉ

Les systèmes de partage de vélos (BSSs) se sont imposés comme un mode de transport urbain prédominant, offrant une alternative durable et économique pour les usagers. Néanmoins, la nature stochastique de la demande des utilisateurs et la forte demande pendant les heures de pointe entraînent souvent des déséquilibres d'utilisation aux stations lorsque les vélos ne sont pas disponibles là où ils sont nécessaires.

Cette thèse de doctorat se concentre sur le développement de stratégies de rééquilibrage dynamiques pour optimiser la distribution des vélos à travers les stations, améliorant ainsi la satisfaction des utilisateurs. La recherche englobe trois projets principaux comme suit :

Premièrement, nous proposons un modèle général de programmation linéaire en nombres entiers (PLNE) pour les problèmes de rééquilibrage multi-périodes qui peut être facilement adapté à différentes hypothèses de modélisation. Le modèle propose une stratégie de rééquilibrage pour chaque véhicule, minimisant la demande perdue, qui est la fonction objectif commune dans la littérature. Une analyse complète de diverses hypothèses de modélisation est présentée, incluant la discrétisation temporelle, la distribution des trajets, les domaines de variables et les séquences d'événements (séquences d'arrivées et de départs des clients, et de rééquilibrage des véhicules pendant chaque période). Nous développons un générateur d'instances pour synthétiser des réseaux de stations réalistes et des trajets clients, ainsi qu'un simulateur à granularité fine pour évaluer la performance opérationnelle des stratégies de rééquilibrage obtenues. Des expériences numériques approfondies sont réalisées, tant sur des données synthétiques que réelles, pour analyser l'efficacité des différentes hypothèses et techniques de modélisation. Cette analyse nous permet d'identifier les hypothèses de modélisation qui fournissent empiriquement les stratégies de rééquilibrage les plus efficaces.

Deuxièmement, nous intégrons les intervalles d'inventaire et les inventaires cibles, deux concepts souvent utilisés par les opérateurs de BSS, dans le modèle de rééquilibrage PLNE, présentant une approche plus pragmatique pour les opérateurs de BSS. Deux modèles, nommés DROB-I et DROB-T, sont proposés avec des fonctions objectifs innovantes pour minimiser les écarts par rapport à l'intervalle d'inventaire prévu et à l'inventaire cible, respectivement. Un algorithme d'apprentissage automatique est employé pour prévoir l'intervalle et la cible, en tenant compte de diverses caractéristiques telles que les conditions météorologiques, les données historiques de trajets et les informations temporelles. Trois modes de réoptimisation - statique, roulant et pliant - sont mis en œuvre et comparés entre les modèles, soulignant l'importance de la réoptimisation en réponse aux mises à jour du système. Des expériences

numériques approfondies sont réalisées, tant sur des données synthétiques que réelles, pour analyser l'efficacité des modèles proposés. Nos résultats montrent que DROB-I et DROB-T surpassent le modèle de base, apportant jusqu'à 34% de réduction sur la demande perdue pour la planification en horizon roulant en plus de temps d'exécution très rapides.

Troisièmement, nous proposons un algorithme basé sur l'apprentissage par renforcement pour les problèmes de rééquilibrage dynamique dans les BSS, formulé comme un processus de décision de Markov. Les espaces d'état et d'action sont conçus dans un cadre temporel continu, permettant une coopération fluide entre plusieurs véhicules de rééquilibrage pour améliorer le réalisme et l'efficacité du rééquilibrage. Pour saisir la dynamique et les incertitudes du système, un simulateur à granularité fine est présenté pour estimer les récompenses sous différents scénarios de demande sensibles aux conditions météorologiques et aux informations temporelles. L'apprentissage profond Q est utilisé pour apprendre la politique qui minimise la demande totale perdue. Des évaluations comparatives démontrent que notre algorithme surpasse les benchmarks, réduisant notamment la demande perdue de 21,5% par rapport à un modèle PLNE.

ABSTRACT

Bike-sharing systems (BSSs) have emerged as a prevalent mode of urban transportation, offering a sustainable and economical alternative for commuters. Nonetheless, the stochastic nature of user trips and high demand during peak hours often lead to station imbalances, where bikes or docks are not available where needed.

This Ph.D. thesis focuses on developing dynamic rebalancing strategies to optimize the distribution of bikes across stations, thus improving user satisfaction. The research encompasses the following three principal projects.

First, we propose a general Mixed-integer Programming (MIP) framework for multi-period rebalancing problems that can be easily adapted to different modeling assumptions. The model proposes a rebalancing strategy for each vehicle, minimizing lost demand, which is the common objective function in the literature. A comprehensive analysis of various modeling assumptions, including time discretization, trip distribution, variable domains, and event sequences (i.e., sequences of customer arrivals, departures, and vehicles' rebalancing during each time-period), is presented. We develop an instance generator to synthesize realistic station networks and customer trips, as well as a realistic fine-grained simulator to evaluate the operational performance of the obtained rebalancing strategies. Extensive numerical experiments are carried out, both on synthetic data and on real-world data, to analyze the effectiveness of various modeling assumptions and techniques. Such analysis allows us to identify the modeling assumptions that empirically provide the most effective rebalancing strategies.

Second, we incorporate inventory intervals and target inventories, two concepts that the BSS operators often use, into the MIP rebalancing model, presenting a more pragmatic approach for BSS operators. Two models, named DROB-I and DROB-T, are proposed with innovative objective functions to minimize deviations from the predicted inventory interval and target inventory, respectively. A Machine Learning algorithm is employed to forecast the interval and target, considering various features such as weather conditions, historical trip data, and temporal information. Three reoptimization modes – static, rolling, and folding planning – are implemented and compared across models, showcasing the significance of reoptimization in response to system updates. Extensive numerical experiments are carried out, both on the synthetic and real-world data, to analyze the effectiveness of the proposed models. Our results illustrate that DROB-I and DROB-T outperform the baseline model, especially reducing up to 34% of the lost demand in the rolling planning with swift runtimes.

Third, we propose a reinforcement learning-based algorithm for dynamic rebalancing problems in BSSs formulated as a Markov decision process. The state and action spaces are crafted within a continuous time framework, enabling seamless cooperation between multiple rebalancing vehicles to enhance the realism and efficiency of rebalancing. To capture the dynamics and uncertainties of the system, a fine-grained simulator is presented to estimate the rewards under different demand scenarios that are sensitive to weather conditions and temporal information. Deep Q learning is utilized to learn the policy, aiming to minimize the total lost demand. Comparative evaluations demonstrate that our algorithm outperforms benchmarks, particularly reducing the lost demand by 21.5% compared to an MIP model.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

BSSs	Bike-Sharing Systems
CP	Constraint Programming
DBRP	Dynamic Bicycle Repositioning Problem
DROB-I	**D**ynamic **R**ebalancing **O**ptimization for **B**SS based on **I**nventory Interval
DROB-LD	**D**ynamic **R**ebalancing **O**ptimization for **B**SS minimizing **L**ost Demand
DROB-T	**D**ynamic **R**ebalancing **O**ptimization for **B**SS based on **T**arget Inventory
DRL	Deep Reinforcement Learning
DQN	Deep Q-Network
GBT	Gradient Boosting Tree
LP	Linear Programming
MARL	Multi-Agent Reinforcement Learning
MDP	Markov Decision Process
MIP	Mixed-Integer Programming
MILP	Mixed-Integer Linear Programming
MINLP	Mixed-integer Nonlinear Programming
ML	Machine Learning
MMDP	Multi-agent Markov Decision Process
NN	Neural Network
PDP	Pick-up and Delivery Problem
RL	Reinforcement Learning
SBRP	Static Bicycle Repositioning Problem
VRP	Vehicle Routing Problem

CHAPTER 1 INTRODUCTION

Bike-Sharing Systems (BSSs) have rapidly ascended in popularity across the globe, serving as a pivotal solution to mitigate traffic congestion and curtail vehicular CO_2 emissions. Despite their success, these systems grapple with the challenge of station imbalance throughout the day. This imbalance occurs when bikes are unevenly distributed across the network, leading to situations where stations are either completely full, preventing users from returning bikes, or completely empty, leaving potential riders without options. To address this issue, BSSs operators deploy rebalancing trucks to redistribute bikes among stations. Due to the stochastic nature of user demand, coupled with the complexity of the associated dynamic planning problem, the need for efficient and adaptive rebalancing strategies is evident.

To enhance decision support for the Dynamic Bicycle Repositioning Problem (DBRP) to mitigate lost demand, a large variety of predictive and prescriptive models have been introduced, predominantly into Mixed-Integer Programming (MIP) and Markov Decision Process (MDP). MIP models have dominated the literature due to their flexibility, availability, and maintenance within industrial decision-making processes. Researchers consider both deterministic (see, e.g., [23, 45, 59]) and uncertain trip demand (see, e.g.,[26, 44, 69]). MDP models, while a more recent addition to the field, have shown promise in their ability to implicitly account for sequential decision-making (see, e.g.,[12, 38]).

MIP models are typically required to simplify the DBRP by discretizing the planning horizon into multiple time-periods and imposing modeling assumptions to remain computationally tractable, which raises questions concerning which assumptions and modeling techniques are more realistic. The diversity in modeling assumptions spans from the planning objective to the actual decisions and practical constraints used within the models. Unfortunately, most of those works have been proposed isolated from the remainder of the literature, which therefore leads to a gap about which configuration of modeling assumptions provides solutions that perform best in practice for operators. Chapter 3 aims to tackle this gap by introducing a general MIP modeling framework that encompasses numerous relevant assumptions. The performance of configurations is estimated via a fine-grained simulator that mimics customer behaviors and rebalancing strategies on a minute-by-minute basis.

Most studies on MIP models focus on minimizing lost demand based on simple predictions, like historical averages, or more advanced Machine Learning (ML) predictions. Such a single point estimate approach heavily relies on the accuracy of trip prediction. In practice, BSS operators, such as BIXI, often use inventory intervals and target inventories to provide a

buffer to station inventories. Such approaches, chosen for the decision processes in reality, have rarely been considered in optimization models, leaving the potential benefits of combining optimization with predicted intervals/targets unexplored. Additionally, in executing optimization models for planning, one can either apply a one-time optimization or utilize a rolling and folding planning with multiple runs. The benefits of these reoptimization modes are not adequately explored, especially in terms of how system status updates influence the planning solutions. Therefore, Chapter 4 proposes two novel objective functions incorporating predicted inventory intervals and target inventories provided by a predictive model that considers weather and temporal features. We assess these models across various reoptimization modes, static, rolling, and folding, on both synthetic data and real-world data.

MIP models for the DBRP typically divide the planning horizon into several equal-duration time-periods, potentially resulting in sub-optimal vehicle coordination, such as limiting each vehicle to visit only one station per time-period. While MDP models do actually not have this obstacle, they are likely to be limited to one single rebalancing vehicle or smaller station networks to remain computationally tractable. Advanced Reinforcement Learning (RL) techniques present a promising avenue to address the intricacies of DBRP, potentially surpassing the constraints of MDP methods, yet this potential remains largely untapped. As a remedy, Chapter 5 proposes a spatio-temporal RL-based algorithm for multi-vehicle DBRP under a non-discrete time framework, aiming to minimize total lost demand. We cast this problem as an MDP, where state and action spaces are crafted to encapsulate system dynamics and uncertainties. A Deep Q-Network (DQN) is applied to estimate the value function with a detailed simulator to gauge immediate rewards across various demand scenarios and learn the rebalancing policy.

1.1 Research Objectives

The primary objective of this thesis is to develop dynamic rebalancing strategies for BSSs in order to reduce lost demand and improve user satisfaction through MIP and RL techniques.

Our research endeavors to address the following key questions:

1. What configurations of modeling assumptions are the best to use when modeling the DBRP as an MIP?

2. How do we evaluate the obtained rebalancing strategies under the first-arrive-first-serve rule as in reality?

3. Is it beneficial to integrate predicted inventory intervals and target inventories into the

objective functions of the optimization model?

4. Do rolling and folding planning for rebalancing models have advantages in terms of reducing lost demand and computing time?

5. How to formulate DBRP as an MDP with multiple vehicles synchronization under a non-discrete time framework?

6. Do RL techniques lead to an improvement in solving DBRP as an MDP?

1.2 Contributions

Configuration of modeling assumptions in multi-period MIP models for DBRP (Chapter 3, [65]) We conduct an in-depth literature review on multi-period MIP models for DBRP, culminating in a flexible modeling framework. We explore diverse modeling assumptions and alternative formulations, including variable domains, time discretization, trip distribution, and sequence of rentals, returns, and rebalancing operations. We evaluate the performance of the most relevant assumptions using a minute-to-minute fine-grained discrete-event simulator to determine optimal configurations for decision makers. Our extensive experiments, based on synthetic and real-world data, primarily focus on lost demand and the simulation-optimization-gap as pivotal metrics. Our analysis provides insights and recommendations for modeling DBRP, aiming at minimizing lost demand. The findings highlight the usage of trip distribution constraints and newly proposed event sequences with limited computing resources and trip distribution constraints in a shorter time-period with adequate computing resources.

The integration of predicted inventory interval and target inventory with MIP models and the advantages of reoptimization with system updates (Chapter 4) We present two novel models, namely **D**ynamic **R**ebalancing **O**ptimization for **B**SS based on **I**nventory Interval (DROB-I) and **D**ynamic **R**ebalancing **O**ptimization for **B**SS based on **T**arget Inventory (DROB-T), that incorporate inventory intervals and target inventories into the objective function. The inventory interval and target inventory are based on trip prediction from the Gradient Boosting Tree (GBT) algorithm, considering weather conditions and temporal information. Tailored for BSSs operators, the presented models tend to introduce robustness into the objective functions and produce planning solutions that are more capable of dealing with demand fluctuations. Our experiments with the proposed models span static, rolling, and folding reoptimization modes using both synthetic and real-world

data. Demonstrating robustness and adaptability, the models significantly reduce lost demand in diverse settings. Specifically, DROB-T achieves up to 34% reduction and DROB-I up to 28%, outperforming the baseline model. Moreover, the rolling and folding planning consistently outperform static planning, underscoring the need for reoptimizing according to system updates (station inventories and trip predictions over time).

A RL algorithm for MDP model in DBRP within continuous time framework and multiple vehicle collaboration (Chapter 5) We introduce a spatio-temporal RL algorithm specifically designed for DBRP involving multiple vehicles. We initially frame this problem as an MDP within a non-discrete time framework, which allows vehicles to operate independently without waiting for each other, enhancing the rebalancing's realism and efficiency. An event-driven simulator is then utilized to gauge rewards between consecutive states over diverse demand scenarios. Leveraging DQN in various configurations, we estimate the value function and learn a rebalancing policy to reduce lost demand. Numerous experiments are conducted on a trip dataset sensitive to weather and temporal factors. Our findings highlight our algorithm's advantage over existing benchmarks, including static rebalancing and a conventional MIP model. Impressively, our method surpasses the MIP benchmark, reducing lost demand by up to 21.53%.

1.3 Thesis Outline

The remainder of this thesis is organized as follows. Chapter 2 provides a comprehensive literature review encompassing the BSSs and its DBRP, MIP for DBRP, MDP for DBRP, and demand predictions. The three contributions highlighted earlier are sequentially presented in Chapters 3 to 5, respectively. Chapter 6 is dedicated to a comprehensive analysis of our contributions as a collective entity. Finally, we discuss our overall conclusions and ideas for future directions in Chapter 7.

CHAPTER 2 LITERATURE REVIEW

Over the past few years, BSSs have emerged as an innovative contribution to alleviate urban transportation challenges, offering a blend of sustainability, affordability, and convenience. As their popularity grows, the operational complexities associated with ensuring a balanced distribution of bikes across various stations have attracted wide attention. Consequently, academic and industry researchers are undertaking rigorous investigations to elucidate the underlying challenges and conceptualize effective methodologies for optimal system rebalancing.

In this chapter, we provide a comprehensive literature review that delves into the essential aspects of DBRP. Section 2.1 introduces the fundamental characteristics of BSSs and the rebalancing problems. Section 2.2 presents the MIP models for dynamic rebalancing, including modeling assumptions, objective functions, and reoptimization. Section 2.3 focuses on the application of MDP and RL in DBRP. In Section 2.4, demand prediction and the concepts of inventory interval and target inventory are summarized.

2.1 Bike-Sharing Systems and Rebalancing Problems

BSSs have evolved into two primary categories: station-based and dockless. Station-based BSSs are characterized by a structured network where bicycles are picked up from and returned to specific docking stations strategically dispersed throughout urban areas. This system offers a more organized layout, ensuring predefined parking spots and often facilitating easy maintenance. In contrast, dockless BSSs provide greater flexibility to users. Bikes in these systems are equipped with self-locking mechanisms, permitting users to initiate and conclude their rides virtually anywhere within designated service areas. While this offers tremendous convenience to users, it also poses challenges related to bike redistribution and potential clutter in public spaces. According to [73], until August 2022, there are 1134 station-based BSSs and 393 dockless BSSs. Station-based systems are by far more common.

Moreover, for dockless systems, researchers tend to group areas and treat each area as a station (see, e.g., [28, 72, 102, 113]). As such, it has been a common practice to approach rebalancing for dockless systems in a similar way as for station-based BSSs. We thus focus on station-based BSSs.

BSSs rebalancing, also known as redistribution, is necessitated by the dynamic and stochastic nature of trip demand, which leads to some stations having a surplus and others a deficit of

bikes [83]. Operators typically employ trucks to rebalance bikes among the stations to cope with this temporal and spatial imbalance. Meanwhile, researchers have explored various rebalancing strategies to provide decision support and improve user satisfaction and system efficiency. The literature for bike-sharing repositioning can be divided into the Static Bicycle Repositioning Problem (SBRP) and DBRP [84]. The SBRP typically performs balancing operations overnight, and user trips during the operating period are not considered. For DBRP, it focuses on intraday rebalancing, and the demand forecast during the operation is important [35]. SBRPs are useful for arranging the inventory of bicycles in the system in preparation for the next day. However, it is limited and may not be able to address the dynamic nature of demand throughout the day. We here focus on the DBRP.

From the perspective of rebalancing operation carriers, the literature mainly focuses on two types of rebalancing in BSSs [96]. User-based rebalancing leverages incentives to encourage users to participate in the rebalancing process [48]. Users are encouraged to voluntarily move bikes from stations with surpluses to stations with shortages. This can be incentivized through reward systems, such as credits or discounts, provided to users who contribute to rebalancing efforts. Such an approach is more common in dockless BSSs. In contrast, operator-based rebalancing involves the active management of a rebalancing fleet (e.g., trucks) that relocates bikes from one station to another. Such an approach is more common to station-based BSSs [51]. Since we concentrate on station-based BSSs, operator-based rebalancing is our research focus.

Current research predominantly tackles the DBRP utilizing either MIP or MDP. The majority of the literature applies MIP models (see, e.g., [46, 74, 111, 114]) owing to their flexibility and applicability within an industrial decision-making process. Alternatively, MDP offers a different approach by conceptualizing DBRP as a problem of making a sequence of decisions [4]. Both MIP and MDP models may benefit from integrating predictive trip information, such as predicted rental/return demand, inventory intervals, and target inventories, either as part of the objective function or within the constraints. We will delve into the literature concerning MIP models in the next section 2.2.

2.2 MIP in Dynamic Rebalancing Problems

In DBRPs, rebalancing operations are performed multiple times during the day and real-time demand flow is considered when rebalancing takes place. The goal is to find rebalancing strategies during the day (especially the peak hours) with minimum cost and demand unsatisfaction. The majority of the literature applies MIP models, within which multi-period models are predominantly utilized. Multi-period models determine the number of bikes that

should be deployed at each station for each period in the whole planning horizon. They benefit from integrated planning over all time-periods and do not suffer from the myopic behavior of single-period models. Although there are various proposed MIP models, its modeling requires assumptions that greatly vary within the literature. In the following sections, we summarize the modeling assumptions, objective functions, and reoptimization modes in the existing literature.

2.2.1 Modeling Assumptions

Due to its modeling flexibility, MIP remains a popular choice compared to MDP based methods. However, the complex planning problems demand significant model simplifications to ensure computational tractability. Notably, diverse modeling assumptions and techniques have been adopted in existing models, especially concerning time discretization, trip modeling, variable domains, trip distribution, and the sequence of events.

Time discretization. The planning horizon is split into discrete time-periods to represent shifts in station status and vehicle positions. Most literature on dynamic rebalancing typically uses equal length time-periods (see, e.g., [20, 69, 114]). It is normally assumed that each vehicle rebalances at most one station during one time-period. The duration of time-periods plays a crucial role in the models and their effectiveness. However, determining the optimal time-period length depends on model intricacy and the targeted planning horizon length.

Several studies, including [45], [69], and [85], investigate the impact of time-period aggregation on rebalancing performance. Shorter time-periods typically enable more frequent rebalancing under the assumption that each vehicle can rebalance one station per time-period, which in turn reduces unmet demand. In this case, time constraints may be needed to ensure that the rebalancing and subsequent transit to the next station are feasible within the stipulated time-period. Essentially, these restrictions prevent truck movements that would be impractical due to time limitations. Notably, [46] and [44] consider time constraints, but within a single-period rolling planning context. Time constraints are also utilized in [23] and [69], but their focus is solely on the transit time of vehicles, neglecting the time necessary for rebalancing operations.

Trip modeling. Each customer trip encompasses one rental and one return demand. Two main variable types have been used in the literature: Origin-Destination (O-D) Variables and Station-Based Variables. Specially, O-D variables denote the departure and arrival stations, coupled with the respective times, while station-based variables separate trip departure (rental demand) and trip arrival (return demand) as two distinct variables.

While station-based models reduce the number of variables, they disconnect rentals from returns. Conversely, O-D models maintain this connection but require many more variables, potentially compromising the computational tractability.

Variable domains. Variables in the rebalancing model may represent routing, rebalancing, and user trips. The routing variables, which are always binary, indicate vehicle positions and routes. Rebalancing variables indicate vehicle inventory and the number of rebalanced bikes. The variable type can affect solution performance. Most models define them as continuous variables (see, e.g., [23, 44, 59, 69, 91]), while in [112] integer rebalancing variables are used. For variables of user trips, most models in the literature use continuous variables, except for [111], where integer O-D variables are used.

Trip distribution. When the demand for rentals or returns of bikes surpasses the available bikes/docks, an optimization model might opt for rentals or returns based on the objective function's preference. Contrarily, in practice, demand is served in the first-come-first-serve fashion. This discrepancy has led several researchers to integrate constraints for achieving a more realistic trip distribution. We summarize assumptions regarding demand distribution as follows:

- **Station-based variables without distribution**. Two distinct variables are used to indicate the rentals and returns for each station during a defined period [59]. Similarly, in [23] two variables denoting shortage and excess of bikes are introduced, essentially equivalent to station-based variables. In these models, demands are greedily satisfied, missing to interlink rentals and returns, leading to potential underestimations of lost demand.

- **Station-based variables with proportional distribution**. In [69], a trip distribution is considered in which the returns are proportional to the rentals based on historical O-D trips. It is also assumed that the number of bikes returned during a specific time-period is not higher than the number of bikes rented in the previous periods multiplied by the corresponding proportion. Similarly, in [108], a return ratio is considered on the number of returns at a station divided by the total number of bikes currently used by customers during the time-period.

- **O-D variables without distribution**. Here, O-D variables encapsulate the number of trips starting at one station and ending at another, strictly within a specified time-period.

- **O-D variables with proportional distribution**. In [45] and [114], a proportionality

similar to [69] is used, implying that the rentals from a given station are confined by the ratio of distribution adjusted with the presently available bikes.

- **Poisson distribution**. In [21], [61], and [91], the arrival of rentals and returns is modeled as a Poisson process, which implicitly models trip uncertainty.

Sequence of events. In a station-based BSSs, both customers and rebalancing vehicles interact with station inventories. While customers can either rent or return bikes, rebalancing trucks can pick up or drop off bikes from stations. However, optimization models often simplify reality by amalgamating all these operations within discrete time intervals.

Some models assume that rentals, returns, and rebalancing operations happen simultaneously within a specific time-period (see, e.g.,[16, 23, 45, 74, 109, 114]). Such an assumption essentially allows for immediate compensation between rentals and returns, resulting in an overly optimistic demand satisfaction within the optimization model. An alternative, more conservative model presumes a sequential occurrence of these events within each time-period, for instance, initiating with returns, followed by rebalancing, and concluding with rentals [44]. In [69], it is suggested that returns take precedence, followed by rentals with rebalancing operations taking place thereafter. In [112], it is assumed that rebalancing happens first and then customers rent and return bikes. In [59] a different approach is adopted by leveraging station-visit events and cumulated demand extremities to discretize the planning horizon. In that model, the time taken for bike pick ups or drop offs is not considered. However, it can be inferred that the model essentially follows a pattern where customers engage in rentals and returns before any rebalancing action takes place.

Finally, in [108] a novel methodology is proposed wherein each time-period is subdivided into more refined segments, with each segment associated with specific rental and return demands. In this paradigm, pick-ups occur at the first segment of each period and drop-offs are scheduled at the end.

2.2.2 Objective Functions

Different decision-makers may have different objectives for their BSSs. The objective functions are mostly used to quantify the costs caused by rebalancing and customers' satisfaction. For example, some choose to use lost demand to measure customers' satisfaction [23, 44] and some choose the number of realized trips [91, 103]. Some studies consider more than one objective along with weights in the objective function. Besides, other factors, such as CO_2 emissions and labor consumption, have also been modeled in some research [92].

The objectives used in the literature are summarized in Table 2.1.

Table 2.1 Objectives of the related literature

	Papers	Objective
Static	[6]	Minimize the cost of transporting and loading
	[8]	Minimize the cost of route
	[17]	Minimize the cost of route
	[25]	Minimize the total cost
	[27]	Minimize the weighted sum of total time and deviation from the target bike number
	[32]	Minimize the total travel and handling cost
	[33]	Minimize the total cost
	[35]	Minimize deviation in the number of bikes at each station
	[37]	Minimize the total cost (penalties for stations and travel cost)
	[58]	Maximizes number of full vehicle loads picked up and delivered to the stations
	[60]	Minimizing the rebalancing tour length (cost)
	[63]	Minimize the total transportation cost
	[81]	Minimize the deviation from target bike number, loading activities number, and total time
	[82]	Minimize the deviation from target bike number, loading activities number, and total time
	[84]	Minimize the weighted sum of the stations' penalty costs and operating costs
	[88]	Minimize makespan
	[94]	Minimize the weighted sum of unmet demands and operational time
	[95]	Minimize the total cost
Dynamic	[12]	Minimize the rate of arrival of unsatisfied users
	[15]	Minimize the vehicles repositioning costs
	[18]	Maximize the probability that a certain amount of users find a bike
	[19]	Minimize the total distance
	[20]	Minimize the time during which stations are unable to meet demand
	[23]	Minimize the total unmet demand
	[43]	Minimize the lost demand and travel cost
	[44]	Minimize the lost demand
	[45]	Minimize the lost demand and travel cost
	[46]	Maximize the probability of meeting the uncertain customer demand
	[59]	Minimize weighted cost of unmet demands, deviation, loading number and time
	[61]	Minimize the rate of arrival of unsatisfied users
	[69]	Minimize the lost demand
	[71]	Minimize the total cost
	[74]	Maximize positive value actions with traveling time and redundant redistribution
	[91]	Maximize the total number of bicycle trips
	[92]	Minimize the weighted sum of total unmet demand, fuel and CO_2 emission cost
	[103]	Maximize the cost saving and users' satisfaction
	[107]	Minimize the gap between the number of bikes after repositioning and the target
	[109]	Minimize the total cost
	[111]	Minimize the total user dissatisfaction and the vehicle traveling cost
	[113]	Maximize matching degree of demand and actual number of bikes in each zone

The metrics used within the objective functions in the literature can be classified into three categories:

- Distance-Based Metrics: A significant portion of the literature has optimized metrics that focus on the traveling distance of vehicles. Paramount among these are the costs associated with traveling, the time taken, and the fuel consumed (e.g., [3, 45, 82, 114]).
- Loading-Based Metrics: Loading and unloading operations have been another focal point (e.g., [50, 95]). Handling costs and time required to load are used in the objective functions to represent the operational workload.
- Demand-Based Metrics: These metrics aim to capture customer satisfaction or dissatisfaction. The literature reveals two dominant strategies. One is to minimize the deviations from a target inventory (e.g., [6, 35, 47, 63, 68, 107]). Another one is to minimize the lost demand or maximize the realized trips (e.g., [69, 92, 108, 111, 112]).

In DBRPs, most studies focus on demand-based metrics, especially minimizing lost demand (e.g., [23, 45, 50, 74, 108, 112, 114]). A few works minimize the deviation from target inventories, which will be further discussed in Section 2.4.

2.2.3 Reoptimization

In DBRP, the planning horizon represents the total duration over which rebalancing solutions are planned by the optimization model. This horizon is segmented into multiple time periods, allowing for the periodic reassessment after updating system parameters, such as station inventories, vehicle locations, and vehicle inventories. Three main reoptimization modes may be employed: static, rolling, and folding planning, with the first two being more prevalent in the context of DBRP. Note that the static here is not the same as the SBRP. Static planning refers to rebalancing strategies for the entire planning horizon without reoptimization after system updates. Static planning aggregates information across multiple time-periods to deduce rebalancing strategies for the entire planning horizon in a single computation (see, e.g. [43, 59, 71, 86, 91, 112]). This is able to capture significant look-ahead information. However, the rebalancing solution obtained under static planning is not responsive to real-time system changes. Moreover, when dealing with a large number of time-periods, it can escalate the computational intricacy, potentially leading to exponential number of variables.

In contrast, rolling planning operates in stages. The optimization model is resolved at each stage based on updated system parameters. A predetermined number of time-periods may be fixed for each stage, ensuring that the computational effort remains manageable while

retaining a certain number of look-ahead time-periods. To benefit from system updates, in [46, 69, 74, 85, 92, 111], a rolling planning with several stages is applied, each of which contains a certain number of time-periods. Especially, some studies consider a single-period rolling model (see [44, 47, 50]). Constantly updating the status of the system enables the rectification of deviations stemming from forecast discrepancies. Additionally, limiting the number of time-periods per stage ensures computational tractability. However, this approach might display short-sightedness, especially when insufficient time-periods are encompassed within one stage. This might lead to prioritized short-term demand satisfaction at the expense of potential long-term demand deficits.

To overcome the short-term focus of rolling planning, folding planning takes into account all the remaining time-periods at each stage. Although it is common in planning problems such as [67], its application in DBRP remains relatively uncharted, to the best of our knowledge, suggesting an opportunity and potential advancements in the field.

2.3 Markov Decision Process in Dynamic Rebalancing Problems

Most MIP models for DBRP proposed in the literature employ time discretization, partitioning the planning horizon into time-periods (Section 2.2). To alleviate the complexity of the models, a series of assumptions are often introduced, notably limiting each vehicle to service only one station within one time-period. Multi-period planning allows for anticipation of station changes and access to look-ahead information, yet it is not trivial to find the ideal time-period lengths. The length significantly impacts the rebalancing strategy and its performance.

When faced with stochastic demand inherent to real-world scenarios, especially when multiple trucks operate to redistribute bikes simultaneously yet independently, multi-period planning may result in suboptimal and impractical solutions. For example, one truck has completed its rebalancing for the current time-period, while another truck is still rebalancing bikes. The truck that completed its current rebalancing task has to wait until the next time-period. Such inconsistencies in operational timings can diminish the realism and applicability of the solutions generated by the model, highlighting the necessity for a modeling approach that better aligns with the dynamic and stochastic nature of bike demand and the asynchronous operations of the rebalancing fleet.

As an alternative, the application of MDP has been explored for crafting dynamic strategies for DBRP. Despite the appeal of MDP, its use has been limited by the immense complexity of the problem space, often rendering the computation of an optimal policy impractical within

a reasonable time. This high dimensionality obstacle is a recurrent theme in the critique of MDP applications within DBRP [61]. We summarize the work on MDP for DBRP in the following.

2.3.1 MDP Models

MDP is a mathematical model for sequential decision-making in stochastic environments, providing a framework for understanding how to make a series of decisions to optimize an objective over time. An MDP is characterized by the ability to capture the uncertainty and temporal aspects inherent in decision-making processes, making it particularly useful for problems like DBRP [89].

Despite its relevance and potential for application in DBRP, MDP adoption in this domain has been relatively recent, with a limited but growing body of work in academic research. In [12] DBRP is conceptualized as an MDP and a dynamic lookahead policy heuristic is introduced, albeit limited to single-vehicle rebalancing operations. In [13], the MDP model is extended to handle multiple vehicles, proposing a coordinated lookahead policy to simultaneously address redistribution and routing decisions. In [61], an approximation method (one-step policy improvement) is employed, to determine which station should be prioritized, intending to minimize the rate of arrival of unsatisfied users who find their station empty or full. However, the considered planning horizon is segmented into periods of equal length.

The MDP introduction for DBRP illustrates its potential, as well as ongoing challenges including computational complexity, granularity of time discretization, and synchronization of multiple vehicles.

2.3.2 Reinforcement Learning for MDP in DBRP

RL has seen significant progress, particularly in its application to multi-agent systems, enabling the development of algorithms that learn high-quality decision-making strategies from interactions with the environment [75]. Such advancements offer great potentials for addressing the complexities of MDP in scenarios like the DBRP, where the need for intelligent, adaptive decision making is critical. DBRP can be modeled as a multi-agent system, with the vehicles acting as agents that make decisions based on the current state of a complex, dynamic environment that includes a network of stations and stochastic user demand. In this setting, vehicles, as agents, determine optimal rebalancing actions, aiming to maximize bike availability across the network while satisfying user demand as efficiently as possible. The dynamic and uncertain BSSs environment makes RL a suitable approach for DBRP, given

its success in similar domains like gaming, finance, and marketing [101].

While our focus here is on operator-based rebalancing, RL has been more frequently applied to user-based rebalancing [96]. For instance, in [78] a Deep Deterministic Policy Gradient-based RL algorithm is used to motivate user behavior through monetary incentives, guiding the redistribution of bikes. In a similar vein, in [29], a differentiated Deep Reinforcement Learning (DRL) framework under MDP is leveraged to optimize incentive pricing for rebalancing. In [49] a dynamic incentivization system powered by RL is introduced, benchmarking three policy-gradient RL methodologies. Building upon this, in [87], the approach is adapted for more realistic environments. However, the users' behaviors are always uncertain, and rebalancing by users can be less efficient without the intervention of operators' rebalancing.

In terms of operator-based rebalancing, in [64] a spatio-temporal RL framework is introduced to address the rebalancing problem. The approach begins with a clustering algorithm designed to group the entire bike-sharing system. Then, a deep neural network is designed to estimate the optimal long-term value function, a crucial component for devising rebalancing strategies. In [101], a distributed RL approach infused with transfer learning techniques is used to make decisions about how many bikes should be moved for each station. The approach pairs each station with an agent responsible for determining the optimal number of bikes to be allocated to the respective station, followed by an aggregate optimization process. The model avoids an exponentially-expanding action space but requires higher run-time processing power to support the parallel learning of all agents. A notable limitation is the absence of a routing optimization component, suggesting that further work is needed to provide realistic rebalancing strategies.

In [72], a Policy Function Approximation (PFA) algorithm for MDP is designed for dynamic rebalancing with a single truck. In [62], a static rebalancing method using a policy gradient-based RL algorithm is proposed to enhance user experience and reduce operational costs. Finally, in [106], a model based on DRL is developed that leverages a DQN to provide prompt and efficient rebalancing actions. This approach is particularly geared towards adapting to the dynamic demands of bike-sharing systems in real time, showing a proactive step towards real-world applications. Among the above literature, RL for operator-based DBRP is limited to single truck rebalancing and small-scale station network, often using unrealistic simulation or time discretization.

2.4 Demand Prediction and Inventory Interval

As an integral component of BSSs optimization, demand prediction is fundamental in anticipating the spatial and temporal patterns of bike usage. The literature on this subject delves into various predictive models that leverage historical data, weather conditions, and temporal factors to forecast demand. These predictive insights are crucial for rebalancing operations and enhancing user satisfaction by ensuring the availability of bikes and docks when and where needed. We first review predictive models for demand in Section 2.4.1. Two concepts, inventory intervals and target inventories based on demand prediction, are further introduced in Section 2.4.2.

2.4.1 Demand Prediction

Demand prediction is a complicated task as it is naturally stochastic and influenced by numerous factors [98]. Literature can be divided into prediction of a system-wide demand level (see, e.g., [30, 42, 105]), on a cluster level (see, e.g., [10, 36, 98]) and on a station level (see, e.g., [11, 31, 54, 79]). This review emphasizes station-level demand predictions, given that rebalancing strategies are typically refined based on individual station dynamics.

A naive demand prediction model estimates future demand by averaging historical trip data. In [45], [69], [111] and [114], a sampling method is adopted to generate expected demand. The model determines the expected demand by drawing samples from distributions of rentals and returns, often Poisson, whose mean is calculated based on historical data. In [39], [68], and [70], it is highlighted that historical mean demand has a high prediction error compared to more sophisticated ML algorithms, as ML algorithms can analyze numerous factors beyond historical trip data, including weather conditions, time of the day, day of the week, and special events. Consequently, ML algorithms can capture intricate patterns and correlations, resulting in significantly more accurate demand predictions compared to the rudimentary historical averaging approach.

Among prediction models for BSSs using ML algorithms, research works (see, e.g., [11, 54, 100]) indicate that GBT performs well in demand prediction compared to other ML algorithms in terms of prediction accuracy. Similarly, comparative analyses by [105] and [70] suggest that both Random Forest and GBT supersede other ML paradigms in the domain of BSSs demand prediction. A recurring theme in these research endeavors is the incorporation of temporal and meteorological attributes, highlighting their pivotal role in shaping demand dynamics within BSSs. In [85], an extensive comparison of different predictions is conducted within their proposed optimization model. The results show that a reliable prediction enables

system operators to cope with the rebalancing needs more effectively.

Specially, in [54], Singular Value Decomposition (a dimensional reduction technique) is leveraged to reduce the dimensionality of the trip data and, hence, make faster predictions and enhance the problem's tractability when dealing with an elevated number of stations.

2.4.2 Inventory Interval and Target Inventory

Inventory intervals and target inventories are crucial components of dynamic rebalancing adopted by BSSs operators. Inventory intervals offer a permissible range of inventory for individual stations, within which the station has a balanced inventory level. The target inventory embodies the ideal number of bikes that operators aim to uphold at each station. The integration of inventory intervals and target inventories in a dynamic rebalancing theme enables operators to decide when and how to redistribute bikes among stations. Inventory intervals and target inventories are normally computed based on predicted demand, either with a naive prediction (e.g., the historical demand mean) (see, e.g., [24, 47, 83, 88]) or a ML-based prediction (see, e.g., [52, 54]). While various works focus on the methodologies to compute inventory intervals and target inventories, a few integrate them into optimization models.

In [83], the target inventories are computed based on the initial inventory that can minimize the estimated lost demand for a given time period. Similarly, in [52] the target inventories are obtained by estimating the initial inventory that reduces the likelihood of a station attaining full or empty statuses. In [24], a simulation approach is applied in which the performance of BSSs is evaluated given an initial inventory. Several initial inventories are tested to select the one with the best performance. In [68], the target inventory represents the initial inventory that can maximize the time window in which the station is considered balanced.

In [88], inventory intervals are seen as the range of the initial inventory that can ensure a minimum service level over a specified time period. To be more specific, these inventory intervals represent the initial inventory values that can guarantee the satisfaction of a sufficient proportion of the total demand. In [54], both target inventories and inventory intervals are computed based on the service level. Two additional hyperparameters, α and β, are introduced to fine-tune the configuration of inventory intervals. The hyperparameter α plays a crucial role in determining whether the target inventory intervals should prioritize rentals, returns, or both. Hyperparameter β influences the gap between the upper and lower bound within the inventory interval and, therefore, it affects the size of the inventory buffer. By leveraging these hyperparameters, operators gain a level of control over the target inventory level and inventory intervals, enabling adjustments in alignment with the desired rebalancing

strategy.

The integration of inventory intervals and target inventories plays a pivotal role in the optimization of BSSs, as reflected in recent scholarly work. These elements are commonly incorporated into optimization models, serving either as key objectives or as constraints.

Target inventories are frequently embedded within the objective functions. The goal is to minimize the deviation between actual station inventories and their targets. For instance, in [59], a dynamic rebalancing model is suggested to penalize deviations from station target inventories. These targets are considered to represent the optimal number of bikes at a station at the end of the rebalancing process. Various metaheuristics are proposed to evaluate efficiency in solving the model. In [47], the approach is designed for curtailing deviations from target inventories. The approach, however, utilizes a single-period rolling planning to align station inventories with target inventories. Moreover, the work in [17] integrates target inventories as a hard constraint, mandating stations to achieve these targets by the end of the rebalancing procedure.

When it comes to inventory intervals, in [88] and [97], they are imposed as strict constraints within the models. Nevertheless, while in [88] the approach is focused on a static single-period problem, in [97] a dynamic multi-period model is adopted. The latter eventually opts to disregard the inventory interval constraints in the empirical analysis, acknowledging the possibility of infeasibility in multi-period settings. Furthermore, in [99] an MIP model is presented seeking to minimize the deviation from inventory intervals within the objective function. This method potentially offers greater operational flexibility compared with using hard constraints. However, the actual efficacy of incorporating inventory intervals into the objective function remains uncertain, as their experimental design also excludes the intervals, suggesting an area open to further research. Thus, the integration of predicted inventory intervals and target inventories with multi-period optimization models remains unexplored.

CHAPTER 3 ARTICLE 1: DYNAMIC REBALANCING OPTIMIZATION FOR BIKE-SHARING SYSTEMS: A MODELING FRAMEWORK AND EMPIRICAL COMPARISON

Authors: Jiaqi Liang, Sanjay Dominik Jena, and Andrea Lodi.
Submitted to European Journal of Operational Research on December 15, 2022.[1]

Abstract Station-based Bike-sharing systems have been implemented in multiple major cities, offering a low-cost and environmentally friendly transportation alternative. As a remedy to unbalanced stations, operators typically rebalance bikes by trucks. The resulting dynamic planning has received significant attention from the Operations Research community. Due to its modeling flexibility, mixed-integer programming remains a popular choice. However, the complex planning problem requires significant simplifications to obtain a computationally tractable model. As a result, existing models have used a large variety of modeling assumptions and techniques regarding decision variables and constraints. Unfortunately, the impact of such assumptions on the solutions' performance in practice remains generally unexplored.

