Dynamic Resource Redistribution and Demand Estimation: An Application to Bike Sharing Systems

Konstantina Mellou
Operations Research Center, Massachusetts Institute of Technology, kmellou@mit.edu

Patrick Jaillet
Operations Research Center, Massachusetts Institute of Technology, jaillet@mit.edu

Shortage of bikes and docks is a common issue in bike sharing systems. To tackle this problem, operators use a fleet of vehicles to redistribute bikes across the network. We propose a model that captures successful user trips in the system, and a new mixed integer programming formulation that solves the dynamic redistribution problem by producing routes and pick-up/drop-off decisions for the vehicles. In order to scale to large instances, we develop a decomposition method based on proper station grouping, accompanied by an optimization with partial information approach, where relevant information for each group (routing and redistribution options) is modeled using piecewise linear concave functions and explicitly included in the model. We test our methods on both synthetic and real-world data, and show that our algorithms can scale to large real-world systems, with short running times that allow for real-time information to be taken into account. Furthermore, since accurate estimation of user demand is essential for efficient redistribution, we also develop data-driven and optimization-based approaches to consider lost and shifted demand. Our methods are general and not tied to the specific application domain; for instance, the optimization with partial information can be applied to any pick-up and delivery vehicle routing problem.

Key words: bike-sharing, dynamic redistribution, demand estimation

History: February 15th, 2019, first version.

1. Introduction

Early precursors of vehicle sharing systems started emerging more than half a century ago, presenting limited success at first, but slow growth, which led to the worldwide phenomenon that is observed in the present years. Millions of people have acknowledged the benefits of these systems and shown their support by substituting a smaller or larger part of their everyday commute with some form of ridesharing. The most common types of vehicles that are met worldwide are cars and bicycles, with recent advances in technology, such as electric vehicles, being gradually incorporated as well.

The idea behind vehicle sharing systems is simple; commuters are offered the benefit of a short-period vehicle rental in order to perform one-way or round trips. Vehicles can be picked-up and returned at specific locations, which are spread across the cities in order to increase the range of service. The charge depends on the duration of the trip, as well as the type of membership selected by the user. Operators usually offer monthly and annual memberships, which are ideal for regular users, as well as daily or short-period memberships designed to facilitate visitors and casual users.

The main purpose of this research is to provide the operator of a vehicle sharing system with an approach to manage major operations that are required for the smooth functioning of the system, by utilizing historic and real-time data. In this work, we assume that the system is already functional, in the sense that we do not consider strategic decisions that need to be made while the system is being constructed. These include the locations and capacity of stations, the number of vehicles, etc., and thorough planning is required in order to ensure that the system specifications are adequate to cover most of the customer demand.

After a system launches, the operator faces many daily challenges in order to keep the service at a high level and meet the expectations of the users. Let us now focus on the particular case of the bike sharing systems, which will allow us to provide more details about specific operations. In a typical bike sharing system, users can pick up a bike from any station with available bikes and return it to any station with an empty dock. However, large flows of riders from residential to industrial and commercial areas in the morning and opposite flows in the evening, as well as many one-way trips, can render the system highly unbalanced in many instances during the day. In particular, very often there are phenomena of *starvation*, when customers wish to pick up a bike from a station that is empty, and *congestion*, when customers want to return their bikes to a station with no empty docks.

In order to deal with this problem, companies deploy a fleet of trucks which perform the *rebalancing* of the network: they pick up bicycles from stations with a surplus of bikes and drop them off at stations that currently require them. Efficient rebalancing is of crucial importance to the operating company. If a customer repeatedly fails to find a bike or is not able to drop it off at their desired destination, they will eventually seek alternative modes of transportation, leading to significant losses for the company and endangering the viability of the system.

The goal of this work is to generate a complete plan of actions for the operating crew that will cover all daily rebalancing operations. An important factor for the success of this project is the plethora of data that is regularly gathered by the operators. Historic data, such as past trips and system statistics, will be used to accurately estimate the user demand, which can in turn be used to generate strategies that optimize the system's performance. At the same time, real-time tracking of all components of the system allows to dynamically readjust daily plans, based on the currently realized scenarios.

The motivation and significance of this research might be obvious from the view of the operator. However, the impact of a viable vehicle sharing system is much greater than that. If users can rely on such a system covering their transportation needs, vehicle ownership statistics will be reduced, decreasing the consumption of valuable natural resources that a vehicle manufacturing requires. Car sharing can solve problems that big cities often face, such as lack of parking spaces. In addition, a transition to using bike sharing systems can contribute significantly to dealing with the problem of congestion, and, at the same time, reduce harmful gas emissions in the atmosphere. A success in the operation of the existing vehicle sharing systems will likely motivate more and more cities to adopt similar transport behaviors, turning them from energy consumers and pollution generators into more eco-friendly transportation networks.

From a theoretical perspective, this topic raises many interesting research questions. The high dimensionality of the system - large number of stations, time intervals of interest, options of routing - renders it intractable for real-life instances. The answer to this type of questions is relevant to many systems, whose operation requires matching demand and supply, especially in the context of a dynamic environment. This might refer to resources that need to travel to various locations in order to serve customers requests, as well as many flexible transportation systems that have currently emerged or been planned for the near future.

The contributions of this work can be summarized along three main axes:

1. We present new models for estimating the actual user demand of bike sharing systems. These include both the lost demand due to lack of bikes and docks at the stations, as well as the shifted demand which is the result of users walking to nearby stations in order to find available resources when none are available at their current location. Both of these aspects are very often not considered in bike sharing literature, but their effect on the rebalancing efficacy is crucial since redistribution plans heavily depend on reliable estimation of the demand.

2. We propose a linear programming model that captures the bike flows that result from all trips that are performed by users throughout the network. This model can be used instead of trip simulations to evaluate the efficiency of the system. It does not depend on the exact arrival order of the users, which is often difficult to estimate, as simulation scenarios might, but instead on the total number of users per time period. Moreover, the fact that this is a linear program makes it possible to be used as a component of more complex models, thus incorporating user flows with other problem aspects.