In this paper, we first systematically survey the literature on rebalancing problems and their modeling assumptions. We then propose a general mixed-integer programming model for multi-period rebalancing problems that can be easily adapted to different assumptions, including trip modeling, time discretization, trip distribution, and event sequences. We develop an instance generator to synthesize realistic station networks and customer trips, as well as a realistic fine-grained simulator to evaluate the operational performance of rebalancing strategies. Finally, extensive numerical experiments are carried out, both on the synthetic and on real-world data, to analyze the effectiveness of various modeling assumptions and techniques. Based on our results, we identify the assumptions that empirically provide the most effective rebalancing strategies in practice. Specifically, a set of specific trip distribution constraints and event sequences ignored in the previous literature seem to provide particularly good results.

Keywords: Facilities planning and design, Bike sharing systems, Dynamic rebalancing, Modeling framework, Mixed-integer programming

[1] A pre-print is available in [65].

3.1 Introduction

Bike-sharing systems (BSSs) are quickly gaining popularity worldwide, as they help to reduce traffic congestion and vehicle CO_2 emissions. Over the past few years, BSSs were implemented in most major cities such as New York, Boston, London, Sydney, Beijing, Paris, Toronto, and Montreal. We focus on station-based BSSs, where stations with specific capacities are installed in the city, holding an inventory of rentable bikes. Users may rent available bikes from these stations and return their bikes to available docks.

Throughout the day, BSS stations are often unbalanced, which leads to demand unsatisfaction, given that rental demand may not be satisfied when a station is empty, and return demand may not be satisfied when the station is full (i.e., no empty docks are available). As a remedy, BSS operators employ trucks to rebalance bikes among stations. However, due to the uncertain rental and return demand, as well as the complexity of the dynamic planning problem, manual planning tends to be sub-optimal.

To provide decision support, the scientific community has provided a large variety of predictive and prescriptive models, aiming at improving station rebalancing. Planning models can roughly be divided into those based on Markov Decision Process (MDP) and those based on Mixed-integer Linear Programming (MILP). The former, more recently applied in the context of BSSs (see, e.g., [12, 13, 38, 61, 72, 90, 101]), implicitly considers uncertainty. However, in order to remain computationally tractable, MDP models rely on state and action spaces limited in size and are therefore constrained to small station networks, small rebalancing fleets, or limited rebalancing decisions. The majority of the literature proposes MILP models to represent the planning problem, where MILP remains the predominant modeling tool due to its flexibility, as well as the availability of a well-established process to integrate and maintain such models and a wide range of solution methods within an industrial decision-making process. For MILP, both deterministic customer demands (see, e.g., [23, 45, 59]) and simplified variants dealing with uncertain demand (see, e.g., [26, 40, 44, 69]) have been considered in the literature. In both cases, due to the complexity of the planning problem and the synergies between customer arrivals and rebalancing operations, its modeling requires assumptions that greatly vary within the literature. These assumptions range from the planning objective to the actual decisions and practical constraints used within the models. Further, customer demands may occur continuously in time. To remain computationally feasible, the planning horizon is typically divided into a set of time-periods, which raises questions concerning the sequence of occurring rental and return demands, as well as the moment of the planned rebalancing operations. As a result, both the planning problems and the mathematical optimization models proposed in the literature are highly diverse. Unfortunately, most of those

works have been proposed isolated from the remainder of the literature, which therefore lacks consensus on which assumptions and modeling techniques are best to use. Thus, operators are rather uncertain about which modeling assumptions should be used in the context of their specific BSS and which modeling techniques provide solutions that perform best in practice.

Objective, scope, and contributions. The objective of this paper is to provide guidelines to both practitioners and academics on which assumptions and techniques are most appropriate and likely to produce rebalancing solutions that perform well in practice. To this end, we provide a systematic review of the modeling assumptions and techniques presented in the literature, mostly focused on multi-period models, and propose a general MILP modeling framework that encapsulates most of the relevant modeling assumptions. While several modeling assumptions have been used in the literature without further justification or comparison, we explicitly discuss the alternatives and provide intuitive insights on which assumptions may be more appropriate in practice. We then provide extensive numerical results to empirically evaluate the realism of the various assumptions and modeling techniques, such as variable domains, time discretization, the distribution of trip demand, and the assumed sequence of bike rentals, bike returns, and rebalancing operations.

We develop a realistic simulator that emulates customer trips and the given rebalancing strategy on a minute-to-minute basis. This simulator evaluates the quality of a proposed rebalancing planning solution and hence the realism of the modeling assumptions and techniques of the associated optimization model. Throughout our experiments, the most relevant combinations of modeling assumptions and techniques are then empirically tested on a large set of synthetically generated problem instances that have been carefully designed to represent different BSS settings and fit the demand patterns observed in real-world trip data from BIXI Montreal.

Based on our modeling framework and empirical results on both synthetic and real-world data, practitioners can derive an optimization model tailored to their BSS environment. In particular, our empirical results suggest that (i) both variable domain and type strongly impact the realism and tractability of the model, (ii) time-related constraints are particularly important when time-periods are short, (iii) a new set of trip distribution constraints performs better than those previously considered in the literature, and (iv) the sequences of trips and rebalancing operations used in the literature are outperformed by the new event sequences proposed in our paper.

We note again that we are less concerned with the possible uncertainty surrounding the input parameters of the models, but rather with the underlying assumptions required to tractably represent the reality within the mathematical model. As such, the presented modeling frame-

work is deterministic (i.e., it uses a single set of expected trip demand) for several reasons. Stochastic, scenario-based formulations are equally subject to such modeling assumptions. Our conclusions are therefore still likely to hold for such problem variants. While we review and discuss the corresponding literature, for the purpose of our study, it is not necessary to explicitly represent uncertainty. Finally, a stochastic problem variant would be computationally intractable, thus forbidding us to obtain close-to-optimal solutions that are required for our analysis.

Outline. The rest of this paper is organized as follows. Section 3.2 reviews related literature on BSS rebalancing problems and summarizes the assumptions. Section 3.3 describes the modeling framework for the multi-period rebalancing problem, including a basic model that can be extended by several constraints according to the various modeling assumptions. Section 3.4 presents the general framework used to evaluate the practical performance of a given multi-period rebalancing strategy. Numerical tests and analyses on synthetic and real-world problem instances are illustrated in Section 3.5. This is followed by the conclusions in Section 3.6.

3.2 A Review of BSS Rebalancing Modeling Assumptions

The literature mainly focuses on two types of rebalancing in BSSs [96]. User-based rebalancing incentivizes users to rent or return bikes at specific stations [48]. Such an approach is more common in dockless BSSs. In contrast, operator-based rebalancing involves the active management of a rebalancing fleet (e.g., trucks) that relocates bikes from one station to another. Such an approach is specific to station-based BSSs [51]. According to a recent statistical report [73], station-based systems are, by far, more common. Even in the case of rebalancing planning in dockless BSSs, it has been a common practice to divide the studying area into different sub-areas, which are then seen as stations (see, e.g., [28, 72, 102, 113]). We, therefore, focus on operator-based rebalancing.

Operator-based bike-sharing rebalancing problems can be divided into static bicycle repositioning problem (SBRP) and dynamic bicycle repositioning problem (DBRP) [77, 84]. SBRP typically rebalance stations overnight, while the operations during the day are not considered. Static rebalancing, therefore, prepares the station inventories for the next day. However, it cannot explicitly react to the demand fluctuation occurring during the day. In contrast, DBRP focus on intraday rebalancing, where customer trips carried out during the day heavily affect the availability of bikes and docks [35]. Indeed, demand satisfaction highly depends on the real-time status of the stations and customers' stochastic rentals and returns, which poses practical challenges [93]. We here focus on dynamic rebalancing planning, which has a higher

impact on practice since it considers continuous rebalancing throughout the day. Given that DBRP consider trip demand and rebalancing operations over the day, the models proposed in the literature either approach this problem using a repeatedly solved single-period model or a multi-period model.

Single-period rolling vs. multi-period planning. A single-period model generally spans a duration between 10 and 60 minutes. Single-period models are typically solved in a rolling horizon fashion throughout the day, while multi-period models are either solved once at the beginning of the planning horizon or several times throughout the day.

Solving a single-period model is computationally easier than a multi-period one. However, single-period models tend to be myopic, i.e., the decisions made at the current time-period cannot take into consideration the consequences in future time-periods. Such models therefore greedily maximize demand satisfaction at the current time-period, even if this results in station inventories that cannot satisfy demand well in the next time-periods. In contrast, multi-period models benefit from integrated planning over all time-periods and avoid suffering from myopic behavior. On the downside, these models may be more difficult to solve, given that they consider several sets of decision variables and constraints for each time-period.

Since we are concerned with finding the model that performs best in practice, we focus on the multi-period planning model. In the following, we summarize the main assumptions made in multi-period planning models proposed in the literature, review existing alternatives and propose some that might not have been considered yet.

3.2.1 Time Discretization and Time Constraint

To represent the change in stations' status and vehicles, the planning horizon is discretized into time-periods. One mainly has two possibilities to discretize the planning horizon.

- **Time-periods of equal length**. The planning horizon is discretized into a set of evenly-spaced time points and the length of each period is the same, which is employed in most multi-period models. In the multi-period planning framework, we could obtain the changes of stations for each time-period and gain the look-ahead information. However, it is hard to define the optimal length of the time-period. It should depend on the model complexity and the length of the planning horizon we focus on.
- **Time-periods of different lengths**. Here, the length of each period in the planning horizon can be different. [59] split each cumulated demand function into weakly-monotonic segments with extreme values that are regarded as end-of-segment events. Further, the arrival of a vehicle at a station to rebalance bikes is referred to as a

station-visit event. These two types of events are sequentially considered to separate the planning horizon.

For multi-period dynamic rebalancing, time-period with equal length is most common in the literature (see, e.g., [20, 69, 114]). Typically, it is assumed that each vehicle rebalances at most one station during one time-period. Selecting an appropriate length of time-period is crucial to the rebalancing strategy and its performance. Several works have investigated the effect of such aggregation on the rebalancing performance (see, e.g., [45, 69, 85]). Generally, aggregations over shorter time-periods allow for more rebalancing operations and, therefore, lower lost demand. However, this typically comes at the cost of increased computing times.

Moreover, when the time-period is short, a time constraint may be required to ensure that the required time for rebalancing and transiting to the next station fits into the length of the time-period. Time constraints, therefore, interdict truck relocations that are unrealistic in practice. [45] and [91] do not use time constraints, while follow-up work [44, 46] apply time constraints within a single-period rolling planning framework. [23] and [69] use time constraints considering only vehicle traveling time, while the handling time at stations is ignored. [59] and [112] consider both traveling and handling time, where the latter is computed as an average value regardless of the number of loading/unloading operations. A summary of how time constraints are handled in the literature, along with other model characteristics, is presented in Table A.1 in the Appendix A.

Note that the introduction of time constraints may make the model difficult to solve. It is therefore crucial to select a proper time-period length that can lead to a reasonable solution time while providing high-quality rebalancing strategies.

3.2.2 Trip Modeling and Variable Domains

Each customer trip contains one bike rental demand and one bike return demand. To model successful trips, mainly two types of variables have been considered:

- **Origin-destination (O-D) variables** $x_{s_1,s_2}^{t_1,t_2}$, contain departure station s_1, arrival station s_2, departure time t_1, and arrival time t_2.
- **Station-based variables** represent the departure of a trip, i.e., satisfied rental demand $x_{s_1}^{+,t_1}$ and the arrival of a trip, i.e., satisfied return demand $x_{s_2}^{-,t_2}$.

Station-based variable models require fewer variables but lack the connection between rentals and returns. The O-D variable models avoid this issue at the expense of a large number of

variables, which may complicate the solutions of the model. A general classification of existing models can be found in Table A.1 in the Appendix A.

Next to the type of variables, the variable domains may also impact the performance of the induced solutions. In the rebalancing model, variables represent three main actions: routing, rebalancing, and the above-discussed user trips. The routing variables typically indicate the location of a vehicle along the planning horizon and the route taken. These variables are always binary. Rebalancing variables represent the inventory of vehicles and the number of bikes to be rebalanced. Most models define them as continuous variables (see, e.g., [23, 44, 59, 69, 91], etc.). [112] use integer rebalancing variables. User trip variables for rentals and returns also interact with the inventories of stations. Most models in the literature use continuous variables, except for [111], which use integer O-D variables.

3.2.3 Trip Distribution

If the rental/return demand exceeds the current inventory of bikes/docks, a basic optimization model (such as the one in Section 3.3.1) may select the rentals/returns opportunistically according to the objective function. In reality, however, demand will be satisfied based on a first-come-first-serve rule. Several works therefore aimed at enforcing a more realistic distribution of the trips by adding specific constraints. We review the existing assumptions on demand distribution below:

- **Station-based variables without distribution**. Two variables are created to present rentals and returns for each station and each period (see, e.g., [59]). Similarly, [23] use two variables to represent the shortage and excess of bikes, which is equivalent to the use of station-based variables. Demands will be greedily satisfied in the optimization model without considering the link between rentals and returns. As a result, the lost demand tends to be underestimated.

- **Station-based variables with proportional distribution**. [69] enforce a trip distribution proportional to the O-D trip demand as $x_{s_2}^{-,t_2} \leq \sum_{t=0}^{t_2-1} \sum_s x_s^{+,t} \frac{F_{s,s_2}^{t,t_2}}{f_s^{+,t}}$. Especially, the proportion is given by the ratio between the number of trips starting at station s in time-period t and ending at station s_2 in time-period t_2 (F_{s,s_2}^{t,t_2}) and the total number of trips starting at station s in time-period t ($f_s^{+,t}$). The authors also assume that the number of bikes returned during a specific time-period is not higher than the number of bikes rented in the previous periods multiplied by the corresponding proportion. Similarly, [108] consider a return ratio on the number of returns at station s divided by the total number of bikes currently used by customers during the time-period t. The

return demand at station s in period t is assumed to be the product of the return ratio and the total number of bikes being used.

- **O-D variables without distribution**. In this case, O-D variables will be created to represent the number of trips starting from one station and ending at another station during one particular time-period.

- **O-D variables with proportional distribution**. [45] and [114] apply a similar proportional distribution as [69]. The constraints $x_{s_1,s_2}^{t_1,t_2} \leq ab_{s_1}^{t_1} \frac{F_{s_1,s_2}^{t_1,t_2}}{f_{s_1}^{+,t_1}}$ imply that the trips rentals from a specific station are limited by the distribution ratio multiplied with the number of currently available bikes ($ab_{s_1}^{t_1}$).

- **Poisson distribution**. [21], [61], and [91] model the arrival of rentals and returns as a Poisson process, which implicitly models trip uncertainty.

These trip distribution constraints, as well as other alternatives, will be discussed in detail in Section 3.3.2.

3.2.4 Sequence of Rebalancing, Bike Rental, and Bike Return Events

In station-based BSSs, where the operator carries out station rebalancing using trucks, both customers and vehicles interact with the station inventories: customers may rent or return bikes, while trucks may drop off or pick up bikes. While in reality, customers and trucks interact with the station inventory at a specific moment in time, an optimization model aggregates these operations within each time-period.

Different assumptions can be made to deal with this issue. Some or all of these events can be assumed to happen simultaneously, allowing rentals and pick-ups to compensate for returns and drop-offs occurring within the same time-period. Such a generous assumption may achieve a high demand satisfaction within the optimization model. In practice, however, this may be overly optimistic and not perform well, i.e., rentals may occur at empty stations before returns and drop-offs, or returns may occur at full stations before rentals and pick-ups. Alternatively, one may assume that these events occur in a pre-defined chronological sequence within each time-period; for example, bike rentals occur first, then bike returns, and finally, the rebalancing operations. While such an assumption is more restrictive concerning demand satisfaction, it assumes that rentals can only be performed if sufficient bikes are available before the returns, and customer demands cannot benefit from the rebalancing operations that are assumed to happen at the end of the time-period.

Let us denote by **(r)** the event of vehicle **rebalancing**, **(a)** the event of customer **arrival** to return bikes, and **(d)** customer **departure**, i.e., bike rental. While models in the literature have assumed different event sequences, theoretically, any combination of these three event types is possible.

Table 3.1 summarizes the possible combinations, where events within the same parentheses are assumed to happen simultaneously. For example, (r)(a)(d) assumes that rebalancing is performed first, then customer arrivals, and then customer departures. In contrast (r+a+d) assumes that all three events happen at the same time. Note also that all sequences reported within the same row in Table 3.1 have the same order of events, but not necessarily within the same time-period. For example, both (r)(a)(d) and (a)(d)(r) assume that arrivals occur after rebalancing and departures occur after arrivals if the sequence is observed over several time-periods, e.g., (r)(a)(d)(r)(a)(d)(r)(a)(d), etc. However, such similarity does not guarantee a similar performance of the obtained solutions.

Table 3.1 Potential sequences of events

Three separate events	(r)(a)(d)	(a)(d)(r) (d)(r)(a)
	(r)(d)(a)	(d)(a)(r) (a)(r)(d)
Two events simultaneously	(r)(a+d)	(a+d)(r)
	(a)(d+r)	(d+r)(a)
	(d)(r+a)	(r+a)(d)
All events simultaneously	(r+a+d)	

Within the existing literature, [59] use a series of station-visit events and extreme values of cumulated demand to discretize the planning horizon. Since the time required to pick up and drop off bikes is neglected, there is no particular order of events. However, their model essentially assumes (a+d)(r), which means that customers first rent and return bikes before vehicles rebalance. [112] assume that rebalancing happens first and then customers rent and return bikes. [44] use sequence (a)(r)(d), while [69] use sequence (a)(d+r). Several other works (see, e.g., [16, 23, 45, 74, 109, 114]) ignore the issue of event sequences, which is equivalent to assuming a simultaneous event sequence (r+a+d). Finally, a different approach is taken by [108], who subdivide each time-period into smaller time-segments and associate rental and return demand to such fine-grained time-segments. Bike pick-ups from rebalancing operations are assumed to happen at the first segment of each time-period, while drop-offs occur at the end. Such an assumption does not fit our classification scheme, but the discretization into time segments resembles the operating mode of our simulator.

Unfortunately, no studies are available exploring the degree of realism and effectiveness of the different event sequences. In Section 3.3.2, we will therefore explicitly review the modeling of the various alternatives and empirically evaluate their effectiveness.

Other assumptions and objective functions can be found in Appendix A.1.

3.3 Multi-period Rebalancing Modeling Framework

We now present a general modeling framework that can be adapted to the various assumptions discussed in Section 3.2. To this end, we first propose a basic multi-period optimization model for dynamic rebalancing. We then formulate the different assumptions, which can be easily incorporated into the basic model.

3.3.1 Formulation of the Basic Model

We first consider a basic multi-period model with minimal assumptions. Its input parameters are summarized in Table 3.2. Namely, S denotes the set of stations, while V denotes the set of available vehicles. Each station $s \in S$ has a total of C_s docks, referred to as its capacity. Each vehicle $v \in V$ has a bike capacity $\hat{C}_v$. Parameters $D_{i,j}$ and $R^t_{i,j}$ denote respectively the distance and transit time between stations i and j at time-period t. We consider a planning horizon with $|T|$ time-periods, where each time-period $t \in T$ has a duration of L_t minutes.

Table 3.2 Input parameters of the optimization model

Input	Definition
S	The set of stations.
V	The set of vehicles.
T	The set of discretized time-periods.
$D_{i,j}$	The distance between station $i \in S$ and $j \in S$.
C_s	The capacity of station $s \in S$.
$\hat{C}_v$	The capacity of vehicle $v \in V$.
L_t	The length (in minutes) of time-period $t \in T$.
d^1_s	The initial number of bikes in station $s \in S$.
$\hat{d}^1_v$	The initial number of bikes in vehicle $v \in V$.
$z^1_{s,v}$	1, if vehicle $v \in V$ is at station $s \in S$ at the beginning of planning; 0, otherwise.
$f^{+,t}_s$	The expected rental demand at station $s \in S$ in period $t \in T$.
$f^{-,t}_s$	The expected return demand at station $s \in S$ in period $t \in T$.
$R^t_{s,s'}$	Transit time for vehicles from station $s \in S$ to station $s' \in S$ in period $t \in T$.
$F^{t,t'}_{s,s'}$	The number of trips from station $s \in S$ at period $t \in T$ to $s' \in S$ at period $t' \in T$.

We formulate the rebalancing problem as a MILP and assume that each vehicle can only visit one station at each period.

The decision variables are summarized in Table 3.3. Variables $r^{+,t}_{s,v}/r^{-,t}_{s,v}$ represent the number of bikes picked up/dropped off at station s by vehicle v during period t. Variable $z^t_{s,v}$ takes value 1 if station s visited by vehicle v at period t; 0 otherwise. For each time-period, intermediate variables are used, such as the number of bikes available at stations/in vehicles,

Table 3.3 Decision variables of the optimization model

Variable	Definition
$\boldsymbol{r_{s,v}^{+,t}}$	The number of bikes picked up at station s by vehicle v in period t
$\boldsymbol{r_{s,v}^{-,t}}$	The number of bikes dropped off at station s by vehicle v in period t
$\boldsymbol{z_{s,v}^{t}}$	1, if vehicle $v \in V$ is at station $s \in S$ at period $t \in T$; 0, otherwise.
d_s^t	The number of bikes available in station $s \in S$ at the beginning of period t
$\hat{d}_v^t$	The number of bikes in vehicle $v \in V$ at the beginning of period t
$p_{s,s',v}^t$	1, if vehicle v is at station s in period t and at station s' in period $t+1$; 0, otherwise
$x_s^{+,t}$	The number of successful bike trips starting from station s in period t
$x_s^{-,t}$	The number of successful bike trips ending at station s in period t

successful trips, and the routes of the vehicles. We employ rental and return demand without enforcing trip distribution, and use **station-based trip** variables $x_s^{+,t}$ and $x_s^{-,t}$.
Then, the basic MILP model reads as follows:

$$\min \quad \sum_{s\in S}\sum_{t\in T}(f_s^{+,t} - x_s^{+,t}) + \sum_{s\in S}\sum_{t\in T}(f_s^{-,t} - x_s^{-,t}) \tag{3.1}$$

$$\text{s.t.} \quad \hat{d}_v^{t+1} = \hat{d}_v^t + \sum_{s\in S}(r_{s,v}^{+,t} - r_{s,v}^{-,t}) \qquad \forall\ v \in V,\ t \in T \tag{3.2}$$

$$d_s^{t+1} = d_s^t - \sum_{v\in V}(r_{s,v}^{+,t} - r_{s,v}^{-,t}) - x_s^{+,t} + x_s^{-,t} \qquad \forall\ s \in S,\ t \in T \tag{3.3}$$

$$\sum_{s\in S} z_{s,v}^t = 1 \qquad \forall\ v \in V,\ t \in T \tag{3.4}$$

$$r_{s,v}^{+,t} + r_{s,v}^{-,t} \le \hat{C}_v z_{s,v}^t \qquad \forall\ s \in S,\ v \in V,\ t \in T \tag{3.5}$$

$$0 \le \hat{d}_v^t \le \hat{C}_v, 0 \le d_s^t \le C_s \qquad \forall\ s \in S,\ v \in V,\ t \in T \tag{3.6}$$

$$0 \le x_s^{+,t} \le f_s^{+,t}, 0 \le x_s^{-,t} \le f_s^{-,t} \qquad \forall\ s \in S,\ t \in T \tag{3.7}$$

$$0 \le r_{s,v}^{+,t}, r_{s,v}^{-,t} \le \hat{C}_v \qquad \forall\ s \in S,\ v \in V,\ t \in T \tag{3.8}$$

$$z_{s,v}^t \in \{0,1\} \qquad \forall\ s \in S,\ v \in V,\ t \in T \tag{3.9}$$

Objective function (3.1) minimizes the total lost rental and return demand in the planning horizon over all stations and time-periods. If required, it can be modified according to the preferences of the BSS operators (see Appendix A.1). Constraints (3.2) ensure that the number of bikes in each vehicle is synchronized with the vehicles' bike pick-ups and drop-offs. Constraints (3.3) manage the station inventory along time, considering the rebalancing operations and successful customer trips (i.e., rentals and returns). Note that we use the sequence (r+a+d) in our basic model. The initial inventory of stations d_s^1 is an input of the optimization model and can be obtained from the operators or through static rebalancing. Constraints (3.4) ensure that each vehicle is at exactly one station at each time-period, which forms the flow of vehicles sequentially. [44] use an alternative constraint: $\sum_{s'} p_{s,s',v}^t - \sum_{s'} p_{s',s,v}^{t-1} = 0\ (\forall\ s,t,v)$, which directly ensures that the flow out of station s for vehicle v at

time-period t is equivalent to the flow of v into the station s at time $t-1$. Both of them indicate the relocation of vehicles along time. Note that, in our model, the vehicle can stay at the same station in the next time-period, i.e., $z^t_{s,v} = z^{t+1}_{s,v} = 1$. Constraints (3.5) ensure that a vehicle only operates at the station where it is currently located. Constraints (3.6) enforce that the number of bikes in each vehicle is limited by its capacity and the number of bikes at each station is within the station's capacity. Constraints (3.7) impose that the number of successful trips is bounded by the expected demand for rentals and returns. Constraints (3.8) force the number of picked-up/dropped-off bikes to respect the vehicle capacity.

The model can easily be reformulated using **O-D** variables $x^{t,t'}_{s,s'}$ instead of station-based variables. In this case, all occurrences of $x^{+,t}_s$ within (3.1)–(3.3) simply have to be replaced by $\sum_{s',t'} x^{t,t'}_{s,s'}$, while Constraints (3.7) have to be replaced by $x^{t,t'}_{s,s'} \leq F^{t,t'}_{s,s'}$.

3.3.2 Formulating Different Modeling Assumptions

We now show how to extend the basic model to account for the additional modeling assumptions discussed in Section 3.2.

Time Constraints.

The basic model assumes that vehicles can relocate to any other stations and carry out the rebalancing operations within the duration of a time-period. When the duration of the time-period is short, the resulting planning solution may become infeasible in practice. Since vehicles may not have sufficient time to relocate and rebalance bikes, time constraints (as discussed in Section 3.2.1) may be added to restrict the vehicle relocation between stations and rebalancing operations to the time available. We formulate time constraints as follows. First, for each pair of stations s and s', vehicle v, and time-period t, Constraints (3.10) enforce variable $p^t_{s,s',v}$ to take value 1 if both variables $z^t_{s,v}$ and $z^{t+1}_{s',v}$ have value 1. Then, time constraints (3.11) guarantee that the transit time between stations and the operation time for picking up/dropping off bikes for each period will not surpass the available time L_t.

$$z^t_{s,v} + z^{t+1}_{s',v} - 1 \leq p^t_{s,s',v} \qquad \forall\ s, s' \in S,\ v \in V,\ t \in T \tag{3.10}$$

$$\sum_{s \in S} \sum_{s' \in S} p^t_{s,s',v} R^t_{s,s'} + op \sum_{s \in S} (r^{+,t}_{s,v} + r^{-,t}_{s,v}) \leq L_t, \qquad \forall\ v \in V,\ t \in T \tag{3.11}$$

$$p^t_{s,s',v} \in \{0,1\} \qquad \forall\ s, s' \in S,\ v \in V,\ t \in T, \tag{3.12}$$

where op is the average operational time to pick up/drop off a single bike. Parameters $|T|$ and L_t can be altered by the decision-maker. In Section 3.5.3, we will test different lengths of time-periods along with the time constraints to explore their impact on the solution performance.

Trip distribution constraints.

We now discuss the proportionality distribution for station-based trip variables. The proportionality distribution constraints for O-D variables can be found in Appendix A.2.

Consider a trip starting at station s_1 in period t_1 and ending at station s_2 in period t_2. The station-based trip variables related to this trip are $x_{s_1}^{+,t_1}$ and $x_{s_2}^{-,t_2}$. The *proportional distribution* can be written as:

$$x_{s_2}^{-,t_2} \leq \sum_{t=0}^{t_2-1} \sum_{s \in S} x_s^{+,t} \frac{F_{s,s_2}^{t,t_2}}{f_s^{+,t}} \qquad \forall\ s_2 \in S,\ t_2 \in T \qquad \text{(TD1)}$$

$$x_{s_1}^{+,t_1} \leq \sum_{t=t_1+1}^{|T|} \sum_{s \in S} x_s^{-,t} \frac{F_{s_1,s}^{t_1,t}}{f_s^{-,t}}, \qquad \forall\ s_1 \in S,\ t_1 \in T \qquad \text{(TD2)}$$

where, as discussed, $\frac{F_{s,s_2}^{t,t_2}}{f_s^{+,t}}$ represents the proportion of all rental demand from s at time-period t that will be returned to s_2 at time-period t_2.

Constraints (TD1) impose that the number of bikes returned to station s_2 during period t_2 is no more than the bikes rented in the previous periods with a proportion of $\frac{F_{s,s_2}^{t,t_2}}{f_s^{+,t}}$. Conversely, constraints (TD2) impose that the number of bikes rented in station s_1 during period t_1 is no more than the bikes returned in the later periods with proportion $\frac{F_{s_1,s}^{t_1,t}}{f_s^{-,t}}$.

Similarly, instead of considering the number of rented/returned bikes, we can consider the number of available bikes/docks. To this end, we rewrite (TD1) and (TD2) to (TD3) and (TD4) by replacing $x_s^{+,t}$ with $ab_{s_1}^{t_1}$ and $x_s^{-,t}$ with $ad_{s_1}^{t_1}$.

$$x_{s_2}^{-,t_2} \leq \sum_{t=0}^{t_2-1} \sum_{s \in S} ab_s^t \frac{F_{s,s_2}^{t,t_2}}{f_s^{+,t}} \qquad \forall\ s_2 \in S,\ t_2 \in T \qquad \text{(TD3)}$$

$$x_{s_1}^{+,t_1} \leq \sum_{t=t_1+1}^{|T|} \sum_{s \in S} ad_s^t \frac{F_{s_1,s}^{t_1,t}}{f_s^{-,t}}, \qquad \forall\ s_1 \in S,\ t_1 \in T \qquad \text{(TD4)}$$

where ab_s^t and ad_s^t are the number of available bikes and docks respectively at station s in period t.

Given that station-based variables do not maintain the link between bike rental and return, we may use Constraints (TD5) below to enforce that the total number of rentals equals the total number of returns. When all trips are assumed to take no longer than one time-period, one may use Constraints (TD6) below to enforce a stronger relationship. Under the same assumption, Constraints (TD6) can also be derived by summing (TD2) all over s_1.

$$\sum_t \sum_s x_s^{+,t} = \sum_t \sum_s x_s^{-,t}, \qquad \text{(TD5)}$$

$$\sum_s x_s^{+,t} = \sum_s x_s^{-,t+1} \qquad \forall t \in T. \qquad \text{(TD6)}$$

Practical toy example. To develop an intuition of the impact of the trip distribution constraints, we consider the following toy example. We consider two time-periods with four stations, each of which has a pair of $[ab_s^t, ad_s^t]$ indicating the current number of bikes available for rentals and docks for returns, visualized in Figure 3.1. The value of ab_s^t is equal to $d_s^t - \sum_v (r_{s,v}^{+,t} - r_{s,v}^{-,t}) + x_s^{-,t}$ and it is analogous for the value of ad_s^t. The numbers circled in red along the arcs represent the trip demands for each station pair. Stations s_1 and s_2 may either have a sufficient (S) or insufficient (I) number of bikes to satisfy rental demand. Further, station s_3 and s_4 may either have a sufficient (S) or insufficient (I) number of empty docks to satisfy return demand. This leads to 4 different configurations shown in Figure 3.1.

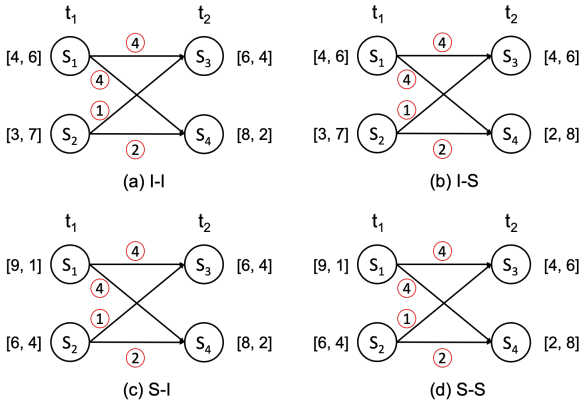

Figure 3.1 Toy example with 4 different situations of bike/dock availability

Ideally, for scenario I-I, a BSS operator would expect to see 4 trips to station s_3 and 2 trips to station s_4 to fill their empty docks, and will not mind whether the trips come from station s_1 or station s_2 as long as they have sufficient bikes. The ideal distribution is similar for scenario S-I. In contrast, in the case of scenario I-S, the operator would expect to see 4 trips from station s_1 and 3 trips from station s_2 such that all the available bikes can be used. However, there is no preference for the destinations of the 4 trips from station s_1. Clearly, scenario S-S is irrelevant since all trip demand is satisfied.

In order to compute the successful trips under different trip distribution constraints, we solved the basic model with each of them. For station-based trip variables, based on Constraints (3.3), (3.6), and (3.7), we have $x_{s_1}^{+,t_1} \leq \min\{f_{s_1}^{+,t_1}, ab_{s_1}^{t_1}\}$ and $x_{s_3}^{-,t_2} \leq \min\{f_{s_3}^{-,t_2}, ad_{s_3}^{t_2}\}$.

Minimizing the lost demand for cases I-I, I-S, and S-I in Figure 3.1 under the different trip distribution constraints results in the departures and arrivals $[x_{s_1}^{+,t_1}, x_{s_2}^{+,t_1}, x_{s_3}^{-,t_2}, x_{s_4}^{-,t_2}]$ indicated in Table 3.4. Here, all ideal trip distributions as expected by the BSS operator are indicated in the first row (an '*' refers to any coherent allocation). Note that all trip

distribution constraints result in the same solution for case S-S since all trip demands can be satisfied. The solutions that are considered coherent with an ideal solution are indicated in bold.

Table 3.4 Trip distribution for 3 different demand scenarios under different trip distribution constraints for station-based variables

Constraints	I-I	I-S	S-I
Ideal solution	[*, *, 4, 2]	[4, 3, *, *]	[*, *, 4, 2]
without TD	[4, 3, 4, 2]	[4, 3, 5, 6]	[8, 3, 4, 2]
(TD1)	[4, 3, 3, 2]	**[4, 3, 3, 4]**	[8, 3, 4, 2]
(TD2)	[4, $\frac{22}{15}$, 4, 2]	[4, 3, 5, 6]	[$\frac{68}{15}$, $\frac{22}{15}$, **4, 2**]
(TD6)	[$x_{s_1}^{+,t_1} + x_{s_2}^{+,t_1} = 6$, 4, 2]	[4, 3, $x_{s_3}^{-,t_2} + x_{s_4}^{-,t_2} = 7$]	[$x_{s_1}^{+,t_1} + x_{s_2}^{+,t_1} = 6$, **4, 2**]
(TD1)+(TD2)	[$\frac{8}{3}$, 1, $\frac{5}{3}$, 2]	[4, $\frac{3}{2}$, $\frac{5}{2}$, 3]	[$\frac{8}{3}$, 1, $\frac{5}{3}$, 2]
(TD1)+(TD6)	[4, 0, 2, 2]	**[4, 3, 3, 4]**	[4, 0, 2, 2]
(TD2)+(TD6)	[4, $\frac{4}{3}$, $\frac{10}{3}$, 2]	[4, 2, 0, 6]	[$\frac{68}{15}$, $\frac{22}{15}$, **4, 2**]
(TD3)	[4, 3, 3, 2]	**[4, 3, 3, 4]**	[8, 3, 4, 2]
(TD4)	[4, $\frac{22}{15}$, 4, 2]	[4, 3, 5, 6]	[$\frac{68}{15}$, $\frac{22}{15}$, **4, 2**]
(TD3)+(TD4)	[4, $\frac{22}{15}$, 3, 2]	**[4, 3, 3, 4]**	[$\frac{68}{15}$, $\frac{22}{15}$, **4, 2**]
(TD3)+(TD6)	[4, 1, 3, 2]	**[4, 3, 3, 4]**	[$x_{s_1}^{+,t_1} + x_{s_2}^{+,t_1} = 6$, **4, 2**]
(TD1)+(TD4)	[4, $\frac{22}{15}$, $\frac{112}{45}$, 2]	**[4, 3, 3, 4]**	[$\frac{68}{15}$, $\frac{22}{15}$, $\frac{124}{45}$, 2]
(TD2)+(TD3)	[$\frac{56}{15}$, $\frac{19}{15}$, 3, 2]	[4, $\frac{29}{15}$, 3, 4]	[$\frac{68}{15}$, $\frac{22}{15}$, **4, 2**]

According to the observed trip distribution, (TD3)+(TD6) and (TD3)+(TD4) have the potential to produce trips that are close to an ideal solution. The combination of (TD1) and (TD2) will be tight for the feasible region, especially for full/empty stations. Constraints (TD1)+(TD6) introduce strict proportional limitations for rentals when the docks are insufficient for returns, which deviates from the ideal solution. Using only (TD1) may result in solutions with more rentals than returns.

Based on this analysis, within our empirical evaluation in Section 3.5, we will consider the station-based variable model without trip distribution constraints, with (TD1), (TD6), (TD3)+(TD6), and (TD3)+(TD4). The trip distribution of O-D variables, as well as the discussion, is illustrated in Table A.3 of the Appendix A.

Sequences of events

The basic station-based model (3.1)–(3.9) implicitly uses event sequence (r+a+d), where the three events are assumed to occur simultaneously. We now show how this model can be modified to take into account the different sequences of rebalancing, rental, and return events. We give several sequence examples which perform well in the following experiments or are used in the literature.

(r)(a)(d): Since the rebalancing operations performed by the vehicles occur at the beginning of each time-period, Constraints (3.13)–(3.16) need to be added, in order to ensure that arrivals consider the rebalancing and departures consider both rebalancing and departures.

$$\sum_v r_{s,v}^{+,t} \leq d_s^t \qquad \forall\ s \in S,\ t \in T \tag{3.13}$$

$$\sum_v r_{s,v}^{-,t} \leq C_s - d_s^t \qquad \forall\ s \in S,\ t \in T \tag{3.14}$$

$$x_s^{+,t} \leq d_s^t - \sum_v r_{s,v}^{+,t} + \sum_v r_{s,v}^{-,t} + x_s^{-,t} \qquad \forall\ s \in S,\ t \in T \tag{3.15}$$

$$x_s^{-,t} \leq C_s - d_s^t + \sum_v r_{s,v}^{+,t} - \sum_v r_{s,v}^{-,t} \qquad \forall\ s \in S,\ t \in T \tag{3.16}$$

(r)(d)(a): Since rebalancing occurs first, we require Constraints (3.13) and (3.14). In order for bike rentals to consider the previous rebalancing, we further require Constraints (3.17). Finally, bike returns are already correctly implemented due to Constraints (3.3) and (3.6).

$$x_s^{+,t} \leq d_s^t - \sum_v r_{s,v}^{+,t} + \sum_v r_{s,v}^{-,t} \qquad \forall\ s \in S,\ t \in T \tag{3.17}$$

(r)(a+d): Here, we use Constraints (3.13) and (3.14) since vehicles operate at the beginning of each period. The restrictions related to the rentals and returns are already satisfied by Constraints (3.3) and (3.6). The constraints required for all remaining event sequences can be found in Appendix A.3.

3.4 Dynamic Rebalancing Evaluation Framework

Evaluation framework for rebalancing strategies. The framework used to obtain rebalancing strategies for given problem instances and to evaluate their estimated performance in practice is visualized in Figure 3.2. The input set contains the trip demand with exact time-stamps, referring, for example, to historical days with similar demand patterns (e.g., weekdays).

In order to analyze the effect of the various modeling assumptions in a controlled environment, we first generate synthetic problem instances, including a station network along with probability distributions of trip demand over a specified time horizon (in our case one day). For each problem instance, an input set and a test set are sampled, containing a certain amount of days of trip demands. The deterministic optimization model then receives as input the expected trip demand (i.e., a single demand scenario). While this point estimate may be provided by a predictive model, we here refrain from using a predictive model to focus on the impact of the modeling assumptions without potential interference from prediction errors. Instead, we compute the expected demand by averaging over the demand

(for each time-period and station) of all days in the input set. Note that, if a stochastic (scenario-based) model was used, each scenario would contain the trip data of a different day.

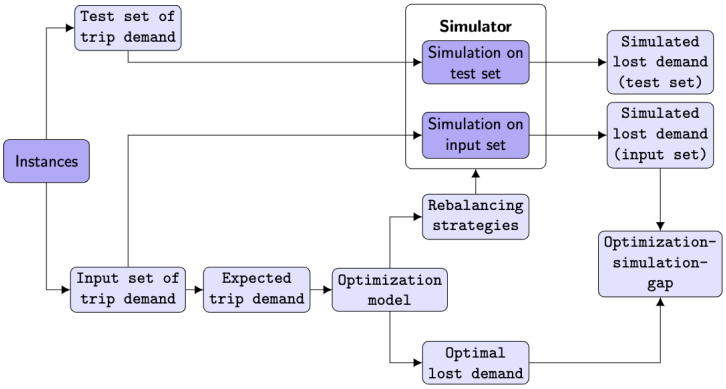

Figure 3.2 Evaluation framework for dynamic rebalancing strategies

The rebalancing strategies obtained from the optimization model are then applied to a simulated BSS, which aims at realistically estimating the performance (e.g., the lost demand) of the planning solutions on the various trip demand days within the test data set.

We also define an optimization-simulation-gap, representing the difference between the number of successful trips in the deterministic optimization model and the average number of successful trips simulated on each day in the input set. This gap therefore estimates the deviation from reality, largely explained by the temporal aggregation of the optimization model versus a FIFO policy that applies in reality.

Specifically, $\left(|\sum_{t,s} x_s^{+,t} - \bar{x}_s^{+,t}|\right) / \left(\sum_{t,s} \bar{x}_s^{+,t}\right)$ and $\left(|\sum_{t,s} x_s^{-,t} - \bar{x}_s^{-,t}|\right) / \left(\sum_{t,s} \bar{x}_s^{-,t}\right)$ compute the relative gaps of rentals and returns, respectively, over the entire planning horizon and all stations, where $x_s^{+,t}$/ $x_s^{-,t}$ are rentals/returns as computed by the optimization model and $\bar{x}_s^{+,t}$/ $\bar{x}_s^{-,t}$ are the average number of successful rentals/returns within the simulator.

Fine-grained simulator. Using a simulator to evaluate the performance of the proposed rebalancing strategies has been a common approach in the literature [44, 46]. Most simulators, however, aggregate the entire demand of each time-period, therefore allowing return demand to cancel out rental demand (and vice-versa). This is overly optimistic and deviates from the first-arrive-first-serve policy in practice. The required operating time for rebalancing is also typically ignored.

Aiming at a more realistic evaluation, we develop a discrete-event simulator taking into account more realistic operational BSS mechanisms. Each time-period used in the optimization model (spanning 30 or 60 minutes) is further discretized into 1-minute *time-slots* (which is

sufficiently fine-grained to be considered real-time in practice). We consider both users' behaviors (rentals and returns) and trucks' operations (pick-ups and drop-offs) as discrete events, each of which is associated with a specific 1-minute time-slot. Rebalancing operations may occur in parallel to rentals and returns and depend on the time the truck arrives at the station. Customer trips and rebalancing operations are therefore considered in chronological order.

For ease of presentation, the operation mode of the simulator is now described verbally. First, the simulator *initializes* the station and vehicle inventories according to the input data. Each rental demand is associated with its respective 1-minute time-slot. Pick-up and drop-off attempts for the first time-period are associated with their respective time-slot, taking into consideration the time to load/unload bikes on/from the truck.

The simulator then scrolls through the time-slots minute by minute, attempting to *perform all scheduled events* for that time-slot. The operating rules in this iterative process can be summarized as follows: (i) A rental demand is satisfied if the station holds at least one bike. In this case, a return demand is created for the destination station and associated with a future time-slot based on an estimated travel time. If station inventory is insufficient, the rental demand is counted as lost. (ii) Analogously, a return demand is satisfied if a free dock is available. Otherwise, the bike is returned to the nearest station with an available dock. However, this return demand is then counted as lost. Note that lost returns can only occur if the corresponding rental demand was successful. (iii) Drop-off and pick-up attempts from rebalancing operations are carried out as best as the available inventory and available docks at the stations and the vehicles allow. (iv) Once a truck has finished the rebalancing attempt, it departs to the next station as prescribed by the planning solution for the next time-period. The arrival event is scheduled for a future time-slot based on the estimated travel time of the vehicle. (v) Once a truck arrives at a new station, it is assumed to immediately start the rebalancing operations. However, rebalancing will not start before the first time-slot associated with the current time-period.

A pseudo-code of the simulator, along with a technical description can be found in Appendix A.5.

3.5 Computational Experiments

We now report on different sets of computational experiments on both synthetic and real-world data to systematically explore the impact of the various modeling assumptions, based on the evaluation framework we proposed in Section 3.4.

In Section 3.5.1, we first elaborate on the synthetic problem instances used throughout the majority of our experiments. In Section 3.5.2, we compare dynamic and static rebalancing, as well as the usage of different variable types. Section 3.5.3 focuses on the impact of time discretization and time constraints. Section 3.5.4 analyzes the importance of the various trip distribution constraints. In Section 3.5.5, we cross-test whether the previous findings still hold when different variable domains are used. Section 3.5.6 focuses on the impact of the assumed event sequence. Finally, Section 3.5.7 empirically validates a variety of such modeling assumptions on real-world data from the Montreal bike-sharing system.

3.5.1 Generation of Synthetic Data and Computing Environment

Even though we have access to real-world trip data from different BSSs, the majority of our experiments are based on synthetically generated instances for a variety of reasons. First, the unobserved demand makes it difficult to obtain accurate rebalancing strategies and evaluate their performance. Second, existing data often contains errors and noise concerning trip and station inventory. Third, rebalancing operations carried out in the BSS alter station inventories, but existing data sets do not provide reliable data on the rebalancing carried out. We therefore develop an instance generator that aims at generating realistic instances with BSS networks of different sizes and characteristics, as well as trip data that is coherent with trips observed in reality (see details in Appendix A.4). Note, however, that we validate the most relevant conclusions within a case study based on real-world data in Section 3.5.7.

For the purpose of our study, we only focus on weekdays, since they have similar demand patterns for work-related trips and demand tends to be much higher than on weekends. We divide the entire daily trip demand into four types. People who live outside city centers and work inside city centers typically use similar origin (outside city centers) and destination stations (within city centers) during peak hours. These trips are denoted as OI trips. Trips of people who live and work outside the centers are referred to as OO trips. In contrast to work-related OI and OO trips, RD trips refer to random trips occurring during the day and RN trips refer to random trips during the night. Such random trips do not have the same origin and destination stations. The departure time for each trip type is characterized by a Beta distribution (see Appendix A.4). The average demand per 30-minute duration for weekdays of one week in July 2019 at BIXI is visualized in Figure 3.3(a). For comparison, the trip demand averaged over 500 days as generated by our instance generator is displayed in Figure 3.3(b) and shows a similar pattern as the trip demand observed at BIXI.

We generate 3 ground truths with different station networks. For each ground truth, we generate 5 instances with the proportions for the four trip types. For each instance, we

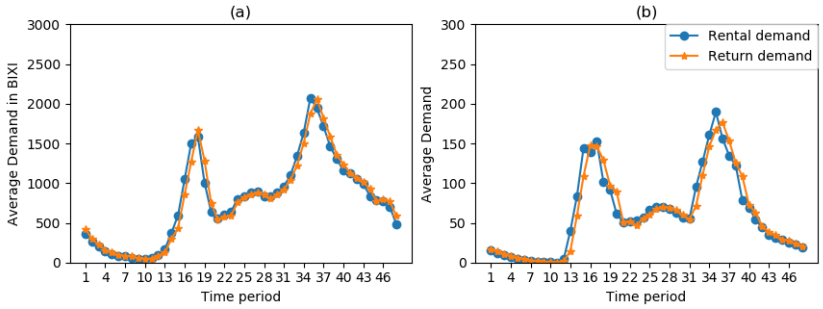

Figure 3.3 Total rental and return demand over 24 hours (48 time-periods) : (a) Average weekday demand at BIXI ; (b) Average demand in the synthetic instance GT1 (Sample of 500 days)

.

generate 5000 weekdays, from which a single expected demand is computed for each time-period and station as input for the optimization model. We then generate 1000 weekdays of trips as test data, on each of which the planned solution will be simulated. Table 3.5 shows the characteristics of the 3 ground truths. The percentage of stations within the city centers and those associated with each trip type are indicated in rows 2 to 5. Ground Truth 3 (GT3) has more work-related trips, compared to Ground Truth 1 (GT1). Ground Truth 2 (GT2) has two city centers under the same trip pattern as GT1. A detailed description can be found in Appendix A.4.

Table 3.5 The parameters for the ground truths

		GT1	GT2	GT3
Station Network	Number of city centers	1	2	1
	City center capacity	26%	35%	26%
Trip Pattern	OI	32% - $\beta(3,8)$	32% - $\beta(3,8)$	55% - $\beta(3,8)$
	OO	32% - $\beta(3,7)$	32% - $\beta(3,7)$	25% - $\beta(3,7)$
	RD	23% - $\beta(3,7)$	23% - $\beta(3,7)$	15% - $\beta(3,7)$
	RN	13% - $\beta(6,8)$	13% - $\beta(6,8)$	5% - $\beta(6,8)$

Our optimization models are solved by IBM ILOG CPLEX on 2.70 GHz Intel Xeon Gold 6258R machines with 8 cores. The stopping criterion for the optimization model is a MIP optimality gap of 0.01% and a maximum running time of 24 hours. In the following, we will report results for GT1 and GT2. While GT3 has been harder to solve, its results demonstrated the same conclusions. Detailed results can be found in Appendix A.6.

3.5.2 Impact of Initial Station Inventory and Trip Variable Types

To quantify the impacts of the initial inventory at stations, we define two baselines as the pre-allocated inventory.

- **Baseline 1: Inventory proportional to rental demands without rebalancing**. The initial inventories of stations at the beginning of a day are set to predefined levels proportional to the rental demands of the first time-period in the planning horizon. We round the values to the closest integer, respecting the station capacities and the total number of bikes available in the system.
- **Baseline 2: Static rebalancing only**. The optimal static rebalancing is obtained by solving the problem given by (A.13)–(A.17) in Appendix A.6.2, where the inventories for the first time-period are decision variables that sum to the total number of available bikes in the system.

In the following, we set the initial inventory of the stations according to these two baselines in the optimization model and run our simulator without dynamic rebalancing for all 3 ground truths. We consider a planning horizon from 6 a.m. to 1 p.m. (i.e., 7 hours) and divide the planning horizon into 14 time-periods with a length of 30 minutes each. We calculate the average rental and return demands for each instance over the input set at each station of each time-period. Rebalancing strategies are obtained through the optimization model and then applied in the simulator to estimate the lost demand on the test set.

Table 3.6 summarizes the results for GT1 and GT2. The results for GT3, as well as for the O-D variables can be found in Appendix A.6.2. We report the optimal value of the objective function as 'O.F. Value' and the running time of the optimization model as 'Opt. Time' in minutes. The 'MIP gap' refers to the optimality gap as reported by CPLEX when the stopping criterion is reached. The lost demand is computed as the relative gap between successful trips and the original demand specified in the problem instances. To be specific, $\frac{\sum_{s,t}(f_s^{+,t}-\hat{x}_s^{+,t})}{\sum_{s,t} f_s^{+,t}}$ defines the lost rental demand over the entire planning horizon, where $\hat{x}_s^{+,t}$ is the number of successful rentals in simulator. Similarly, the lost return demand is computed as $\frac{\sum_{s,t}(\hat{x}_s^{+,t}-\hat{x}_s^{-,t})}{\sum_{s,t} \hat{x}_s^{+,t}}$, where $\hat{x}_s^{-,t}$ is the number of successful returns in simulator. Since, in practice, return demand does not exist when the corresponding rental demand is unsuccessful, the lost returns are only associated with successful rentals $\hat{x}_s^{+,t}$. Lost return demand has to be interpreted critically since the relative lost return may be high when the lost rental is low (which doesn't indicate a low-quality planning solution). In our result analysis, we, therefore, emphasis on the lost rental.

For each ground truth, the initial inventory observed from Baselines 1 and 2 is directly applied to the simulator without any rebalancing operations, whose average lost demand over 5 instances is reported as 'Baseline 1/2 without rebal.'.

Table 3.6 Station-based model with baseline 1 and baseline 2 (60 stations, 4 trucks, 30 mins)

	Baselines, Configuration, Trip Modeling	O.F. Value	Opt. Time (mins)	MIP Gap (%)	Lost Demand (%) Rental	Return
GT1	Baseline 1 without rebal.	-	-	-	26.05	10.27
	Baseline 2 without rebal. (static)	-	-	-	10.40	11.79
	Baseline 1 dyn.rebal. station-based	44.2	1440	0.10	11.11	2.12
	Baseline 2 dyn.rebal. station-based	0.8	<1	0.00	8.78	7.99
GT2	Baseline 1 without rebal.	-	-	-	21.22	6.85
	Baseline 2 without rebal. (static)	-	-	-	11.61	3.18
	Baseline 1 dyn.rebal. station-based	6.4	294	0.01	11.22	1.78
	Baseline 2 dyn.rebal. station-based	0.5	<1	0.00	9.46	1.78

According to the first two rows for each GT in Table 3.6 (baselines without rebalancing), the initial station inventory seems important to the performance of the BSS. Compared to Baseline 1, static rebalancing (Baseline 2) can significantly improve the lost demand. However, static rebalancing is still insufficient to meet customer demand.

Based on the initial station inventory from Baselines 1 and 2, rows 3-4 for each GT compare the impact of the two different strategies with additional dynamic rebalancing. For the dynamic rebalancing, we use model (3.1)–(3.9) without trip distribution and time constraints.

The lost rental is improved when dynamic rebalancing is applied. The performance of dynamic rebalancing varies substantially between the two baselines, which highlights the importance of optimizing the initial station inventory before the dynamic rebalancing. For GT1, the lost rental for dynamic rebalancing with Baseline 1 is higher, leading to a decrease in actual return demands. Given that we only consider lost returns for successful rentals, it is possible that the relative lost return is small when the relative lost rental is high (i.e., only a few successful rentals).

Since strategies based on Baseline 2 outperform those based on Baseline 1, in the following experiments, we will use Baseline 2 to define the initial inventory of each station. Note that we use pre-defined initial locations and inventories for trucks because previous experiments have shown that such assumptions do not affect the performance (see Appendix A.6.1 for details).

3.5.3 Impact of Time Discretization and Time Constraints

We now explore the impact of the length of time-periods and the use of time constraints. We consider two lengths of time-periods: 30-minute and 60-minute, and with or without time constraints. The comparative results for station-based trip variables are summarized in Table 3.7. The results of similar experiments using O-D variables can be found in Appendix A.6.3.

Table 3.7 Station-based model with/without time constraints in 30/60 mins (60 stations, 4 trucks)

	Time	Time	O.F.	Time	MIP	Lost Demand (%)		Opt-sim-gap (%)	
	Period (mins)	Constraints	Value	(mins)	Gap (%)	Rental	Return	Rental	Return
GT1	30	No	0.8	<1	0.00	8.78	7.99	9.62	15.88
	30	Yes	0.9	69	0.00	8.13	6.06	8.84	12.69
	60	No	9.5	<1	0.00	8.50	4.39	8.52	11.24
	60	Yes	9.5	2	0.00	8.73	5.85	8.80	13.27
GT2	30	No	0.5	<1	0.00	9.46	1.78	10.42	9.21
	30	Yes	0.5	28	0.00	8.48	1.90	9.24	8.18
	60	No	10.4	<1	0.00	9.72	1.20	9.88	8.90
	60	Yes	10.5	1	0.00	9.63	1.40	9.76	9.02

As shown in Table 3.7, 30-minute time-periods allow for more rebalancing operations within the optimization model, leading to smaller optimal objective function values than those for cases with 60-minute time-periods. Note again, that in our simulator, we postpone the rebalancing operations if the truck cannot reach the station in time due to long relocation distances. Using 30-minute time-periods, the lost rental without time constraints may therefore be worse than in the case of 60-minute time-periods. Coherently, time constraints with 30-minute time-periods give the best overall performance. Thus, if we have a tolerance for optimization time and the distances between stations tend to be large, a short time-period (30 mins) with time constraints seems beneficial. For longer time-periods, time constraints do not seem necessary. Using the model with O-D variables, we reach similar conclusions (see Appendix A.6.3).

3.5.4 Impact of Trip Distribution Constraints

We implement optimization models with various TD constraints as discussed in Section 3.3.2 for station-based trip variables with Baseline 2. The results of station-based trip variables are shown in Table 3.8. The results of O-D variables can be found in Table A.15 of the Appendix A.

Table 3.8 Station-based model with different trip distribution constraints (60 stations, 4 trucks, 30mins)

	Constraints	O.F. Value	Time (mins)	MIP Gap (%)	Lost Demand (%)		Opt-sim-gap (%)	
					Rental	Return	Rental	Return
GT1	(TD1)	600.3	1440	1.41	3.65	6.34	0.56	40.07
	(TD6)	137.7	<1	0.00	9.94	11.22	5.89	14.28
	(TD3)+(TD6)	240.3	<1	0.00	8.61	6.56	0.01	2.35
	(TD3)+(TD4)	170.5	1440	0.13	6.15	1.57	0.11	1.82
	None	0.8	<1	0.00	8.78	7.99	9.62	15.88
GT2	(TD1)	588.6	1440	1.48	4.78	1.76	1.20	40.70
	(TD6)	101.3	<1	0.00	10.86	3.21	8.40	7.45
	(TD3)+(TD6)	165.0	<1	0.00	8.68	2.08	3.14	0.94
	(TD3)+(TD4)	150.0	1440	0.06	7.34	1.20	2.05	0.56
	None	0.5	<1	0.00	9.46	1.78	10.42	9.21

According to Table 3.8, the use of Constraints (TD1) performs best in terms of lost rental for both input and test sets even if optimality has not been proven within 24h. This suggests that (TD1) reflects the flow of rentals more realistically as also shown by the low rental Opt-sim-gap. Since (TD1) imposes a strict restriction for returns, the lost return of the optimization model is quite high, and lost rental is low. That leads to a small opt-sim-gap for rental but a large one in return. Constraints (TD3)+(TD4) also provide a good performance. Both sets of Constraints (TD1) and (TD3)+(TD4) include the proportionality characteristics of the trip flow. They improve performance, but they require longer computing times.

Based on these experimental results, in the following, we will restrict our analysis to station-based variables with (TD1), (TD3)+(TD4), and (None).

3.5.5 Impact of Variable Domains

If the variable domains were selected such that they represent more realistically the BSS operations, trip variables would be binary, while station and vehicle inventory, as well as the rebalancing variables, would be integer. However, using such variable domains may result in models that may be difficult to solve and restricted by certain trip distribution constraints, severely underestimating successful trips. We now explore the impact of using different variable domains. In our models, routing variables $z_{s,t}^t$ are always binary. Let variables d_v^t, $r_{s,v}^{+,t}$, and $r_{s,v}^{-,t}$ be referred to as rebalancing variables and variables d_s^t, $x_s^{+,t}$, and $x_s^{-,t}$ be referred to as station variables. In the previous experiments, both rebalancing and station variables have been continuous, which we denote as an *All-continuous* model. Here, we also consider the other two cases: the *All-integer* model and the *Partially-integer* model. In the All-integer

model, both rebalancing and station variables are integers. In the Partially-integer model, the station variables are continuous, while the rebalancing variables are integers.

Table 3.9 Station-based model with different variable domains and trip distribution constraints for GT1 (30 stations, 2 trucks, 30 mins)

Variable Domains	Constraints	O.F. Value	Time (mins)	MIP Gap (%)	Lost Demand (%)		Opt-sim-gap (%)	
					Rental	Return	Rental	Return
All-continuous	(TD1)	262.7	1440	0.18	2.93	1.86	0.82	43.73
	(TD3)+(TD4)	82.7	19	0.00	6.00	0.87	1.62	2.86
	None	0.5	<1	0.00	8.95	4.73	9.83	11.80
Partially-integer	(TD1)	263.6	1440	0.33	2.95	1.29	0.75	44.11
	(TD3)+(TD4)	82.8	50	0.00	5.78	1.47	1.96	2.32
	None	0.5	<1	0.00	6.72	3.56	7.20	7.81
All-integer	(TD1)	656.5	140	0.08	3.89	4.62	30.43	81.39
	(TD3)+(TD4)	408.3	<1	0.00	9.22	1.96	29.56	29.60
	None	380.5	<1	0.00	10.50	7.65	24.88	22.36

Since the optimization models with 60 stations and trip distribution constraints (TD1) and (TD3)+(TD4) cannot be solved within the given time limit, we also carry out experiments with the 30-station network to reliably explore the impact of such constraints coupled with different variable domains. Specifically, we consider two ground truths with 30 stations using the same configurations as GT1 (see Table 3.5).

The results of the corresponding experiments for GT1 under different trip distribution constraints are summarized in Table 3.9. Surprisingly, the All-integer model is more tractable. Although we have not conducted a complete analysis of this behavior, we observed that more cuts are generated by CPLEX for the All-integer model, which helps in improving the dual bound. More precisely, under Constraints (TD1), three of the five instances of the All-integer model are solved to optimality within 24 hours. For the All-continuous and Partially-integer models under Constraints (TD1), none of the instances has been solved to optimality within the time limit. However, the MIP gaps are relatively small. As previously concluded from Table 3.8, we again observe that Constraints (TD1) provide the lowest lost demand, and this is consistent for all types of variable domains. In terms of the performance for different variable domains, even though it is fast to solve, the All-integer model provides the worst performance, which indicates that restricting station variables to integer values may result in too conservative trips. The Partially-integer model introduces an improvement and performs best in most of the cases, especially with constraints (TD3)+(TD4) and (None). The results for O-D variables can be found in Appendix A.6.4.

3.5.6 Impact of Event Sequences

Recall that using no particular event sequence, which equals (r+a+d), within the All-continuous station-based model without trip distribution constraints yields an average rental loss of 8.78% on the test set of the 60-station network instances (see Table 3.8). Based on the model (3.1)–(3.9) with continuous variables and the event sequences reviewed in Section 3.3.2, we now analyze the impact of such event sequences for problem instances on the 60-station network in Table 3.10.

Table 3.10 Station-based model with different sequences of events (60 stations, 4 trucks, 30 mins)

	Sequences	O.F.	Time	MIP	Lost Demand(%)		Opt-sim-gap(%)	
	of events	Value	(mins)	Gap (%)	Rental	Return	Rental	Return
GT1	(r)(a)(d)	1.0	288	0.00	7.91	5.22	8.62	1.39
	(a)(d)(r)	3.3	298	0.00	7.64	4.63	8.19	11.83
	(d)(r)(a)	13.4	1399	0.04	6.02	5.81	5.38	9.89
	(r)(d)(a)	8.2	1013	0.03	5.61	3.26	5.36	6.53
	(d)(a)(r)	15.1	879	0.01	6.40	3.48	5.70	7.65
	(a)(r)(d)	2.0	<1	0.00	7.51	2.79	8.12	8.04
	(r)(a+d)	0.8	<1	0.00	8.49	6.86	9.28	14.12
	(a+d)(r)	1.6	<1	0.00	7.42	6.92	7.98	12.87
GT2	(r)(a)(d)	3.2	288	0.00	8.05	3.61	8.71	9.35
	(a)(d)(r)	1.9	864	0.49	8.33	5.11	8.98	11.55
	(d)(r)(a)	17.5	1440	0.16	6.75	5.51	5.88	10.14
	(r)(d)(a)	11.9	1440	0.08	6.39	3.79	5.93	7.77
	(d)(a)(r)	20.0	1341	0.13	6.38	3.77	5.26	7.74
	(a)(r)(d)	3.8	<1	0.00	7.87	1.93	8.50	7.25
	(r)(a+d)	0.7	<1	0.00	8.40	3.29	9.15	9.63
	(a+d)(r)	1.9	576	0.03	7.98	3.66	8.53	9.56

Sequences (d)(r)(a), (r)(d)(a), and (d)(a)(r) perform best for lost rental and return with a small opt-sim-gap. Although GT3 instances are hard to solve, these three sequences still perform well (see Table A.17 in Appendix A). In contrast, the sequences used in the literature (i.e., (r+a+d), (a)(r)(d), and (a+d)(r)) have performed less well in our experiments. Instead, sequences (d)(r)(a), (r)(d)(a), and (d)(a)(r) may be a better choice, reducing lost rental by an additional 1%-2%.

Table 3.11 shows the results of the same experiments for the problem instances on a 30-station network and 60-minute time-periods. Here, all instances have been solved to optimality and lead to the same conclusions. Particular sequences help reduce the lost rental: sequences (d)(r)(a) and (r)(d)(a) reduce the lost rental to around 7.45% from 9.12% in the test set.

Table 3.11 Station-based model with different sequences of events (30 stations, 2 trucks, 60 mins)

	Sequences	O.F.	Time	MIP	Lost Demand (%)		Opt-sim-gap (%)	
	of events	Value	(mins)	Gap (%)	Rental	Return	Rental	Return
GT1	(r)(a)(d)	10.4	<1	0.00	8.95	2.09	8.83	7.94
	(a)(d)(r)	15.0	<1	0.00	9.38	1.37	8.47	7.63
	(d)(r)(a)	37.7	<1	0.00	7.43	1.23	1.22	6.01
	(r)(d)(a)	33.1	<1	0.00	7.45	1.10	1.93	6.10
	(d)(a)(r)	42.1	<1	0.00	7.89	1.81	1.04	7.00
	(a)(r)(d)	11.3	<1	0.00	8.62	1.92	8.29	7.34
	(r)(a+d)	5.3	<1	0.00	9.81	2.08	9.89	9.94
	(a+d)(r)	9.8	<1	0.00	9.53	2.48	8.72	10.05
	(r+a+d)	5.3	<1	0.00	9.12	2.55	9.07	9.76

In Section 3.5.4, we have concluded that it is beneficial to use trip distribution constraints (TD1) when no particular event sequence is used. The results discussed above suggest that it is beneficial to use a specific event sequence, such as (r)(d)(a) when no trip distribution constraints are used.

Table 3.12 Station-based model with (TD1) and sequences of events (30 stations, 2 trucks, 60 mins, GT1)

	Sequences	O.F.	Time	MIP	Lost Demand (%)		Opt-sim-gap (%)	
	of events	Value	(mins)	Gap (%)	Rental	Return	Rental	Return
All-continuous	(r)(d)(a)	499.4	22.9	0.00	6.62	4.09	14.36	65.34
	(d)(a)(r)	515.1	7	0.00	6.78	5.66	16.50	65.35
	(a+d)(r)	503.4	6	0.00	7.10	5.14	14.33	65.12
	(r+a+d)	485.5	237.16	0.00	6.32	3.87	12.37	65.22
Partially-integer	(r)(d)(a)	500.1	18	0.00	6.72	3.22	14.41	65.62
	(d)(a)(r)	515.6	5	0.00	6.73	6.32	16.69	65.08
	(a+d)(r)	504.1	7	0.00	6.97	6.59	14.57	64.65
	(r+a+d)	485.9	440.67	0.00	6.64	3.74	12.15	65.13

We now explore the combination of those two modeling assumptions and further backtest on different variable domains. To this end, Table 3.12 summarizes the results for GT1 problem instances on the 30-station network using trip distribution constraints (TD1) and various event sequences for All-continuous and Partially-integer variable domains. The models for both variable domains seem to perform similarly well in terms of lost rentals, while the All-continuous models tend to be solved faster.

When comparing with the results in Table 3.11, the introduction of Constraints (TD1) further decreases the lost rental for any of the event sequences. However, the improvement for

(r+a+d) is the highest, indicating that using Constraints (TD1) without any specific event sequence may be the best option if the longer computing time is acceptable. Operators may therefore opt for event sequence (r)(d)(a) with all-continuous variables without trip distribution constraints if a quick solution is required, or Constraints (TD1) without a specific event sequence if higher computing times can be tolerated.

3.5.7 Case Study on BIXI Montreal Data

We now validate our previous findings by means of a case study based on real-world data from BIXI Montreal [1].

Data preprocessing. We focus on the 2019 season. To ensure a coherent association of historical arrivals and rentals to the given station IDs, we first discarded stations that changed locations throughout the 2019 seasons by more than 1 km (which is common given that the operator relocates stations due to events, construction, or holidays). The resulting network contained 606 stations, which, obviously, would result in optimization models that are too large to solve directly. Given that vehicle relocation is time-consuming, efficient rebalancing solutions typically relocate locally rather than over large distances. It is therefore reasonable to assume that the network can be subdivided into sub-clusters (see, e.g., [16, 37, 43, 53, 56, 68], etc.).

To this end, we first cluster stations via k-means based on demand pattern similarity (see, e.g., [16, 43, 56]), ensuring the inclusion of city center stations and work-related trips. Similar to [37], we also limit the maximal distance between stations that belong to the cluster, which is motivated by the fact that the vehicles only travel within one cluster. We selected a cluster with 53 stations distributed around the downtown and Plateau areas (see Figure 3.4), which has approximately the same number of total rentals and returns [85] (a requirement, which is obviously satisfied in closed BSSs systems).

We focus on days with high demand, i.e., weekdays within the summer season of 2019, which have therefore not been affected by the COVID-19 pandemic. Outlier days with extremely low numbers of trips (e.g., due to bad weather or special events) are excluded, resulting in a total of 50 days. Aligned with the experiments on synthetic problem instances, we here consider a planning horizon from 7 a.m. to 2 p.m. (7 hours) with 14 time-periods, each of which spans 30 mins.

Empirical results. With the objective of validating the most relevant conclusions drawn from the experiments on synthetic problem instances, we here use the station-based model. As opposed to O-D variables, the use of station-based rental and return variables $x_s^{+,t}$ and

Figure 3.4 Considered station cluster from BIXI Montreal (generated through Google Maps[2])

$x_s^{-,t}$ also enables us to aim at capturing trips to and from all stations in the original network, not only those included in the considered cluster. We further focus on the impact of variable domains, the event sequences, and the best-performing trip distribution constraints (TD1). Note that the use of time constraints is not necessary, given that all stations within the clusters are located sufficiently close to each other.

Trip constraints (TD1) consider the proportion of successful rentals that depart from any station to a specific station. Given that our model only uses stations within the considered cluster $C \subset S$, this proportion cannot be computed. We may rewrite the right-hand side of (TD1) as $\sum_{t=0}^{t_2-1} \sum_{s \in C} x_s^{+,t} \frac{F_{s,s_2}^{t,t_2}}{f_s^{+,t}} + \sum_{t=0}^{t_2-1} \sum_{s \in S \setminus C} x_s^{+,t} \frac{F_{s,s_2}^{t,t_2}}{f_s^{+,t}}$, where the first term accounts for successful trips from within the cluster, and the second term refers to rentals from stations outside the considered cluster. We replace $x_s^{+,t} (s \in S \setminus C)$ by $f_s^{+,t} (s \in S \setminus C)$ within the second term, therefore optimistically assuming that rentals outside the cluster are all satisfied. The adapted trip constraint then writes as follows:

$$x_{s_2}^{-,t_2} \leq \sum_{t=0}^{t_2-1} \sum_{s \in C} x_s^{+,t} \frac{F_{s,s_2}^{t,t_2}}{f_s^{+,t}} + \sum_{t=0}^{t_2-1} \sum_{s \in S \setminus C} F_{s,s_2}^{t,t_2} \qquad \forall\, s_2 \in C,\ t_2 \in T.$$

Table 3.13 summarizes the results of the corresponding experiments. Here, Baseline 2 refers to the model that only carries out (overnight) static rebalancing (see Section 3.5.2). In contrast to the results on synthetic problem instances, the improvement through dynamic rebalancing (as opposed to Baseline 2 only) is not impressive when no trip distribution constraints are used. The small improvement may be explained by the fact that the here-considered real-world data only contains successful trips. All other observations are well aligned with the results for synthetic instances. Using partially-integer variable domains without trip distribution constraints and without a specific event sequence reduces the es-

Table 3.13 Station-based model (53 stations, 4 trucks, 30 mins, 50 days high demand, Baseline 2)

	Sequences of events	(TD1)	O.F. Value	Time (mins)	MIP Gap (%)	Lost Demand (%)		Opt-sim-gap (%)	
						Rental	Return	Rental	Return
Baseline 2	-	-	-	-	-	8.32	6.12	-	-
All-continuous	(r)(d)(a)		0.0	<1	0.00	7.43	6.15	20.95	18.04
	(d)(a)(r)		0.0	<1	0.00	7.19	6.13	20.93	18.07
	(a+d)(r)		0.0	<1	0.00	8.32	6.11	20.75	18.05
	(r+a+d)		0.0	<1	0.00	9.42	7.65	20.82	18.30
Partially-integer	(r)(d)(a)		0.0	<1	0.00	7.54	7.22	20.58	18.55
	(d)(a)(r)		0.0	<1	0.00	7.36	6.14	20.92	18.05
	(a+d)(r)		0.0	<1	0.00	8.32	6.12	20.75	18.05
	(r+a+d)		0.0	<1	0.00	8.32	6.12	20.75	18.05
All-continuous	(r)(d)(a)	✓	453.4	<1	0.00	5.70	6.67	20.98	25.50
	(d)(a)(r)	✓	453.4	<1	0.00	3.88	7.21	20.83	25.29
	(a+d)(r)	✓	453.4	<1	0.00	5.09	4.99	20.66	25.36
	(r+a+d)	✓	453.4	<1	0.00	6.31	6.23	20.84	25.29
Partially-integer	(r)(d)(a)	✓	453.5	<1	0.00	4.53	6.96	20.91	25.27
	(d)(a)(r)	✓	453.5	<1	0.00	2.82	6.59	20.60	25.42
	(a+d)(r)	✓	453.4	<1	0.00	2.89	6.34	20.61	25.25
	(r+a+d)	✓	453.4	<1	0.00	5.79	6.07	20.70	25.36

timated lost demand (compare Table 3.9). Further, the use of adapted trip distribution constraints (TD1) positively impacts lost demand (compare Table 3.9, and Tables 3.11 and 3.12). Interestingly, the combination of using partially-integer domains with trip constraints seems particularly effective on real-world data, providing the lowest lost demand among all configurations. Finally, event sequences (r)(d)(a) and (d)(a)(r), which have been found to be among the best-performing sequences on synthetic instances, here also perform well with both all-continuous variables and partially integer variables in lost rental and return without trip distribution constraints. Finally, we note that all models have been solved to optimality within 1 minute, which is, of course, sufficiently fast for use in practice.

3.6 Conclusions

In this paper, we aimed to disentangle and structure the various modeling assumptions and constraints used in the literature on Mixed-Integer Programming models for BSSs rebalancing optimization. To this end, we first surveyed the literature according to modeling techniques and assumptions, with a particular focus on multi-period models. We then introduced a modeling framework, rooted in a basic model, and showed how to adapt it to the various modeling assumptions. Finally, we evaluated the performance of the planned

solutions induced by the different model variants as realistically as possible. Specifically, we generated different ground truths that propose BSSs networks and trip patterns matching observed trip patterns at BIXI Montreal. We then developed a fine-grained discrete-event simulator for truck movement, rebalancing operations, as well as bike rentals and returns on a minute-to-minute basis.

3.6.1 Summary Recommendation

Based on the simulation results on a large set of test instances, we focused on two performance measures to analyze the appropriateness of the various modeling assumptions: the lost rental and return demand observed throughout the simulation, and the simulation-optimization-gap that indicates the deviation between the lost demand observed as estimated by the optimization model and by the simulator. Extensive numerical experiments on problem instances with networks including 30 and 60 stations and three different ground truths were carried out. Experiments have also been carried out on real-world data with 53 stations from the BIXI Montreal 2019 summer season. While one is required to be more careful when drawing conclusions from such results, given that the observed trip data refers to trips satisfied in the past, the corresponding conclusions tend to align. Based on these results, our principal conclusions can be summarized as follows:

(i) On the synthetic data sets, adding dynamic rebalancing to static rebalancing reduces the lost rental demand by an additional 2% (e.g., from 10.43% to 8.79% in Table 3.6).

(ii) Using station-based trip variables instead of more detailed trip variables based on origin-destination pairs generally appears to be competitive and results in faster solution times.

(iii) Shorter time-periods tend to allow for planning more rebalancing operations but may require time constraints to ensure that the resulting rebalancing is time-feasible in practice.

(iv) Trip distribution constraints, especially (TD1), reflect more realistically the trip flow observed in practice and may strongly improve the lost demand on both synthetic and real-world data (e.g., from 8.78% to 3.65% in Table 3.8; from 9.42% to 6.31% in Table 3.13); further, the best performing trip distribution constraint(s) are not necessarily those used in the literature.

(v) Using integer variables exclusively for truck routes, while keeping all other variables continuous (even the pick-up and drop-off decisions in the rebalancing operations) generally approximates reality sufficiently well; in some specific cases, it is beneficial to impose integrality on the rebalancing variables, while it does not seem beneficial to use integer variables for all decisions that, in reality, would also be integer.

(vi) Exploring the various sequences in which bike rentals, bike returns, and rebalancing operations may occur yields interesting conclusions. In particular, event sequences that have not been studied in the literature perform particularly well and tend to reduce lost rental by an additional 2% - 3% (Tables 3.10 and 3.11). Coupling specific event sequences, in particular (d)(a)(r), with our proposed trip distribution constraints (TD1) provides consistently low lost demand in short computing times on both synthetic and real-world instances.

The optimization model for the real-world problem instance has been solved within 1 minute of computing time, while the solution time for synthetic instances may vary strongly with the model configuration. When computing resources are limited and quick decisions are required, using a combination of trip distribution constraints with one of the newly proposed event sequences, in particular the above-mentioned combination of (TD1) with sequence (d)(a)(r), seems to be recommended (compare Table 3.12).

With unrestricted resources for computing time and memory, we would recommend applying trip distribution constraints with a short time-period for both synthetic data and real-world data. When computing resources are limited and quick decisions are required, we recommend introducing event sequences (r)(d)(a) or (d)(a)(r). Within the experiments on real-world data, partially-integer variable domains, along with constraints (TD1) and event sequence (d)(a)(r) performed particularly well.

Note that we have used a time limit of 24h in order to be capable of solving the models to optimality, allowing us to draw conclusions on their degree of realism and potential performance in practice. Solving models multiple times throughout the day (albeit over a smaller time horizon) in sufficiently short computing times may require the use of parallel computing, specialized solution methods (e.g., mathematical decomposition), or a combination of both.

3.6.2 Future Work

Having explored the different modeling assumptions and techniques both from methodological and empirical standpoints, this work aimed at shedding light on the modeling jungle and guiding both practitioners and academics in future research on multi-period rebalancing optimization.

While concepts such as the sequences of events cannot be found in most of the related classical optimization problems, such as the Pickup-and-Delivery Problem, such characteristics are not exclusive to BSSs. Indeed, any planning problem in which the interaction between customers and system operator is aggregated in discretized time-periods (e.g., car-sharing, multi-mode transportation planning with synchronization) may exhibit similar event sequences, which

may be worth studying.

Certain future research directions may be particularly worthwhile. First, while we have identified the formulations that are likely to provide well-performing planning solutions, solving those models in real-time throughout the day may be challenging; therefore, decomposition algorithms may be employed to speed up the solution time. Second, the proposed models minimize total lost demand. Certain BSS operators consider target intervals, which may be interesting to consider within the objective function.

Finally, while we have intentionally focused on the deterministic planning problem which assumes as input a single set of customer demands, some of our conclusions may also hold for models that consider several demand scenarios simultaneously. Models explicitly considering the underlying uncertainty and demand probability distribution are worthwhile research directions, in particular for the model variants that have shown to be more realistic here.

Acknowledgments

The authors are thankful to BIXI Montreal for providing real-world trip data and for their support throughout several discussions. We also thank Compute Canada for providing the computational resources to carry out the numerical experiments. The work of the second author was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada under grant 2017-05224. We are indebted to three anonymous referees whose detailed comments helped us to significantly improve the paper.

CHAPTER 4 ARTICLE 2: DYNAMIC REBALANCING FOR BIKE-SHARING SYSTEMS UNDER INVENTORY INTERVAL AND TARGET PREDICTIONS

Authors: Jiaqi Liang, Maria Clara Martins Silva, Daniel Aloise, and Sanjay Dominik Jena.
Submitted to Transportation Research Part C on November 14, 2023.[1]

Abstract Bike-sharing systems have become a popular transportation alternative. Unfortunately, station networks are often unbalanced, with some stations being empty, while others being congested. Given the complexity of the underlying planning problems to rebalance station inventories via trucks, many mathematical optimization models have been proposed, mostly focusing on minimizing the unmet demand. This work explores the benefits of two alternative objectives, which minimize the deviation from an inventory interval and a target inventory, respectively. While the concepts of inventory intervals and targets better fit the planning practices of many system operators, they also naturally introduce a buffer into the station inventory, therefore better responding to stochastic demand fluctuations. We report on extensive computational experiments, evaluating the entire pipeline required for an automatized and data-driven rebalancing process: the use of synthetic and real-world data that relies on varying weather conditions, the prediction of demand and the computation of inventory intervals and targets, different reoptimization modes throughout the planning horizon, and an evaluation within a fine-grained simulator. Results allow for unanimous conclusions, indicating that the proposed approaches reduce unmet demand by up to 34% over classical models.

Keywords: bike-sharing system, dynamic rebalancing, inventory intervals, target inventories, reoptimization modes, mixed-integer programming

4.1 Introduction

The pursuit of environmentally friendly transportation modes has increased considerably in the last few years, with Bike-sharing systems (BSSs) having emerged as a notable choice. As of 2022, there were over 1,900 BSSs in operation comprising almost 9 million bikes [76], offering cities opportunities to reduce carbon emissions, traffic relief, and improve the quality of life for their residents [22].

[1] A pre-print is available in [66].

A particular challenge in the management of BSSs are station imbalances. Most users follow consistent travel patterns, often commuting toward commercial areas (such as city centers) during morning peak hours and returning to residential zones after work. Such behavior often results in stations being full or empty, leading to *lost demand*, i.e., rental demand that cannot be met due to an empty station or return demand that cannot be met as a result of a full station. Occasional user trips introduce stochasticity, further aggravating station imbalances. As a remedy, BSSs operators often redistribute bikes among the stations, typically, via trucks, a process known as *rebalancing*. Rebalancing imbalanced stations has been proven to be more cost-effective than alternative solutions, such as adding more stations or installing additional docks [91]. The development of effective rebalancing strategies has therefore become a crucial research field with the potential to significantly improve user satisfaction. Two primary rebalancing schemes have been acknowledged in the literature: overnight station rebalancing and intraday rebalancing. The latter is often referred to as the Dynamic Bicycle Repositioning Problem (DBRP) [77, 84]. In contrast to overnight rebalancing, the DBRP involves continuous intraday rebalancing operations in parallel to the user trips occuring throughout the day. Given its higher impact on demand satisfaction, we here focus on this problem.

While a few recent works use Markov Decision Process (MDP) to address the problem variants [13, 61, 62, 64, 89], the majority of the literature applies Mixed-integer Programming (MIP) models [46, 74, 111, 114], given that they are flexible and widely used within industrial decision-making processes. Among MIP models, multi-period models benefit from an integrated planning over all time-periods and do therefore not suffer from the myopic behavior of single-period models.

Most models aim at minimizing lost demand or maximizing successful trips [46, 69, 92, 108, 111, 112], with both rental and return demands estimated either by naive predictions (e.g., the historical mean) [43, 45, 69, 111] or more sophisticated Machine Learning (ML) techniques [85, 112]. However, minimizing lost demand may result in sub-optimal rebalancing solutions if the predicted rental and return are not accurate enough. BSSs operators, like BIXI Montreal, often use *inventory intervals* and *target inventories* within their rebalancing planning. The inventory interval of a station refers to the acceptable range of its inventory, ensuring that the station maintains a level of available bikes and free docks. The target inventory of a station represents the desired number of bikes at that station to ensure optimal service.

While many methods have been proposed to compute inventory intervals and target inventories (see e.g., [52, 54, 68, 83]), only a few have incorporated them into optimization models. Notably, most relevant studies [47, 59, 88] either focus on single-period models or minimize

the deviation of target inventory at the end of the planning process only. Although [97] and [99] attempt to introduce intervals into multi-period models, they relax the intervals by using station capacities directly in their experiments. Furthermore, the intervals and targets in these models are often determined without considering weather conditions, which significantly impact user behavior. As a result, the benefits of combining optimization algorithms with intervals or targets in the objective function for multi-period models are still to be determined.