3. We introduce a new mixed integer programming formulation to solve the dynamic rebalancing problem. We suggest decomposition techniques based on appropriate station grouping and incorporation of partial information for each group to the global rebalancing model. This achieves smaller model sizes that still contain enough information for each station group, and we manage to produce rebalancing plans that scale to real-world instances of actual bike sharing systems. The core ideas of our approach can be extended to dockless bike-sharing (or scooter-sharing) systems by proper space discretization, as well as be applied to the solution of any pick-up and delivery vehicle routing problem.

In the sections that follow, we start by presenting an overview of the literature on bike sharing systems and existing rebalancing approaches. Then, in Section 3, we introduce a model that accounts for lost and shifted demand given historic data of past trips. Section 4 focuses on modeling the user trips across the network, and Section 5 studies the rebalancing of the system. Computational experiments follow in Section 6 to evaluate the efficacy of the methods.

2. Related Work

Research in the area of Bike Sharing Systems (BSS) is relatively recent. DeMaio (2009) presents a comprehensive history of BSS, and Shaheen et al. (2010) discuss their evolution around the world, their business models, as well as their social and environmental effects. For a review of BSS, readers can also refer to Fishman (2015), which includes aspects such as growth, usage, and impact. Lin and Yang (2011), Martinez et al. (2012) and Nair and Miller-Hooks (2014) focus on BSS design, identifying the number of stations that are required and their optimal locations across the city, while Freund et al. (2017) modify already operational systems by reallocating their docks to increase their efficiency. The sections that follow focus on the specific themes of the BSS literature that are closer to our work.

Performance Analysis and Decision Support. This first stream of research is peripheral to our work and mainly focuses on providing insights that help decision making in BSS. Shaheen et al. (2011) conduct a survey to better understand the early adoption of BSS and users' travel behavior. Raviv et al. (2013) model the user behavior in the face of lack of resources and they introduce a user dissatisfaction measure to evaluate station performance. Tao and Pender (2017) conduct an analysis on various performance measures for bike distribution across the system, while Vogel and Mattfeld (2010) study the effect of various levels of redistribution efforts on user satisfaction.

User Demand. Accurate demand modeling is essential for the efficiency of the redistribution process. Shu et al. (2013) model the number of customers traveling between each pair of stations at each time period as a Poisson process. A similar approach is employed by Nair and Miller-Hooks (2011), after they observe that in the dataset of a Singapore car-sharing system they use, both the number of vehicles leaving a station and arriving at a station at each time period can be accurately modeled with Poisson distributions. Vogel et al. (2011) cluster stations with similar behavior throughout the day and use data from realized trips aggregated per hour to generate typical customer demand. Finally, Borgnat et al. (2011) and Singhvi et al. (2015) take into account various parameters, such as information about the weather, special events in the area, taxi usage, to forecast the number of rentals across the system.

Our work differs from many of these approaches in that we do not focus on modeling demand based on observed trips or predicting future demand. Instead, we wish to reveal the actual underlying user demand of the system by taking into account trips that were not accomplished because of resource shortages. Lack of resources leads to customers leaving the system or walking to nearby stations. Considering only the completed trips leads to a misrepresentation of the actual demand, since this user behavior can affect the resources that are required at each station. O'Mahony and Shmoys (2015) consider only the first of these two aspects (users that left the system) and obtain an estimation for this lost demand by examining the usual behavior of the stations. In our approach, we incorporate an extra dimension, the daily demand trends of each station, and show that this leads to decreased estimation error in all instances that we examined. Finally, we provide a method to estimate the shifted demand of the system, and obtain in the end the actual user demand for each station. This shifted demand is also considered in the recent work by Goh et al. (2019). The authors propose a rank-based choice model to reveal the primary user demand of the system, which differs from our work as we follow a linear programming approach to estimate the probability of walking between stations.

Redistribution without Routing. A different stream of research focuses on the redistribution procedure. Some studies aim to propose the desired configuration of the system at each time period based on the anticipated demand and identify the number of vehicles that need to be carried from one station to the other in order to achieve it. Nair and Miller-Hooks (2011) present a MIP that involves joint chance constraints to determine the least cost redistribution that will meet most of the upcoming demand in the application of car-sharing systems. Shu et al. (2013) formulate a linear program to estimate the performance of the system and the benefits of redistribution. Both of these approaches assume that redeployment of vehicles between any two stations is always possible, which might not be the case in an actual vehicle sharing system. Moreover, since they do not take into account the exact routing of the carriers, they do not provide a detailed plan for the execution of the redistribution. This contrasts with our approach which produces a complete routing and redistribution plan.

Static Redistribution. Another set of studies tackles the problem of static rebalancing with routing, where the goal is to find the optimal routing of the carriers that will achieve a desired configuration, considering no customer demand during the redistribution. This is the case of systems where redistribution is performed only at the end of each day, in order to prepare the system for the demand of the following day.

Schuijbroek et al. (2017) provide a MIP formulation for the problem, as well as a cluster-first route-second heuristic to reduce its running time. Raviv et al. (2014) wish to minimize user dissatisfaction in conjunction with the cost of repositioning. They introduce multiple formulations and techniques, such as arc deletion when alternative paths of same cost exist, which allow them to solve the problem in reasonable time even for larger instances. Rainer-Harbach et al. (2013) propose a variable neighborhood search approach to generate candidate routes for the carriers, while Di Gaspero et al. (2013) present a hybrid method that combines constrained programming with ant colony optimization. Chemla et al. (2013) propose an integer programming formulation and they use a branch-and-cut algorithm to solve its relaxation and tabu search to obtain an upper bound on the optimal solution of the problem. A tabu search approach is also proposed by Ho and Szeto (2014) in order to solve the static rebalancing problem and obtain good quality solutions in short running times. Dell'Amico et al. (2014) suggest multiple formulations for the problem and a branch-and-cut solving approach, while Forma et al. (2015) present a 3-step heuristic to obtain a static repositioning plan for the system.

The research described in this section differs from our approach on the assumption that the system is not being used by customers while the rebalancing takes places. In our work, we consider the system being in full operation during the redistribution procedures, and this complicates the problem further since user movements keep changing the state of the system and this needs to be taken into account.