In a similar vein, while some of the existing models have been carried out in a rolling fashion (see, e.g., [46, 47, 74, 85, 92]), literature has not yet quantified the benefits of integrating system status update through reoptimization (i.e., rolling and folding planning) over classical static planning of multi-period rebalancing models. In this paper, we aim at filling these gaps and approach the DBRP by incorporating inventory intervals and target inventories.

Contributions. Our work evaluates the entire pipeline required for an automatized and data-driven rebalancing process in BSSs. The main contributions can be summarized as follows. (i) We propose two optimization models that integrate inventory intervals and target inventories into the objective functions, concepts that are often already used within the decision-making process of BSS operators. In contrast to classical models that minimize unmet demand, the proposed models tend to ensure a buffer in the station inventories and are therefore more capable of dealing with demand fluctuations. (ii) We propose a realistic instance generator, generating varying weather conditions for different days along with trip data that is historically coherent with such conditions. (iii) We conduct an extensive comparison among three multi-period models with different objective functions for DBRP, including the classical objective that minimizes unmet demand and the two proposed objectives that minimize the deviations from inventory intervals and target inventories. Demand predictions are obtained from an advanced machine learning model, capable of making sufficiently accurate predictions based on weather and temporal features. Inventory and target inventories are computed such that they maximize the desired service-level. The performance is estimated by a fine-grained discrete-event simulator. Our models demonstrate a remarkable robustness to cope with trip fluctuations, reducing lost demand by up to 34% as compared to the model minimizing lost demand. (iv) We empirically compare the impact of employing different reoptimization modes (i.e., static and rolling planning) for all models. The results indicate a clear advantage of reoptimizing over the planning horizon, reducing the lost demand by at least 30% on average, without necessarily increasing the computing time. (iv) We compare the impact of using perfect information and less accurate demand predictions on the performance of the planning models. Interestingly, our proposed optimization models

remain remarkably robust. (v) A case-study on real-world data is considered, confirming the benefits of the proposed approaches.

Outline. This paper is organized as follows. Section 4.2 reviews relevant literature for BSSs in objective functions in rebalancing models, inventory intervals and target inventories, trip prediction, and reoptimization modes. Section 4.3 reviews the baseline model that minimizes unmet demand and introduces two dynamic rebalancing models minimizing the deviations from inventory intervals and target inventories. Numerical experiments and analysis on synthetic and real-world data are presented in Section 4.4. This is followed by the conclusions in Section 4.5.

4.2 Literature Review for Rebalancing Problems in BSSs

This section reviews the literature related to the here considered planning problem and our contributions, focusing on the objective functions used within DBRP models, inventory intervals and targets, demand prediction, and reoptimization modes.

4.2.1 Objective Functions in Rebalancing Models

We first review existing MIP models and their objective functions used in dynamic rebalancing. Metrics used in the objective functions can be classified into three different types (see Appendix of [65]): distance-based metrics, loading-based metrics, and demand-based metrics. Distance-based metrics are associated with the travelling distance of vehicles, mainly including travelling costs, travelling time, and fuel consumption (see e.g., [3, 45, 82, 114]). Loading-based metrics are associated with the number of handling (i.e., loading and unloading) operations (see e.g., [50, 95]). Handling cost or time reflects the workload of operations. Finally, demand-based metrics concern the dissatisfaction of customers. Some studies consider more than one aspect in their objective functions (see, e.g., [43, 50, 59, 74, 112]).

We here focus on demand-based metrics, which have been more relevant in the literature, and to which our contributions are directly related. The most common approach for demand-based metrics is to minimize the lost rental and return demand, which is also equivalent to maximizing successful trips (see e.g., [23, 45, 46, 50, 69, 92, 108, 111, 112, 114]). Concurrently, there have been a few attempts to minimize the deviation between the inventories of the stations and their target inventories [47, 59] or inventory intervals [99], which still holds considerable potential for further exploration. Our work focuses on inventory intervals and target inventories, for which the related literature is reviewed next.

4.2.2 Inventory Intervals and Target Inventories

As opposed to minimizing lost demand, the concepts of inventory intervals and target inventories have been found to be useful within the rebalancing decision-making process. Inventory intervals define an acceptable range of the bike inventory at individual stations, whereas target inventories represent specific inventory levels that operators aim to uphold at each station. Both are typically designed to ensure a high rate of demand satisfaction.

Even though inventory intervals and target inventories are often used concepts in the planning processes of BSS operators, only a few works have incorporated them into optimization models. When using target inventories, objective functions typically minimize the deviations between the station inventories and the specified targets. Here, [47] consider a single-period model, executed in a rolling planning, that minimizes such inventory deviations. Further, [59] propose a multi-period rebalancing model for which they compare various metaheuristics. Next to multiple criteria (such as minimizing costs) considered within the objective function, the model also aims at aligning the final station inventory at the end of the planning horizon with a specific target inventory. Finally, the model introduced by [17] also uses target values, but instead of minimizing deviations, they use hard constraint to ensure that station inventories equal those targets at the end of the rebalancing process. As a result, the minimization of the deviations between station inventories and target values at each of the time-periods, as a mean of improving demand satisfaction, has not yet been considered and deserves further investigation.

Inventory intervals have been implemented as hard constraints [88, 97] into rebalancing models, requiring that station inventories remain within a specific range. Here, [88] focus on the static rebalancing problem with a single time-period, while [97] propose a multi-period model. However, the authors do not consider these constraints in their computational experiments. Given that hard constraints may easily lead to infeasible outcomes, [99] present a multi-period model that, among several other criteria, minimizes the deviation of station inventories from predefined inventory intervals within the objective function. Unfortunately, their experimental setup also ignores such intervals, leaving the actual effectiveness of inventory intervals within the optimization model unexplored.

While the works cited above assume that target inventories and inventory intervals are given, a few works also propose how to effectively compute them. Target inventories have often been computed such that they reduce the probability of a station reaching both the full and empty status, typically requiring the prior estimation of rental and return distribution on historical data [47, 52, 83]. Inventory intervals have been computed in a similar fashion [54, 88], selecting those that minimize the likelihood of a station becoming either empty or

full. Finally, different to those approaches, [24] and [55] simulate the performance of several sets of initial inventories and select the one that performs best.

All works cited above compute target inventories and inventory intervals based on historical trip demand. However, they disregard external factors such as weather conditions, the importance of which has been widely acknowledged in the literature (see, e.g., [34, 42, 54, 57, 68, 76]). To this end, [54] extend the notion of service-levels proposed by [88], computing both target inventories and inventory intervals based on the service level and the predicted demand. The latter is estimated based on machine learning models trained on both temporal and weather data, therefore holding the potential of providing better performing target inventories and inventory intervals. The authors also introduce two additional hyperparameters, α and β, allowing operators to align inventory intervals and target inventories with their priorities for either rentals or returns (see Appendix B.1 for more details), hence making it an attractive approach to operators.

4.2.3 Demand Prediction for BSSs

Demand prediction is an essential step in the rebalancing process, enabling operators to anticipate which stations require higher inventories to better serve trip demand. Accurately predicting rentals and returns is challenging, as it is influenced by numerous factors [98]. Literature on demand prediction in BSSs can be divided into approaches predicting at global demand level (see e.g., [30, 42, 105]), at the level of station-clusters (see e.g., [10, 36, 98]), and at the level of individual stations (see e.g., [11, 31, 54, 79, 85]). We here focus on works predicting demand at station level, which is required for rebalancing operations since they are tailored considering the rentals and returns for each station.

Different techniques have been used to predict demand in BSSs. The average of historical trips (i.e., rentals and returns) can be used to estimate future demand [5, 24, 43, 47, 83, 88, 111], which can be seen as a naive predictor. Alternatively, rental and return (often Poisson) distributions have been estimated from historical trip data, from which trip demand is then sampled and used within the optimization models (see, e.g., [45, 69, 111, 114]). While such prediction methods centered on historical mean demand have been quite popular, several studies (see, e.g., [39, 68, 70] haven suggested that such methods may result in high prediction errors when compared to ML models, as the latter can take into considerations features beyond historical trip data, including weather conditions, time of day, day of the week and the occurrence of special events.

Generally, ML algorithms can capture intricate patterns and correlations and may, therefore, result in significantly more accurate demand predictions compared to the rudimentary his-

torical averaging approach. Among works using ML algorithms in the context of BSSs, many concluded that gradient boosted trees result in particularly accurate predictions (see e.g., [11, 54, 100]). Random forests and gradient boosted trees, in particular, have been found to provide competitive prediction accuracy in the context of rental and return predictions [70, 105]. Typically, such models integrate weather and temporal features, highlighting their importance to accurately predict demand. Here, [54] utilize Singular Value Decomposition, a dimension reduction technique, to reduce the dimensionality of the trip data, eliminate noise and improve both time required and the accuracy of hourly station-level rental and return predictions (details can be found in Appendix B.1). These reasons make this model an attractive option to estimate the future trip demand for our rebalancing optimization models.

4.2.4 Reoptimization Modes: Static, Rolling, and Folding

In practice, both single-period and multi-period models can be implemented in several ways. A simple, yet common approach (see, e.g., [43, 59, 71, 86, 91, 112]) is to optimize once over the entire planning horizon (i.e., all considered time-periods) and then implement all rebalancing decisions (i.e., the number of bikes dropped off and picked up at each station) as planned for all time periods.

While multi-period models can represent the consequences of decisions made at early time-periods, when executed within a static planning, they do not benefit from updated system information (such as station inventories or improved demand predictions). Therefore, practitioners often tend to reoptimize the rebalancing decisions throughout the planning horizon. Specifically, rolling planning (also called rolling window planning) considers the reoptimization over several time-periods at predefined reoptimization stages. Figure 4.1 (A) depicts the rolling planning, where, at each reoptimization stage, the green squares indicate time periods

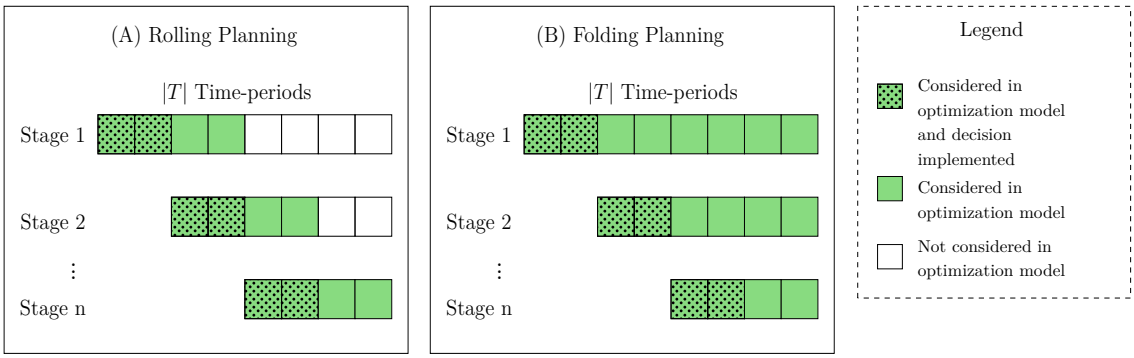

Figure 4.1 The structure of rolling and folding planing

considered in the optimization models, and the green dotted squares indicate decisions of the time-periods actually executed in practice. This approach has been popular, with works implementing rolling planning for both multi-period (see, e.g., [46, 69, 74, 85, 92, 111]) and single-period models (see e.g., [44, 47, 50]). This allows for correcting ineffective planning, e.g., due to forecasting inaccuracies. In addition, considering only a subset of all time-periods within the planning window has the advantage of resulting in a more tractable optimization model.

If model decisions at early time-periods may impact decisions several time-periods ahead, one may want to consider a model with a longer planning horizon. Folding planning therefore optimizes on the remaining planning horizon, as depicted in Figure 4.1 (B). While this is generally a common approach in multi-period models [67], to the best of our knowledge, this approach has not yet been explored for BSSs rebalancing planning.

4.3 Dynamic Rebalancing Models

In this section, we formulate the DBRP as MIP models. Section 4.3.1 first describes a multi-period model, which minimizes the unmet rental and return demand, the more popular objective in the literature (as discussed in Section 4.2.1). We then propose two models integrating inventory intervals and target inventories into the objective functions in Section 4.3.2.

4.3.1 Multi-period Rebalancing Model Minimizing Lost Demand

We consider the model with multiple time-periods from [65]. The input parameters are listed in Table 4.1 and decision variables are shown in Table 4.2. We denote S as the set of stations, while V denotes the set of available vehicles. Each station $s \in S$ has a capacity of C_s docks and each vehicle $v \in V$ can hold at most $\hat{C}_v$ bikes. We consider a planning horizon with $|T|$ time-periods, where each time period $t \in T$ represents a duration of L_t minutes.

We assume that a vehicle can visit only one station per time-period. As a result, a vehicle can visit at most T stations during the entire planning horizon. The decision variables $r^{+,t}_{s,v}$ and $r^{-,t}_{s,v}$ represent the number of bikes vehicle v picks up and drops off, respectively, at station s during period t. Furthermore, binary variable $z^t_{s,v}$ takes value 1 if and only if vehicle v visits station s at time-period t. For each time-period, intermediate variables are used: the number of bikes available at stations and in vehicles, successful trips, and vehicle routes. The resulting MIP model is expressed as follows:

Table 4.1 Input parameters of the optimization model

Input Parameters	Definition
S	The set of stations.
V	The set of vehicles.
T	The set of discreted time-periods.
C_s	The capacity of station $s \in S$.
$\hat{C}_v$	The capacity of vehicle $v \in V$.
L_t	The duration (in minutes) of time period $t \in T$.
d_s^1	The initial number of bikes at the station $s \in S$.
$\hat{d}_v^1$	The initial number of bikes in vehicle $v \in V$.
$z_{s,v}^1$	The initial location of each vehicle $v \in V, s \in S$.
$f_s^{+,t}$	The expected rental demand at station $s \in S$ in period $t \in T$.
$f_s^{-,t}$	The expected return demand at station $s \in S$ in period $t \in T$.

Table 4.2 Decision variables of the optimization model

Variables	Definition
d_s^t	The number of bikes available at station $s \in S$ at the beginning of period $t \in T$.
$\hat{d}_v^t$	The number of bikes in vehicle $v \in V$ at the beginning of period $t \in T$.
$x_s^{+,t}$	The number of successful rentals starting from station $s \in S$ in period $t \in T$.
$x_s^{-,t}$	The number of successful returns ending at station $s \in S$ in period $t \in T$.
$r_{s,v}^{+,t}$	The number of bikes picked up at station $s \in S$ by vehicle $v \in V$ in period $t \in T$.
$r_{s,v}^{-,t}$	The number of bikes dropped off at station $s \in S$ by vehicle $v \in V$ in period $t \in T$.
$z_{s,v}^t$	$z_{s,v}^t = 1$, if vehicle $v \in V$ visits station $s \in S$ in period $t \in T$; 0 otherwise.

$$\min \quad \sum_{s\in S}\sum_{t\in T}(f_s^{+,t} - x_s^{+,t}) + \sum_{s\in S}\sum_{t\in T}(f_s^{-,t} - x_s^{-,t}) \tag{4.1}$$

$$\text{s.t.} \quad \hat{d}_v^{t+1} = \hat{d}_v^t + \sum_{s\in S}(r_{s,v}^{+,t} - r_{s,v}^{-,t}) \qquad \forall\ v \in V,\ t \in T \tag{4.2}$$

$$d_s^{t+1} = d_s^t - \sum_{v\in V}(r_{s,v}^{+,t} - r_{s,v}^{-,t}) - x_s^{+,t} + x_s^{-,t} \qquad \forall\ s \in S,\ t \in T \tag{4.3}$$

$$\sum_{s\in S} z_{s,v}^t = 1 \qquad \forall\ v \in V,\ t \in T \tag{4.4}$$

$$r_{s,v}^{+,t} + r_{s,v}^{-,t} \le \hat{C}_v z_{s,v}^t \qquad \forall\ s \in S,\ v \in V,\ t \in T \tag{4.5}$$

$$0 \le \hat{d}_v^t \le \hat{C}_v \qquad \forall\ v \in V \tag{4.6}$$

$$0 \le d_s^t \le C_s \qquad \forall\ s \in S \tag{4.7}$$

$$0 \le x_s^{+,t} \le f_s^{+,t}, 0 \le x_s^{-,t} \le f_s^{-,t} \qquad \forall\ s \in S,\ t \in T \tag{4.8}$$

$$0 \le r_{s,v}^{+,t}, r_{s,v}^{-,t} \le \hat{C}_v \qquad \forall\ s \in S,\ v \in V,\ t \in T \tag{4.9}$$

$$z_{s,v}^t \in \{0,1\} \qquad \forall\ s \in S,\ v \in V,\ t \in T. \tag{4.10}$$

The objective function (4.1) minimizes the total lost demand, i.e., the unmet expected demand for both rentals and returns, over the entire planning horizon at all stations. Con-

straints (4.2) compute the number of bikes in each vehicle v in period $t+1$ based on the number of bikes in the previous period and the number of picked up/ dropped off bikes. Constraints (4.3) compute the number of bikes in period $t+1$ at each station s as the sum of the number of bikes of that station in the previous period, the number of bikes rebalanced by vehicles, and those moved by users (i.e., successful rentals and returns). Constraints (4.4) ensure that each vehicle v can only be at one station at each time-period. Constraints (4.5) ensure that a vehicle can perform operations at a station only when it is present at that station. Constraints (4.6) impose that the number of bikes in each vehicle is bounded by its capacity. Constraints (4.7) are the capacity constraints for the stations. Constraints (4.8) bound the number of successful trips by the expected rental and return. Finally, constraints (4.9) enforce that the pick-up and drop-off operations respect the vehicle's capacities.

The above model, denoted as **D**ynamic **R**ebalancing **O**ptimization for **B**SS minimizing **L**ost **D**emand (DROB-LD), derives rebalancing strategies for the entire planning horizon, i.e., it decides how many bikes each vehicle should pick up or drop off at which station. The model can be easily implemented in different reoptimization modes (static, rolling, and folding planning) with alterable length of planning horizons and duration of time-periods, depending on the requirements of the decision-maker.

4.3.2 Rebalancing Models Based on Inventory Interval and Target Inventory

Even though the above used objective minimizing the lost demand is quite popular in the literature, its performance is sensitive to the accuracy of the expected rentals $f_s^{+,t}$ and returns $f_s^{-,t}$. Rather than minimizing the deviation from such a point estimate, we propose to minimize the deviation from either the inventory interval or the target inventory. This approach provides a buffer for the station inventories, allowing them to maintain reasonable inventories even when the trip prediction is less accurate, and to be better prepared for fluctuations of the stochastic demand.

To this end, we propose two multi-period models with novel objective functions: **D**ynamic **R**ebalancing **O**ptimization for **B**SS based on **T**arget Inventories (DROB-T) and **D**ynamic **R**ebalancing **O**ptimization for **B**SS based on **I**nventory Intervals (DROB-I). The parameters and variables used in both models are depicted in Table 4.3.

The objective function of DROB-T (4.11) aims at minimizing the total deviations between station inventories and target values, thus yielding the following formulation:

$$\begin{aligned} \min \quad & \sum_{s\in S}\sum_{t\in T} |\ell_s^t - d_s^t| && (4.11)\\ \text{s.t.} \quad & (4.2)-(4.10). \end{aligned}$$

Table 4.3 Parameters and variables that define inventory intervals and target inventories

Input	Definition
ℓ_s^t	The target inventory of station $s \in S$ at period $t \in T$
$\underline{\ell}_s^t$	The lower bound for the inventory interval of station $s \in S$ at period $t \in T$
$\overline{\ell}_s^t$	The upper bound for inventory interval of station $s \in S$ at period $t \in T$
Variables	**Definition**
$e_s^{+,t}$	The number of bikes above the upper bound at station $s \in S$ in period $t \in T$
$e_s^{-,t}$	The number of bikes below the lower bound at station $s \in S$ in period $t \in T$

DROB-I is designed to keep the station inventories as much as possible within the computed intervals. To this end, DROB-I is formulated as the objective function (4.12), along with constraints (4.2)-(4.10) and (4.13)-(4.15) as defined below:

$$\min \quad \sum_{s\in S}\sum_{t\in T} e_s^{-,t} + e_s^{+,t} \tag{4.12}$$

$$\text{s.t.} \quad \underline{\ell}_s^t - e_s^{-,t} \le d_s^t \qquad \forall\ s \in S,\ t \in T \tag{4.13}$$

$$d_s^t \le \overline{\ell}_s^t + e_s^{+,t} \qquad \forall\ s \in S,\ t \in T \tag{4.14}$$

$$e_s^{-,t} \ge 0, e_s^{+,t} \ge 0 \qquad \forall\ s \in S,\ t \in T \tag{4.15}$$

$$(4.2) - (4.10).$$

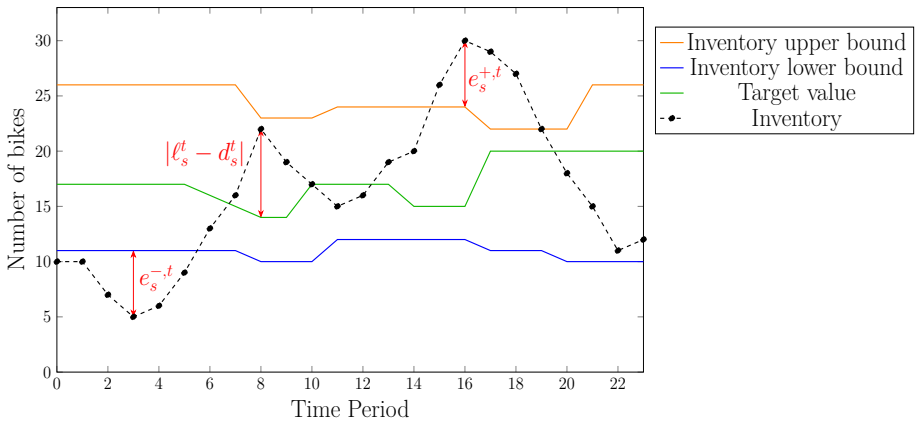

Figure 4.2 Example of deviations from inventory intervals and target inventory

Figure 4.2 exemplifies the inventory of a station s, as well as the lower bound and upper bound of the inventory interval, and its target inventory. For DROB-I, the excess of inventory at each time-period ($e_s^{+,t}$ and $e_s^{-,t}$, respectively), with respect to the inventory interval $[\underline{\ell}_s^t, \overline{\ell}_s^t]$, is computed by constraints (4.13) and (4.14). For DROB-T, Figure 4.2 also illustrates the deviation $|\ell_s^t - d_s^t|$ from the current target value. By minimizing these deviations, DROB-I and DROB-T aim at providing safety buffers to the station inventory and are therefore more likely of being capable to deal with stochastic demand fluctuations. We next present two toy

examples to provide an intuition of the potential benefits of the two new objective functions.

Table 4.4 Comparative analysis of three objective functions for rebalancing operations

Objective Function	Example 1		Example 2	
	Dropped off Bikes	Expected Lost Demand*	Dropped off Bikes	Expected Lost Demand*
DROB-LD (4.1)	1	0.33	0	0.67
DROB-T (4.11)	8	0	5	0
DROB-I (4.12)	2	0	2	0

* The expected lost demand is calculated considering all possible chronological sequences of rentals and returns derived from historical trip data. For simplicity, we assume that the probability of a rental occurring before a return equals to that of a return happening before a rental.

Example 1. Consider an empty station with 10 docks. For a given time-period at a given day, historical rentals follow a uniform distribution ranging from 0 to 2, while no returns have been observed. A predictive model is used to predict the expected demand and target value and inventory interval are computed as to ensure a sufficient service level (see Section 4.4.1 and Appendix B.1 for details). As a result, an estimation of 1 rental and no return is obtained, directly used in model DROB-LD. For DROB-T, the computed target value is 8, while for DROB-I, the computed inventory interval is [2, 10]. Table 4.4 summarizes the number of bikes dropped off at that station according to each of the three models. Furthermore, the table reports the expected lost demand if rental demand is uniformly distributed between 0 to 2. Here, DROB-T and DROB-I drop off at least 2 bikes, accounting for the potential demand of 2 rentals. In contrast, DROB-LD drops off only 1 bike, and therefore lacks 1 bike when the rental demand is 2.

Example 2. Consider that the same empty station, for another time-period and given day, has a uniform distribution between 0 and 2 for both rentals and returns. Both the estimated rental and return are therefore 1. The computed target value is 5, whereas the inventory interval is [2, 8]. In this case, DROB-T and DROB-I still drop off at least 2 bikes and therefore do not induce any unmet rental demand. In contrast, DROB-LD implicitly assumes that returns cancel rentals, and thus does not drop off any bikes, which may result in unmet demand when the rental demand is 1 or higher.

4.4 Experiments and Results

We now employ computational experiments to explore the benefits of the proposed models. Section 4.4.1 introduces the synthetic data and reports on the corresponding empirical results. A case study on real-world data is then presented in Section 4.4.2.

Computational environment. All optimization models are solved using IBM ILOG CPLEX v20.1.0.0 on 2.70 GHz Intel Xeon Gold 6258R machines with 8 cores. Optimization terminates once the MIP gap reaches 0.01% or the time limit of 24 hours is reached.

4.4.1 Experiments on Synthetic Data

We here focus on experiments carried out on synthetic problem instances. To this end, we first introduce the instance generator that generates weather-dependent trip data. We then detail the experimental set-up, including the machine learning model used to predict rental demand, the computation of inventory intervals and targets, and the simulator. This section also summarizes the computational results, comparing the performance of the various planning models. Finally, we explore the performance of the planning models under the assumptions that predictions are less accurate.

Synthetic dataset

Even though we have access to real-world trip data, we synthetically generate instances for several reasons. First, the available real-world trip data lacks information on unobserved demand. Second, existing data may contain noise related to trip and station inventory data. Finally, rebalancing operations conducted by operators impact station inventory, but data on such operations is not openly available.

The general data generation process of the here employed instance generator is depicted in Figure 4.3. We extend the instance generator proposed by [65] (corresponding to the third box in Figure 4.3: "*Trip data for the generated station network*"), which is capable of generating diverse station networks and trip data, based on predefined *trip patterns & distributions* that align with those observed in real-world BSSs. Whereas the instance generator of [65] generates trip data under the same weather conditions (specifically, assuming high demand during summer months), we here explicitly acknowledge the strong correlation between weather conditions and trip demand. As such, we extend this instance generator as follows. (i) We introduce varying weather conditions into our generator (corresponding to the first box "*Weather data*" in Figure 4.3) by estimating statistical distributions that represent the

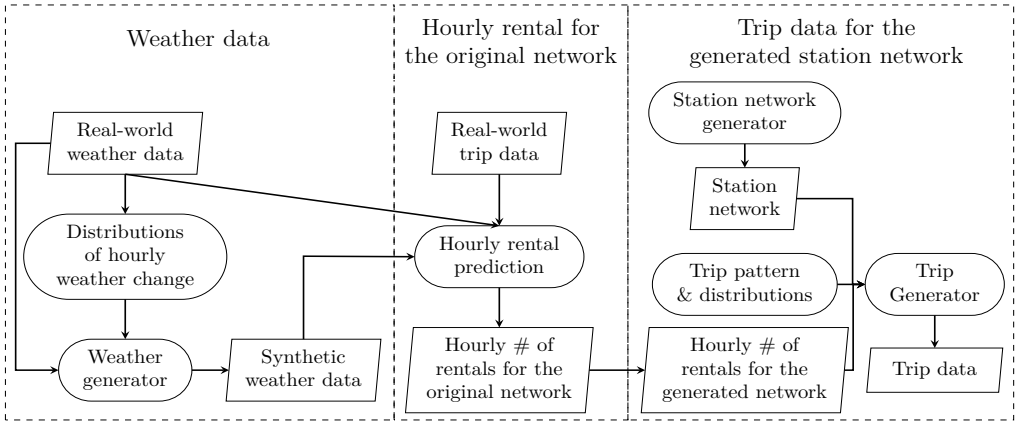

Figure 4.3 Process to generate weather data, station network and trip data

hourly changes of weather conditions estimated on real-world weather data, and then sample new weather conditions from these distributions. (ii) We then compute the system-wide level of trip demand that is correlated to such weather conditions (corresponding to the second box "*Hourly rental for the original network*" in Figure 4.3), i.e., the hourly number of rentals for the entire network. Finally, individual trip demand is generated for the generated station network. Each of these components is next explained in detail.

Generation of weather data. Given that we ultimately aim at generating trip data that resembles periods of the demand peak season, we are interested in generating weather conditions for the months of June to August, which tend to have high user demand. We therefore procure *Real-world weather data*[2] from Montreal for June, July, and August from 2017 to 2020, resulting in a total of 368 days (including both weekdays and weekends). We select two features of utmost importance (see, e.g., [34, 41, 42, 57]): temperature and humidity. We analyze the temperature and humidity differences between consecutive hours throughout the day and divide the day into four distinct time segments such that temperature and humidity tend to remain relatively stable within each of which (0 am – 5 am, 6 am – 11 am, 12 pm – 5 pm, and 6 pm – 11 pm). The hourly differences are then used to estimate normal *Distributions of hourly weather change* for each time segment. The estimated distributions of temperature change for each time segment can be found in Appendix B.2.

Synthetic weather data has been generated for a total of 500 days as follows. For each hour of the original 368 days, we use the temperate and humidity that originally occurred at that day and add a change of temperature and humidity, respectively, sampled from the corresponding distributions. To obtain a total of 500 days, we repeat the process with the first

[2]https://climate.weather.gc.ca

132 of the original 368 days. To ensure that the generated weather conditions are sufficiently realistic and avoid drastic fluctuations, we introduce constraints to keep the temperature and humidity within 5.5°C–36°C and 15%–99%, respectively. We denote the final set of the 500 generated days with synthetic weather data as the *Synthetic weather data*.

Generation of hourly rental demand. Using the temperature, humidity, hour, and weekday as features, we estimate the *Hourly rental prediction* model. This linear regression model is trained using the *Real-world weather data* and *Real-world trip data* (also see Section 4.4.2) and captures the correlation between time and weather conditions and the total demand level for the entire station network. The trained regression model is then used to estimate the total system-wide number of *Hourly # of rentals for the original network* that depends on the *Synthetic weather data* throughout each day. Given that this total number of rentals has been estimated on the original network from the *Real-world trip data* (here, the BIXI network with over 600 stations), this number is then scaled to the number of stations used in the here considered *Station network* (which has 60 stations). These hourly system-wide rental demands (*Hourly # of rentals for the generated network*) then serve as input to generate the detailed trip data.

Generation of station network and individual trip data. We generate two ground truth problem instances, denoted GT1 and GT2, each of which contains a *station network*, as well as hourly weather data and detailed trip information for 500 days. In both instances, the network contains 60 stations with different numbers of city center stations. The stations within city centers are equipped with 40 docks, while those outside city centers have 20 docks each.

Generated trips contain the origin station, the destination station, the departure time, and the arrival time. We consider four trip patterns of user behaviors with origins and destinations outside (O) and inside (I) city centers: (i) users who live outside city centers and work inside city centers typically use similar origin (outside city centers) and destination stations (inside city centers) during peak hours (OI trips); (ii) users who live and work outside city centers (OO trips); (iii) random non-work related trips occurring during the day (RD trips), and (iv) random non-work related trips occurring during the night (RN trips).

The characteristics of instance ground truths GT1 and GT2 are described in Table 4.5. Although the proportions of work-related (i.e., city center related) trips are identical in GT1 and GT2, the latter has more city center stations (12 stations in 2 city centers, as opposed to 9 stations in 1 city center). As such, work related trips in GT2 are distributed over a larger

Table 4.5 Characteristics of the two considered ground truth instances

Instance		GT1	GT2
Network (60 stations)	# of city centers	1	2
	# of stations per city center	9	6
	City center capacity	26%	35%
Trip Pattern	*OI*	32%	32%
	OO	32%	32%
	RD	23%	23%
	RN	13%	13%

number of stations, which are therefore less stressed.

To sample individual trips for the considered *station network*, we assume that each trip type follows a particular temporal distribution, indicating the probabilistic time at which the rental occurs (as detailed in [65]). For each day, the *trip generator* then sequentially samples trips (i.e., origin-destination pairs and exact time stamps) from the *Trip pattern & distributions* (as defined by the ground truth) until the *hourly # of rentals for the generated network* is met, resulting in the final set of *Trip Data*.

Model performance based on regular trip predictions

Experimental set-up. The 500 generated days for GT1 and GT2 are separated into training set, validation set and test set as follows. The first 250 days are allocated to calibrate the gradient boosted tree introduced in [54], capable of predicting hourly station demand and trained on trips (time of rental and arrival/departure stations), weather conditions (temperature and humidity), and temporal data (day of the week, hour of the day, and a binary indicator for holidays). The subsequent 100 days are used for the validation and fine-tuning of the gradient boosted tree and inventory intervals. Details on the training of the gradient boosted tree can be found in Appendix B.1.

The remaining 150 days constitute the test set on which the optimization algorithms are executed. We consider a planning horizon from 7 a.m. to 3 p.m., discretized into 8 time-periods, each with a duration of one hour. For each of the 150 days, we assume to have access only to its corresponding weather conditions, but not to the exact trip (i.e., rental and return) demand. This is a reasonable assumption in practice, where one can assume to have access to a reasonably accurate weather prediction. Based on such weather conditions, the trained gradient boosted tree then predicts the hourly rental and return demand for each station ($f_s^{+,t}$ and $f_s^{-,t}$), used within model DROB-LD. The inventory intervals and target values, used within models DROB-I and DROB-T, are then computed based on the predicted rental demand (details can also be found in Appendix B.1). For each of the 150 days, the rebalancing

planning solutions provided by the various models are then evaluated in the simulator (see Section 4.4.1) on the exact trip data. Note, again, that the optimization models only have access to demand predictions (based on weather data), whereas the simulator evaluates on the exact trip demand of the days in the test set.

In all experiments, 4 vehicles are available to rebalance the stations, each with a capacity for 40 bikes. The initial inventory of stations is obtained by solving an overnight rebalancing problem (equivalent to the one used in [65]).

Each of the optimization models can be executed in different reoptimization planning modes. In *static planning*, the optimization model is solved once for the entire planning horizon. The rebalancing strategies of the first 6 (out of 8) time-periods are then executed within the simulator to estimate the lost demand. The *rolling planning* has 3 optimization stages, each of which contemplates 4 time-periods. At each stage, the rebalancing decisions of the first 2 time-periods are executed within the simulator, as illustrated in Figure 4.1 (A). The *folding planning* uses all the remaining time-periods at each stage, as depicted in Figure 4.1 (B). The fine-grained discrete-event simulator from [65] here used employs a chronological first-arrive-first-serve rule, for both user rentals and returns, as well as rebalancing vehicles (i.e., pick-ups and drop-offs). Events are discretized events into 1-minute time-slots, which results in a particularly detailed and realistic simulation.

Computational Results. Table 4.6 illustrates the average lost demand and computing time (over the test set) for all the models and reoptimization modes (**S**tatic, **R**olling, and **F**olding). We report the computing times required to solve the optimization models as 'Opt. Time' (in minutes). The lost rental demand is computed as the relative gap between successful rentals and the original rental demand specified in the instances over the entire planning horizon, i.e., $\frac{\sum_{s,t}(f_s^{+,t}-\hat{x}_s^{+,t})}{\sum_{s,t} f_s^{+,t}}$, where $\hat{x}_s^{+,t}$ is the number of successful rentals in the simulator. The lost return demand is computed as $\frac{\sum_{s,t}(\hat{x}_s^{+,t}-\hat{x}_s^{-,t})}{\sum_{s,t} \hat{x}_s^{+,t}}$, where $\hat{x}_s^{-,t}$ is the number of successful returns in simulator. Since, in practice, return demand does not exist when the corresponding rental demand is unsuccessful, the lost returns are only associated with successful rentals $\hat{x}_s^{+,t}$. We also present the relative difference ($\Delta(\%)$) of the rental, return, and total lost demand of DROB-I and DROB-T when compared to DROB-LD under the respective reoptimization mode.

Table 4.6 allows for the following observations:

1. **Comparison of proposed models.** From Table 4.6, models DROB-I and DROB-T generally outperform DROB-LD for both GT1 and GT2. For example, on GT1, DROB-I reduces the lost demand from 7.65% to 5.66% in rolling planning, while being

Table 4.6 Results of dynamic rebalancing models for GT1 and GT2 under regular prediction

Instance	Model	Reopt. Mode	Opt.Time (min.)	Lost Demand (%)					
				Rental	Δ(%)	Return	Δ(%)	Total	Δ(%)
GT1	DROB-LD	S	0.19	13.14		2.74		8.31	
		R	0.02	11.94		2.76		7.65	
		F	0.21	11.54		2.87		7.48	
	DROB-I	S	0.18	10.68	-18.72	3.79	38.32	7.43	-10.59
		R	0.10	8.18	-31.49	2.90	5.07	5.66	-26.01
		F	0.39	8.39	-27.30	2.84	-1.05	5.74	-23.26
	DROB-T	S	1.53	9.98	-24.05	3.97	44.89	7.14	-14.08
		R	0.15	8.20	-31.32	1.52	-44.93	5.01	-34.51
		F	1.79	8.21	-28.86	1.61	-43.90	5.06	-32.35
GT2	DROB-LD	S	0.41	8.27		1.68		5.13	
		R	0.06	8.16		1.67		5.06	
		F	0.44	8.11		1.62		5.01	
	DROB-I	S	0.04	7.76	-6.17	2.51	49.40	5.24	2.14
		R	25.24	6.31	-22.67	1.35	-19.16	3.92	-22.53
		F	229.15	5.87	-27.62	1.12	-30.86	3.57	-28.74
	DROB-T	S	0.13	8.79	6.29	0.63	-62.50	4.91	-4.29
		R	0.28	6.96	-14.71	0.58	-65.27	3.89	-23.12
		F	0.32	6.97	-14.06	0.57	-64.81	3.90	-22.16

solved within seconds. While under static planning, DROB-I and DROB-T tend to outperform DROB-LD, they consistently outperform DROB-LD by a higher rate under rolling planning (reducing total lost demand by 23.26%-34.51%)

2. **Comparison of different reoptimization modes.** The rolling and folding planning consistently result in lower lost demand than the static planning, likely due to the fact that they update the station inventories before reoptimizing at every reoptimization stage (see Figure 4.1). Updating the inventory narrows the gap between the estimated inventories in the optimization model and the observed inventories during the simulation, allowing the optimization models to make more informed decisions. For example, rolling and folding planning in DROB-I on GT1 reduce the lost demand from 7.43% to 5.66% and 5.74%, respectively, over the static planning. Note that updating weather forecast may have significat impact in the rolling planning, which will be discussed later. In terms of computing times, even though most models have been solved within 1-2 minutes, the folding planning requires much longer computing times for GT2.

3. **Comparison between GT1 and GT2.** While the general conclusions and tendencies

are the same for GT1 and GT2, for GT1, models present more unmet demand than for GT2. Indeed, GT2 has more stations located in city centers, a region with high work-related demand, resulting in more evenly distributed trip patterns for these stations. Moreover, the larger number of city center stations translates into greater availability of docks in the network, as stations located in this area contain twice the number of docks than stations located in other regions of the city.

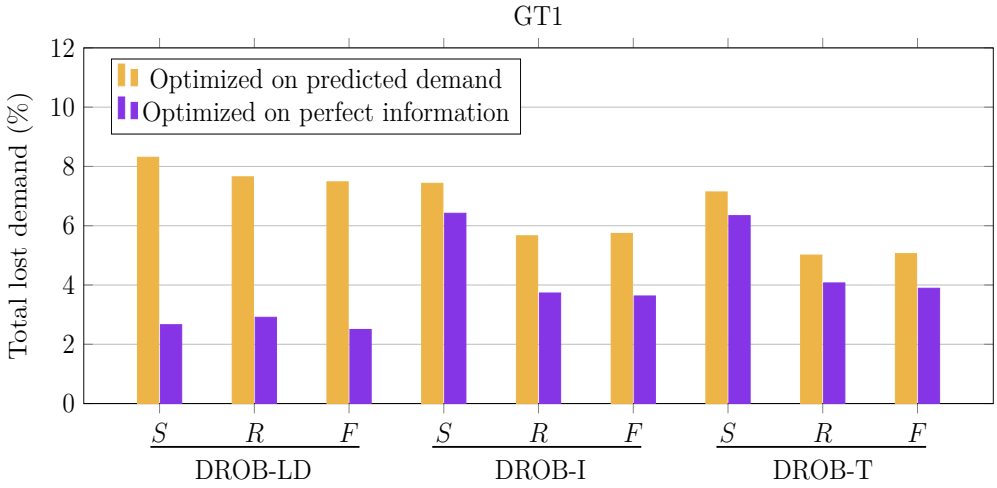

Figure 4.4 Results of total lost demand for GT1 using regular prediction and ground truth

Results based on perfect trip prediction. We further carry out experiments with perfect trip information for each day of the test set, i.e., the model optimizes on the exact rental and return demand on which its rebalancing policy is later evaluated within the simulator. These experiments establish an empirical performance bound and provide insights into the efficiency of the optimization models and reoptimization modes. Figure 4.4 presents the total lost demand obtained using the predicted trip demand (i.e., the same as used for Table 4.6) and the perfect information for GT1. Unsurprisingly, if perfect information was available, DROB-LD would consistently outperform DROB-I and DROB-T, given that an inventory safety buffer would be unnecessary. However, in reality, demand is stochastic. In this case, DROB-I and DROB-T can provide more robust station inventories, allowing them to deal with the stochastic trip demand. Interestingly, DROB-I and DROB-T still benefit from inventory updates (rolling and folding planning) under perfect information, as opposed to DROB-LD. Indeed, the former two models rely on the current inventory levels to update their objective functions, while the objective of DROB-LD remains unchanged, even when station inventories change. As a result, reoptimization for DROB-LD is not beneficial. Finally, the results also enable us to derive insights into the empirical bounds on the potential gains

achieved through the utilization of a more accurate predictive model. While the gains are substantial (∼5% of lost demand) for DROB-LD, they are much smaller (∼1-2%) for DROB-I and DROB-T. While using a more accurate predictions may obviously lead to reduced unmet demand, we will next investigate how those models perform when predictions are less accurate.

Model performance based on noisy prediction

The optimization models used in our previous experiments have taken as input demand predictions and interval predictions that have assumed a perfect weather forecast. In practice, weather forecasts for the next 2 to 8 hours can be prone to inaccuracies. In a similar vein, having access to a predictive model with sufficiently high accuracy may not always be possible. We will now investigate the performance of the various models under the assumption that demand and interval predictions are less accurate. To this end, we deliberately introduce noise into the performed trip predictions. Since accurately predicting demand becomes increasingly challenging as we project further into the future, we introduce more noise to later time-periods.

Noisy predictions. We consider two types of effects caused by noises over demand predictions: (i) overestimation, e.g., due to a forecast of overly favorable weather conditions and, therefore, expecting a higher number of trips than will actually occur; (ii) underestimation, e.g., due to a forecast of adverse weather conditions, therefore predicting a lower number of trips than will actually occur.

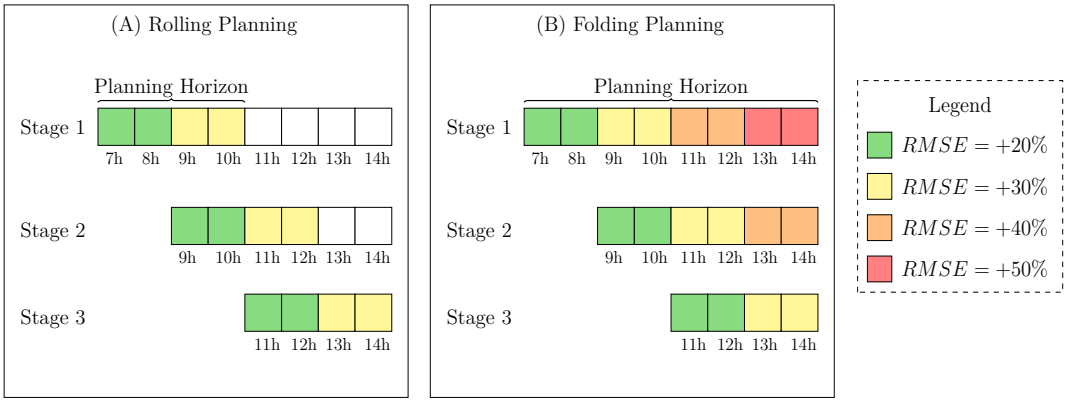

Figure 4.5 Noise added to the demand prediction over the planning horizon of the rolling and folding planning

To create noisy predictions for the demand at station s at time t, we sample noise from a

normal distribution. Its mean μ is determined by the original predicted number of rentals (or returns), while its standard deviation is strategically adjusted to achieve a predetermined increase in the Root Mean Squared Error (RMSE) for these noisy predictions in comparison to μ. This approach is applied throughout the planning horizon for each stage in the rolling and folding planning as illustrated in Figure 4.5. Values sampled above the mean are used to create overestimating predictions, whereas values below the mean are sampled to create underestimating predictions. Thus, an underestimating forecast consistently predicts lower demand and an overestimating forecast consistently predicts higher demand. Note that static planning operates under the same conditions as the first stage of folding planning.

Results. Tables 4.7 and 4.8 show the results for the three optimization models under underestimating and overestimating predictions, respectively. Same as in Table 4.6, the relative difference of DROB-I and DROB-T when compared to DROB-LD is reported ($\Delta(\%)$).

Table 4.7 Results of dynamic rebalancing models for GT1 and GT2 under underestimating predictions

Instance	Model	Reopt. Mode	Opt.Time (min.)	Lost Demand (%)					
				Rental	$\Delta(\%)$	Return	$\Delta(\%)$	Total	$\Delta(\%)$
GT1	DROB-LD	S	0.02	13.97		4.56		9.62	
		R	0.02	11.90		3.54		7.99	
		F	0.04	11.57		3.56		7.81	
	DROB-I	S	11.45	10.31	-26.20	4.37	-4.17	7.51	-21.93
		R	0.13	8.23	-30.84	2.82	-20.34	5.65	-29.29
		F	21.13	7.73	-33.19	1.68	-52.81	4.84	-38.03
	DROB-T	S	0.20	11.89	-14.89	2.55	-44.08	7.52	-21.83
		R	0.30	9.98	-16.13	1.11	-68.64	5.78	-27.66
		F	2.84	10.03	-13.31	1.04	-70.79	5.78	-25.99
GT2	DROB-LD	S	0.01	11.66		2.69		7.46	
		R	0.01	10.85		1.36		6.39	
		F	0.02	10.44		1.38		6.17	
	DROB-I	S	0.23	8.6	-26.24	2.48	-7.81	5.68	-23.86
		R	0.19	7.02	-35.30	1.40	2.94	4.32	-32.39
		F	1.94	6.99	-33.05	1.39	0.72	4.30	-30.31
	DROB-T	S	21.30	9.42	-19.21	1.27	-52.79	5.55	-25.60
		R	0.79	7.42	-31.61	0.56	-58.82	4.13	-35.37
		F	21.88	7.48	-28.35	1.16	-15.94	4.45	-27.88

Based on Tables 4.7 and 4.8, we summarize our observations as follows:

Table 4.8 Results of dynamic rebalancing models for GT1 and GT2 under overestimating predictions

Instance	Model	Reopt. Mode	Opt.Time (min.)	Lost Demand (%)					
				Rental	Δ(%)	Return	Δ(%)	Total	Δ(%)
GT1	DROB-LD	S	1.50	12.09		2.14		7.44	
		R	0.04	10.79		2.12		6.71	
		F	1.52	10.50		2.18		6.58	
	DROB-I	S	0.08	11.05	-8.60	3.77	76.17	7.62	2.42
		R	0.06	8.75	-18.91	1.62	-23.58	5.35	-20.27
		F	1.41	8.59	-18.19	0.86	-60.55	4.91	-25.38
	DROB-T	S	0.06	11.05	-8.60	3.94	84.11	7.71	3.63
		R	0.10	8.77	-18.72	1.24	-41.51	5.19	-22.65
		F	0.26	8.75	-16.67	1.21	-44.50	5.16	-21.58
GT2	DROB-LD	S	1.31	7.58		0.77		4.32	
		R	0.06	6.97		0.93		4.07	
		F	1.46	6.87		0.85		3.97	
	DROB-I	S	0.03	8.04	6.07	1.64	112.99	4.98	15.28
		R	0.03	7.00	0.43	0.60	-35.48	3.93	-3.44
		F	0.07	7.01	2.04	0.60	-29.41	3.93	-1.01
	DROB-T	S	0.08	8.63	13.85	1.56	102.6	5.27	21.99
		R	0.05	7.04	1.00	0.45	-51.61	3.88	-4.67
		F	0.16	7.06	2.77	0.46	-45.88	3.89	-2.02

1. **Comparison of models.** Although DROB-LD shows performance improvement in the case of overestimating predictions, DROB-I and DROB-T consistently demonstrate lower lost demand in most cases. Especially within rolling and folding planning, DROB-I and DROB-T outperform DROB-LD considerably. Their advantage is particularly pronounced when optimizing on underestimating predictions.

2. **Comparison of different reoptimization modes.** Generally, the improvement of lost demand when transitioning from static planning to folding and rolling planning is more significant under perturbed trip predictions than under noise-free predictions (see Table 4.6). This confirms the importance of such reoptimization planning modes when less accurate predictions are used.

3. **Comparison between predictions.** Underestimating trip predictions results in higher lost demand compared to noise-free predictions, since fewer rebalancing operations are triggered. In contrast, overestimating predictions may lead to less lost demand, especially notable for DROB-LD. This is explained by the fact that overesti-

mating predictions triggers more rebalancing operations in DROB-LD. We report the number of rebalancing operations (number of bikes picked up and dropped off) over the planning horizon in Figure 4.6. Indeed, overestimating predictions results in more rebalancing operations to meet the high demand. Overall, it appears that DROB-I and DROB-T are less sensitive to prediction noise than DROB-LD, given that they are designed to introduce a buffer into the optimized stations' inventories.

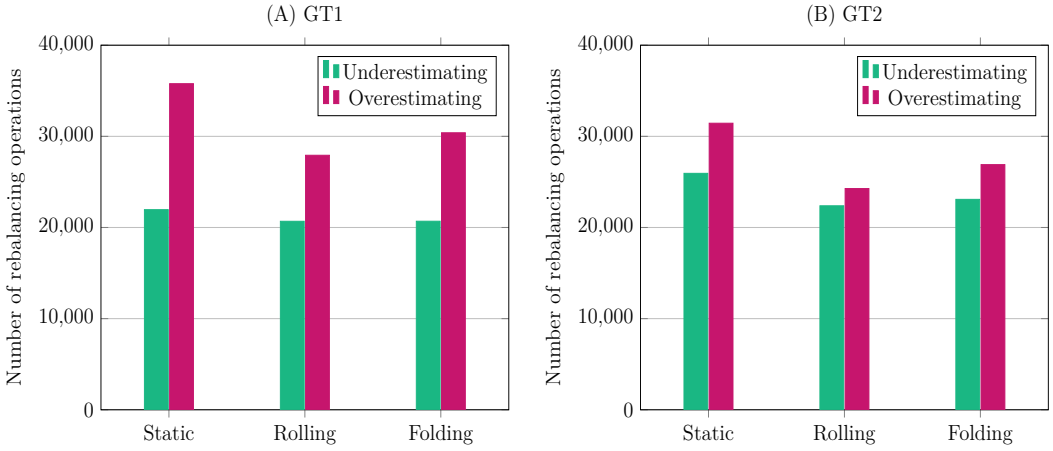

Figure 4.6 Number of rebalancing operations carried out in GT1 and GT2 for the DROB-LD

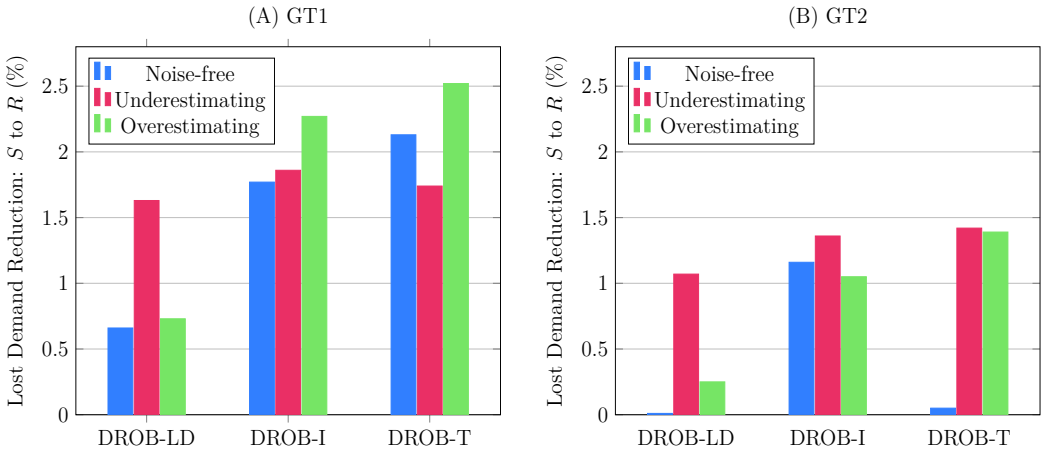

Figure 4.7 Improvement of the total lost demand from static to rolling planning

Remarks. Overall, the results indicate that DROB-LD is much more susceptible to the accuracy of demand predictions than DROB-I and DROB-T as it directly considers such predictions in its objective function. By introducing a buffer to the stations inventories, the performance of DROB-I and DROB-T remains more stable among the different prediction

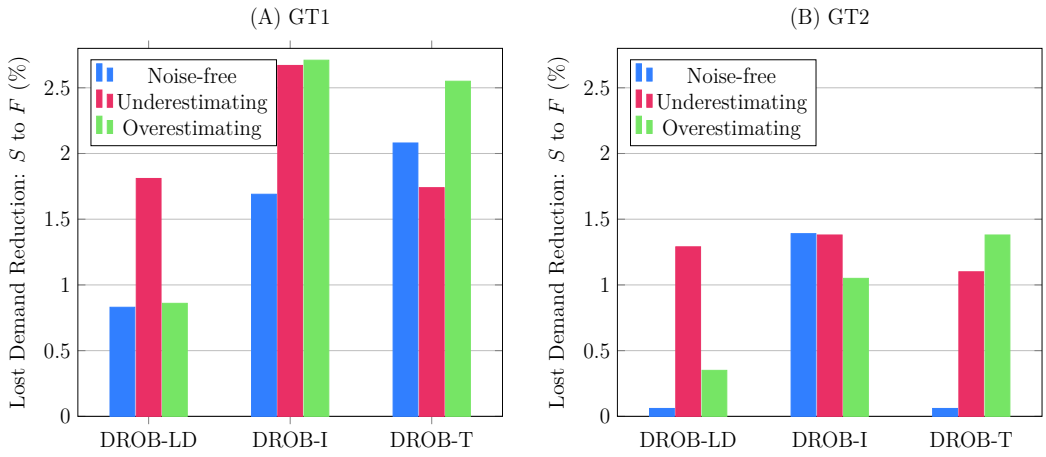

Figure 4.8 Improvement of the total lost demand from static to folding planning

approaches. To visualize the improvement of rolling/folding planning over static planning, we illustrate the difference in total lost demand between static and rolling planning in Figure 4.7, and between static and folding planning in Figure 4.8 for both GT1 and GT2. It is worth noting that the difference in total lost demand in these figures consistently shows positive values, meaning that in all experiments, rolling and folding planning consistently outperform static planning. These improvements are even more significant in the experiments with underestimating and overestimating predictions. This can be attributed to the fact that, in addition to updating the station inventory, a more accurate trip prediction is updated before re-optimization in each stage (see Figure 4.5).

4.4.2 Experiments on Real-world Data

In this section, we describe the real-world dataset used to validate the effectiveness of model DROB-I, which, on synthetic data, has demonstrated consistently low lost rentals while maintaining reasonable computing times. We first describe the real-world dataset. We then describe the results. The experimental set-up and planning horizon here considered are the same as in the experiments on synthetic data.

Real-world dataset

The real-world data consists of weather, temporal, station, and trip data. Weather and temporal data are directly provided by the official website of the Government of Canada, including temperature and humidity. The temporal data contains the date, hour (0h -23h), year (2019), and weekday (Monday to Friday). The trip and station data are provided by

BIXI[3]. The trip data contain the origin station, start time, destination station, and arrival time of each trip, while the station data contain the location and station capacity (i.e., the number of docks).

We only focus on trips during weekdays from May to September 2019. Selecting trips before 2020 ensures that analyzed trip patterns are not affected by the COVID-19 pandemic. Weekdays are chosen due to their typically consistent work-related trip patterns. The first 21 days of each month, excluding the weekends, constitute the training dataset. The remaining days of May are used for validation and the remaining days from June to September are assigned to the test dataset. The initial inventory for stations at 7 a.m. is also collected from BIXI dataset and serves as input for the optimization models.

For BIXI's station network, we exclude stations that have been relocated more than 1 km from their original locations by the operator during specific events, constructions, or holidays. As a result, 606 stations out of originally 620 remain in our experiments. This network is too large to be directly solved by general-purpose solvers. As a remedy, literature often divides the network into smaller clusters. Rebalancing is then performed within a specific cluster or between different clusters (see, e.g., [16, 37, 43, 53, 56, 68]).

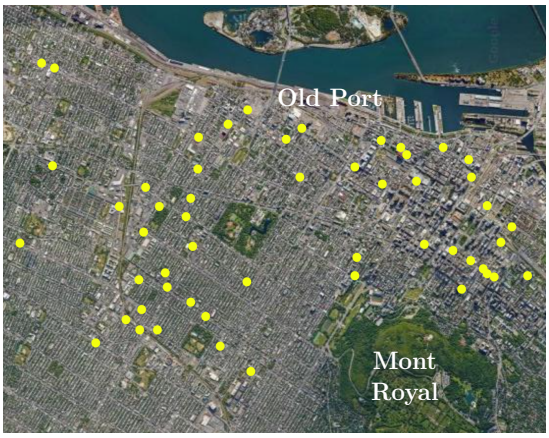

Figure 4.9 Cluster of BIXI stations located in Montreal (Canada)

We follow the approach of [65] to cluster the stations according to their trip behaviour using k-means. We then select a cluster around the downtown and plateau areas in which the total number of rentals is approximately the same as the total number of returns. This cluster has 53 stations, including several city center stations and therefore contains work-related trips. Given that the distances between stations inside the cluster are limited, vehicles have

[3]https://bixi.com/en/open-data-2/

sufficient time available to relocate and rebalance bikes within each time-period. The stations in the selected cluster are visualized in Figure 4.9.

Results on Real-world data

Given that, on synthetic data, DROB-I within rolling planning outperformed DROB-T on lost rental under all the predictions and consistently had swift computing times, we here focus on comparing the performance of DROB-I and DROB-LD. Experiments are carried out in a rolling planning, which aligns with practice and accommodates the need for swift runtime, while also allowing for real-time system updates.

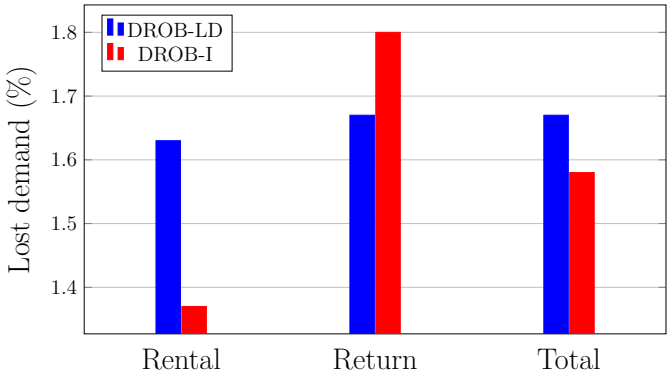

Figure 4.10 Lost demand of DROB-I and DROB-LD on a cluster from BIXI

The results are visualized in Figure 4.10. Detailed results for each day can be found in Appendix B.3. DROB-I performs better on both lost rental and total lost demand, while the lost return is higher than for DROB-LD. Note that lower lost rental demand (and therefore more successful trips) also results in more return demand, which explains that DROB-I suffers from a slightly higher lost return. Such observations align with previous results on synthetic data. The results on real-world data confirm the benefits of model DROB-I, providing robustness to the station inventories and reducing the total lost demand. In terms of computing time, both models can be solved to optimality within 1 minute.

In contrast to the results on synthetic data, the improvement provided by DROB-I is not impressive. Such smaller improvement may be explained by the fact that the here-considered real-world data only contains successful trips. We also carried out static planning for the DROB-LD and DROB-I. However, under static planning, DROB-LD runs out of memory for many cases, given that the model contains more time-periods. The total lost demand of DROB-I under static planning has been 1.98%, shortly higher than the one under rolling planning (1.58%), highlighting once again the benefits of the rolling planning for DROB-I.

4.5 Conclusions

In this work, we have proposed two objective functions for multi-period rebalancing models for Bike-sharing systems, DROB-I and DROB-T, incorporating inventory intervals and target inventories. The resulting models provide an alternative to classical models minimizing unmet demand and are particularly suitable for BSSs operators that use (often manually computed) inventory intervals and targets to guide their rebalancing process.

Our work evaluates the entire pipeline required for an automatized and data-driven rebalancing process. Instead of relying on manual input, we estimate rental and return demand for each hour and station in a data-driven fashion, using a machine learning model that has been shown to provide reasonably accurate results based on historical data related to time, weather and user trips. Inventory intervals and targets are then derived such that they maximize the desired service-level.

Our empirical analysis explore the capability of the proposed planning solutions to meet customer demands under three key characteristics of the planning process: the used optimization model, the employed reoptimization mode, and the impact of highly accurate (or inaccurate) demand predictions. The obtained planning solutions are then evaluated within a fine-grained minute-by-minute discrete-event simulator. A series of experiments on synthetic data allows for three key conclusions: First, our proposed models exhibit remarkable robustness compared to DROB-LD, the classical model minimizing unmet demand. DROB-T leads to a reduction in lost demand of up to 34%, while DROB-I decreases lost demand by up to 28%. Second, such robustness is also observed when demand predictions are less accurate, as these models introduce a conservative buffer into the station inventories, capable of better dealing with stochastic demand fluctuations. Third, there is a pronounced benefit in reoptimizing the rebalancing decisions throughout the planning as opposed to executing the optimization model only once and implementing a static planning solution for the entire planning horizon. Allowing for updated system information, the improvement via a rolling or folding planning has been found to be consistently in the order of 15-20% as opposed to static planning for DROB-LD. For DROB-I and DROB-T, the improvement tends to be higher than 30%, clearly indicating the benefits of such additional reoptimization effort. Finally, the benefits of our proposed models observed on synthetic data are also verified on a case study with real-world data from a BIXI Montreal.

Our approach may be highly attractive to system operators, not only due to their superior performance, but also due to their fit within the existing decision-making processes, as inventory intervals and targets are often used concepts in practice. In addition, we hope that the

here proposed weather generator inspires future research to evaluate planning approaches in a more complex and realistic manner. Given the benefits of the here proposed models, we believe that the development of tailored solution methods (such as mathematical decomposition methods) may be promising research directions, highly useful for both academia and practitioners to approach rebalancing in large-scale station networks.

Acknowledgements

The authors are thankful to BIXI Montreal for providing real-world trip data and for their support throughout several discussions. We also thank the Digital Research Alliance of Canada for providing the computational resources to carry out the numerical experiments. The work of the fourth author was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada under grant 2017-05224 and the Fonds de Recherche du Québec - Nature et technologie (FRQNT).

CHAPTER 5 A REINFORCEMENT LEARNING APPROACH FOR DYNAMIC REBALANCING IN BIKE-SHARING SYSTEMS

5.1 Introduction

The blossoming of Bike-sharing systems (BSSs) in smart cities worldwide marks a significant stride towards easing traffic congestion and curtailing automobile emissions. By 2022, the global landscape boasted over 1,900 operational BSSs, collectively fielding close to 9 million bikes [76]. Despite their widespread adoption, optimizing rebalancing strategies remains a critical research area for further exploration and refinement. Due to the stochastic nature of user behaviors and high demand in certain areas during peak hours, the frequent imbalance of stations (full/empty) results in user dissatisfaction and lost demand. To mitigate this, BSSs operators actively redistribute bikes across stations using vehicles.

The optimization problem for bike-sharing repositioning can be divided into static bicycle repositioning problem (SBRP) and dynamic bicycle repositioning problem (DBRP) [84]. The DBRP stands out for its critical impact on user satisfaction, primarily because it encompasses intraday rebalancing operations that take into account the interference of user demand, in contrast to SBRP. We focus on DBRP, given its significance in ensuring continuous rebalancing throughout the day.

Existing works primarily address DBRP through Mixed-Integer Programming (MIP) or Markov Decision Process (MDP). The majority of the literature applies MIP models (see, e.g., [46, 74, 111, 114], etc.) leveraging their flexibility and applicability within an industrial decision-making process. However, the MIP model typically simplifies DBRP by discretizing the planning horizon into multiple time-periods and making certain assumptions for tractability. This may restrict the capability to effectively coordinate multiple vehicles (e.g., restricting each vehicle to rebalance only one station at a time) and hinder the ability to capture optimal sequential decisions. In contrast, MDP presents a viable alternative, framing DBRP as a sequential decision-making problem [4]. Nevertheless, MDP models usually depend heavily on the dimensions of the state and action spaces, confining their applicability to limited scenarios, such as deterministic demands, single-vehicle operations, or very small station networks. Advanced Reinforcement Learning (RL) techniques offer a potential to solve MDP models for DBRP, an area that remains underexplored.

Our study aims at proposing a spatio-temporal RL-based algorithm for dynamic rebalancing problems in BSSs with multiple vehicles under a continuous time framework without dis-

cretization. We formulate DBRP as a Multi-agent Markov Decision Process (MMDP) whose state and action spaces are designed to capture the dynamics and uncertainties of the system. We utilize a Deep Q-Network (DQN) to estimate total expected rewards of rebalancing actions, targeting to minimize lost demand. A fine-grained simulator is developed to obtain immediate rewards under different demand scenarios. Experiments are conducted on a trip dataset generated based on historical trips, weather conditions, and temporal factors.

The main contributions of this study can be summarized as follows: (1) We introduce a spatio-temporal RL-based algorithm for DBRP, aiming at minimizing lost demand for networks of up to 60 stations. (2) We design a highly realistic simulator to estimate the rewards under the first-arrive-first-serve rule, handling different demand scenarios generated from weather and temporal information. (3) We construct a DQN algorithm to estimate the total lost demand for DBRP, especially in contexts involving simultaneous rebalancing operations by multiple vehicles without time discretization. (4) We conduct an experimental evaluation whose results demonstrate the effectiveness of our algorithm in reducing lost demand, compared to the benchmarks.

The remaining of this paper is organized as follows. Section 5.2 reviews related literature on BSSs DBRP and summarizes the existing MDP models and RL algorithms for it. Section 5.3 describes the model formulation based on MDP under continuous time frameworks. Section 5.4 presents the methodology to estimate the value function. A simulator considering stochastic demand is discussed to calculate the immediate rewards. Numerical tests and analyses are illustrated in Section 5.5. This is followed by the conclusions in Section 5.6.

5.2 Literature Review

In this section, we first review the DBRP in BSSs (Section 5.2.1). We then illustrate the existing MDP models for DBRP in Section 5.2.2. Finally, the RL techniques used for DBRP related problems are summarized in Section 5.2.3.

5.2.1 Dynamic Bicycle Repositioning Problems

Rebalancing strategies in BSSs are primarily categorized into two types: user-based rebalancing and operator-based rebalancing [96]. User-based rebalancing entails incentivizing users to rent or return bikes at specific stations, as discussed by [48]. This strategy is predominantly utilized in dockless BSSs. In contrast, operator-based rebalancing involves the active participation of a dedicated rebalancing fleet (commonly vehicles) to redistribute bikes, which is prevalent in station-based BSSs. According to a recent statistical report by [76], station-

based systems are currently the more dominant ones. Even when addressing rebalancing challenges in dockless BSSs, researchers tend to divide the station network into sub-clusters, each of which can be regraded as an individual station (see, e.g., [28, 72, 102, 113]). Given the prevalence and relevance of station-based systems, our study opts to concentrate on operator-based rebalancing in station-based BSSs.

In DBRPs, rebalancing operations are performed multiple times during the day and real-time demand flow is considered when rebalancing takes place. The goal is to find rebalancing strategies during the day (especially during peak hours) with minimum lost of demand (unsatisfaction).

The MIP models in DBRP usually employ time discretization, partitioning the temporal dimension into specific periods. Time-periods with equal length is most common in the literature (see, e.g., [20, 69, 114]). To simplify the complexity of MIP models, a series of assumptions are often introduced, notably constraining each vehicle to rebalance at a maximum of one station per time-period. While multi-period planning allows for anticipation of station changes and access to lookahead information, the critical challenge lies in determining the optimal length of time-periods. The length significantly impacts the rebalancing strategy and its performance. In real-world applications with the stochastic nature of the demand, especially when multiple vehicles are engaged in simultaneous yet independent rebalancing activities, multi-period planning may result in suboptimal and impractical solutions. For example, one vehicle has completed its rebalancing for the current period, while another vehicle is still rebalancing bikes. Such disparity in operational timings shows the inefficiency and unreality of the solutions generated by multi-period planning, highlighting the necessity for a modeling approach that better aligns with the dynamic and uncertain nature of bike demand and the asynchronous operations of the rebalancing fleet.

Alternatively, MDP has been applied to derive a dynamic bike repositioning strategy. However, MDP models can be limited by the immense complexity of the problem space, often rendering the computation of an optimal policy impractical within a reasonable time. This high dimensionality obstacle is a recurrent theme in the critique of MDP applications within DBRP [61]. We summarize the work on MDP for DBRP in the following section.

5.2.2 Markov Decision Process

MDP is a mathematical model for sequential decision-making in stochastic environments, providing a framework for understanding how to make a series of decisions to optimize certain objectives over time. Due to characteristics, DBRP can be represented as a sequential decision-making problem [89], where the agents, represented by the vehicles, interact within

a complex environment, including a network of stations and the stochastic behavior of users. At each decision epoch, the vehicles make rebalancing decisions regarding the system status, with the ultimate goal of optimizing the availability of bikes across the network and catering to user demand effectively. Despite its relevance and potential for application in DBRP, the adoption of MDP in this domain has been relatively recent, with a limited but growing body of work in academic research.

[12] conceptualized DBRP as an MDP and introduced a dynamic lookahead policy heuristic using online simulations, albeit limited to scenarios involving a single rebalancing vehicle. Building upon this, [13] extended the model to incorporate multiple vehicles, proposing a coordinated lookahead policy to simultaneously address inventory and routing decisions. [61] applied MDP to develop the decision-support tool for DBRP, aiming to minimize the rate of arrival of unsatisfied users who find their station empty or full. An approximation method, one-step policy improvement, was applied to determine which station should be prioritized. However, their approach segments the planning horizon into periods of equal length, which is similar to the majority of MIP models.

Multi-agent Markov Decision Process (MMDP) is an extension of the traditional MDP, where multiple decision-making agents operate concurrently. Within an MMDP framework, each agent aims to determine its optimal strategy, while considering both its actions and those of other agents. As multi-agent systems become more prevalent in various domains such as robotics, transportation, and gaming, the importance of developing effective and scalable algorithms for MMDPs continues to grow [104].

While MDP provides a powerful framework for addressing DBRP, its application in this domain is still evolving. The existing literature demonstrates the MDP potential for rebalancing strategies and highlights the need for further research to overcome challenges related to computational complexity, the choice of time discretization, and vehicle synchronization for more accurate and efficient decision-making.

5.2.3 Reinforcement Learning

RL has witnessed profound advancements in the realm of multi-agent systems, offering promising avenues for algorithmically learning effective decision-making strategies ([75]). This is particularly relevant for the MDP complexities encountered in DBRP, where the need for intelligent, adaptive solutions is paramount.

Rebalancing shares many environmental characteristics with gaming, finance, and marketing, where RL is typically applied, as highlighted by [101]. Meanwhile, the success of RL on

similar problems summarized in [9] motivates the use of RL in BSSs. Applying RL to DBRP could potentially unlock more adaptive solutions, allowing the system to learn from past experiences and adjust its strategies dynamically in response to changing environment, given the stochastic nature of user demand in BSSs.

The majority of the studies about RL is focusing on user-based rebalancing. [78] presented a RL algorithm based on deep deterministic policy gradient algorithm to offer users monetary incentives and alternative pick-up/drop-off stations to rebalance the system. Similarly, [29] adapted a deep reinforcement learning framework for an MDP model to learn the differentiated incentive price for rebalancing bikes. [49] proposed a dynamic incentivization system based on RL and compared three policy-gradient based RL methods. [87] extended the study by [49] and applied it to a more realistic environment. However, users' behaviors are always uncertain and rebalancing by users can be less efficient without the intervention of operators' rebalancing.

In terms of operator-based rebalancing, [64] proposed a spatio-temporal RL framework to tackle the rebalancing problem. They first proposed a clustering algorithm to group the entire bike-sharing system. Then, they designed a deep Neural Network (NN) to estimate the optimal long-term value function to obtain the reposition strategies. [101] proposed a distributed RL solution with transfer learning to make decisions about how many bikes should be moved for each station, where a station-agent pair for each station is created. Each agent finds the optimal solution for each station and aggregates subsequently. The model avoids an exponentially expanding action space but requires a significant processing power to support the parallel learning of all agents. The routing part is not included and needs to further consolidate the rebalancing strategies for the system. [72] designed a PFA algorithm for an MDP model of dynamic rebalancing with one single vehicle. [62] proposed a static rebalancing method using a policy gradient-based RL method to enhance user experience and reduce operational costs. [106] proposed a Deep Reinforcement Learning (DRL) based rebalancing model and produced rebalancing solutions with DQN under discrete time slots.

[72] and [90] focused on the DBRP with one single vehicle, while [64] considered smaller clusters with less than 30 stations with multiple vehicles. The Paper [106] is close to our study, yet a simulator within time discretization was employed, ignoring the order of rentals, returns, and rebalancing operations. We therefore formulate operator-based DBRP with multiple vehicles as an MDP within continuous time framework in Section 5.3 and develop a RL algorithm with a fine-grained simulator to learn reblancing policies in Section 5.4.

5.3 Multi-agent Dynamic Rebalancing System

In this section, we formulate DBRP as an MMDP, as well as our continuous time framework.

5.3.1 Problem Definition

We consider a BSS with a set of stations N and a fleet of vehicles V to rebalance the bikes among the stations. Each station $n \in N$ has a capacity C_n and each vehicle $v \in V$ has a capacity $\hat{C}_v$. The initial state of the system is defined by the bike inventory d_0^n at each station and the inventory p_0^v and location z_0^v of each vehicle v. The distance and transit time between any two stations i and j are $D_{i,j}$ and $R_{i,j}$, respectively. The loading/unloading time per bike is represented by β. The system encounters lost demand when rental or return requests cannot be satisfied due to the absence of bikes or available docks, respectively. The primary goal is to optimize the rebalancing strategies for vehicles across the station network, minimizing lost demand. The parameters are summarized in Table 5.1.

Table 5.1 Notation of BSSs and rebalancing system

Parameters	Definition
N	The set of stations
V	The set of vehicles
C_n	The capacity of station $n \in N$
$\hat{C}_v$	The capacity of vehicle $v \in V$
$D_{i,j}$	The distance between station $i \in N$ and $j \in N$
$R_{i,j}$	The transit time between station $i \in N$ and $j \in N$
β	The time for loading/unloading a bike
d_0^n	The initial number of bikes in station $n \in N$
p_0^v	The initial number of bikes in vehicle $v \in V$
z_0^v	The initial location (station) of each vehicle $v \in V$

5.3.2 Multi-agent Markov Decision Process Model

DBRP can be formulated as an MDP, and more specifically as an MMDP to accommodate multiple rebalancing vehicles within the system. An MMDP is characterized by the tuple $(\boldsymbol{S}, \boldsymbol{A}, W, R, \gamma)$, as shown in Table 5.2. In an MMDP, a sequence of decision epochs occurs with the changes of system states. At each decision epoch, an action is made and leads to a new state. In the following sections, we provide a detailed elaboration of the critical components of our MMDP model.

State Space

At each decision epoch $k \in K$, there is an associated state $\boldsymbol{S_k} \in \boldsymbol{S}$ including comprehensive information about time, stations, and vehicles. We denote $\boldsymbol{S_k} = (t_k, \boldsymbol{d_k}, \boldsymbol{H_k})$, where t_k is the current time and $\boldsymbol{d_k} = (d_k^n, \forall n \in N)$ represents the inventory at each station. The vehicle information is denoted as $\boldsymbol{H_k} = (b_k^v, g_k^v, p_k^v, m_k^v, o_k^v, \forall v \in V)$, where b_k^v is the current station that vehicle v is or has just departed from and g_k^v is the next station that the vehicle v is traveling to, p_k^v indicates its current inventory, and m_k^v indicates the estimated time of arrival of vehicle v at the station g_k^v. Note that if vehicle v is still rebalancing at station b_k^v, the number of remaining operations for that station is stored in o_k^v. To facilitate clarity and ease of reference, the notations used in the model are comprehensively listed and described in Table 5.2.

Table 5.2 Notation of MMDP model for rebalancing

Symbol	Definition
K	The sequence of decision epochs
$\boldsymbol{S}$	The set of states
T	The set of time at each decision epoch, $T = \{t_1, ..., t_{\|K\|}\}$
$\boldsymbol{d_k}$	The number of bikes available at stations, $\boldsymbol{d_k} = (d_k^n, \forall n \in N)$
b_k^v	The station that vehicle $v \in V$ is or just departed from at time t_k
g_k^v	The station that vehicle $v \in V$ is heading to at time t_k
p_k^v	The inventory of vehicle $v \in V$ at time t_k
m_k^v	The estimated arrival time of vehicle v at the station g_k^v at time t_k
o_k^v	Remaining rebalancing operations for vehicle $v \in V$ at time t_k
$\boldsymbol{A}$	The set of actions
l_k^v	The rebalancing decision at the current station for vehicle $v \in V$ at time t_k
z_k^v	The routing decision for vehicle $v \in V$ at time t_k
Π	The set of policies
γ	Discount factor

Action Space

An action is selected at each decision epoch $k \in K$, which indicates both rebalancing and routing decisions. Let $\boldsymbol{A}$ denote the set of actions and $\boldsymbol{a_k} \in \boldsymbol{A}$ is represented by (l_k^v, z_k^v). The rebalancing decision l_k^v is the number of bikes to be rebalanced at the current station and the routing decision z_k^v is the next station to visit for the vehicle $v \in V$. A positive l_k^v indicates that vehicle v should pick up l_k^v bikes while a negative l_k^v implies that vehicle v should drop off $|l_k^v|$ bikes. Note that here DBRP is formulated to accommodate various operational time frameworks, where an action can be made for all the vehicles $v \in V$ at the same time (normally with time discretization) or for only one specific vehicle. In our continuous time framework (see Section 5.3.3), one action at decision epoch $k \in K$ is exclusively associated with a specific

vehicle upon its arrival at the allocated station. Distinctly, the particular vehicle triggering a decision epoch may vary, which ensures that decision-making is a dynamic and ongoing process, responsive to global system status.

Transition Function

When a decision is made for state $\boldsymbol{S_k}$ at each decision epoch $k \in K$, the transition to the next state $\boldsymbol{S_{k+1}}$ is triggered. $\boldsymbol{S_{k+1}}$ is achieved based on the current state $\boldsymbol{S_k}$, decision $\mathbf{a}_k$, and a stochastic transition function W_{k+1}. In other words, W_{k+1} reveals the probability that an action $\mathbf{a}_k$ is taken under $\boldsymbol{S_k}$ and transit to $\boldsymbol{S_{k+1}}$. The transition function contains the users' demand information where uncertainty occurs. We apply a simulator to represent the transition between two successive states under different demand scenarios, which will be presented in Section 5.4.2.

Reward and Objective Function

At each decision epoch $k \in K$, an action $\boldsymbol{a_k}$ is taken and the system transits to $\boldsymbol{S_{k+1}}$. Accompanying this transition is the realization of an immediate reward r_{k+1}, which is computed by $S_k, \boldsymbol{a_k}, W_{k+1}$ and is the negative value of the lost demand across all stations within the interval $[t_k, t_{k+1}]$.

The expected discounted return is $R_k = \mathbb{E}[\sum_{j=0}^{K-k-1} \gamma^j r_{k+j+1}]$, indicating the reward accumulated by the actions in the long run. The discount factor $\gamma \in [0, 1]$ defines how important the rewards are from the uncertain future. A solution is a policy $\pi \in \Pi$ that maps the current state to an action, whose overall goal is to find the optimal policy maximizing the discounted return.

5.3.3 Continuous Time Framework for MMDP

Based on the above notation, our model framework is built in a multi-agent way to realize fully cooperative rebalancing decision-making tasks. Each vehicle is regarded as an agent and the decisions for the vehicles will affect each other. We present our framework as illustrated in Figure 5.1.

This framework allows for a dynamic and immediate response to the current state $\boldsymbol{S_k}$. Specifically, as soon as a vehicle reaches its next designated station, a rebalancing action is instantly generated and executed for that particular vehicle based on the state of the system. This approach eliminates the need for any waiting time for the vehicles, even when other vehicles are simultaneously in operation.

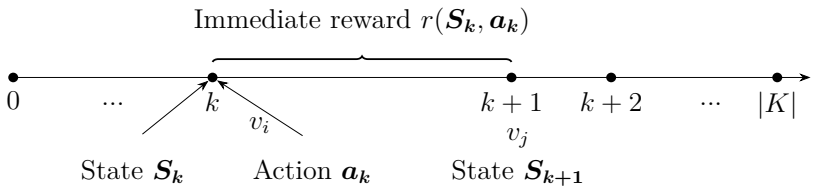

Figure 5.1 Continuous time framework of MMDP

When vehicle $v_i \in V$ arrives in the station $g_k^{v_i}$, the other vehicles are still rebalancing bikes at their current stations or are on the way to their next stations. As Figure 5.1 shows, the arrival of vehicle v_i triggers the decision epoch k and an action $\mathbf{a}_k = (l_k^{v_i}, z_k^{v_i})$ is generated for v_i. Completing action $\mathbf{a}_k$, the system transits to the next state $\boldsymbol{S_{k+1}}$, and an immediate reward is obtained. In our case, the immediate reward is the negative value of lost demand (rentals and returns) in the time segment $[t_k, t_{k+1}]$. The duration between two decision epochs depends on the global system changes.

The corresponding action-value function (Q-function) is the expected return of a state-action pair given the policy π, which is denoted as (5.1), the optimal Q-function is as (5.2), namely

$$Q^{\pi}(\boldsymbol{S_k}, \boldsymbol{a_k}) = \mathbb{E}[R_k | \boldsymbol{S_k}, \boldsymbol{a_k}] \tag{5.1}$$

$$Q^{*}(\boldsymbol{S_k}, \boldsymbol{a_k}) = \max_{\pi} Q^{\pi}(\boldsymbol{S_k}, \boldsymbol{a_k}). \tag{5.2}$$

The optimal greedy policy can be inferred by (5.3) via maximum Q-value, namely.

$$\boldsymbol{a_k^*} = \arg\max_{\boldsymbol{a_k}} Q^{*}(\boldsymbol{S_k}, \boldsymbol{a_k}). \tag{5.3}$$

The decision-making process focuses solely on the action for a single vehicle. As a result, the complexity of the action space is significantly reduced compared to a scenario where actions for all vehicles would need to be determined simultaneously. Additionally, it reflects realistic operational conditions. It is crucial to note that although there is no direct communication or coordination between the vehicles, the status of all vehicles is within the current state, facilitating an indirect form of situational awareness. In other words, each vehicle operates autonomously, making decisions based solely on the global state of the system (i.e., knowing the status of all the other vehicles) as opposed to engaging in collaborative strategies with the other vehicles.

5.4 MARL Methodology with Deep Q-Network

Multi-Agent Reinforcement Learning (MARL) is a research field aiming to find high-quality solutions for multiple agents interacting with each other. In this section, we present our DQN framework to solve the MMDP model for DBRP. The pipeline is shown in Figure 5.2.