Dynamic Redistribution. In the dynamic problem, redistribution is performed multiple times during the day and demand is not neglected when rebalancing takes place. Ghosh et al. (2017) propose a MIP formulation for the problem, which they solve by using Lagrangian Dual Decomposition combined with an abstraction approach in the case of large instances. Contardo et al. (2012) focus on the peak hours of the day. They formulate the problem and use two decomposition methods, Dantzig-Wolfe and Benders decomposition, to obtain lower and upper bounds for medium and large instances. O'Mahony and Shmoys (2015) study the rebalancing problem both for the (static) overnight version as well as during the rush hour. They propose a matching approach that matches stations that need bikes with stations that need docks in order to transfer the resources between them. Caggiani and Ottomanelli (2013) also provide an optimization model to reduce the redistribution costs which they test on small system instances.

Our work also tackles the dynamic rebalancing of the network. However, our goal is to provide solutions that can scale to large real-world systems in running times that are fast enough to allow real-time information to be taken into account. In our experiments, we take this type of information into account every thirty minutes of the planning horizon, but this can take place in much shorter time intervals (especially during rush hours) so as the most up-to-date state of the system is always considered.

3. Demand Estimation

While BSS operators record a variety of data about the system such as outages and past trips data, an important component remains missing; *lost user demand.* This demand occurs whenever a user is unable to complete a desired trip due to either the absence of bikes at the origin station, or the absence of docks at the destination, and they decide to leave the system without using it. This behavior may cause inadequate planning on the system operator's part, since this user intention is never recorded, hence leading to instances of underestimating the actual demand.

Another natural consequence of encountering empty and full stations is the *shift of demand* towards nearby locations. Overlooking this event results in overestimating the number of unserved customers, since some of them have actually been satisfied from other locations. Moreover, this phenomenon causes an artificial demand increase for stations at the receiving end of this shift. This will manifest in a greater need of resources, misleading the rebalancing operations. Therefore, we now focus on identifying the extent of this lost and relocated demand, in order to determine the actual demand of the system.

3.1. Observed Demand

The *observed demand* consists of all trips that are completed in a system. For each pair of stations (i,j) and time periods (t,t'), let $d_{i,j}^{t,t'}$ denote the number of users picking up a bike from station i at time t and returning it to station j at time t'. The observed demand is known to the BSS operators, since the successful trips are recorded in the system data. Using this information, we can also calculate the *observed outgoing (resp. incoming) demand* from station i at time t, which is defined as the total number of users who left (resp. arrived at) station i at time t, $d_{out,i}^{t} = \sum_{j,t'} d_{i,j}^{t,t'}$ (resp. $d_{in,i}^{t} = \sum_{j,t'} d_{j,i}^{t',t}$).

3.2. Lost Demand

Lost demand refers to the users who failed to perform a trip due to lack of bikes at their origin station or lack of docks at their desired destination. For each pair of stations (i,j) and periods (t,t'), we use $ld_{i,j}^{t,t'}$ for the number of users that failed to go from station i at time t to station j at t' due to lack of resources at i or at j. The *lost outgoing demand* $ld_{out,i}^{t}$ and *lost incoming demand* $ld_{in,i}^{t}$ are defined similarly to the previous section, but using the lost demand instead of the observed one.

In the following analysis, we focus on the lost outgoing demand, but the same methods can be applied for the estimation of the lost incoming demand. Consider a station i which is empty from time t_s to time t_e on a particular day. Naturally, there is no observed outgoing demand during that time. However, there is no way to discover with certainty whether this is because there are no users wanting to depart from that station or whether the lack of bikes prohibits any users to do so. It turns out that the usual behavior of the station that time of the day (*average station behavior*) as well as the behavior around the interval when bikes are available (*daily demand trends*) can offer some intuition.

3.2.1. Average Station Behavior. The usual station behavior can be used as an indicator of the number of users that wish to depart from the station. This is based on the assumption that days do not differ much from one another, as well as the fact that a station is not empty every day at the same time. For the outage interval $[t_s, t_e]$ of station i, consider the average observed demand at $[t_s, t_e]$ at i on the days where i is not empty during $[t_s, t_e]$. Let that average demand be denoted by $\text{AVG\_}ld_{out,i}^{t_s,t_e}$, alternatively referred to as the average station behavior.

3.2.2. Daily Demand Trends. Another aspect to be considered is the daily trend of the demand. There are days that are slower, for example due to weather conditions, and others that are busier. By examining the user behavior right before and after the interval, we can get an estimate of the interpolated demand during the outage.

Definition 1. We define the *outgoing demand rate* $r_{out,i}(t_1, t_2)$ as the average number of users per minute that want to depart from station i during the interval $[t_1, t_2]$.

$$r_{out,i}(t_1, t_2) = \frac{1}{t_2 - t_1} \cdot \sum_{t=t_1}^{t_2} d_{out,i}^t \tag{1}$$

We then proceed to provide an estimate for the demand rate $r_{out,i}(t_s, t_e)$ for the outage interval $[t_s, t_e]$, by using the demand rate the hour right before and after this interval. In particular:

$$r_{out,i}(t_s, t_e) = \frac{1}{2}\left[r_{out,i}(t_s - 60, t_s) + r_{out,i}(t_e, t_e + 60)\right] \tag{2}$$

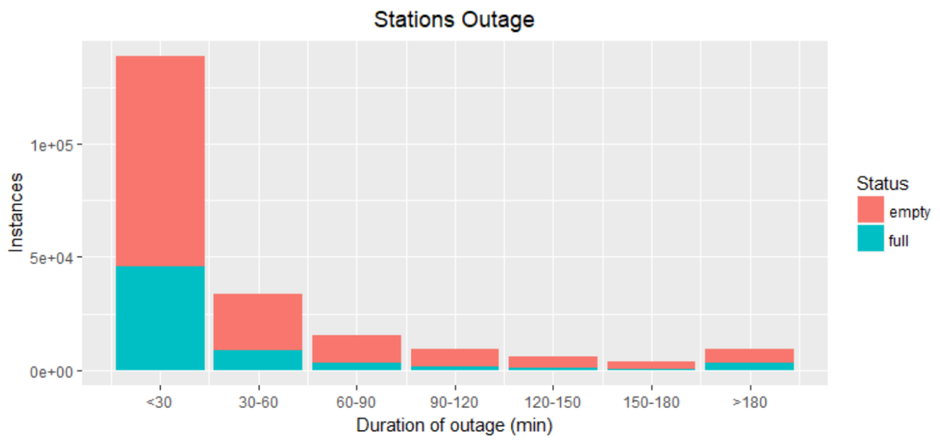

Figure 1 Duration of outage events. Most of the intervals of empty and full stations are short, lasting less than thirty minutes.