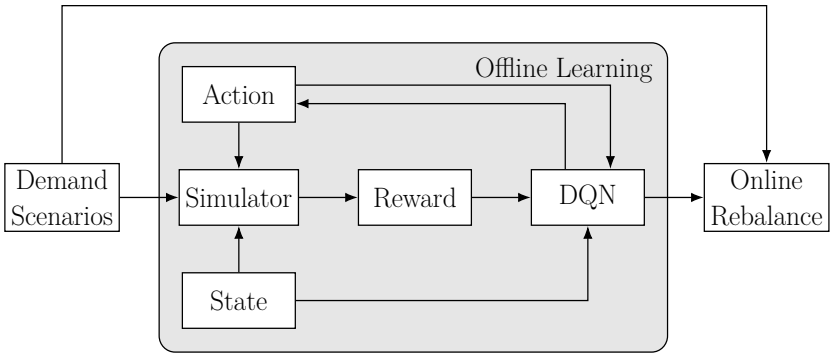

Figure 5.2 Dynamic rebalancing pipeline

In the framework depicted in Figure 5.2, we design a fine-grained *simulator* as an interactive environment, where rebalancing, rentals, and returns are executed under the first-arrive-first-serve rule. The rebalancing operations are decided by the *action* that the agents take and the rentals and returns are obtained under various *demand scenarios*. Our simulator allows for computing immediate rewards between two consecutive decision epochs as realistically as possible. The *reward*, along with the system *state* and the action, are then collected for training our *DQN*. Concurrently, the simulator updates the system state that is then fed back to the agents, forming a critical feedback loop through their interaction with the environment. The DQN is trained through *offline learning*, utilizing a training set responsive to weather and temporal information. Once the offline learning is complete, the refined rebalancing policy encapsulated within the trained DQN is applied in the *online rebalance* phase. This phase involves deploying the policy on a test set, enabling us to assess its performance.

5.4.1 State and Action Space Design

Building on Sections 5.3.3, we formulate our model within a multi-agent framework. When a vehicle arrives at a new station, it immediately triggers a rebalancing action for it based on the current state, often referred to as the observation, without waiting the completion of rebalancing from other vehicles. We here demonstrate the details and structure of state and action space adapted for our DQN MARL algorithm.

At each decision epoch, the current state encapsulates station inventory, current time, and

the status of all vehicles. The station inventory feature gathers the inventory information of all the stations and is quantified as a vector where the size $|N|$ corresponds to the total number of stations, and each element within this vector reflects the current inventory of each station. The current time is denoted as a singular value, encapsulating the precise timestamp of the current state.

The status of the vehicle fleet details the current station, next station, inventory, estimated time of arrival, and remaining rebalancing operations for each vehicle. An example is illustrated in Figure 5.3. Note that when a vehicle v arrives at a station, the next station of this vehicle is not determined yet. In our current implementation, we temporarily set the value of this feature to its current station until the routing decision for the next station is made. For vehicles that are still in transit between two stations, we also temporarily fix their current station feature to the station where they left from for convenience. Both the current station and next station for each vehicle are represented as one-hot vectors of length $|N|$ to facilitate the representation of data in a format that NNs can easily process and learn from, where $|N|$ is the total number of stations and a singular '1' indicates the active station of the feature. Conversely, the vehicle inventory, estimated time, and remaining operations are encoded in vectors of length $|V|$, with $|V|$ being the total number of vehicles, displaying the current vehicle inventory, the estimated time of arrivals at the next station, and the remaining rebalancing operations.

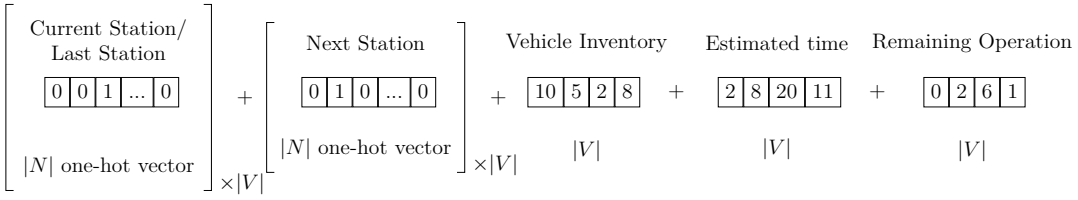

Figure 5.3 An example for an observation of vehicle fleet status

Moreover, forthcoming trip set $I = \{[t_d(i), s_d(i), t_a(i), s_a(i)]\}$ is integrated into the observation in our algorithm, which serves as input to our simulator (see Section 5.4.2). Each trip $i \in I$ contains origin station $s_d(i)$, departure time $t_d(i)$, destination station $s_a(i)$, and arrival time $t_a(i)$.

In terms of action, at decision epoch k, the vehicle will make the decision $\boldsymbol{a_k} = (l_k^v, z_k^v)$ with loading decision l_k^v and route decision z_k^v. To reduce the set of loading decisions, we set three levels of target inventory proportion: $\mu_1 = 10\%$, $\mu_2 = 50\%$, and $\mu_3 = 90\%$. The target inventory levels are: $\mu_1 C_{g_k^v}$, $\mu_2 C_{g_k^v}$, and $\mu_3 C_{g_k^v}$, once the vehicle arrives at station g_k^v. The feasible loading decisions depend also on the vehicle's capacity $\hat{C}_v$, inventory p_k^v, and station

inventory $d_k^{g_k^v}$. It is possible that the station cannot reach the target inventory. Thus, the loading decision variables are

$$(l_k^v)_i = \begin{cases} \min\{\hat{C}_v - p_k^v, d_k^{g_k^v} - \mu_i C_{g_k^v}\} & \text{if} \quad \mu_i C_{g_k^v} < d_k^{g_k^v} \\ \max\{-p_k^v, d_k^{g_k^v} - \mu_i C_{g_k^v}\} & \text{if} \quad \mu_i C_{g_k^v} > d_k^{g_k^v} \\ 0 & \text{otherwise.} \end{cases} \tag{5.4}$$

Recall that $l_k^v < 0$ is for dropping off. The first case in (5.4) is that the vehicle needs to pick up bikes from the station. The case when the vehicle needs to drop off bikes for the station is shown as the second one in (5.4). The third case corresponds to a vehicle arriving at a station whose target inventory level has already been satisfied, thus the vehicle neither picks up nor drops off.

For the route decision variable z_k^v, an agent has the flexibility to choose any station as its destination. However, a unique constraint is applied such that no two vehicles can be directed towards the same station simultaneously.

5.4.2 Simulator

After taking an action $\boldsymbol{a_k}$ under a state $\boldsymbol{S_k}$, an immediate reward (lost demand) is estimated as the negative lost demand in a duration of t_k to t_{k+1}. We here develop a discrete-event simulator as the environment to compute lost demand under the first-arrive-first-serve rule. The pseudo-code for our simulator is given in Algorithm 1. Maximizing the long-term reward is essentially minimizing the total lost demand. Note that the same simulation is also applied to evaluate the policy on the test set.

As vehicle v arrives at station g_k^v at t_k, an action $\boldsymbol{a_k}$ will be decided instantly. The system status, such as station inventory and vehicle fleet information, can be obtained through the current observation. A queue set W is created to record individual upcoming events such as rentals, returns, and rebalancing operations. Each event within set W is detailed by a tuple $[w_t(m), w_s(m), w_i(m), w_v(m)]$, where $w_t(m)$ specifies the time the event is to occur, $w_s(m)$ identifies the involved station, $w_i(m)$ denotes the type of event—whether it is a demand for rental (d), for return (a), for a pick-up (p), or for a drop-off (f)—and $w_v(m)$ associates the event with a particular vehicle, if applicable. Note that the value of $w_v(m)$ has no impact on rental or return events and is set to 0. The events from queue W are sorted by event time $w_t(m)$ in ascending order and executed in sequential order of time, ensuring the first-come-first-serve rule for user demand and interaction of rebalancing operations. Once the completion of an event occurs, the first element in queue W will be pulled out as the next

Algorithm 1: Simulator with a certain action and state

Input : I, $D_{s,s'}$, $R_{s,s'}$, $\hat{C}_v$, C_s, β , $\boldsymbol{a_k}$, $\boldsymbol{S_k}$
Initialization: W; e_t; $reward = 0$; $time = w_t(1)$; $s = w_s(1)$; $indicator = w_i(1)$; $v = w_v(1)$; $i = 1$;
while $time < e_t$ *and* $W \neq \emptyset$ **do**
  $sign = 0$;
  if $indicator = d$ **then**
    if $d_k^s > 0$ **then**
      $d_k^s = d_k^s - 1$; $W = W \cup \{[t_a(i), s_a(i), a, 0]\}$;
    else
      $reward = reward - 1$;
    $i = i + 1$;
  else if $indicator = a$ **then**
    if $C_s - d_k^s > 0$ **then**
      $d_k^s = d_k^s + 1$;
    else
      $reward = reward - 1$; $s' = \arg\min_{d_k^{s'} < C_{s'}} D_{s,s'}$; $d_k^{s'} = d_k^{s'} + 1$;
  else if $indicator = p$ **then**
    if $d_k^s > 0$ *and* $p_k^v < \hat{C}_v$ **then**
      $d_k^s = d_k^s - 1$; $p_k^v = p_k^v + 1$;$o_k^v = o_k^v - 1$;
      if $o_k^v = 0$ **then**
        $sign = 1$;
    else
      Remove w in W whose $w_s = s, w_i = p, w_v = v$; $sign = 1$;
  else
    if $p_k^v > 0$ *and* $d_k^s < C_s$ **then**
      $d_k^s = d_k^s + 1$; $p_k^v = p_k^v - 1$; $o_k^v = o_k^v - 1$;
      if $o_k^v = 0$ **then**
        $sign = 1$;
    else
      Remove w in W whose $w_s = s, w_i = f, w_v = v$; $sign = 1$;
  if $sign = 1$ **then**
    $m_k^v = time + R_{s,g_k^v}$;
    if $m_k^v < e_t$ **then**
      $e_t = m_k^v$; Remove elements in W whose $w_i = d$;
  $W = W \backslash \{[time, s, indicator, v]\}$;
  $time, s, indicator, v = w_t(m), w_s(m), w_i(m), w_v(m)$ $(m = \arg\min w_t(m))$;
end
Update $\boldsymbol{S_{k+1}}$;
Output : $reward$ and $\boldsymbol{S_{k+1}}$

event to be executed by the system.

We first initialize W with all rental demands from I (see Section 5.4.1) and rebalancing operations (pick-up/drop-off) from o_k^v happening in the duration of t_k to e_t, where e_t is the estimated time point for the next arrival of a vehicle (next decision epoch) derived from m_k^v. The corresponding returns of successful rentals are created and added to W in real time.

As in reality, the simulator processes the events in W in chronological order. When a rental demand arises and the station holds at least one available bike, we update the station inventory and add the corresponding return demand to the waiting set W. Otherwise, the customer fails to rent a bike when there is no inventory in the station, resulting in a lost rental demand. When a return demand occurs but the station has no available docks, we assume that the customer returns the bike at the nearest station with available docks immediately. However, a lost return demand will be counted. The station inventories are updated accordingly. For pick-up/drop-off rebalancing operation events, we verify whether sufficient bikes/docks at the station and space/bikes within the vehicle are available. When a vehicle completes its rebalancing task or lacks the necessary resources to continue, it departs for the next station. Then, the estimated arrival time at the next station for this vehicle is calculated. If this estimated arrival is sooner than the current estimated time point of next decision epoch e_t, we update e_t accordingly. Ultimately, e_t stands as the time when the next decision epoch will occur.

During the simulation, the state is continuously updated in real time. It culminates in the generation of the succeeding state $\boldsymbol{S_{k+1}}$ and an immediate reward, quantified as the negative of the total lost demand until the next decision epoch.

5.4.3 Policy Evaluation and Network

Q-learning is a value-based off-policy Temporal-Difference (TD) RL method. In the realm of value-based RL, the central objective is to learn a value function that predicts the expected return of taking a specific action in a particular state, guiding the agent towards the most rewarding outcomes. Specifically, with a Q-value function, $Q(\boldsymbol{S}, \boldsymbol{a})$ indicates the return after taking an action $\boldsymbol{a}$ in a given state $\boldsymbol{S}$, and a policy π^* that maximizes reward accordingly to (5.3) can be obtained. The optimal Q-function obeys the following Bellman equation [7]:

$$Q^*(\boldsymbol{S_k}, \boldsymbol{a_k}) = \mathbb{E}[r_{k+1} + \gamma \max_{\boldsymbol{a_{k+1}}} Q^*(\boldsymbol{S_{k+1}}, \boldsymbol{a_{k+1}})], \tag{5.5}$$

where $\boldsymbol{S_k}$ and $\boldsymbol{a_k}$ are the current state and action, $\boldsymbol{S_{k+1}}$ and r_k are the next state and immediate reward after taking action $\boldsymbol{a_k}$, and $\boldsymbol{a_{k+1}}$ is the action that achieves maximal Q-

value on state $\boldsymbol{S_{k+1}}$. The expectation is computed over the distribution of immediate reward r_k and next state $\boldsymbol{a_{k+1}}$.

TD error is the discrepancy between the currently estimated Q-value for a given state-action pair and the new estimate based on the observed reward and the best possible future Q-value, namely

$$\delta = r_{k+1} + \gamma \max_{\boldsymbol{a_{k+1}}} Q(\boldsymbol{S_{k+1}}, \boldsymbol{a_{k+1}}) - Q(\boldsymbol{S_k}, \boldsymbol{a_k}). \tag{5.6}$$

The basic idea of Q-learning is to update the Q-value function by minimizing the TD error. The key updating rule is

$$Q(\boldsymbol{S_k}, \boldsymbol{a_k}) \leftarrow Q(\boldsymbol{S_k}, \boldsymbol{a_k}) + \alpha(r_{k+1} + \gamma \max_{\boldsymbol{a}_{k+1}} Q(\boldsymbol{S_{k+1}}, \boldsymbol{a_{k+1}}) - Q(\boldsymbol{S_k}, \boldsymbol{a_k})), \tag{5.7}$$

where α denotes the learning rate.

The Q-learning update rule exemplifies the TD approach. It adjusts the value (Q-value) of a state-action pair based on the actual reward received and the projected value of the next state. This rule essentially allows the agent to forecast future rewards and use this foresight to make informed decisions in the present. Being off-policy, Q-learning enables the agent to gain insights from exploratory actions, which may not immediately appear optimal, thereby enriching the learning process and allowing the policy to evolve beyond the limits of the agent's existing strategy.

DQN is an advancement in RL that combines Q-learning with deep NNs. In DQN, two neural networks are employed: the prediction network and the target network. The prediction network is responsible for approximating the Q-value of any state-action pair, and it is updated iteratively to reduce the discrepancy between predicted Q-values and target Q-values. The target network plays a pivotal role in ensuring the stability of the learning process by supplying target Q-values for the updates of the prediction network. Specifically, the target Q-value for a state-action pair is given by $r + \gamma \max_{a'} Q(S', a'; \theta^{target})$, where S' is the subsequent state. The loss function is given by the TD error of the predicted Q-value and the target Q-value

$$L(\theta^{pred}) = \mathbb{E}[((r_k + \gamma \max_{\boldsymbol{a}} Q(\boldsymbol{S_{k+1}}, \boldsymbol{a}; \theta^{target})) - Q(\boldsymbol{S_k}, \boldsymbol{a_k}; \theta^{pred}))^2], \tag{5.8}$$

where θ^{target} and θ^{pred} are the parameters of the target network and prediction network, respectively.

During the training phase, the agents explore the state-action space by using an epsilon-

greedy strategy: Initially, a high epsilon value is set, meaning the agent is more likely to take a random action, ensuring sufficient exploration. As training progresses, epsilon is gradually decreased, shifting the agent's behavior from exploration to exploitation, where the action that maximizes the Q-value from the prediction network has a higher probability to be selected. As the agent interacts with the environment, the accumulated experiences, consisting of state, action, reward, and next state, are stored in a replay buffer. These experiences are then randomly sampled in mini-batches for training the target network. At each iteration, a mini-batch of training data is sampled from the replay buffer and the training loss is computed by (5.8). The parameters of the prediction network are updated via gradient descent to minimize the loss, while the target network's parameters are updated less often, usually by directly copying the prediction network's weights. This delayed update mechanism for the target network ensures that the learning process remains stable and prevents divergence.

We employ the same NN architecture for the two Q-value networks: first, an input layer matches the dimensions of the observation. Then, we have two fully connected dense layers followed by a ReLU (Rectified Linear Unit) activation function. Finally, an output layer corresponding to the number of actions allows the network to output a Q-value for each possible action based on a given state.

5.5 Experiments and Results

In this section, we present the results of our computational investigation. In Section 5.5.1, we first introduce the dataset and the benchmarks. Then, in Section 5.5.2, we demonstrate the training process for DQN. Finally, we compare the evaluation on the test set across our algorithm and benchmarks.

5.5.1 Dataset and Baselines

Dataset. Although real-world trip data is generally available, we opt for synthetic instance generation due to several considerations. The existing real-world data does not account for unobserved demand and may contain inconsistencies or noise. Moreover, the rebalancing operation carried out by operators is not available, which highly affects station inventory. Consequently, we utilize the instance generator introduced in Chapter 4 that offers trip data influenced by both weather patterns and temporal aspects. The generator first generates synthetic weather based on real-world weather data distributions. Subsequently, hourly rentals are predicted using a linear regression model, trained on real-world weather and trip data. Finally, the trip data is produced based on these hourly rentals, in conjunction with trip

distributions and considered station networks (station locations and capacities). In this case, the trip data is diverse and facilitates DQN to learn a rebalancing policy that accounts for various demand scenarios.

The synthetic dataset is generated based on the weather and temporal data sourced directly from the official Government of Canada website[1], which details temperature, humidity, day, hour, year, and weekday. The real-world trips and station information, that we used as input to the generator, are from BIXI, a BSS in Montreal. We only focus on weekdays from May to September 2019 to ensure the data remains uninfluenced by the COVID-19 pandemic and is consistent. Details can be found in Chapter 4. We generate 150 days, each of which contains detailed trips (origin station, departure time, destination station, arrival time). The first 100 days in the dataset are used for training. The remaining 50 days are used for evaluation as a test set. The station network contains 60 stations with 9 city-center stations. The stations within city centers are equipped with 40 docks, while those outside city centers have 20 docks each.

Four vehicles are responsible for rebalancing the stations, each with a capacity of 40 bikes. The initial inventory of stations is obtained by solving a static rebalancing problem from [65], where the inventories for the first time-period are decision variables that sum to the total number of available bikes in the system.

Baselines. We benchmark our algorithm with the following two baselines.

- **Static Rebalancing:** Static rebalancing only happens during the night and the goal is to set the inventories for the beginning of the next day (7 a.m. in our case). The solution is obtained by solving a MIP model from [65]. Then, we simulate all the days from our test set in the simulator without any bike rebalancing and the average lost demand is reported.
- **Dynamic Rebalancing:** We apply the basic multi-period dynamic rebalancing MIP model from [65] with a 30-minute length of time-periods. The average number of rentals and returns computed from training set serves as input for the MIP model. The produced rebalancing strategy indicates how many bikes to pick up or drop off at which station by each vehicle for each time-period. The performance of the rebalancing strategy is also evaluated on the test set in our simulator under the first-arrive-first-serve rule.

[1] https://climate.weather.gc.ca

5.5.2 Experimental Results

We consider a planning horizon of 4 hours from 7 a.m to 11 a.m, which is also considered as an episode in DQN. The training phase in DQN is performed on a single GPU Tesla V100-PCIE-32GB.

DQN Training Process

We first investigate the performance of different variants utilizing distinct activation functions for the output layer. The DQN without activation function is denoted as DQN. Leaky ReLU, Parameterized ReLU (PReLU), and Exponential Linear Unit (ELU) [80] are applied. We report episodic return, i.e., the cumulative reward obtained in an episode, and Q-value during the training process in Figure 5.4. Each step represents one episode which is 4 hours (7 a.m to 11 a.m) in our case. The parameters can be found in Table 5.3.

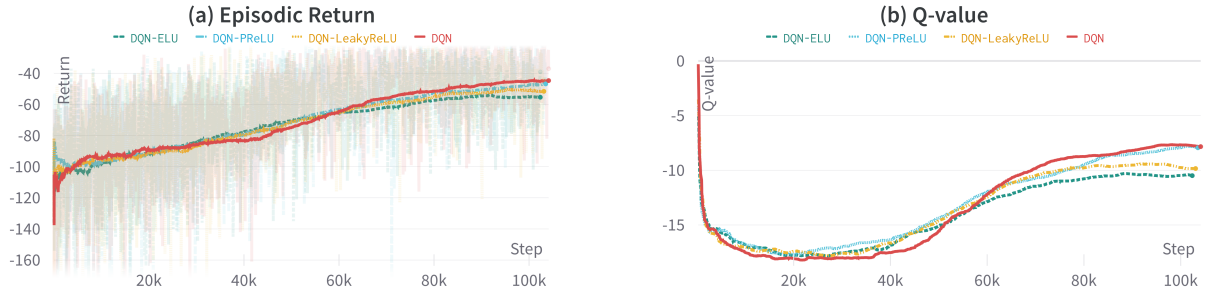

Figure 5.4 Episodic return and Q-value of DQN variants

Table 5.3 Parameters in training process

Parameters	Values
Total time step	3000000
Learning rate	2.5e-4
Buffer size	10000
γ	0.99
Batch size	256
Exploration rate ϵ	$1 \rightarrow 0.05$
Exploration fraction	0.5
1st layer neuron	1024
2nd layer neuron	512

The results in Figure 5.4 show that all variants converge over time, both in terms of episodic return and Q-value, indicating stable learning dynamics. The episodic return indicates the cumulative reward obtained in an episode and shows an increasing trend throughout the training.

Among the tested variants, standard DQN stands out slightly in terms of Q-value, suggesting it is likely to be a bit more effective or optimistic. DQN-LeakyReLU, while still showing a positive trend, lags a bit behind its competitors, particularly in episodic return.

We then report more metrics for the best performed model, namely DQN, in Figure 5.5 and Figure 5.6. The agents in our algorithm appear to be effectively learning the dynamic rebalancing task in BSSs. This is evidenced by the rising Q-values, reducing TD loss, see Figure 5.5, and improving episodic returns, see Figure 5.6.

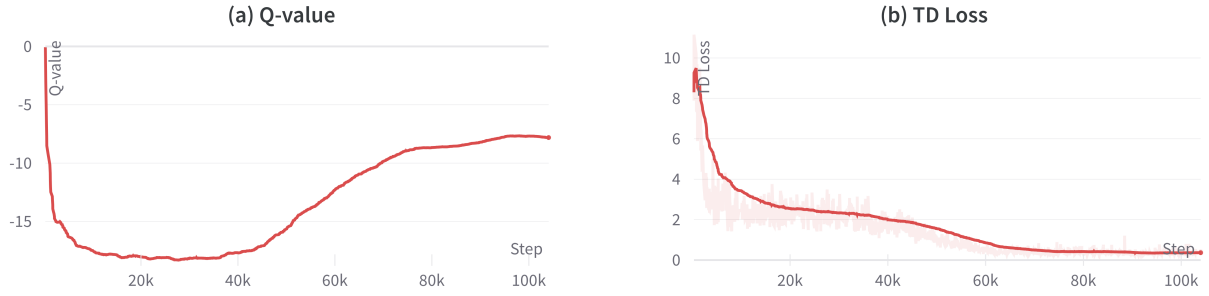

Figure 5.5 Q-value and TD loss for DQN

As in Figure 5.5(a), the Q-value decreases first when agents begin to explore the state-action space and encounters negative rewards, which is typically characterized by exploration, where the agent is still learning about the environment and the consequences of its actions. Then the agents start to learn from the experiences and the Q-value increases steadily, leading to a better rebalancing policy. The function approximation of the policy's Q-value also improves over time as the agent learns to better predict expected returns from its actions. TD loss in Figure 5.5(b) decreases sharply in the initial stages of training and then begins to stabilize towards the end, indicating the agents are becoming more accurate in predicting Q-values as training progresses, thereby reducing the TD error.

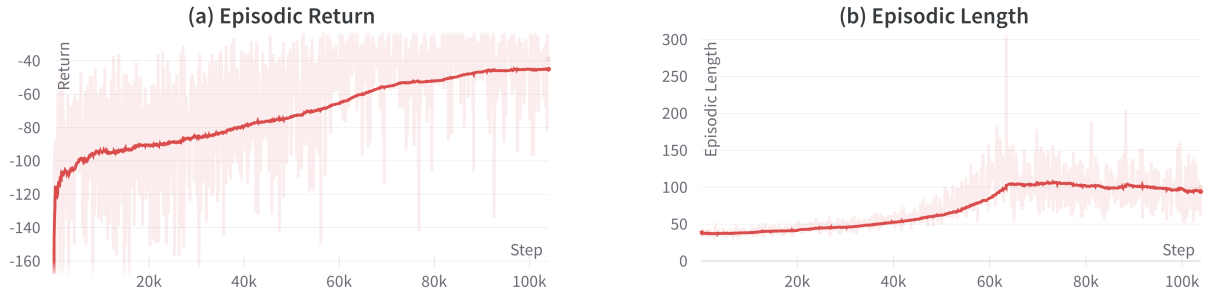

Figure 5.6 Episodic return and episodic length for DQN

The episodic return in Figure 5.6(a) suggests that the agent's policy is becoming more effec-

tive over time at achieving higher rewards in the rebalancing task. The episodic length reflects the number of stations that all the agents (4 vehicles) manage to visit within one episode, increasing from approximately 50 to 100, as shown in Figure 5.6(b). Note that the vehicles start from city-center stations, they are likely to rebalance within neighboring city-center stations to address the fluctuating and typically elevated demand patterns characteristic of such regions. This increasing trend indicates that the agents learn to optimize rebalancing operations by covering stations within the same operation to reduce lost demand.

The metrics illustrated offer a valuable perspective on the models' performance throughout the training phase, capturing the progression and refinement of the learning process. However, it is crucial to understand the trained models' capabilities on the test set. Therefore, the subsequent section reports the evaluation results on the test set, which is important to show practical efficacy and robustness when faced with new and potentially more complex demand scenarios.

Evaluation

Table 5.4 depicts the average episodic lost demand (lost rental and lost return) of our algorithm and benchmarks on the test set. The training time for DQN and its variants is approximately 14 hours. We report two distinct scenarios on DQN, each with different values of ϵ. Unlike the epsilon-greedy strategy used during the training phase, here, ϵ is set to a constant value, dictating the probability that a random action is chosen over the predicted best action. When $\epsilon = 0$, the action yielding the maximum Q-value from the trained network is consistently selected. Given that the static rebalancing and the dynamic MIP model do not have a stochastic strategy, their performance are showcased only under the $\epsilon = 0$ condition.

From Table 5.4, the static rebalancing displays the highest lost demand of 112.82 compared to the dynamic models. This empirically demonstrates the natural advantage of dynamic rebalancing strategies in reducing lost demand, as they adjust based on the real-time state of the environment.

Table 5.4 Lost demand on test set

Model	Lost Demand	
	$\epsilon = 0$	$\epsilon = 0.05$
Static Rebalancing	112.82	-
Dynamic MIP Model	44.76	-
DQN-ELU	50.13	51.13
DQN-PReLU	36.58	44.21
DQN-LeakyReLU	44.53	48.27
DQN	35.12	42.69

The DQN showcases the best performance with a lost demand of 35.12 when $\epsilon = 0$. The DQN-PReLU and DQN-LeakyReLU also show competitive results, potentially offering superior learning dynamics during the training phase. Introducing randomness in action selection through $\epsilon = 0.05$ results in an increase in lost demand across all DQN models. In this case, the agents are no longer solely exploiting their learned knowledge but are also exploring potentially suboptimal actions to discover new strategies. Notably, most of the DQN models outperform the static rebalancing and dynamic MIP model even when exploration is introduced, which underscores the DQN models' superior ability to generalize and adapt to the test environment despite the introduction of randomness in the decision-making process.

Almost all the variants of DQN consistently outperform the Dynamic MIP Model, highlighting underscoring the robustness and capability of RL. Particularly, DQN reduces lost demand by 21.5% compared to the Dynamic MIP model. Since the episode is divided into 8 time-periods in MIP model and each vehicle can only visit one station per time-period, the maximum number of rebalancing stations is 32 with 4 vehicles. However, according to Figure 5.6(b), the number of rebalancing stations can be more than 50. The continuous time framework enables the vehicle fleet to rebalance stations more frequently and efficiently since the vehicles do not have to wait for each other.

5.6 Conclusions

In this chapter, we developed a spatio-temporal RL algorithm for DBRP with multiple vehicles. We first formulated the problem as an MDP under a continuous time framework, allowing vehicles to independently and cooperatively rebalance stations without the need to wait for each other, which enhances the realism and efficiency of the rebalancing process. We then developed an advanced simulator, following the first-arrive-first-serve rule, to calculate rewards between two successive states across varied demand scenarios. To solve the MDP model for DBRP, we proposed a DQN framework to learn effective rebalancing strategies with the aim of minimizing lost demand. Our method was evaluated with a diverse trip dataset responsive to weather conditions and temporal factors. The results have shown that our algorithm consistently outperforms the benchmarks, including static rebalancing and a traditional MIP model, by reducing the lost demand up to 21.53% compared to the MIP baseline.

This study shows the potential of RL in DBRP, outlining various intriguing avenues for future research. Although our method showcases computational efficiency within a 60-station system, it may be extended to a larger system directly or through the integration of clustering methods, allowing the algorithm to be applied to large-scale BSSs effectively. The extension

of our method could also involve additional state parameters such as vehicle initial locations, offering a more granular and responsive approach to rebalancing.

CHAPTER 6 GENERAL DISCUSSION

The objective of this thesis is to offer robust decision support for the DBRP in BSSs, aiming to enhance operational efficiency and elevate user satisfaction. The research unfolded across three projects, employing a mix of MIP, discrete-event simulation, and RL, each targeting specific facets of dynamic rebalancing problems.

In the realm of DBRP in BSSs, a myriad of diverse MIP models have been proposed with various modeling assumptions and techniques. Chapter 3 aims at disentangling the literature and providing an empirical comparison of the impact of the major existing assumptions and alternatives. We proposed a general multi-period MIP modeling framework tailored for accommodating a range of modeling assumptions with a fine-grained simulator to assess the performance. To provide a comprehensive and unbiased comparison, the most prevalent objective function of minimizing the lost demand is used and the optimization models are executed in static planning, i.e., one-time optimization. We introduced a generator capable of creating diverse station networks and customer trip scenarios to provide a variety of instances. The historical average of the generated demand serves as input in our proposed optimization model. Extensive experiments are conducted to dissect the effectiveness of various modeling assumptions and techniques.

Chapter 4 advances the foundational MIP model established in Chapter 3 by introducing a trip prediction derived from a ML algorithm that considers weather conditions and temporal features, as opposed to relying on a naive historical average. Two pivotal concepts used by BSS operators, inventory intervals and target inventories, are computed based on trip prediction and integrated into the objective functions of the MIP model. By shifting the focus from minimizing lost demand through a single point estimate to minimizing deviations from inventory interval and target, the model gained more robustness, showing an improved capacity to accommodate demand fluctuations and yield resilient planning solutions. Two reoptimization modes, rolling and folding, are finally employed to investigate the benefits of updating system status throughout the rebalancing process, compared to the static planning in Chapter 3.

The MIP models deployed in Chapter 3 and Chapter 4 discretize the planning horizon into multiple time-periods, a common approach in the literature. Because this discretization introduces challenges in achieving optimal vehicle coordination, Chapter 5 addresses the DBRP from a different perspective by modeling it as an MDP, designing state and action spaces that represent the system dynamics and inherent uncertainties. We proposed a spatio-temporal

RL-based algorithm within a continuous time framework, enabling vehicles to conduct rebalancing at stations both individually and collectively. Acknowledging the stochasticity of demand, a simulator adapted from Chapter 3 and Chapter 4 under the first-arrive-first-serve rule is applied to estimate the rewards under different demand scenarios. DQN is employed to estimate the Q-value and deduce the optimal policy, aiming to minimize the total lost demand. A series of experiments is implemented on a dataset sensitive to weather fluctuations, ensuring comprehensive evaluation and validation of the proposed algorithm.

CHAPTER 7 CONCLUSION

This thesis focuses on decision-support for dynamic rebalancing for BSSs to enhance user satisfaction and operational efficiency. The modeling assumptions and techniques are first studied within a general MIP framework. Then, two novel objective functions are proposed by integrating predicted inventory intervals and target inventories derived from demand prediction. Finally, a RL algorithm is introduced to solve the MDP-based DBRP with multiple vehicles.

7.1 Summary of Works

In Chapter 3, we provided a thorough literature review focusing on multi-period MIP models for DBRP, leading to the development of a versatile modeling framework. We demonstrated the various existing modeling assumptions and the formulations of other alternatives. We evaluated the performance of the planned solutions induced by the different model variants as realistically as possible with a fine-grained discrete-event simulator on a minute-to-minute basis, in order to find the best configurations for decision-makers. Our analysis, conducted over extensive numerical experiments with varying station networks and demand scenarios, concentrated on lost demand and the simulation-optimization-gap as key performance metrics. The results indicated that when computing resources are constrained, a combination of trip distribution constraints with the newly proposed event sequences, such as (d)(a)(r) (departure, arrival, rebalancing), is recommended. In contrast, with ample computing resources, applying trip distribution constraints with shorter time periods emerged as the optimal configuration for both synthetic and real-world data. Notably, for rapid decision-making scenarios with limited resources, event sequences (r)(d)(a) or (d)(a)(r) were highlighted as preferable, with the latter performing exceptionally well on real-world data when combined with partially-integer variable domains and constraints (TD1) (see, Chapter 3).

Chapter 4 introduced two innovative models designed to incorporate inventory intervals and target inventories estimated by demand prediction from the GBT algorithm. These models offer high-quality rebalancing strategies suitable for use by BSSs operators. We assessed these models across static, rolling, and folding reoptimization modes on both synthetic and real-world datasets. The proposed models showcased significant robustness and flexibility, leading to a notable reduction in lost demand across various scenarios. Model DROB-T reduced lost demand by up to 34%, while model DROB-I achieved up to a 28% reduction compared to the baseline. Additionally, the performance of rolling and folding planning

consistently surpassed that of static planning, emphasizing the importance of continuously updating the system status, including station inventory and trip predictions, particularly in response to weather changes. These results highlight the indispensable role of inventory intervals and target inventories, and reoptimization, ensuring a more reliable and effective rebalancing strategy.

Chapter 5 devised a spatio-temporal RL algorithm tailored to address DBRP with multiple trucks in BSSs. We first modeled this problem as an Multi-agent Markov Decision Process (MMDP) within a continuous non-discrete time framework. This framework facilitates the independent and cooperative rebalancing of stations by the vehicles, eliminating the necessity for them to wait for one another and thereby augmenting both the realism and efficiency of the rebalancing process. Then, an event-based simulator was introduced to estimate the rewards between two successive states across a variety of demand scenarios under the first-arrive-first-serve rule. To estimate the value functions and derive effective rebalancing strategies aimed at minimizing lost demand, we applied the DQN with different configurations. Experiments were carried out on a diverse trip dataset responsive to weather conditions and temporal factors. The results demonstrate the superiority of our algorithm over existing benchmarks, including both static rebalancing methods and a traditional MIP model. Notably, our approach achieved a remarkable reduction in lost demand, outperforming the MIP baseline by up to 21.53%.

7.2 Limitations and Future Research

Both Chapter 3 and Chapter 4 focus on MIP models with exact solutions, which inherently restricts the problem's size and scalability, rendering the direct application of these models to large-scale BSSs impractical. In our experiments on real-world data, we circumvented this issue by segmenting the entire network into smaller clusters of manageable size. However, this segmentation highlights the models' limitation in handling extensive, unclustered networks directly. Chapter 5 takes strides towards addressing the DBRP with more trucks and stations compared to the majority of existing literature on MDP in DBRP. Nevertheless, to maintain computational tractability, we conduct our experiments on a 60-station network, indicating a persistent challenge across chapters in scaling our proposed models to large-scale BSSs networks. Future research may explore decomposition methods [74] or heuristic algorithms [59] to overcome these scalability challenges and extend the applicability of our models to larger, more complex BSSs.

Chapter 3 delves into diverse modeling assumptions and techniques from both theoretical and practical viewpoints, showcasing that concepts like event sequences are prevalent in various

classical optimization problems beyond BSSs, such as the Pick-up-and-Delivery Problem. Planning problems in car-sharing and multimodal transportation with synchronization, where interactions among customers and system operators are aggregated into discrete time-periods, display similar event sequences, and warrant exploration.

Chapter 4 presents an innovative approach to DBRP in BSSs, integrating inventory intervals and target inventories within the objective functions, which provide a buffer for station inventory when dealing with demand fluctuations. In practical applications, operators employ these inventory intervals to generate alerts for stations that fall outside of the designated range. Building upon this real-world application, future research could explore incorporating this alert-based strategy directly into our model to address tractability in large-scale systems, offering an alternative to current clustering methods.

Chapter 5 shows the capabilities of RL in DBRP domain, suggesting a plethora of promising directions for forthcoming research. One of the primary areas for further refinement lies in the incorporation of additional state parameters. For instance, including the initial locations of the vehicles as a part of the learning policy significantly enhances the model's responsiveness and adaptability. Additionally, there may be an opportunity to revise the network's architecture, e.g., layers, network types, and activation functions, to accommodate larger BSSs.

REFERENCES

[1] "Bixi open data," 2023. [Online]. Available: https://bixi.com/en/open-data-2/

[2] "A set of bixi stations, google maps, google," 2023, accessed: August 2023.

[3] H. Akova, S. Hulagu, and H. B. Celikoglu, "Static bike repositioning problem with heterogeneous distribution characteristics in bike sharing systems," *Transportation Research Procedia*, vol. 62, pp. 205–212, 2022.

[4] O. Alagoz, H. Hsu, A. J. Schaefer, and M. S. Roberts, "Markov decision processes: a tool for sequential decision making under uncertainty," *Medical Decision Making*, vol. 30, no. 4, pp. 474–483, 2010.

[5] R. Alvarez-Valdes, J. M. Belenguer, E. Benavent, J. D. Bermudez, F. Muñoz, E. Vercher, and F. Verdejo, "Optimizing the level of service quality of a bike-sharing system," *Omega*, vol. 62, pp. 163–175, 2016.

[6] S. M. Arabzad, H. Shirouyehzad, M. Bashiri, R. Tavakkoli-Moghaddam, and E. Najafi, "Rebalancing static bike-sharing systems: a two-period two-commodity multi-depot mathematical model," *Transport*, vol. 33, no. 3, pp. 718–726, 2018.

[7] R. Bellman, "A markovian decision process," *Journal of Mathematics and Mechanics*, pp. 679–684, 1957.

[8] M. Benchimol, P. Benchimol, B. Chappert, A. De La Taille, F. Laroche, F. Meunier, and L. Robinet, "Balancing the stations of a self service "bike hire" system," *RAIRO-Operations Research*, vol. 45, no. 1, pp. 37–61, 2011.

[9] Y. Bengio, A. Lodi, and A. Prouvost, "Machine learning for combinatorial optimization: a methodological tour d'horizon," *European Journal of Operational Research*, vol. 290, no. 2, pp. 405–421, 2021.

[10] P. Borgnat, P. Abry, P. Flandrin, C. Robardet, J.-B. Rouquier, and E. Fleury, "Shared bicycles in a city: A signal processing and data analysis perspective," *Advances in Complex Systems*, vol. 14, no. 03, pp. 415–438, 2011.

[11] N. Boufidis, A. Nikiforiadis, K. Chrysostomou, and G. Aifadopoulou, "Development of a station-level demand prediction and visualization tool to support bike-sharing systems' operators," *Transportation Research Procedia*, vol. 47, pp. 51–58, 2020.

[12] J. Brinkmann, M. W. Ulmer, and D. C. Mattfeld, "Dynamic lookahead policies for stochastic-dynamic inventory routing in bike sharing systems," *Computers & Operations Research*, vol. 106, pp. 260–279, 2019.

[13] ——, "The multi-vehicle stochastic-dynamic inventory routing problem for bike sharing systems," *Business Research*, vol. 13, no. 1, pp. 69–92, 2020.

[14] B. P. Bruck, F. Cruz, M. Iori, and A. Subramanian, "The static bike sharing rebalancing problem with forbidden temporary operations," *Transportation Science*, vol. 53, no. 3, pp. 882–896, 2019.

[15] L. Caggiani and M. Ottomanelli, "A dynamic simulation based model for optimal fleet repositioning in bike-sharing systems," *Procedia-Social and Behavioral Sciences*, vol. 87, pp. 203–210, 2013.

[16] G. C. Calafiore, C. Bongiorno, and A. Rizzo, "A robust mpc approach for the rebalancing of mobility on demand systems," *Control Engineering Practice*, vol. 90, pp. 169–181, 2019.

[17] D. Chemla, F. Meunier, and R. W. Calvo, "Bike sharing systems: Solving the static rebalancing problem," *Discrete Optimization*, vol. 10, no. 2, pp. 120–146, 2013.

[18] D. Chemla, F. Meunier, T. Pradeau, R. Wolfler Calvo, and H. Yahiaoui, "Self-service bike sharing systems: simulation, repositioning, pricing," Mar. 2013, working paper or preprint. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00824078

[19] J. Chen, Z. Yang, P. Cheng, and Y. Shu, "Rebalancing bike-sharing system with deep sequential learning," *IEEE Intelligent Transportation Systems Magazine*, vol. 13, no. 4, pp. 92–98, 2020.

[20] F. Chiariotti, C. Pielli, A. Zanella, and M. Zorzi, "A dynamic approach to rebalancing bike-sharing systems," *Sensors*, vol. 18, no. 2, p. 512, 2018.

[21] ——, "A bike-sharing optimization framework combining dynamic rebalancing and user incentives," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 14, no. 3, pp. 1–30, 2020.

[22] Y. Chumin, O. O'Brien, P. DeMaio, R. Rabello, S. Chou, and T. Benicchio, "The meddin bike-sharing world map report," https://bikesharingworldmap.com/reports/bswm_mid2021report.pdf, 2021, accessed: 2023-08-04.

[23] C. Contardo, C. Morency, and L.-M. Rousseau, *Balancing a dynamic public bike-sharing system*. Cirrelt Montreal, Canada, 2012, vol. 4.

[24] S. Datner, T. Raviv, M. Tzur, and D. Chemla, "Setting inventory levels in a bike sharing network," *Transportation Science*, vol. 53, no. 1, pp. 62–76, 2019.

[25] M. Dell'Amico, E. Hadjicostantinou, M. Iori, and S. Novellani, "The bike sharing rebalancing problem: Mathematical formulations and benchmark instances," *Omega*, vol. 45, pp. 7–19, 2014.

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

[27] L. Di Gaspero, A. Rendl, and T. Urli, "A hybrid aco+ cp for balancing bicycle sharing systems," in *International Workshop on Hybrid Metaheuristics*. Springer, 2013, pp. 198–212.

[28] Y. Du, F. Deng, and F. Liao, "A model framework for discovering the spatio-temporal usage patterns of public free-floating bike-sharing system," *Transportation Research Part C: Emerging Technologies*, vol. 103, pp. 39–55, 2019.

[29] Y. Duan and J. Wu, "Optimizing rebalance scheme for dock-less bike sharing systems with adaptive user incentive," in *2019 20th IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 2019, pp. 176–181.

[30] S. V. E, J. Park, and Y. Cho, "Using data mining techniques for bike sharing demand prediction in metropolitan city," *Computer Communications*, vol. 153, pp. 353–366, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366419318997

[31] W. El-Assi, M. Salah Mahmoud, and K. Nurul Habib, "Effects of built environment and weather on bike sharing demand: a station level analysis of commercial bike sharing in toronto," *Transportation*, vol. 44, no. 3, pp. 589–613, 2017.

[32] G. Erdoğan, G. Laporte, and R. W. Calvo, "The static bicycle relocation problem with demand intervals," *European Journal of Operational Research*, vol. 238, no. 2, pp. 451–457, 2014.

[33] G. Erdoğan, M. Battarra, and R. W. Calvo, "An exact algorithm for the static rebalancing problem arising in bicycle sharing systems," *European Journal of Operational Research*, vol. 245, no. 3, pp. 667–679, 2015.

[34] E. Eren and V. E. Uz, "A review on bike-sharing: The factors affecting bike-sharing demand," *Sustainable Cities and Society*, vol. 54, p. 101882, 2020.

[35] H. M. Espegren, J. Kristianslund, H. Andersson, and K. Fagerholt, "The static bicycle repositioning problem-literature survey and new formulation," in *International Conference on Computational Logistics*. Springer, 2016, pp. 337–351.

[36] S. Feng, H. Chen, C. Du, J. Li, and N. Jing, "A hierarchical demand prediction method with station clustering for bike sharing system," in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*. IEEE, 2018, pp. 829–836.

[37] I. A. Forma, T. Raviv, and M. Tzur, "A 3-step math heuristic for the static repositioning problem in bike-sharing systems," *Transportation Research Part B: Methodological*, vol. 71, pp. 230–247, 2015.

[38] C. Fricker and N. Gast, "Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity," *Euro Journal on Transportation and Logistics*, vol. 5, no. 3, pp. 261–291, 2016.

[39] J. Froehlich, J. Neumann, N. Oliver *et al.*, "Sensing and predicting the pulse of the city through shared bicycling." in *Twenty-first International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 9, no. Jul, 2009, pp. 1420–1426.

[40] C. Fu, N. Zhu, S. Ma, and R. Liu, "A two-stage robust approach to integrated station location and rebalancing vehicle service design in bike-sharing systems," *European Journal of Operational Research*, vol. 298, no. 3, pp. 915–938, 2022.

[41] C. Gallop, C. Tse, and J. Zhao, "A seasonal autoregressive model of vancouver bicycle traffic using weather variables," *i-Manager's Journal on Civil Engineering*, vol. 1, no. 4, p. 9, 2011.

[42] K. Gebhart and R. B. Noland, "The impact of weather conditions on bikeshare trips in washington, dc," *Transportation*, vol. 41, no. 6, pp. 1205–1225, 2014.

[43] S. Ghosh, P. Varakantham, Y. Adulyasak, and P. Jaillet, "Dynamic redeployment to counter congestion or starvation in vehicle sharing systems," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 25, 2015, pp. 79–87.

[44] S. Ghosh, M. Trick, and P. Varakantham, "Robust repositioning to counter unpredictable demand in bike sharing systems," in *Proceedings of the 25th International Joint Conference on Artificial Intelligence IJCAI*, 2016, pp. 3096–3102.

[45] S. Ghosh, P. Varakantham, Y. Adulyasak, and P. Jaillet, "Dynamic repositioning to reduce lost demand in bike sharing systems," *Journal of Artificial Intelligence Research*, vol. 58, pp. 387–430, 2017.

[46] S. Ghosh, J. Y. Koh, and P. Jaillet, "Improving customer satisfaction in bike sharing systems through dynamic repositioning," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, 2019, pp. 5864–5870.

[47] M. D. Gleditsch, K. Hagen, H. Andersson, S. J. Bakker, and K. Fagerholt, "A column generation heuristic for the dynamic bicycle rebalancing problem," *European Journal of Operational Research*, 2022.

[48] Z. Haider, A. Nikolaev, J. E. Kang, and C. Kwon, "Inventory rebalancing through pricing in public bike sharing systems," *European Journal of Operational Research*, vol. 270, no. 1, pp. 103–117, 2018.

[49] S.-S. Ho, M. Schofield, and N. Wang, "Learning incentivization strategy for resource rebalancing in shared services with a budget constraint," *Journal of Applied and Numerical Optimization*, vol. 3, no. 1, pp. 105–114, 2021.

[50] R. Hu, Z. Zhang, X. Ma, and Y. Jin, "Dynamic rebalancing optimization for bike-sharing system using priority-based moea/d algorithm," *IEEE Access*, vol. 9, pp. 27 067–27 084, 2021.

[51] Z. Hu, K. Huang, E. Zhang, Q. Ge, and X. Yang, "Rebalancing strategy for bike-sharing systems based on the model of level of detail," *Journal of Advanced Transportation*, vol. 2021, 2021.

[52] J. Huang, H. Sun, H. Li, L. Huang, A. Li, and X. Wang, "Central station-based demand prediction for determining target inventory in a bike-sharing system," *The Computer Journal*, vol. 65, no. 3, pp. 573–588, 2022.

[53] J. Huang, Q. Tan, H. Li, A. Li, and L. Huang, "Monte carlo tree search for dynamic bike repositioning in bike-sharing systems," *Applied Intelligence*, pp. 1–16, 2022.

[54] P. Hulot, D. Aloise, and S. D. Jena, "Towards station-level demand prediction for effective rebalancing in bike-sharing systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 378–386.

[55] G. Héctor, L. Rafael, and A. Ramirez-Nafarrate, "A simulation-optimization study of the inventory of a bike-sharing system: The case of mexico city ecobici's system," *Case Studies on Transport Policy*, vol. 9, no. 3, pp. 1059–1072, 2021.

[56] Y. Jin, C. Ruiz, and H. Liao, "A simulation framework for optimizing bike rebalancing and maintenance in large-scale bike-sharing systems," *Simulation Modelling Practice and Theory*, vol. 115, p. 102422, 2022.

[57] K. Kim, "Investigation on the effects of weather and calendar events on bike-sharing according to the trip patterns of bike rentals of stations," *Journal of Transport Geography*, vol. 66, pp. 309–320, 2018.

[58] C. Kloimüllner and G. R. Raidl, "Full-load route planning for balancing bike sharing systems by logic-based benders decomposition," *Networks*, vol. 69, no. 3, pp. 270–289, 2017.

[59] C. Kloimüllner, P. Papazek, B. Hu, and G. R. Raidl, "Balancing bicycle sharing systems: an approach for the dynamic case," in *European Conference on Evolutionary Computation in Combinatorial Optimization*. Springer, 2014, pp. 73–84.

[60] B. Lahoorpoor, H. Faroqi, A. Sadeghi-Niaraki, and S.-M. Choi, "Spatial cluster-based model for static rebalancing bike sharing problem," *Sustainability*, vol. 11, no. 11, p. 3205, 2019.

[61] B. Legros, "Dynamic repositioning strategy in a bike-sharing system; how to prioritize and how to rebalance a bike station," *European Journal of Operational Research*, vol. 272, no. 2, pp. 740–753, 2019.

[62] G. Li, N. Cao, P. Zhu, Y. Zhang, Y. Zhang, L. Li, Q. Li, and Y. Zhang, "Towards smart transportation system: A case study on the rebalancing problem of bike sharing system based on reinforcement learning," *Journal of Organizational and End User Computing (JOEUC)*, vol. 33, no. 3, pp. 35–49, 2021.

[63] Y. Li, W. Szeto, J. Long, and C. Shui, "A multiple type bike repositioning problem," *Transportation Research Part B: Methodological*, vol. 90, pp. 263–278, 2016.

[64] Y. Li, Y. Zheng, and Q. Yang, "Dynamic bike reposition: A spatio-temporal reinforcement learning approach," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1724–1733.

[65] J. Liang, S. D. Jena, and A. Lodi, "Dynamic rebalancing optimization for bike-sharing systems: A modeling framework and empirical comparison," GERAD, HEC Montréal, Canada, Les Cahiers du GERAD G–2023–47, 2023.

[66] J. Liang, M. C. Martins Silva, D. Aloise, and S. D. Jena, "Dynamic rebalancing for bike-sharing systems under inventory interval and target predictions," CIRRELT, Canada, Research Paper CIRRELT-2023-37, 2023.

[67] S. Lin, F. Y. Chen, Y. Li, and Z.-J. M. Shen, "Dynamic inventory control with covariates: Risk constraints, regularization, and folding-horizon plan," *Yanzhi and Shen, Zuo-Jun Max, Dynamic Inventory Control with Covariates: Risk Constraints, Regularization, and Folding-horizon Plan (February 14, 2022)*, 2022.

[68] J. Liu, L. Sun, W. Chen, and H. Xiong, "Rebalancing bike sharing systems: A multi-source data smart optimization," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1005–1014.

[69] M. Lowalekar, P. Varakantham, S. Ghosh, S. D. Jena, and P. Jaillet, "Online repositioning in bike sharing systems," in *Twenty-Seventh International Conference on Automated Planning and Scheduling*, 2017.

[70] Á. Lozano, J. F. De Paz, G. Villarrubia Gonzalez, D. H. D. L. Iglesia, and J. Bajo, "Multi-agent system for demand prediction and trip visualization in bike sharing systems," *Applied Sciences*, vol. 8, no. 1, p. 67, 2018.

[71] C.-C. Lu, "Robust multi-period fleet allocation models for bike-sharing systems," *Networks and Spatial Economics*, vol. 16, no. 1, pp. 61–82, 2016.

[72] X. Luo, L. Li, L. Zhao, and J. Lin, "Dynamic intra-cell repositioning in free-floating bike-sharing systems using approximate dynamic programming," *Transportation Science*, 2022.

[73] R. Meddin, P. DeMaio, O. O'Brien, R. Rabello, C. Yu, J. Seamon, and T. Benicchio. (2022) The meddin bike-sharing world map. [Online]. Available: http://bikesharingworldmap.com/

[74] K. Mellou and P. Jaillet, "Dynamic resource redistribution and demand estimation: An application to bike sharing systems," *Available at SSRN 3336416*, 2019.

[75] A. OroojlooyJadid and D. Hajinezhad, "A review of cooperative multi-agent deep reinforcement learning," *arXiv preprint arXiv:1908.03963*, 2019.

[76] O. O'Brien, P. DeMaio, R. Rabello, S. Chou, and T. Benicchio, "The meddin bike-sharing world map report,"

https://bikesharingworldmap.com/reports/bswm_mid2022report.pdf, 2022, accessed: 2023-08-04.

[77] A. Pal and Y. Zhang, "Free-floating bike sharing: Solving real-life large-scale static rebalancing problems," *Transportation Research Part C: Emerging Technologies*, vol. 80, pp. 92–116, 2017.

[78] L. Pan, Q. Cai, Z. Fang, P. Tang, and L. Huang, "A deep reinforcement learning framework for rebalancing dockless bike sharing systems," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 1393–1400.

[79] Y. Pan, R. C. Zheng, J. Zhang, and X. Yao, "Predicting bike sharing demand using recurrent neural networks," *Procedia computer science*, vol. 147, pp. 562–566, 2019.

[80] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[81] M. Rainer-Harbach, P. Papazek, B. Hu, and G. R. Raidl, "Balancing bicycle sharing systems: A variable neighborhood search approach," in *European conference on evolutionary computation in combinatorial optimization*. Springer, 2013, pp. 121–132.

[82] M. Rainer-Harbach, P. Papazek, G. R. Raidl, B. Hu, and C. Kloimüllner, "Pilot, grasp, and vns approaches for the static balancing of bicycle sharing systems," *Journal of Global Optimization*, vol. 63, no. 3, pp. 597–629, 2015.

[83] T. Raviv and O. Kolka, "Optimal inventory management of a bike-sharing station," *IIE Transactions*, vol. 45, no. 10, pp. 1077–1093, 2013.

[84] T. Raviv, M. Tzur, and I. A. Forma, "Static repositioning in a bike-sharing system: models and solution approaches," *EURO Journal on Transportation and Logistics*, vol. 2, no. 3, pp. 187–229, 2013.

[85] K. Roshan Zamir, "Dynamic repositioning for bikesharing systems," Ph.D. dissertation, 2020.

[86] H. Sayarshad, S. Tavassoli, and F. Zhao, "A multi-periodic optimization formulation for bike planning and bike utilization," *Applied Mathematical Modelling*, vol. 36, no. 10, pp. 4944–4951, 2012.

[87] M. Schofield, S.-S. Ho, and N. Wang, "Handling rebalancing problem in shared mobility services via reinforcement learning-based incentive mechanism," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2021, pp. 3381–3386.

[88] J. Schuijbroek, R. C. Hampshire, and W.-J. Van Hoeve, "Inventory rebalancing and vehicle routing in bike sharing systems," *European Journal of Operational Research*, vol. 257, no. 3, pp. 992–1004, 2017.

[89] Y. Seo, "A dynamic rebalancing strategy in public bicycle sharing systems based on real-time dynamic programming and reinforcement learning," Ph.D. dissertation, Doctoral dissertation. Seoul National University, South Korea, 2020.

[90] Y.-H. Seo, D.-K. Kim, S. Kang, Y.-J. Byon, and S.-Y. Kho, "Rebalancing docked bicycle sharing system with approximate dynamic programming and reinforcement learning," *Journal of Advanced Transportation*, vol. 2022, 2022.

[91] J. Shu, M. C. Chou, Q. Liu, C.-P. Teo, and I.-L. Wang, "Models for effective deployment and redistribution of bicycles within public bicycle-sharing systems," *Operations Research*, vol. 61, no. 6, pp. 1346–1359, 2013.

[92] C. Shui and W. Szeto, "Dynamic green bike repositioning problem–a hybrid rolling horizon artificial bee colony algorithm approach," *Transportation Research Part D: Transport and Environment*, vol. 60, pp. 119–136, 2018.

[93] ——, "A review of bicycle-sharing service planning problems," *Transportation Research Part C: Emerging Technologies*, vol. 117, p. 102648, 2020.

[94] W. Szeto, Y. Liu, and S. C. Ho, "Chemical reaction optimization for solving a static bike repositioning problem," *Transportation Research Part D: Transport and Environment*, vol. 47, pp. 104–135, 2016.

[95] Q. Tang, Z. Fu, D. Zhang, H. Guo, and M. Li, "Addressing the bike repositioning problem in bike sharing system: A two-stage stochastic programming model," *Scientific Programming*, vol. 2020, 2020.

[96] C. M. Vallez, M. Castro, and D. Contreras, "Challenges and opportunities in dock-based bike-sharing rebalancing: a systematic review," *Sustainability*, vol. 13, no. 4, p. 1829, 2021.

[97] P. Vogel and P. Vogel, *Service network design of bike sharing systems*. Springer, 2016.

[98] P. Vogel, T. Greiser, and D. C. Mattfeld, "Understanding bike-sharing systems using data mining: Exploring activity patterns," *Procedia-Social and Behavioral Sciences*, vol. 20, pp. 514–523, 2011.

[99] P. Vogel, B. A. Neumann Saavedra, and D. C. Mattfeld, "A hybrid metaheuristic to solve the resource allocation problem in bike sharing systems," in *Hybrid Metaheuristics: 9th International Workshop, HM 2014, Hamburg, Germany, June 11-13, 2014. Proceedings 9*. Springer, 2014, pp. 16–29.

[100] X. Wu, C. Lyu, Z. Wang, and Z. Liu, "Station-level hourly bike demand prediction for dynamic repositioning in bike sharing systems," in *Smart transportation systems 2019*. Springer, 2019, pp. 19–27.

[101] I. Xiao, "A distributed reinforcement learning solution with knowledge transfer capability for a bike rebalancing problem," *arXiv preprint arXiv:1810.04058*, 2018.

[102] C. Xu, J. Ji, and P. Liu, "The station-free sharing bike demand forecasting with a deep learning approach and large-scale datasets," *Transportation research part C: emerging technologies*, vol. 95, pp. 47–60, 2018.

[103] H. Xu, F. Duan, and P. Pu, "Dynamic bicycle scheduling problem based on short-term demand prediction," *Applied Intelligence*, vol. 49, no. 5, pp. 1968–1981, 2019.

[104] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang, "Mean field multi-agent reinforcement learning," in *International conference on machine learning*. PMLR, 2018, pp. 5571–5580.

[105] Y.-C. Yin, C.-S. Lee, and Y.-P. Wong, "Demand prediction of bicycle sharing systems," 2012, [Online].

[106] Z. Yin, Z. Kou, and H. Cai, "A deep reinforcement learning model for large-scale dynamic bike share rebalancing with spatial-temporal context," in *The 12th International Workshop on Urban Computing*, 2023.

[107] A. Yoshida, Y. Yatsushiro, N. Hata, T. Higurashi, N. Tateiwa, T. Wakamatsu, A. Tanaka, K. Nagamatsu, and K. Fujisawa, "Practical end-to-end repositioning algorithm for managing bike-sharing system," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 1251–1258.

[108] P.-S. You, "A two-phase heuristic approach to the bike repositioning problem," *Applied Mathematical Modelling*, vol. 73, pp. 651–667, 2019.

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

[110] B. Zhang, X. Li, and F. Saldanha-da Gama, "Free-floating bike-sharing systems: New repositioning rules, optimization models and solution algorithms," *Information Sciences*, vol. 600, pp. 239–262, 2022.

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

[112] J. Zhang, M. Meng, Y. D. Wong, P. Ieromonachou, and D. Z. Wang, "A data-driven dynamic repositioning model in bicycle-sharing systems," *International Journal of Production Economics*, vol. 231, p. 107909, 2021.

[113] X. Zhang, H. Yang, R. Zheng, Z. Jin, and B. Zhou, "A dynamic shared bikes rebalancing method based on demand prediction," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 238–244.

[114] X. Zheng, M. Tang, Y. Liu, Z. Xian, and H. H. Zhuo, "Repositioning bikes with carrier vehicles and bike trailers in bike sharing systems," *Applied Sciences*, vol. 11, no. 16, p. 7227, 2021.

APPENDIX A APPENDIX FOR THE FIRST ARTICLE

A.1 Other Assumptions and Objective Functions

A.1.1 Initial Location and Inventory of Vehicles

At the beginning of the planning horizon, vehicles are located at specific stations or a depot with a predefined inventory. Existing works assume that the initial locations and inventories of vehicles are always known and fixed [see, e.g., 23, 45, 69]. The total number of available bikes for rebalancing may be uniformly or arbitrarily assigned to each vehicle. In contrast, some studies, such as [59] and [112], consider a depot and use it as the initial location of the vehicles.

Table A.1 Modeling assumptions in MIP-based multi-period rebalancing models

References	Time Constraints		Initial Location and Load of Trucks		Trip Distribution		Sequences of Events	Variable Type
	Traveling	Handling	Fixed	Flexible	Proportionality	Poisson		
[23]	✓		✓				(r+a+d)	Station-based
[45]			✓		✓		(r+a+d)	O-D
[44, 46]	✓	✓	✓				(a)(r)(d)	Station-based
[59]	✓	✓	✓				(a+d)(r)	Station-based
[69]	✓		✓		✓		(a)(d+r)	Station-based
[91]				✓		✓	(r+a+d)	O-D
[74]	✓	✓	✓		✓		(r+a+d)	O-D
[112]	✓	✓	✓				(r)(a+d)	Station-based
[114]			✓		✓		(r+a+d)	O-D

Intuitively, better solutions may be obtained if the model can explicitly decide on the initial location and inventory of the vehicles [see, e.g., 91].

We numerically explore the impact of these initial settings. Given that these settings rather concern the definition of the planning problem, but not the modeling assumptions, the surrounding discussions can be found in Appendix A.6.1. Throughout all other experiments, we applied fixed vehicle location and inventory as it has been common in the literature.

Finally, to summarize the various modeling assumptions used for the existing multi-period models in the literature, Table A.1 classifies the existing works by their various alternatives.

A.1.2 Objective Functions

Decision-makers may have different objectives for their rebalancing strategies. The most common objective for dynamic rebalancing is to minimize lost demand, which is used within our experiments. The objective is part of the problem definition and not within the scope of our paper. Further, models with different objectives are difficult to compare directly. Therefore, we refrain from evaluating the impact of using different objectives in this paper.

We summarize the various objectives in Table A.2, classifying the measurements into several main aspects and marking with '✓' when the objective function of the corresponding reference contains a particular aspect. We also emphasize the modeling techniques the analyzed papers used to respect the rebalancing problem, mainly including Linear Programming (LP), MILP, Mixed-integer Nonlinear Programming (MINLP), Constraint Programming (CP), NN, and MDP. Some models may be non-linear since they have non-linear terms in the objective function or in some constraints.

Concerning the criterion used within the objective function, distance-based metrics are associated with the traveling distance of vehicles, mainly including traveling cost, traveling time, and fuel consumption. Loading-based metrics are associated with the number of handling (loading/unloading) operations. Researchers normally consider handling costs or time in the objective functions, which reflects the required workload of such operations. Two metrics aim at representing the dissatisfaction of customers: one minimizes the deviations from a target value, while another minimizes the lost demand (or, equivalently, maximizes the successful trips). Besides these popular metrics, some other factors have been considered, such as the cost of holding bikes by rebalancing vehicles, parking costs, CO_2 emissions, costs of using trucks (related to the number of trucks employed), and the number of visits of full vehicle loads. For example, [32] consider the number of bikes held by the trucks during each rebalancing movement and include the total holding cost in the objective function. [6] add to the objective the usage cost of employing each truck for rebalancing. [35] add to the objective a parking time for each station visit using the instances from an operator in Norway. [58] consider only full rebalancing vehicle loads among stations and maximize the total number of full vehicle loads picked up and delivered to the stations, which is indicated as the number of visits in Table A.2.

Given that SBRP mainly focus on night-time operation scenarios, where the dynamic demand is less important, their objectives mainly aim at minimizing the costs of the rebalancing operations based on traveling distance, the number of loading/unloading operations, and deviations from pre-defined target numbers of bikes (see Table A.2. Note that some works exclude the deviation from target values or satisfied demand from the objectives, but

Table A.2 Objectives and modeling technique of related literature

	Reference	Objective fuctions									Modeling Technique
		Distance-based metrics[1]	Loading-based metrics[2]	Number of Bikes in truck	Parking	CO_2	Target Value Dev[3]	Lost demand/ Satisfied demand	Number of trucks	Number of visits	
Static	[6]	✓					✓				MILP
	[3]	✓									MILP
	[8]	✓									MILP
	[14]										MILP
	[17]	✓									MILP
	[19]	✓									NN
	[25]	✓									MILP
	[27]	✓	✓				✓				CP
	[32]	✓		✓							MILP
	[33]	✓									MILP
	[35]	✓	✓		✓		✓				MILP
	[37]	✓						✓			MILP
	[58]									✓	MILP
	[60]	✓									MINLP
	[63]	✓					✓				MILP
	[81, 82]	✓	✓				✓				Combinatorial
	[84]	✓						✓			MILP
	[88]	✓	✓								MILP
	[94]	✓	✓				✓				MINLP
	[95]	✓	✓				✓				MILP
	[110]	✓							✓		MILP
Dynamic Multi-period	[12]							✓			MDP
	[23]							✓			MILP
	[45]	✓						✓			MILP
	[59]	✓	✓				✓	✓			MINLP
	[61]							✓			MDP
	[69]							✓			MILP
	[91]			✓				✓			LP
	[108]	✓						✓			LP
	[112]	✓						✓	✓		MILP
	[114]	✓						✓			MILP
Rolling Period	[44, 46]							✓			MILP
	[47]						✓				MILP
	[50]	✓	✓					✓	✓		MILP
	[74]	✓						✓			MILP
	[92]					✓		✓			Combinatorial
	[107]						✓				MINLP
	[111]	✓						✓			MINLP

[1] These factors are based on the traveling distance, such as traveling cost, traveling time, and fuel consumption.

[2] These factors are based on the number of loading/unloading operations, such as handling cost and handling time.

[3] Deviations from a target value.

implement particular constraints to guarantee the satisfaction of user demand to some degree [17, 25, 88, et al.].

In DBRP, rebalancing operations are performed multiple times during the day and real-time trip flow is considered when rebalancing takes place. Due to the more complex nature of DBRP, their objectives tend to include more aspects considered by operators, especially concerning the demand unsatisfaction. Generally, the existing objective functions quantify the distance-based metrics and customers' satisfaction, including the traveling cost, handling cost, and lost demand. Some focus on maximizing the profits of successful trips [e.g. 45]. [74] consider jointly traveling cost and handling operations, which adds immediate value to the rebalancing. Among the studies considering more than one aspect in their objective functions, [59] and [74] attribute a weight to each of them.

A.2 Proportionality Constraints

In the case where O-D variables $x_{s_1,s_2}^{t_1,t_2}$ are used to represent successful trips, Constraints (TD5) are automatically guaranteed. The proportional distribution can then be written as

$$x_{s_1,s_2}^{t_1,t_2} \leq ab_{s_1}^{t_1} \frac{F_{s_1,s_2}^{t_1,t_2}}{f_{s_1}^{+,t_1}} \qquad \forall\ s_1, s_2 \in S, t_1, t_2 \in T. \tag{TD7}$$

Constraints (TD7) imply that rentals from a station have to respect the transition probability when rental demand exceeds the number of available bikes at that station. [45] use such constraints with $ab_{s_1}^{t_1} = d_{s_1}^{t_1}$.

Note that summing (TD7) over t_1 and s_1, the left-hand side becomes $x_{s_2}^{-,t_2}$, which results in (TD3). Thus, station-based trip variables with proportional distribution and O-D variables with proportional distribution essentially represent the same type of trip distribution within the model.

For O-D variables in the Practical toy example of Section 3.3.2, the constraints enforcing the flow to be no more than the demand of each route and the returns to be less than the available docks can be combined as $x_{s_1,s_2}^{t_1,t_2} \leq \min\{F_{s_1,s_2}^{t_1,t_2}, ad_{s_2}^{t_2}\}$. To facilitate the comparison of the solutions, we present the solution $[x_{s_1,s_3}^{t_1,t_2}, x_{s_1,s_4}^{t_1,t_2}]$ of the O-D variables model in the equivalent format of station-based trip variables, e.g., $x_{s_1}^{+,t_1} = x_{s_1,s_3}^{t_1,t_2} + x_{s_1,s_4}^{t_1,t_2}$. The results are shown in Table A.3.

For O-D variables, constraints (TD7) mainly work for the case, where the rental demands cannot be satisfied. [23] applied station-based variable without any trip distribution constraints. [45] use (TD7). [69] use constraints that are similar to Constraints (TD1).

Table A.3 Trip distribution for 3 different demand scenarios under different trip distribution constraints (O-D variables)

	Constraints	I-I	I-S	S-I
Ideal solution		[*, *, 4, 2]	[4, 3, *, *]	[*, *, 4, 2]
O-D Variables	Without (TD7)	$[4,\ 2,\ x_{s_3}^{-,t_2} + x_{s_4}^{-,t_2} = 6]$	$[4,\ 3,\ x_{s_3}^{-,t_2} + x_{s_4}^{-,t_2} = 7]$ $x_{s_2,s_3}^{t_1,t_2} = 1, x_{s_2,s_4}^{t_1,t_2} = 2$	$[x_{s_1}^{+,t_1} + x_{s_2}^{+,t_1} = 6,\ 4,\ 2]$
	With (TD7)	$[x_{s_1}^{+,t_1} + x_{s_2}^{+,t_1} = 5,\ 3,\ 2]$ $x_{s_1,s_3}^{t_1,t_2} = 2, x_{s_2,s_3}^{t_1,t_2} = 1$	$[4,\ 3,\ 3,\ 4]$ $x_{s_1,s_3}^{t_1,t_2} = 2, x_{s_1,s_4}^{t_1,t_2} = 2$	$[x_{s_1}^{+,t_1} + x_{s_2}^{+,t_1} = 6,\ 4,\ 2]$

We give another practical toy example here. The 4 different configurations are shown in Figure A.1.

Practical toy example with 3 stations

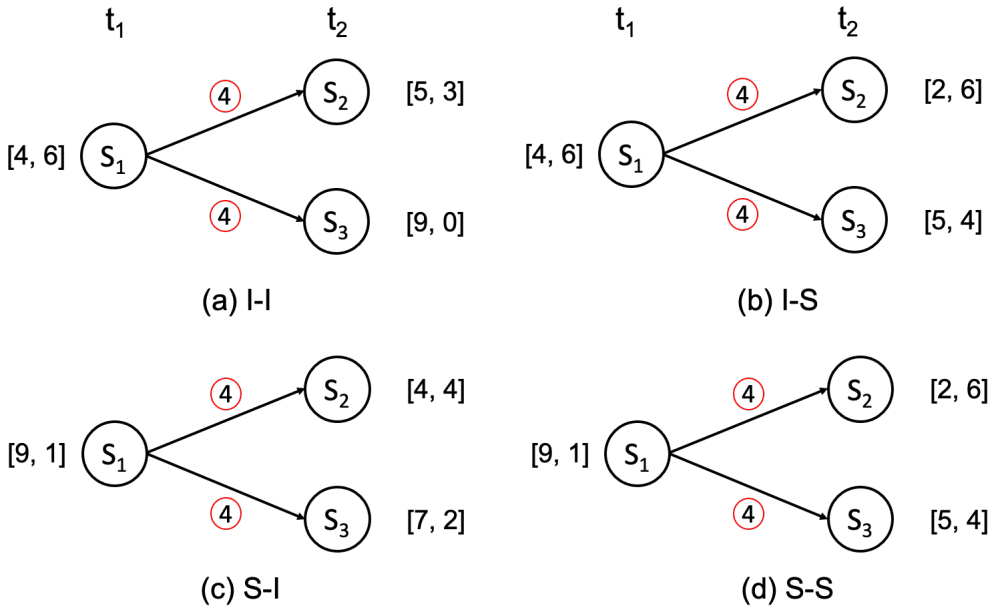

Figure A.1 Trip flow example with 3 stations

As we can see from Table A.4, constraints (TD2) add a strict limitation to the trip flow when there is no dock available in one of the stations. If one of the return variables can be realized is zero (the station is full), then it may cause all the variables equal to zero based on (TD1) and (TD2). In all 4 circumstances, the solutions under constraints (TD6), (TD2)+(TD6) with station-based variables, and no proportional constraints with O-D variables can be the same. Furthermore, the solution under constraints (TD3)+(TD6) is equivalent to the solution under constraints (TD7). Generally, (TD2)+(TD3) and (TD3)+(TD6) perform well in this case. (TD6), (TD4), and (TD2) also give a reasonable solution.

Table A.4 Solutions under different circumstances

$[x_{s_1}^{+,t_1}, x_{s_2}^{-,t_2}, x_{s_3}^{-,t_2}]$	Constraints	I-I	I-S	S-I
	Ideal solution	[3, 3, 0],[2, 2, 0]	[4, 2, 2]	[6, 4, 2]
Station-based variable	without TD	[4, 3, 0]	[4, 4, 4]	[8, 4, 2]
	(TD1)	[4, 2, 0]	**[4, 2, 2]**	[8, 4, 2]
	(TD2)	**[3, 3, 0]**	[4, 4, 4]	**[6, 4, 2]**
	(TD6)	**[3, 3, 0]**	[4, $x_{s_2}^{-,t_2} + x_{s_3}^{-,t_2} = 4$]	**[6, 4, 2]**
	(TD1)+(TD2)	[0, 0, 0]	**[4, 2, 2]**	[4, 2, 2]
	(TD1)+(TD6)	[0, 0, 0]	**[4, 2, 2]**	[4, 2, 2]
	(TD2)+(TD6)	**[3, 3, 0]**	[4, $x_{s_2}^{-,t_2} + x_{s_3}^{-,t_2} = 4$]	**[6, 4, 2]**
	(TD3)	[4, 2, 0]	**[4, 2, 2]**	[8, 4, 2]
	(TD4)	**[3, 3, 0]**	[4, 4, 4]	**[6, 4, 2]**
	(TD3)+(TD4)	[3, 2, 0]	**[4, 2, 2]**	**[6, 4, 2]**
	(TD3)+(TD6)	**[2, 2, 0]**	**[4, 2, 2]**	**[6, 4, 2]**
	(TD1)+(TD4)	[3, 1.5, 0]	**[4, 2, 2]**	[6, 3, 2]
	(TD2)+(TD3)	**[2, 2, 0]**	**[4, 2, 2]**	**[6, 4, 2]**
O-D variable	without (TD7)	**[3, 3, 0]**	[4, $x_{s_2}^{-,t_2} + x_{s_3}^{-,t_2} = 4$]	**[6, 4, 2]**
	(TD7)	**[2, 2, 0]**	**[4, 2, 2]**	**[6, 4, 2]**

A.3 Formulations for Event Sequences

The rest of the event sequences are demonstrated here.

- **(a)(d)(r):** Here, rebalancing operations occur at the end of each time-period, which is enforced by Constraints (A.1)–(A.4).

$$\sum_v r_{s,v}^{+,t} \leq d_s^t + x_s^{-,t} - x_s^{+,t} \qquad \forall\ s \in S,\ t \in T \qquad \text{(A.1)}$$

$$\sum_v r_{s,v}^{-,t} \leq C_s - d_s^t - x_s^{-,t} + x_s^{+,t} \qquad \forall\ s \in S,\ t \in T \qquad \text{(A.2)}$$

$$x_s^{+,t} \leq d_s^t + x_s^{-,t} \qquad \forall\ s \in S,\ t \in T \qquad \text{(A.3)}$$

$$x_s^{-,t} \leq C_s - d_s^t \qquad \forall\ s \in S,\ t \in T \qquad \text{(A.4)}$$

- **(d)(r)(a):** Rebalancing operations occur between rentals and returns, which is enforced by Constraints (A.5) and (A.6). Bike rentals occur at the beginning of the period and are only restricted by the current capacity of the station, as indicated by Constraints (A.7). Bike returns occur at the end and are limited by $x_s^{-,t} \leq C_s - d_s^t + \sum_v r_{s,v}^{+,t} - \sum_v r_{s,v}^{-,t} + x_s^{+,t}$, which can be achieved by replacing d_s^{t+1} in Constraints (3.6) ($d_s^{t+1} \leq C_s$)

with the right-hand side of Constraints (3.3).

$$\sum_v r_{s,v}^{+,t} \leq d_s^t - x_s^{+,t} \qquad \forall\ s \in S,\ t \in T \qquad \text{(A.5)}$$

$$\sum_v r_{s,v}^{-,t} \leq C_s - d_s^t + x_s^{+,t} \qquad \forall\ s \in S,\ t \in T \qquad \text{(A.6)}$$

$$x_s^{+,t} \leq d_s^t \qquad \forall\ s \in S,\ t \in T \qquad \text{(A.7)}$$

For the class of event sequences reported in the second row of Table 3.1, the corresponding modifications are as follows:

- **(d)(a)(r):** We first use Constraints (A.7) since rentals occur first. Then, to implement correct returns, we add Constraint (A.8). Finally, given that the vehicles are assumed to rebalance after bike rentals and returns, we also use vehicle constraints (A.1) and (A.2).

$$x_s^{-,t} \leq C_s - d_s^t + x_s^{+,t} \qquad \forall\ s \in S,\ t \in T \qquad \text{(A.8)}$$

- **(a)(r)(d):** Returns occur at the beginning of the period and consider the current inventory, as ensured by Constraints (A.4). We also add Constraints (A.9) and (A.10), since vehicles rebalance bikes after returns. As customers rent bikes at the end of each time-period, we have to enforce $x_s^{+,t} \leq d_s^t - \sum_v r_{s,v}^{+,t} + \sum_v r_{s,v}^{-,t} + x_s^{-,t}$, which is explicitly satisfied when replacing d_s^{t+1} in Constraints (3.6) ($0 \leq d_s^{t+1}$) by the right hand side of Constraints (3.3).

$$\sum_v r_{s,v}^{+,t} \leq d_s^t + x_s^{-,t} \qquad \forall\ s \in S,\ t \in T \qquad \text{(A.9)}$$

$$\sum_v r_{s,v}^{-,t} \leq C_s - d_s^t - x_s^{-,t} \qquad \forall\ s \in S,\ t \in T \qquad \text{(A.10)}$$

We now consider event sequences in which rentals and returns are assumed to happen simultaneously (see the third row in Table 3.1):

- **(a+d)(r):** The constraints for rebalancing are the same as for **(a)(d)(r)** and **(d)(a)(r)**, i.e., Constraints (A.1) and (A.2). The restrictions for rentals and returns are enforced by using Constraints (A.3) and (A.8).

Finally, two more classes of event sequences allow for rebalancing to occur simultaneously with either rentals or returns (see the fourth and fifth rows respectively in Table 3.1).

- **(a)(d+r):** Customers return bikes first, requiring the use of Constraints (A.4). Rebalancing then simultaneously occurs with rentals, which can be implemented using Constraints (A.1) and (A.2).
- **(d+r)(a):** Here, employing Constraints (A.5) and (A.6) is sufficient.
- **(d)(r+a):** Customers rent bikes first, requiring Constraints (A.7). Rebalancing then simultaneously occurs with returns, requiring Constraints (A.1) and (A.2).
- **(r+a)(d):** It suffices to add Constraints (A.9) and (A.10).

A.4 Generation of Problem Instances

In this appendix, we provide more details on the generator for the synthetic problem instances.

A.4.1 Station Network and Operating Settings

We first generate the station network. The parameters are defined in Table A.5. We consider a rectangular study area defined by the latitude and longitude values for each of its 4 vertices. This area is divided into a total of *num_grid* grids, each of which can be assigned to at most one station. In our studies, we use a rectangular study area with 150 × 150 grids (i.e., *num_grid* = 22500) with latitude values from 45.4 to 45.65 and longitude values from -73.71 to -73.49 (approximating the Montreal island area). The total number of stations, the number of city centers, and the total capacity of the station network can be set and changed according to what kind of instances we focus on.

We assume that there are either one or two city centers in the study area, including stations that have high return demands during morning peak hours and high rental demands during afternoon peak hours. If there is only one city center, its central grid is randomly selected within a square spanning grid 53 to 98 on both the x- and the y-axis of the study area. If there are two city centers, one central grid is selected randomly within the square spanning grids from 30 to 75, and the other one within the square spanning grids from 75 to 120 on both the x- and the y-axis of the study area. Each city center is then defined as an area of $ran\_cc \times ran\_cc$ grids around its central grid.

Each station network has a total of *num_station* stations (set either to 30 or 60). Regular stations are assumed to have a capacity of $cap\_os = 20$ docks. Given that city center stations typically have a much larger capacity, we here assume a capacity of $cap\_cs = 40$ docks for each city center station. We consider that the total capacity of all the stations (i.e.,

Table A.5 The input and output of station generation

Input	Study area: minimum longitude and latitude, maximum longitude and latitude
	num_grid: the number of grids *num_station*: total number of stations *num_cc*: the number of city centers *total_c*: the total capacity of all the stations *ran_cc*: the range of city centers (in number of grids) *per_cc*: the proportion of the total network capacity located within city centers *cap_cs*: capacity of a city center station *cap_os*: capacity of a regular station
Output	Locations of stations Distances between station pairs Number of city center stations Number of regular stations

the number of docks in the entire network) is roughly proportional to the total number of stations *num_station*, as observed within the network of BIXI Montreal which had a total capacity of 14,078 with 617 stations in 2019. Therefore, we set the total capacity *total_c* to 1,369 for 60 station networks and 685 for 30 station networks. Note that *total_c* will be used only to compute the number of regular and city center stations. The number of city center stations is set to $\lfloor total\_c \times per\_cc/cap\_cs \rceil$ (where *per_cc* is defined in Table 3.5 for each of the ground truths). The remaining stations are assumed to be regular (i.e., out of the city center) stations. We then randomly assign stations to the grids as follows:

- **City center stations:** We randomly select grids inside the city center area as locations for city center stations. We show an example with 60 stations and two city centers in Figure A.2. The dotted boxes are the range of city centers and the blue dots are the city center stations.
- **Regular stations:** The remaining stations will be randomly assigned to the grids outside the city center areas, indicated as green dots in Figure A.2.

Once the locations of stations are fixed, we compute the distance between each station pair. For the operating settings, we define the parameters in Table A.6.

Table A.6 The settings of BSSs and rebalancing fleet

n_bikes	the total number of bikes in the stations
n_trucks	the number of trucks available for rebalancing
cap_truck	the capacity of each truck
n_bikeT	the total available number of bikes for rebalancing trucks at the beginning

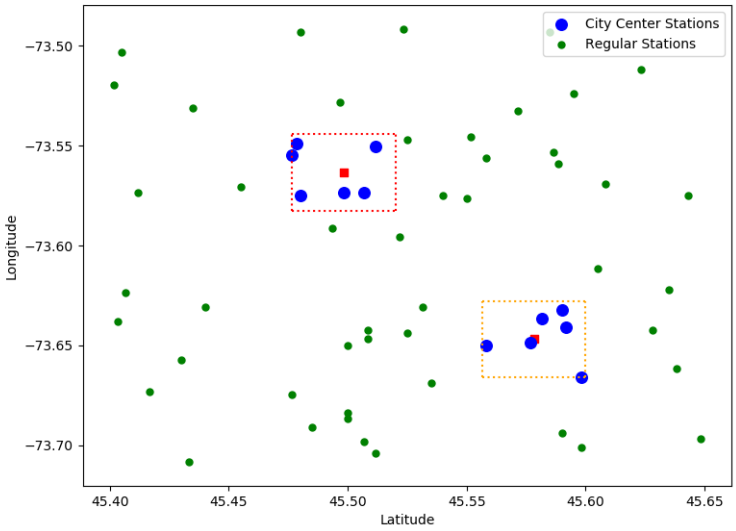

Figure A.2 Visualization of station network used by GT2 with 2 city centers, indicating regular stations (green dots), city center stations (blue dots), and the city center central grid (red rectangle). Note that each grid has a rectangular form.