Assuming that there is no outage during the intervals $[t_s-60,t_s]$ and $[t_e,t_e+60]$, this can be calculated using (1). The opposite case is addressed later in this section. Then, an estimate of the lost demand based on the trends $\text{TREND\_}ld^{t_s,t_e}_{out,i}$ for the outage interval $[t_s,t_e]$ can be obtained as:

$$\text{TREND\_}ld^{t_s,t_e}_{out,i} = r_{out,i}(t_s,t_e)\cdot(t_e-t_s) \tag{3}$$

Outage duration. One assumption of this approach is that demand does not change drastically in a very short amount of time. Therefore, a well-founded concern is whether it can work well when the outage intervals are long, for example lasting a few hours. Computational experiments based on real data reveal that this is not an issue since the vast majority of outages last less than thirty minutes, as is illustrated in Figure 1, while many of the longer lasting outages take place during the night, where demand is already negligible. Note here that empty station intervals during low operation periods such as nighttime are not considered indicators of lost demand and are not taken into account.

Artificial outage endings. Consider a user returning their bike at an empty station that is very soon picked up by another customer. This action will cause the outage interval to be split in two parts, when essentially it is the same event. Thus, before estimating the lost demand, we merge outage intervals that are a few minutes apart. Any recorded users between the initial intervals are then deducted from the estimated lost demand of the merged interval.

Lack of information around the outage. We discussed how the outage demand rate can be calculated based on the demand rates of the hours preceding and following the event. In case that we do not have information about the complete hour due to another outage, we can compute the demand rate based on a shorter time interval or look slightly further in the future or past, where information becomes available.

3.2.3. Lost Demand Estimation. The lost demand estimate is calculated as a convex combination of the estimates obtained by the average station behavior and the daily demand trends.

$$ld^t_{out,i} = \lambda \cdot \text{ AVG\_}ld^t_{out,i} + (1-\lambda) \cdot \text{ TREND\_}ld^t_{out,i} \tag{4}$$

A data-driven approach is followed in order to select the value of λ that minimizes the MSE of the estimated demand. More details and results are presented in the Section 6 as part of the computational experiments.

The main characteristic of this approach is that we take into account both the average demand of the station as well as the specific demand trends of the day that encountered the outage. Since not all days are the same, by using information from the particular day, as well as focusing on the time periods that are closer to the outage interval, we can better estimate the system behavior during the outage, as is also illustrated in experiments on real-world datasets in Section 6.

This method implicitly considers various factors that influence biking. For instance, estimated lost demand in rainy days will probably be lower, since the traffic volume is generally reduced during these days. On the other hand, for outages that emerge in the middle of a rush hour, if the preceding observed demand is large, this will be captured in the estimation of the lost demand as well. Having calculated the number of unserved customers for each outage event, we assume that their arrival times are uniform over the outage interval and generate the lost demand trips.

3.3. Shifted Demand

The next step is to identify the extent of relocated demand in the system resulting from station outages. Let E_t denote the set of stations i that are empty at time t. For each station i, we define its *l-neighborhood* $N_{i,l}$ as the set of stations whose distance from i is at most l:

$$N_{i,l} = \{j | distance(i,j) \leq l\}, \tag{5}$$

and its *active l-neighborhood at time t* as the subset of $N_{i,l}$ that have the resources to serve relocated demand from station i at time t (i.e. are not empty in the case of outgoing demand):

$$N^t_{i,l} = \{j | j \in N_{i,l}, j \notin E_t\} \tag{6}$$

Shifting probability. We introduce a model for the probability of a user shifting to a different location. We assume that there is some limit l_{max} on the distance that users are willing to walk. So, if there is no station with available bikes close enough, users just leave the system without using it. In case such a station exists, users select to walk with probability p. More specifically, for station i and time t, the *probability of shifting* is given by:

$$P^t_i(\text{user shifts}) = \begin{cases} 0 \text{ if } N^t_{i,l_{max}} = \emptyset \\ p \text{ otherwise} \end{cases} \tag{7}$$

Shifting coefficients. If $|N^t_{i,l_{max}}| > 1$, the user has more than one alternative stations to walk to. In that case, they select each station with probability decreasing with respect to its distance. In particular, out of the customers that have selected to walk from station i at time t, the probability of them going to station j is given by:

$$q^t_{ij} = P^t_i(\text{goes to } j | \text{user shifts}) = \frac{1}{Z^t_i \cdot distance(i,j)}, \quad \forall i,t,j \in N^t_{i,l_{max}} \tag{8}$$

where $Z^t_i = \sum_{j \in N^t_{i,l_{max}}} 1/distance(i,j)$ is a scaling factor, so that the coefficients sum to 1.

Shifting distance multiplier. In this model, we assume that users can walk to any station with available resources in the l_{max} radius. However, when some of the candidate stations are really far compared to the others, a rational user will not consider them when selecting a station to walk to. In order to include this behavior in the model, assume the closest non-empty station to station i at time t is at distance $r^t_{min,i}$. A user will then walk only to stations up to distance $\alpha \cdot r^t_{min,i}$, where $\alpha \geq 1$ is a parameter of the model called *shifting distance multiplier*. If $\alpha = 1$, a user can only go to their closest available station, while if $\alpha = \infty$ a user can go to any station of the network within the allowed walking limit l_{max}. The integration of the shifting distance multiplier to the model we described so far requires simply the replacement of l_{max} with $\min(l_{max}, \alpha \cdot r^t_{min,i})$ in $N^t_{i,l_{max}}$ in equations (7) and (8).

In this model, users only walk to stations that currently have resources available. This assumes that users have a prior knowledge of the state of each station, which is true in most real-world systems since they offer real-time tracking of the stations that can inform users of the availability of bikes at each one of them.

In order to compute the value of p, we formulate the linear program *Shifted Demand* (SD). For each station i that is not empty at time t and has demand $d^t_{out,i} > 0$, we introduce a variable x^t_i representing the actual outgoing demand that was originally intended for that station, i.e. is not the result of customer movements from nearby empty stations. We also define for each station the average value of x^t_i over the same time of all remaining days where i is not empty, denoted by $\bar{x}_i(t)$. The model formulation follows.

$$\text{(SD)}: \quad \min \quad \sum_{i,t} |x^t_i - \bar{x}_i(t)| \tag{9}$$

$$\text{s.t.} \quad d^t_{out,i} = x^t_i + \sum_{j: i \in N^t_{j,l_{max}}} p \cdot q^t_{ji} \cdot ld^t_{out,j} \qquad \forall i,t : d^t_{out,i} > 0 \tag{10}$$

$$\bar{x}_i(t) = \text{avg}(x^t_i) \qquad \forall i,t \tag{11}$$

$$0 \leq p \leq 1 \tag{12}$$

According to constraint (10), the observed demand $d^t_{out,i}$ at each station consists of the demand x^t_i that was actually intended for that station combined with part of the demand for the nearby empty stations. The latter is the lost demand for these empty stations, which is denoted as $ld^t_{out,j}$ for station j, and can be computed according to the previous Section 3.2. Out of this lost demand, the probability of users going to station i is given by the probability p of people choosing to walk (instead of abandoning the system), multiplied by the probability q^t_{ji} that they select the specific station i to walk to.

Constraint (11) connects $\bar{x}_i(t)$ and x^t_i, but for simplicity of notation not all details are presented in the model: Each time t is a timestamp, consisting of a day and a time period, and $\bar{x}_i(t)$ is the average of x^t_i that refer to the same time period over all days for which station i is not empty. The objective function takes $\bar{x}_i(t)$ as a reference for each station i and time t, and considers the absolute differences of each x^t_i with the corresponding $\bar{x}_i(t)$. The goal is to minimize the sum of these absolute differences.

Solving (SD) provides a value for p which is the estimated probability that people walked to nearby stations. We now need to readjust the demand of the stations that received this relocated demand, since it was originally directed to other locations. For each station i with an observed demand $d^t_{out,i}$, the actual demand is given by $\max(x^t_i, 0)$.

The fact that $d^t_{out,i}$ includes some of the users for other stations may have influenced the demand trends and the average behavior based on which the lost demand was calculated in Section 3.2. Since $d^t_{out,i}$ might be larger than the actual demand for that station, this might lead to a slight overestimation of the lost demand. For that reason, one might consider running another iteration of lost demand estimation, as described in Section 3.2, considering now the actual demand $\max(x^t_i, 0)$, for each station i and time t.

4. Modeling User Trips

Before presenting our approach for the rebalancing optimization of the network, we propose a linear program which we use in order to model the user trips that are realized throughout the system. We assume that users perform a trip only if there is an available bike at their origin station and a dock at their destination; otherwise they do not wait but leave the system instead. We do not

Table 1 Notation used in the formulation of the model (UT).

Symbol	Interpretation
d_{ij}^t	Demand for bikes from node i to node j at time period t
f_{ij}^t	Bikes that are moved by customers from node i to node j at time period t
b_i^t	Number of bikes at station i at the beginning of time period t
C_i	Bike capacity of each station i
B	Total number of bikes in the system

allow demand shifting (people walking to nearby stations) in this model since we wish to use it to evaluate the efficiency of our methods and the goal is to eliminate the need for demand shifting through appropriate placement of resources.

Furthermore, we assume that each trip is completed within a single time period, which we define to be thirty minutes long. This assumption is not far from reality as 86.77% of trips in a five-month past trips dataset (May to September 2017 for Capital Bikeshare, Washington DC) that we tested indeed had duration less than thirty minutes. The model can still be generalized to longer trips by changing appropriately the indexing of the variables and constraints, which we will not attempt here for the benefit of simplicity of notation. Finally, we assume that each bike and dock is used by at most one customer per time period. A relaxation of this assumption is discussed in Appendix A.

The model notation is provided in Table 1 and the formulation follows.

$$\text{(UT)}: \quad \text{maximize} \quad \text{SUCCESSFUL\_TRIPS}(f) \tag{13}$$

$$\text{subject to} \quad f_{ij}^t \le d_{ij}^t \qquad \forall i,j,t \tag{14}$$

$$\sum_j f_{ij}^t \le \text{AVAILABLE\_BIKES}(i,t) \qquad \forall i,t \tag{15}$$

$$\sum_j f_{ji}^t \le \text{AVAILABLE\_DOCKS}(i,t) \qquad \forall i,t \tag{16}$$

$$b_i^{t+1} = b_i^t + \sum_j f_{ji}^t - \sum_j f_{ij}^t \qquad \forall i,t \tag{17}$$

$$b_i^t \le C_i \qquad \forall i,t \tag{18}$$

$$f_{ij}^t \le \frac{d_{ij}^t}{\sum_k d_{ik}^t} b_i^t \qquad \forall i,j,t \tag{19}$$

$$\sum_i b_i^0 \le B \tag{20}$$

$$f_{ij}^t, b_i^t \in \mathbb{R}_+, \qquad \forall i,j,t \tag{21}$$

According to (14), for each edge and time period the flow of bikes is bounded by the user demand. (15) and (16) require that a trip is only realized if there are available bikes at the origin station and available docks at the destination. Constraint (17) is a flow conservation constraint and (18) a capacity constraint for the bikes of each station. Constraint (19) is a proportionality constraint; since we do not consider the exact arrival order of users within the same time period, this fairness constraint ensures that the number of bikes traveling to each destination is proportional to the corresponding demand (idea based on Ghosh et al. (2017)). In (20) the number of bikes is bounded by the total bikes in the system B (this can be replaced with equality if all bikes must be used, otherwise the solution provides the optimal number of bikes in the system). Finally, if an initial configuration of the system is provided, this is taken into account by setting appropriately the values of b_i^0. In the opposite case, the optimal values of b_i^0 are calculated by solving the above LP. Notice that this is a relaxed version of the problem, i.e. the variables are real numbers and not integers. This is further discussed at the end of the section.

4.1. Available Bikes and Docks

If we assume that each bike and dock can be used only once per time period, then the available bikes for each period are the ones currently present at the station, and similarly for the docks.

$$\text{AVAILABLE\_BIKES}(i,t) = b_i^t \quad \forall i,t \tag{22}$$

$$\text{AVAILABLE\_DOCKS}(i,t) = C_i - b_i^t \quad \forall i,t \tag{23}$$

This is what will be used for the computational experiments and the remaining sections of this paper, but since this approach might be considered conservative, we present a way we can relax this assumption by considering the expected number of arrivals until a station becomes empty or full in Appendix A.