Finally, the total number of available bikes in the system, n_bikes, is 608, which is proportional to those observed at BIXI Montreal in 2019. We assume that 4 trucks (n_trucks) are available to rebalance the bikes. The capacity of each truck, cap_truck, is set to 40 and the number of available bikes for trucks to employ, n_bikeT, is set to 80.

A.4.2 Bike Trips

Based on the generated station information, we generate the trip data. Analyzing the trip data from BIXI, we found that the demand has a similar pattern on weekdays with a morning peak and an afternoon peak that are mainly caused by work-related trips. At weekends, there is less regularity and trips seem more random. We therefore focus on weekdays only. To provide adequate but diverse problem instances and to fully explore the impacts, we generate trips based on real-world data with adaptable parameters, instead of directly applying real-world data.

Parameters to define trip demands. The parameters used for the trip generation are defined in Table A.7. Each trip contains an origin station, a destination station, a departure time, and an arrival time. We set an average total number of trips avg_trips per weekday, which can be estimated from historical trip data in real BSSs. Note that $per\_io + per\_oo + per\_rd + per\_rn = 100$. After having the fixed number of trips of different types, we generate the trip data according to their characteristics. We assume that

the departure time of each trip type follows a particular distribution (Dis), specifying the probability that a trip starts at a specific time. This allows us to model demand changes throughout the day while preserving uncertainty.

Table A.7 The input and output of trips generation

Input	*avg_trip*: The average number of trips per weekday *per_oi*: The percentage of OI trips *per_oo*: The percentage of OO trips *per_rd*: The percentage of RD trips *per_rn*: The percentage of RN trips *per_w*: The percentage of work-related trips (OI and OO) expected to happen *Dis*: The set of distributions for different types of trips *dur_trip*: The interval for the length of trips
Output	All trips with origin stations, destination stations, departure time, and arrival time

The rules for trip generation are as follows:

- RD **and** RN **trips** We choose the origin station and destination station randomly among all stations for each random trip. The departure time is sampled from the corresponding distribution (one distribution for RD trips and one for RN trips). The duration of the trip is selected from the interval *dur_trip* at uniformly random, allowing us to also compute the arrival time for this trip. RD and RN trips are assumed to be one-way without any corresponding returns, as opposed to work-related trips. We repeat this process to obtain all required trips for each day. The total number of RD and RN trips will be $avg\_trips * per\_rd$ and $avg\_trips * per\_rn$ for each weekday.

- OI **trips and** OO **trips** Work-related trips normally have stable O-D pairs that include one trip from home to work and another one back to home with the same origin/destination stations. For each O-D pair, we first generate one trip from home to work. The stations are selected randomly according to the type of trips. For example, the origin station of an OI trip is selected randomly among the stations outside city centers, and the destination station is chosen from city center stations. Then, a corresponding return trip is generated with the origin and destination stations reversed. The process is the same for OO trips, except that both the origin and destination stations need to be selected among the stations outside city centers. The departure time obeys the assumed distribution. Differently from random trips, work-related trips happen during morning peak hours and afternoon peak hours. Thus, we use two different distributions for OI trips and two distributions for OO trips to imitate the two peaks. Using the departure time under the particular distribution and duration *dur_trip*, the arrival time is computed.

Random trips RD and RN may vary a lot from one day to another, so we generate them from scratch for each weekday. However, work-related trips have distinctive characteristics. They do not vary too much since users tend to commute between the same O-D pairs. Nevertheless, the demand for each weekday may change slightly because users may choose alternative means of transportation or not go to work for some personal reason on certain days. Thus, we consider an additional processing step for work-related trips. We generate a total of $avg\_trips * per\_oi$ OI and $avg\_trips * per\_oo$ OO trips that form a work-related trip set and we fix this set for all weekdays of a given problem instance. We then consider a probability per_w that a person is actually taking the bike for a route. For each day, the final demand for work-related trips is based on this fixed set. We then uniformly sample a random value between 0 to 1 for each work-related trip in the set. If the value is smaller than or equal to per_w, we select the corresponding trip with its rental and return demand. The actual number of work-related trips OI and OO at each weekday will slightly vary but will average to about $avg\_trips * per\_oi * per\_w$ and $avg\_trip * per\_oo * per\_w$, respectively.

Parameter values to generate trips. Since BIXI has a total number of around 33,300 trips per weekday on a 617 stations network, we set avg_trip to 3,240 for our 60 station network. The probability per_w for work-related trips is set to 0.85, which means that a user has an 85% chance to choose the bike, and its demand is generated. We assume that the duration of each trip is within $[5, 30]$ minutes (dur_trip).

We now describe the settings of the distributions in Dis. We illustrate the weekday average demand of one week at BIXI Montreal from July 2019 in Figure A.3(a). An example of generated synthetic data is demonstrated in Figure A.3(b)–(f), averaging over 500 days for 24 hours discretized into 48 time-periods. Since our work mainly focuses on rebalancing optimization, we use Beta distributions and linear transformations to fit our demand curve to the one of BIXI. We apply two Beta distributions ($\alpha = 3, \beta = 8$) for the departure time of OI trips in morning peak hours and afternoon peak hours. The random value x generated by the Beta distribution is shifted into departure time using a linear transformation $ax + b$. For example, we set $a = 530$ and $b = 340$ for the OI trips during the morning hours to shift random numbers into the interval of [340, 870] minutes. After transformation, we obtain Figure A.3(c) with two peaks representing the morning and evening rush hours. Similarly, two Beta distributions ($\alpha = 3, \beta = 7$) with $a = 550$ and $b = 900$ for OO trips are employed, as shown in Figure A.3(d).

For RD trips happening around 10 a.m. - 9 p.m., we use a Beta distribution ($\alpha = 3, \beta = 7$) with $a = 900$ and $b = 560$, as illustrated in Figure A.3(e). A Beta distributions ($\alpha = 6, \beta = 8$) with $a = 1200$ and $b = 750$ is shifted to represent RN trips, mainly during 4 p.m. - 5 a.m.,

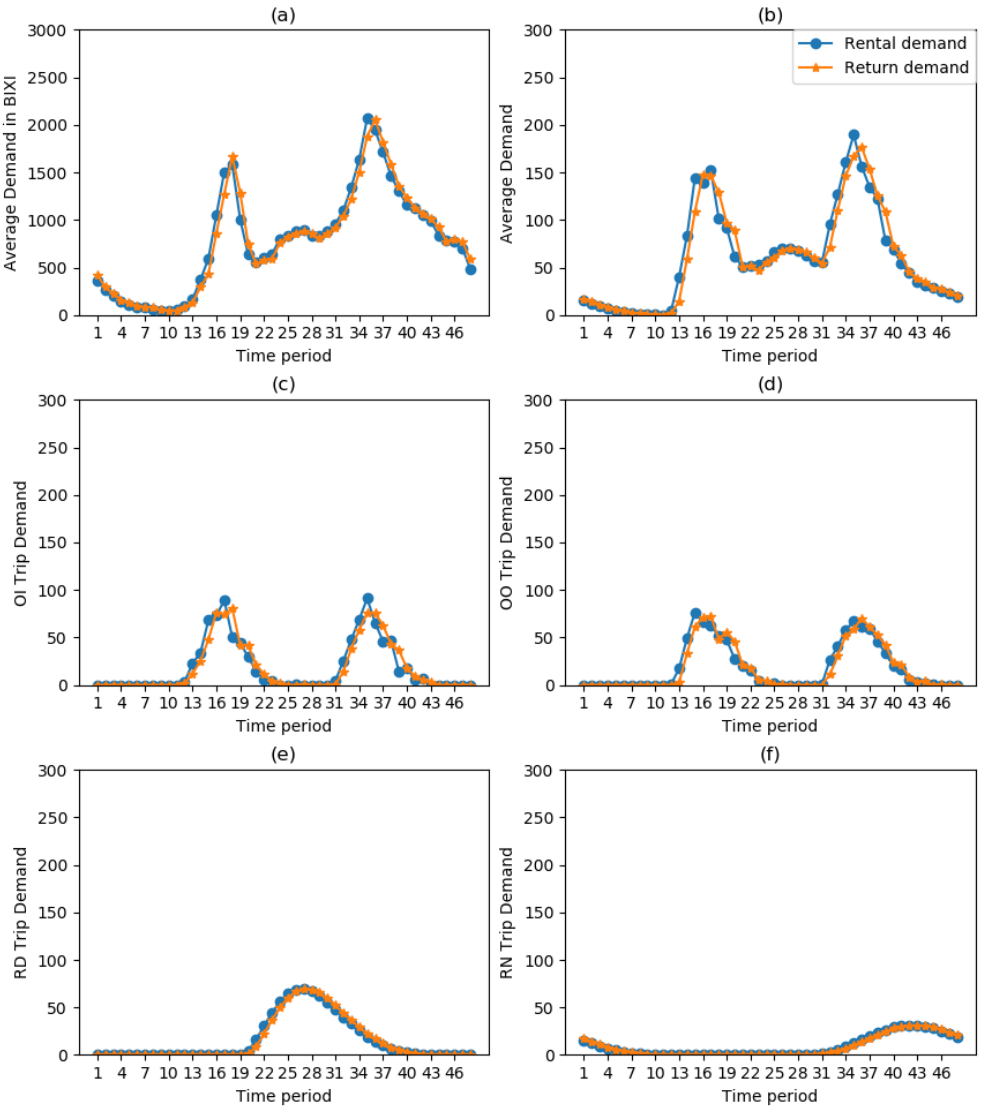

Figure A.3 Demand information for 500 days in 24 hours (48 time-periods)

as depicted in Figure A.3(f). Summing the demand of the four trip types, Figure A.3(b) illustrates the total average demand over 500 days.

A.4.3 Ground Truths for Experiments

To test the models under different station environments and trip patterns, we generate 3 ground truths based on the above-explained settings of parameters. Some of the parameters have the same values in the three ground truths, namely:

- *num_grid*: 150x150
- *num_station*: 60
- *n_trucks*: 4
- *cap_truck* :40
- *n_bikeT* : 80
- *avg_trip*: 3630
- *per_w*: 85%.

The parameters specified for each of the ground truths are defined in Table 3.5 in Section 3.5.1.

A.5 Pseudo-Code for the Simulator

The pseudo-code for our simulator is given in Algorithm 2. The initial inventory of each station d_s^0 and each truck d_v^0 are given as inputs, along with the capacity for each station C_s and each truck $\hat{C}_v$. Rebalancing strategies $r\_pick(s,t,v)$ and $r\_drop(s,t,v)$ are obtained from the optimization model. Trip sequence N is composed of individual trips $\{[t_d(n), s_d(n), t_a(n), s_a(n)], n \in N\}$, including rental and return demands with $t_d(n)$ ascending. For a trip n, $t_d(n)$ is the time of departure, $s_d(n)$ is the departure station, $t_a(n)$ is the arriving time, and $s_a(n)$ is the arrival station. A waiting set of M events $W = \{[w_t(m), w_s(m), w_i(m), w_v(m)], m \in M\}$ stores the upcoming demands and rebalancing operations. The elements of W change with real-time system status. Each element $w \in W$ has an indicator $w_i(m)$ that either represents a rental demand ($w_i(m) = d$), a return demand ($w_i(m) = a$), a pick-up operation ($w_i(m) = p$), or a drop-off operation ($w_i(m) = f$). Moreover, $w_t(m)$ is the departure time of the event, $w_s(m)$ is the station, and $w_v(m)$ is the truck. Note that the value of $w_v(m)$ has no impact on demand events and is set to 0. The waiting set will be sorted in non-decreasing order of $w_t(m)$ and its first element is the event that will be processed next. Parameter $D_{s,s'}$ represents the distance between two stations s and s'. The transit time (in minutes) between two stations during time-period t is given by $R^t_{s,s'}$ and the average operation time for picking up or dropping off one bike is given by *op*.

We denote the inventory of the station s as $Avails\_bike(s)$ and the inventory of the vehicle

Algorithm 2: Simulator for the rebalancing strategy $r\_pick(s,t)$ and $r\_drop(s,t)$

Input : $\{[t_d(n), s_d(n), t_a(n), s_a(n)], n \in N\}$, $D_{s,s'}$, $r\_pick(s,t)$, $r\_drop(s,t)$, d_s^0, d_v^0, C_s, $\hat{C}_v$, $R_{s,s'}^t$, and op. T is the planning horizon.

Initialization: $Lost\_rental(t,s) = 0$; $Lost\_return(t,s) = 0$; $Avails\_bike(s) = d_s^0$; $Availv\_bike(v) = d_v^0$; $W = \{[w_t(m), w_s(m), w_i(m), w_v(m)], m \in M\}$ with all the rental demands from N and the operation events of the first time-period sorted; $time = w_t(1)$; $s = w_s(1)$; $indicator = w_i(1)$; and $v = w_v(1)$.

$n = 1$;
while $time \leq T$ *and* $W \neq \emptyset$ **do**
  Find corresponding time-period t based on $time$;
  $sign = 0$;
  if $indicator = d$ **then**
    if $Avails\_bike(s) > 0$ **then**
      $Avails\_bike(s) = Avails\_bike(s) - 1$;
      $W = W \cup \{[t_a(n), s_a(n), a, 0]\}$;
    else
      $Lost\_rental(t,s) = Lost\_rental(t,s) + 1$;
    $n = n + 1$;
  else if $indicator = a$ **then**
    if $C_s - Avails\_bike(s) > 0$ **then**
      $Avails\_bike(s) = Avails\_bike(s) + 1$;
    else
      $Lost\_return(t,s) = Lost\_return(t,s) + 1$;
      Find s' closest to s with available docks based on $D_{s,s'}$;
      $Avails\_bike(s') = Avails\_bike(s') + 1$;
  else if $indicator = p$ **then**
    if $Avails\_bike(s) > 0$ *and* $Availv\_bike(v) < \hat{C}_v$ **then**
      $Avails\_bike(s) = Avails\_bike(s) - 1$;
      $Availv\_bike(v) = Availv\_bike(v) + 1$;
      if *All the rebalancing operations are done for* s **then**
        $sign = 1$;
    else
      Remove the elements in W whose $w_s = s, w_i = p, w_v = v$;
      $sign = 1$;
  else
    if $Availv\_bike(v) > 0$ *and* $Avails\_bike(s) < C_s$ **then**
      $Avails\_bike(s) = Avails\_bike(s) + 1$;
      $Availv\_bike(v) = Availv\_bike(v) - 1$;
      if *All the rebalancing operations are done for* s **then**
        $sign = 1$;
    else
      Remove the elements in W whose $w_s = s, w_i = f, w_v = v$;
      $sign = 1$;
  if $sign = 1$ **then**
    Create W' of v for $t+1$ based on $R_{s,s'}^t$, and op;
    $W = W \cup W'$;
  $W = W \backslash \{[time, s, indicator, v]\}$;
  $time, s, indicator, v = w_t(m), w_s(m), w_i(m), w_v(m)$ where $w_t(m)$ is the minimum in W;
end

Output : $Lost\_rental(t,s)$ and $Lost\_return(t,s)$

v as $Availv\_bike(v)$. The lost demand during a period t for a station s will be counted in $Lost\_rental(t, s)$ and $Lost\_return(t, s)$ respectively. We first initialize W with all rental demands from N with $w_i = d$ and $w_v = 0$.

The corresponding returns of successful rentals and rebalancing events are created and added to W in simulated real-time. For each truck and time-period, we create $r\_pick(s, t, v)/r\_drop(s, t, v)$ consecutive events with ($w_i = p$ or $w_i = f$ respectively). Rebalancing starts as soon as the truck arrives at the station, but not before the first minute associated with time-period t. A truck leaves for the next station as soon as it finishes the rebalancing operations at the current station. If the truck reaches the next station before the end of the current time-period, it waits until the beginning of the next time-period before starting the rebalancing operations. Since rebalancing and relocation may be scheduled continuously one after another, some rebalancing operations may be delayed due to the previous operations.

As in reality, the simulator processes the events in W in chronological order. When a rental demand occurs and the station holds at least one available bike, we update the station inventory and add the corresponding return demand to the waiting set W. Otherwise, the customer is assumed to leave the system and a lost rental demand is counted. When a return demand occurs but the station has no available docks, we assume that the customer returns the bike at the nearest station with available docks. However, a lost return demand will be counted. For pick-up/drop-off rebalancing operation events, we verify whether sufficient bikes/docks at the station and space/bikes within the truck are available. Rebalancing is carried out as close as possible to the originally planned operations. The inventories of the station and the truck are updated accordingly. After partially/fully successive rebalancing, the truck departs for the next station.

A.6 Complimentary Experiments

A.6.1 Initial Settings for Vehicles

In our basic model, the initial location and inventory of each vehicle are fixed. The fixed initial locations are at stations 1, 16, 31, and 46. Each truck has the same amount of bikes, i.e., 20 bikes. However, the operator may have the possibility and desire to specify an initial location and inventory for the trucks. To this end, Constraints (A.11) and (A.12) are created. Constraint (A.11) implies that the number of vehicles located at specific stations at the first time-period is equal to the number of vehicles num_v in the system, which, along with the Constraints (3.4), means that the trucks can be assigned to any station at the beginning of rebalancing. Constraint (A.12) indicates that the total number of bikes in all vehicles equals

the total number of bikes *av_bike* initially available in vehicles. Here, *av_bike* is set to 80, while the inventory at each vehicle is optimized.

$$\sum_{s,v} z^1_{s,v} = num\_v \tag{A.11}$$

$$\sum_{v} \hat{d}^1_v = av\_bike \tag{A.12}$$

Note that, if we need to consider a central depot in our system, this depot can be represented as an additional station with a particular capacity in our optimization model.

For experiments, we consider 30-minute time-periods in the station-based model without trip distribution constraints. Table A.8 summarizes the results for 3 Ground truths. Compared to the initial inventory, the initial location has obvious impacts on lost demand for dynamic rebalancing. The case with fixed initial inventory and flexible location has the best performance for lost rentals. The results highlight that the initial location of the trucks is important, assuming that each truck holds sufficient bikes. For GT3 with too many work-related trips, the models are hard to solve to optimality. The trip demand is highly concentrated during the peak hours, which makes MIP gaps hard to reach 0.01%. Under these MIP gaps, the system can still benefit from the flexibility of initial locations.

Table A.8 Station-based model with different initial vehicle settings (60 stations, 80 available bikes for 4 trucks, 30 mins)

	Initial	Initial	O.F.	Time	MIP	Lost Demand (%)	
	Location	Inventory	Value	(mins)	Gap (%)	Rental	Return
GT1	Fixed	Fixed	0.8	<1	0.00	8.78	7.99
	Fixed	Flexible	0.8	<1	0.00	8.39	8.07
	Flexible	Fixed	0.1	<1	0.00	8.26	5.89
	Flexible	Flexible	0.1	<1	0.00	8.30	5.60
GT2	Fixed	Fixed	0.5	<1	0.00	9.46	1.78
	Fixed	Flexible	0.5	<1	0.00	9.59	2.37
	Flexible	Fixed	0.2	<1	0.00	8.50	1.93
	Flexible	Flexible	0.1	<1	0.00	8.92	1.58
GT3	Fixed	Fixed	179.1	1440	2.46	20.33	21.67
	Fixed	Flexible	177.1	1440	2.33	19.89	20.86
	Flexible	Fixed	162.1	1440	2.52	17.46	16.66
	Flexible	Flexible	161.3	1440	2.37	17.78	17.19

When the BSSs network is small, a fixed initial setting is easier for the operators with an acceptable performance since no adjustment is needed for trucks before the beginning of

dynamic rebalancing. However, if the station network is complex, the flexible initial setting may be beneficial to obtain a better performance of lost demand.

A similar conclusion is observed in Table A.9 for the O-D variable model. The flexible initial location and fixed inventory give the best performance for GT1 and GT3. For GT2, the flexible location and flexible inventory obtains the best results. Since there are more city center stations in GT2 and the network is more complex, the flexibility of the rebalancing fleet has more advantages over the fixed one. In general, O-D variables seem to work better than the station-based trip variables for GT1 and GT3 with only one city center.

Table A.9 O-D model with different initial settings (60 stations, 80 available bikes for 4 trucks, 30 mins)

	Initial	Initial	O.F.	Time	MIP	Lost Demand (%)	
	Location	Inventory	Value	(mins)	Gap (%)	Rental	Return
GT1	Fixed	Fixed	52.8	<1	0.00	7.98	5.97
	Fixed	Flexible	52.8	<1	0.00	8.31	8.40
	Flexible	Fixed	52.4	<1	0.00	7.73	4.76
	Flexible	Flexible	52.5	<1	0.00	8.38	7.68
GT2	Fixed	Fixed	51.5	288	0.00	9.95	2.27
	Fixed	Flexible	51.6	289	0.00	9.82	2.35
	Flexible	Fixed	50.9	<1	0.00	9.36	2.07
	Flexible	Flexible	50.8	<1	0.00	9.33	2.02
GT3	Fixed	Fixed	227.2	1440	2.44	19.79	21.33
	Fixed	Flexible	223.2	1440	2.53	19.92	21.19
	Flexible	Fixed	207.5	1440	2.74	16.00	18.72
	Flexible	Flexible	208.6	1440	2.64	17.08	18.32

A.6.2 Initial Station Inventory and Trip Variable Types

The static rebalancing (Baseline 2) model is presented as (A.13)–(A.17).

$$\min \quad \sum_{t,s}(f_s^{+,t} - x_s^{+,t}) + \sum_{t,s}(f_s^{-,t} - x_s^{-,t}) \tag{A.13}$$

$$d_s^{t+1} = d_s^t - x_s^{+,t} + x_s^{-,t} \qquad \forall\ s \in S,\ t \in T \tag{A.14}$$

$$\sum_s d_s^1 = n \tag{A.15}$$

$$0 \le d_s^t \le C_s \qquad \forall\ s \in S,\ t \in T \tag{A.16}$$

$$0 \le x_s^{+,t} \le f_s^{+,t}, 0 \le x_s^{-,t} \le f_s^{-,t}, \qquad \forall\ s \in S,\ t \in T \tag{A.17}$$

where n is the total number of bikes in the system.

The results for the O-D model with baseline 1 and baseline 2, as well as for GT3 are illustrated in Table A.10.

Table A.10 O-D model with baseline 1 and baseline 2 and station-based model for GT3 (60 stations, 4 trucks, 30 mins)

	Baselines, Configuration, Trip Modeling	O.F. Value	Opt. Time (mins)	MIP Gap (%)	Lost Demand (%)	
					Rental	Return
GT1	Baseline 2 dyn.rebal. O-D	52.8	<1	0.00	7.98	5.97
GT2	Baseline 2 dyn.rebal. O-D	51.5	288	0.00	9.95	2.27
GT3	Baseline 1 without rebal.	-	-	-	33.26	33.83
	Baseline 2 without rebal. (static)	-	-	-	28.42	33.02
	Baseline 1 dyn.rebal. station-based	232.0	1440	2.60	23.24	18.24
	Baseline 2 dyn.rebal. station-based	179.1	1440	2.46	20.33	21.67
	Baseline 2 dyn.rebal. O-D	227.2	1440	2.44	19.79	21.33

The O-D variable model seems to provide slightly less lost demand, but at the cost of larger computing time due to increased model size.

A.6.3 Time Discretization and Time Constraints

Table A.11 shows the results of GT3 with different time-period lengths and time constraints. The conclusion is consistent with Table 3.7 in Section 3.5.3.

Table A.11 Station-based model for GT3 with/without time constraints in 30/60 mins (60 stations, 4 trucks)

	Time Period (mins)	Time Constraints	O.F. Value	Time (mins)	MIP Gap (%)	Lost Demand (%)		Opt-sim-gap (%)	
						Rental	Return	Rental	Return
GT3	30	No	179.1	1440	2.46	20.33	21.67	16.56	49.23
	30	Yes	272.5	1440	4.69	19.29	20.57	12.58	38.29
	60	No	408.4	1440	1.00	21.28	22.65	11.57	34.98
	60	Yes	409.0	1440	1.73	21.66	22.40	12.00	35.28

We now explore the impacts of time constraints and time-period length for the O-D variable model.

The results for the model with O-D variables are summarized in Table A.12. The conclusion is similar to the station-based trip variable model. However, when time constraints are applied, the running times for the models are much longer and the experiments for GT2 run out of memory.

We also carry out the same experiments on a 30 stations network. The results for them are shown in Table A.13. Given that most of the stations can be reached within 30 minutes,

Table A.12 O-D model with/without time constraints in 30/60 mins (60 stations, 4 trucks)

	Time Period (mins)	Time Constraints	O.F. Value	Time (mins)	MIP Gap (%)	Lost Demand (%)		Opt-sim-gap (%)	
						Rental	Return	Rental	Return
GT1	30	No	52.8	<1	0.00	7.98	5.97	5.03	11.73
	30	Yes	53.0	465	0.00	7.95	6.43	4.95	12.18
	60	No	71.9	1440	0.09	8.40	4.74	4.69	9.94
	60	Yes	71.9	1440	0.10	8.44	4.03	4.75	9.17
GT3	30	No	227.2	1440	2.44	19.79	21.33	13.89	44.67
	30	Yes	328.0	1440	6.05	20.85	21.22	10.60	40.32
	60	No	494.3	1440	3.44	22.52	21.69	5.70	34.94
	60	Yes	496.0	1440	3.88	22.50	20.61	5.57	32.92

time constraints are less effective. Compared to Table 3.7, a short time-period and time constraints are a good combination to guarantee enough rebalancing operations and timely arrivals for a larger studying area. However, for small-scale BSSs, like 30 densely distributed stations, it is not worth applying time constraints, resulting in a marginal improvement and a long optimization time.

Table A.13 Station-based variable model with/without time constraints in 30/60 mins (30 stations, 2 trucks)

	Time Period(mins)	Time Constraints	O.F. Value	Time (mins)	MIP Gap (%)	Lost Demand (%)		Opt-sim-gap (%)	
						Rental	Return	Rental	Return
GT1	30	No	0.5	<1	0.00	8.95	4.73	9.83	11.80
	30	Yes	0.5	<1	0.00	8.90	4.21	9.76	11.17
	60	No	5.3	<1	0.00	9.12	2.55	9.07	9.76
	60	Yes	5.3	<1	0.00	8.93	1.35	8.85	2.03

A.6.4 Trip Distribution and Variables Domains for O-D Model

The experiments for GT3 of the station-based model are carried out and demonstrated in Table A.14. Although GT3 is hard to solve, Constraints (TD1) perform best in lost rental as in Table 3.8.

Concerning the model with O-D variables as shown in Table A.15, adding trip distribution constraints is even harder to solve due to a large number of variables and constraints. Using Constraints (TD7) reduces both the lost rental and the opt-sim-gap. However, the running time is significantly longer.

We finally present the results for trip distribution constraints with different variable domains for the O-D model in Table A.16. Two instances out of five cannot be solved in the All-

Table A.14 Station-based model for GT3 with different trip distribution constraints (60 stations, 4 trucks, 30mins)

	Constraints	O.F. Value	Time (mins)	MIP Gap (%)	Lost Demand (%)		Opt-sim-gap (%)	
					Rental	Return	Rental	Return
GT3	(TD1)	875.8	1440	5.21	17.71	26.18	3.19	13.49
	(TD6)	205.7	1440	1.75	19.93	22.20	16.31	46.16
	(TD3)+(TD6)	304.0	1440	2.71	19.62	20.94	11.61	37.83
	(TD3)+(TD4)	373.7	1440	4.03	20.42	22.48	8.67	39.39
	None	179.1	1440	2.41	20.33	21.67	16.56	49.23

Table A.15 O-D model with different trip distribution constraints(60 stations, 4 trucks, 30mins)

	Constraints	O.F. Value	Time (mins)	MIP Gap (%)	Lost Demand (%)		Opt-sim-gap (%)	
					Rental	Return	Rental	Return
GT1	(TD7)	55.3	1440	4.90	6.86	6.50	3.65	10.90
	None	52.8	<1	0.00	7.98	5.97	5.03	11.73
GT2	(TD7)	53.3	869	0.07	7.94	2.10	4.84	7.11
	None	51.5	288	0.00	9.95	2.27	7.24	9.75
GT3	(TD7)	248.7	1440	4.90	19.79	22.06	12.94	44.85
	None	227.2	1440	2.42	19.79	21.33	13.89	44.67

continuous model within 24 hours under constraints (TD7) for both the All-continuous model and the Partially-integer model. The Partially-integer model outperforms the All-continuous model with an obvious improvement in lost demand with or without (TD7). Since the MIP gap is really close to 0.01%, we can conclude that trip distribution constraints (TD7) give the best performance. The All-integer model for O-D variables is not considered here since its integer requirement for trip variables $x_{s,s'}^{t,t'}$ are too strict.

Table A.16 O-D model with different variable domains and trip distribution constraints for GT1 (30 stations, 2 trucks, 30 mins)

Variable Domain	Const-raints	O.F. Value	Time (mins)	MIP Gap (%)	Lost Demand (%)		Opt-sim-gap (%)	
					Rental	Return	Rental	Return
All-continuous	(TD7)	23.0	585	0.03	8.37	4.11	5.37	9.89
	None	22.7	288	0.00	10.07	4.60	7.39	12.58
Partially-integer	(TD7)	23.0	864	0.03	4.12	1.40	0.72	2.16
	None	22.7	288	0.00	7.26	4.40	4.12	8.92

A.6.5 Sequences of Events

The results of event sequences for GT3 are shown in Table A.17. Compared to Table 3.10, sequences (d)(r)(a), (r)(d)(a), and (d)(a)(r) are still perform well.

Table A.17 Station-based model with different sequences of events (60 stations, 4 trucks, 30 mins)

	Sequences	O.F.	Time	MIP	Lost Demand(%)		Opt-sim-gap(%)	
	of events	Value	(mins)	Gap (%)	Rental	Return	Rental	Return
GT3	(r)(a)(d)	246.9	1440	2.35	19.57	21.98	15.14	41.38
	(a)(d)(r)	295.4	1440	2.32	20.59	20.63	14.24	39.82
	(d)(r)(a)	302.8	1440	3.22	19.22	21.70	8.94	41.28
	(r)(d)(a)	280.3	1440	2.94	19.60	20.68	11.93	39.47
	(d)(a)(r)	318.8	1440	3.12	19.23	19.74	8.57	36.62
	(a)(r)(d)	259.6	1440	1.91	19.45	21.89	15.08	39.42
	(r)(a+d)	236.7	1440	2.39	19.41	22.60	15.27	42.86
	(a+d)(r)	285.0	1440	2.24	19.87	19.70	13.46	38.00

APPENDIX B APPENDIX FOR THE SECOND ARTICLE

B.1 Trip Prediction and Inventory Interval

The methodology for predicting trips and calculating inventory intervals and target inventories is adapted from the model presented by [54]. The rentals and returns ($f_s^{+,t}$ and $f_s^{-,t}$) are predicted on an hourly basis for each station. The model utilizes a Gradient Boosting Tree, which incorporates weather conditions (temperature and humidity) and temporal information (the day of the week, hour of the day, and holidays) as learning features. In this model, a Singular Value Decomposition (SVD) technique is applied to reduce the dimension of the trip data. This process results in faster predictions, enhancing tractability when dealing with an elevated number of stations. The SVD also elevates the accuracy of the model, indicated by a lower Root Mean Square Error (RMSE), as it effectively eliminates noise and outliers from the trip data. Figure B.1 illustrates the model's pipeline.

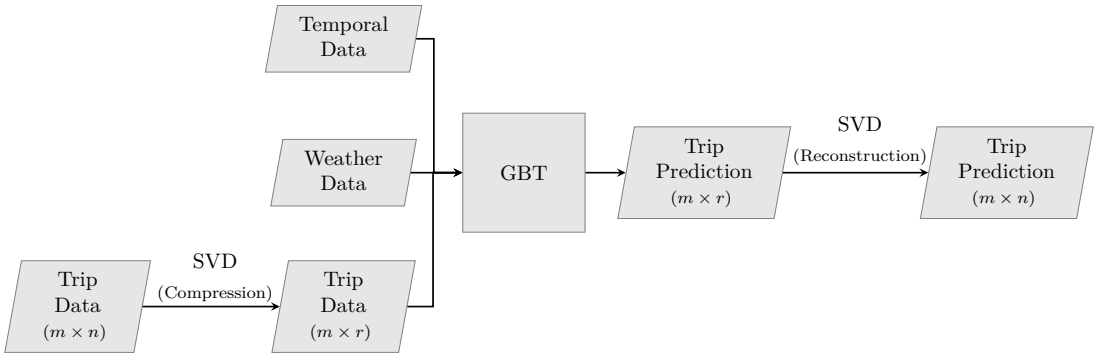

Figure B.1 Pipeline of the predictive model

Based on the predicted rental and return demand, the expected proportion of satisfied trips, known as *service level*, is computed for a given initial inventory. Assuming a station s with initial inventory i and capacity C_s, the rental and the return service levels for a time period $[t, t+\Delta]$ are computed as:

$$SL_s^{+,t}(i) = \frac{\int_t^{t+\Delta} f_s^{+,t}(1 - p_s^t(i,0))dt}{\int_t^{t+\Delta} f_s^{+,t}dt} \tag{B.1}$$

$$SL_s^{-,t}(i) = \frac{\int_t^{t+\Delta} f_s^{-,t}(1 - p_s^t(i,C_s))dt}{\int_t^{t+\Delta} f_s^{-,t}dt}, \tag{B.2}$$

where $f_s^{+,t}$ and $f_s^{-,t}$ represent the rental and return rates, respectively, for station s at time t. Further, $p_s^t(i,0)$, and $p_s^t(i,C_s)$ represent the probability that station s becomes empty and

full, respectively, given an initial inventory i at time t.

The *overall service* level is computed as (B.3)

$$SL_s^t(i) = min\{SL_s^{+,t}(i), SL_s^{-,t}(i)\} \tag{B.3}$$

The minimum and maximum service levels, for a station s in time period $[t, t+\Delta]$, can then be computed depending on the initial inventory at time t as follows:

$$SL_s^{min,t} = min_{i\in\{0,...,C_s\}}(SL_s^t(i)) \tag{B.4}$$

$$SL_s^{max,t} = max_{i\in\{0,...,C_s\}}(SL_s^t(i)). \tag{B.5}$$

A threshold Ω is created to establish an acceptable service level for the time horizon $[t, t+\Delta]$:

$$\Omega_s^t = SL_s^{min,t} + \beta(SL_s^{max,t} - SL_s^{min,t}), \tag{B.6}$$

in which the hyperparameter β controls the proximity of the threshold Ω_s^t to either the minimum service level or the maximum service level. In practice, this hyperparameter influences the gap between the upper and the lower bound of the inventory interval.

The inventory interval for station s for time period $[t, t+\Delta]$ is then defined as

$$\mathcal{I}_s = \{i \in \{0, ..., C_s\} | \mathcal{L} \leq i \leq \mathcal{U}\}, \tag{B.7}$$

where $\mathcal{L} = \min\{i \in \{0, ..., C_s\} | SL_s^t(i) \geq \Omega_s^t\}$, and $\mathcal{U} = \max\{i \in \{0, ..., C_s\} | SL_s^t(i) \geq \Omega_s^t\}$. Finally, the target inventory for station s at time-period $[t, t+\Delta]$ is set to the initial inventory that results in the maximum service level (i.e., $SL_s^{max,t}$).

B.2 Weather Generator

The weather generator in Figure 4.3 utilizes normal distributions derived from the temperature and humidity differences between consecutive hours throughout the day, capturing the change in weather conditions over time.

Histograms of the temperature differences observed between two consecutive hours of 4 periods during the day (0 am – 5 am, 6 am – 11 am, 12 pm – 5 pm, and 6 pm – 11 pm) are depicted in Figure B.2. The overlaid red curves illustrate the normal distributions in which the parameters are computed using the Maximum Likelihood Estimator.

Figure B.2(A) and Figure B.2(C) display narrower distributions, suggesting less variability

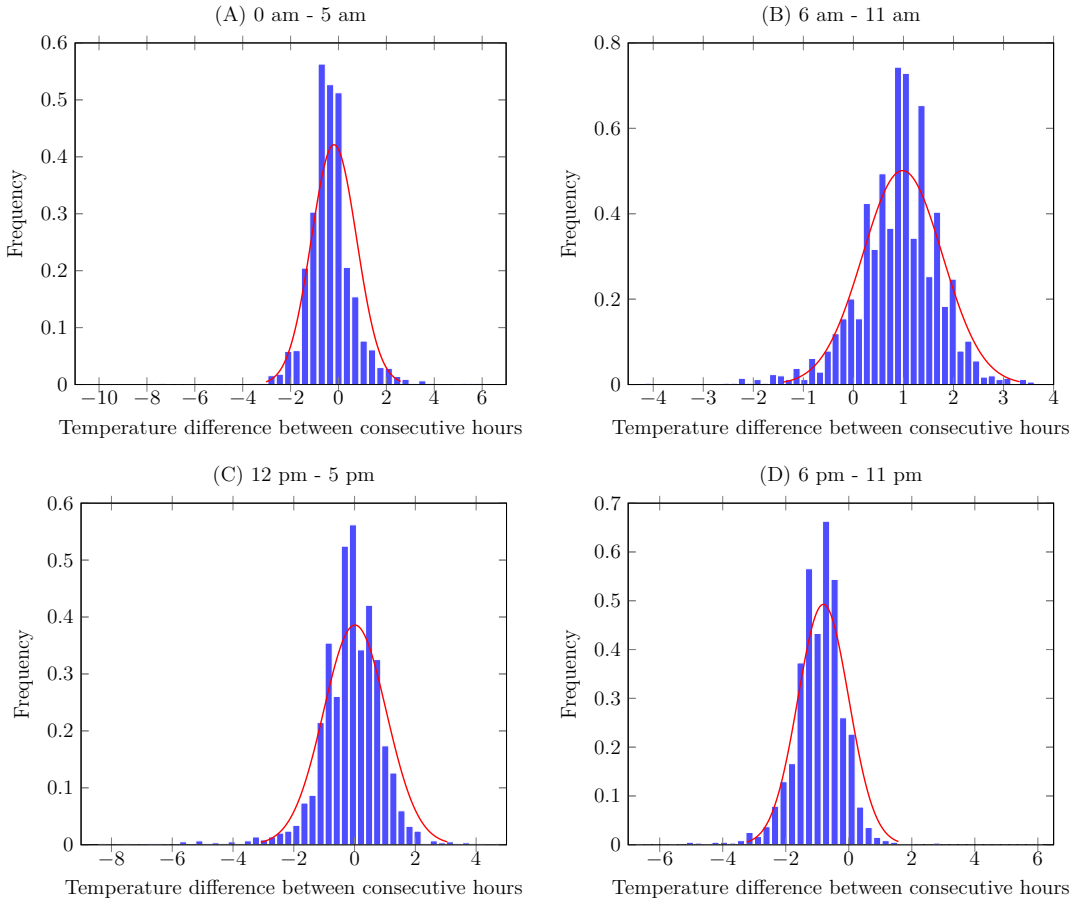

Figure B.2 Distributions of temperature changes between consecutive hours for four time-periods

in temperature change, whereas Figure B.2(B) and Figure B.2(D) exhibit wider spreads, indicating greater fluctuation. The distributions for humidity differences are obtained using the same approach.

B.3 Experimental Results on BIXI Cluster

The daily lost demand of models DROB-LD and DROB-I on the considered BIXI cluster are detailed in Table B.1.

Table B.1 Lost demand for DROB-LD and DROB-I on BIXI cluster

Day	Model	Opt.Time (min)	MIP Gap (%)	Lost demand (%)		
				Rental	Return	Total
Day 1	DROB-LD	0.00	0.00	0.85	7.52	4.40
	DROB-I	0.01	0.00	0.00	2.26	1.20
Day 2	DROB-LD	0.66	0.00	0.50	0.55	0.52
	DROB-I	0.41	0.00	0.50	0.55	0.52
Day 3	DROB-LD	5.41	0.00	0.73	0.98	0.85
	DROB-I	0.01	0.70	0.73	1.96	1.33
Day 4	DROB-LD	0.06	0.00	2.96	1.51	2.27
	DROB-I	0.01	0.00	2.09	1.51	1.81
Day 5	DROB-LD	0.02	0.00	1.74	2.33	2.01
	DROB-I	0.01	0.00	1.90	4.83	3.27
Day 6	DROB-LD	0.01	0.00	0.16	0.18	0.17
	DROB-I	0.01	0.00	0.00	1.41	0.66
Day 7	DROB-LD	0.04	0.00	0.35	2.92	1.56
	DROB-I	0.01	0.00	0.17	3.31	1.65
Day 8	DROB-LD	0.01	0.00	2.66	0.65	1.76
	DROB-I	0.01	0.00	2.66	2.39	2.54
Day 9	DROB-LD	0.09	0.00	1.75	0.80	1.30
	DROB-I	0.01	0.00	1.92	1.59	1.77
Day 10	DROB-LD	0.02	0.00	2.90	2.49	2.71
	DROB-I	0.01	0.00	2.90	2.49	2.71
Day 11	DROB-LD	0.01	0.00	1.71	0.73	1.23
	DROB-I	0.01	0.00	1.54	1.63	1.58
Day 12	DROB-LD	0.02	0.00	0.65	1.65	1.12
	DROB-I	0.01	0.00	1.13	0.74	0.95
Day 13	DROB-LD	0.13	0.00	3.58	1.42	2.52
	DROB-I	0.03	0.00	2.21	0.18	1.22
Day 14	DROB-LD	0.01	0.00	2.62	1.32	1.99
	DROB-I	0.01	0.00	2.62	1.81	2.23
Day 15	DROB-LD	0.01	0.00	0.85	2.21	1.50
	DROB-I	0.01	0.00	0.68	0.37	0.53
Day 16	DROB-LD	0.02	0.00	0.53	3.03	1.81
	DROB-I	0.02	0.00	0.00	3.03	1.55
Day 17	DROB-LD	0.11	0.00	2.56	1.12	1.84
	DROB-I	0.01	0.00	2.28	0.00	1.13
Day 18	DROB-LD	0.36	0.00	2.80	0.76	1.79
	DROB-I	0.03	0.00	2.24	3.82	3.02
Day 19	DROB-LD	0.01	0.00	2.34	0.20	1.29
	DROB-I	0.01	0.00	0.78	1.62	1.19
Day 20	DROB-LD	0.01	0.00	0.40	1.08	0.72
	DROB-I	0.01	0.00	1.00	0.43	0.72