4.2. Objective Function

If we maximize the number of successful trips, while respecting constraints (14)-(21), we get a good representation of the trips that were performed in the system. This is the case because each time a user is looking for a bike or a dock and they find one available, then they are going to use it. So, in reality, the number of successful trips is the largest number of trips that can be performed given demand and availability constraints, as well as the first come first served rule.

As a result, one natural option for the objective function is $\text{SUCCESSFUL\_TRIPS}(f) = \sum_{t,i,j} f_{ij}^t$. However, that would not provide the desired result, since it does not ensure that the arrival order is being respected. There are cases where the system may prevent users that arrive earlier from taking a bike if that can be later used in a better way (by serving more trips), as illustrated by the following example.

Example 1. Assume for simplicity that we have a BSS with three stations, one bike initially located at station 1, and three periods of demand with $d_{12}^1 = 1$, $d_{13}^2 = 1$, and $d_{32}^3 = 1$ (recall d_{ij}^t denotes how many people want to go from i to j at time t). In an actual system with such demand, the first customer would take the available bike, and customers 2 and 3 would not use the system as there are no bikes at their origin (see Figure 2). However, the objective of maximizing the number of people served leads to a solution where user 1 does not take the available bike, but it is instead used by customers 2 and 3 (see Figure 3). This way, two users travel successfully, but that does not agree with the first come first served rule.

In order to impose the rule of first come first served, we need to "prioritize" customers. This prioritization can be achieved by setting weights for each customer, which must be selected appropriately, to ensure that solutions like the one in the example will not be optimal for the model.

Proposition 1. *The objective function SUCCESSFUL_TRIPS$(f) = \sum_{t,i,j} 3^{T-t} f_{ij}^t$, where T is the number of time periods, enforces the first come first served (FCFS) principle for the users.*

Proof of Proposition 1. Let the objective function be of the form $\text{SUCCESSFUL\_TRIPS}(f) = \sum_{t,i,j} w^{T-t} f_{ij}^t$, where the weights w need to be selected. Consider there are enough bikes to cover

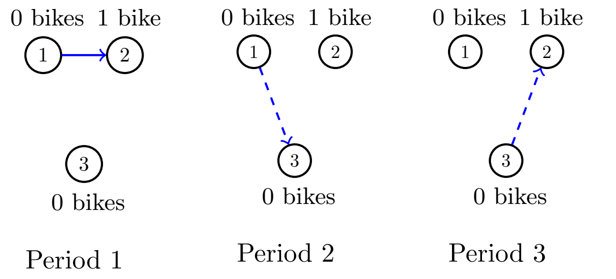

Figure 2 Case I: When the first come first served principle is satisfied, user 1 takes the available bike, while users 2 and 3 do not, leading to one successful trip. (Normal edges are used for successful trips and dashed edges for the unsuccessful ones.)

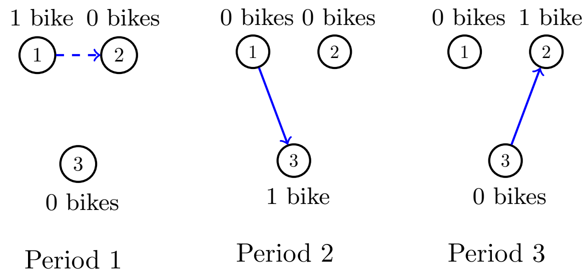

Figure 3 Case II: Results of the model without the first come first served principle. In order to maximize the total successful trips, the solution of the model withholds the bike from user 1, in order to satisfy users 2 and 3.

the demand at station i at time t. Then, a bike may not be used only if it offers greater payoff by remaining at the same place and being used in later time periods. Similarly, due to the interconnected nature of the resources (bikes and docks), the dock that remained empty at another station j due to the bike not getting there, can be then used to satisfy future incoming demand.

Let V_{use} denote the value that is added to the system if the bike is used at time period t and V_{stay} the value in case it remains at station i during period t. Recall that each bike can be used only once per time period. V_{stay} obtains its highest value if the specific bike at i as well as the dock at j are used by customers during all subsequent periods t', with $t < t' \leq T$. Since each bike and dock can be used once per time period, each successful use corresponds to a flow of one unit, so:

$$V_{stay} \leq \sum_{t'=t+1}^{T} w^{T-t'} + \sum_{t'=t+1}^{T} w^{T-t'} = 2\sum_{t'=t+1}^{T} w^{T-t'} = 2\frac{w^{T-t}-1}{w-1} \tag{24}$$

V_{use} is at least as much as the value gained by the user that rents the bike at time t. Of course, it can be higher if the bike is used by other people in the following time periods.

$$V_{use} \geq w^{T-t} \tag{25}$$

It is easy to see that for $w \geq 3$, we have:

$$V_{use} \geq w^{T-t} > 2\frac{w^{T-t}-1}{w-1} \geq V_{stay} \tag{26}$$

$V_{stay} < V_{use}$ implies that FCFS is satisfied, since an available bike not being taken by a user cannot be part of the optimal solution. We thus select $w = 3$ for solver numerical accuracy purposes. $\square$

4.3. Discussion

The optimal solution of (UT) is not necessarily integral, due to proportionality constraints (19). However, this is not an issue as our goal is to use this model as a measure of the performance of the system. A different approach is to perform a simulation of user trips. For that, an arrival order of the customers needs to be specified, in order to determine which ones are served in case of lack of resources.

In our method, we take into account only the number of users that arrive at each station during each time period - for example each half hour - and no specific order within the period is required. The proportionality constraint (19) then attempts to balance the various demand scenarios that arise from different arrival orders, by upper-bounding the number of trips to each destination. The advantage of this approach is that it does not depend on the exact arrival order, which is generally harder to estimate than just the number of users per period. We can think of the solution we get as the result of running multiple simulations on different demand scenarios within each period, and taking an average of their performance. Moreover, even if a similar proportionality approach is incorporated as part of a simulation procedure, the fact that this is a linear programming model expands its capabilities, as it can be also incorporated as part of more complicated models that combine user flows with other problem aspects.

One valid concern is the numerical issues that might arise when solvers are used to get a solution for the model. If there are many time periods per day, the objective function might obtain very large values as the objective weights grow exponentially with the number of periods. This issue is easy to tackle by solving the model in more than one stages. For example, we can solve it for the first half of the time periods, set the final system configuration as the initial state of the second run, and then solve the model separately for the second half of the periods. More stages can be used as well if required, in order to achieve a desired numerical accuracy. A solution for the model can be obtained very fast, so introducing more than one stages does not significantly change the total time that is required.

5. System Rebalancing

5.1. Preliminaries

Redistribution operations are planned based on the current configuration of the system, as well as the future needs. In this section, we provide a brief description of the main aspects that influence rebalancing decisions.

Unserved Customers. We first consider the unserved demand if no rebalancing is performed at the system. This will give an indication about the stations that have the greatest need for bikes or docks. Given the current configuration of the system, and the demand that is expected for the remainder of the day, we use the linear program (UT) of Section 4 to find which users successfully completed their trip. Then, for each station, we calculate the number of extra bikes or docks that are required to address the unserved outgoing demand $\text{UNSERVED\_DEMAND}(out, i, t)$ and incoming demand $\text{UNSERVED\_DEMAND}(in, i, t)$ for each station i and time period t.

Unused Resources. Since the rebalancing takes place simultaneously with the user movements, it should not interfere with their actions. In particular, bikes and docks that are used by customers cannot participate in the rebalancing during the same time period. So, for each station i and time t, we compute the number of unused bikes $\text{UNUSED\_BIKES}(i, t)$ and docks $\text{UNUSED\_DOCKS}(i, t)$ that are available for redistributing.

Rebalancing Needs. At this point, we have calculated the amount of customers that failed to take a trip, as well as the number of resources that are unused at their current location, and, thus, can be transferred elsewhere. We are now ready to make suggestions to the trucks regarding the number of bikes they need to pick up or drop off at each station they may visit. In particular, if we are currently planning the rebalancing to take place during period t, we can look at the unserved demand at time $t+1$, that is $\text{UNSERVED\_DEMAND}(out, i, t+1)$ for bikes and $\text{UNSERVED\_DEMAND}(in, i, t+1)$ for docks, which guide the rebalancing targets for the truck. At the same time, we look at the available bikes that can be picked up/dropped off at each station (UNUSED_BIKES and UNUSED_DOCKS for times t and $t+1$), so that the rebalancing operation is actually feasible. One drawback of this method is that it is very myopic: it only considers the following time period. So, instead, we actually consider the unserved demand and the unused resources further in the future (for example for the upcoming five or ten time periods) and determine the value of current resource needs BIKES_NEEDED(i,t) and DOCKS_NEEDED(i,t) for each station.

5.2. Scalability

In actual systems, the large number of stations often requires multiple vehicles working simultaneously to serve various parts of the network. This extends the redistribution capacity of the operator, but also creates coordination needs that increase the complexity of the problem, making it computationally intractable. Moreover, when redistribution is planned for the whole day in advance, a large number of time periods creates extra challenges for the operator. We will now discuss some methods to address these issues.

5.2.1. Geographic Segmentation. One common approach towards tackling the multi-vehicle scenario is to partition the area of service into regions with exactly one vehicle assigned to them. This is beneficial for drivers, as they can get familiar with the area more quickly, and thus they can become more efficient. It also ensures that the vehicles are adequately spread out to cover the rebalancing needs of most parts of the system.

The main challenge consists of identifying regions that achieve geographical proximity of stations and equally balanced work-load for assigned vehicles. The method that will be used is a variation of the k-median algorithm with a local search heuristic. There have been similar studies in the literature both from a theoretical viewpoint, as well as applied in the area of facility location problems. More on this topic can be found in Kanungo et al. (2014) and Arya et al. (2014).

Let k denote the number of available trucks, which also determines the number of station clusters that we wish to create. S is the set of all stations, S_l the stations of cluster l, C the set of centers of the clusters, and c_l the station-center of cluster l. At any point, each station is assigned to the cluster with the closest center. The general outline of the algorithm is the following.

1. Initialization: Select k stations at random out of S that will serve as the initial centers C for the k clusters.

2. While the termination criteria have not been met, iterate:

- Randomly pick one center $c_l \in C$ and remove it from set of centers: $C \leftarrow C \setminus \{c_l\}$.
- Calculate the clustering cost when c_l is replaced by any station $s \in S \setminus C$.
- Let $s^* \in S \setminus C$ be the station that minimizes the above cost.
- Update the set of centers as $C \cup \{s^*\}$.

Clustering Cost. For each cluster, we consider the work-load of its assigned truck. This is a combination of the expected number of redistributed resources, as well as the distances that it needs to traverse. We use w_i for the expected number of bikes that are required to be picked up or dropped off for station i, and $dist_{ij}$ for the distance between stations i and j. Then, we define the work load of each cluster l as:

$$\text{WORK\_LOAD}(l) = \sum_{i \in S_l} w_i + \alpha \sum_{i,j} \frac{dist_{ij}}{|S_l|} \tag{27}$$

Ideally, all clusters should have the same work load, since we want to ensure that the work is well distributed among the trucks. In order to achieve that, we can consider the maximum work load over all clusters, and aim to minimize it, reducing in that way the work of the busiest truck. The cost of clustering is in that case:

$$\text{CLUSTERING\_COST} = \max_l \text{WORK\_LOAD}(l) \tag{28}$$

Termination. At any point, the cost of the clustering solution is equal to the largest work-load among the vehicles. If this cost can be decreased by swapping a cluster center with some other station, then this replacement takes place during the iteration step of the algorithm. The cost is nonincreasing during the execution of the algorithm, and since it cannot improve indefinitely, a convergence to a local minimum is guaranteed given an adequate number of iterations. The algorithm terminates either when there has been no improvement for a set number of iterations or when a time limit, if any, has been reached. Since the execution time is very short, it can potentially be run multiple times will different random initializations and keep the best solution among them.

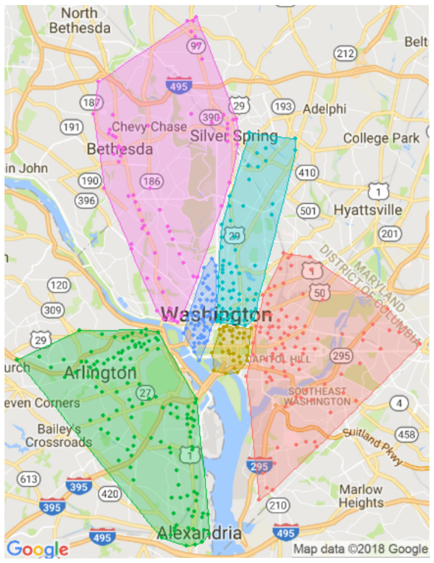

Figure 4 Geographic decomposition of stations into areas. A separate vehicle is assigned to each area.

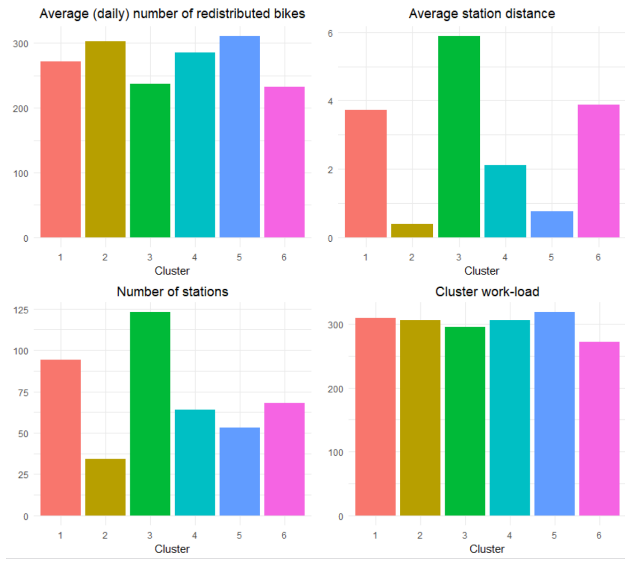

Figure 5 Decomposition statistics: resources to be redistributed, average distance, number of stations and total work-load per cluster with $\alpha = 10$.

5.2.2. Rolling Time Horizon. If redistribution decisions need to be taken for the entire day, a large number of time periods leads to an exponential number of different scenarios. Optimizing over such a large solution space is intractable, so we propose to focus on actions for the imminent future. This choice is also reinforced by the fact that in real systems future user demand is not fully known in advance. So, planning far in the future is not always worthwhile, since the configuration of the system at that time may be very different from the one currently expected.

As a result, we use a rolling time horizon. Each time we look at the current state of the system and plan the rebalancing for the upcoming time periods. When the realization of the actual demand occurs for these periods, we consider the new state of the system, and roll the rebalancing window to solve for the next periods. Despite focusing on a few periods for the rebalancing, we still consider the demand ahead. In that way, all the information we possess about future demand is taken into account when relocating the resources.

5.3. Formulation

In this section, we introduce the Dynamic Rebalancing (DR) model. The decision variables include variables for the routing of the truck: binary variables x_{ij} expressing whether the truck visits station j right after station i, truck arrival times t_i at each station i, station visit duration dur_i, as well as bikes b_i^T in truck upon arrival of the truck at station i. Moreover, the number of bikes to be picked up and dropped off from station i correspond to decision variables PU_i and DO_i respectively. We further specify the picked up bikes by categorizing them as PU_i^+ for bikes that add immediate value to the rebalancing as they vacate docks for customers to use, and PU_i^o for bikes that it is not necessary to be picked up from i, but they are in order to later be dropped off at other stations that need them. We consider the latter pick-up to be of neutral value to the rebalancing since the station where the pick-up takes place does not have an immediate gain from it. Similarly, we have DO_i^+ and DO_i^o for the drop-offs with positive and neutral value respectively.

We wish to maximize the positive value actions, while allowing the neutral type ones to take place where needed. Weights γ and δ in the objective are small constants in order to jointly minimize traveling time and redundant redistribution, M is a large constant, τ_0 is the initial time of the rebalancing period that the truck is available, ζ_0 is the initial bike load of the truck, τ_{action} is the time that is required for each bike pick-up and drop-off, $\tau_{station}$ the overhead time per station visit (for parking, etc.), and τ_{period} the length of the rebalancing period. We introduce a dummy node for the truck, denoted with index 0, S is the set of indices that corresponds to the station nodes and $S_0 = S \cup \{0\}$ the set that also includes the dummy node.

$$\text{(DR)}: \quad \max \quad \sum_{i \in S} PU_i^+ + \sum_{i \in S} DO_i^+ - \gamma \sum_{i \in S_0, j \in S} dist_{ij} x_{ij} - \delta \sum_{i \in S} (PU_i + DO_i) \tag{29}$$

$$\text{s.t.} \quad \sum_{j \in S_0} x_{ij} = 1 \qquad \forall i \in S_0 \tag{30}$$

$$\sum_{j \in S_0} x_{ji} = 1 \qquad \forall i \in S_0 \tag{31}$$

$$t_j \geq t_i + dur_i + dist_{ij} \cdot x_{ij} - M \cdot (1 - x_{ij}) \qquad \forall i, j \in S, i \neq j \tag{32}$$

$$t_i \geq dist_{0i} \cdot x_{0i} + \tau_0 \qquad \forall i \in S \quad (33)$$

$$t_i \leq \tau_{period} \qquad \forall i \in S \quad (34)$$

$$dur_i \geq \tau_{station} \cdot (1 - x_{ii}) + \tau_{action} \cdot PU_i + \tau_{action} \cdot DO_i \qquad \forall i \in S \quad (35)$$

$$b_j^T \geq b_i^T + PU_i - DO_i - M \cdot (1 - x_{ij}) \qquad \forall i, j \in S, i \neq j \quad (36)$$

$$b_j^T \leq b_i^T + PU_i - DO_i + M \cdot (1 - x_{ij}) \qquad \forall i, j \in S, i \neq j \quad (37)$$

$$b_i^T \geq \zeta_0 \cdot x_{0i} \qquad \forall i \in S \quad (38)$$

$$b_i^T \leq C^T - (C^T - \zeta_0) \cdot x_{0i} \qquad \forall i \in S \quad (39)$$

$$b_i^T \leq C^T \qquad \forall i \in S \quad (40)$$

$$b_i^T + PU_i - DO_i \leq C^T \qquad \forall i \in S \quad (41)$$

$$b_i^T + PU_i - DO_i \geq 0 \qquad \forall i \in S \quad (42)$$

$$PU_i \leq \text{UNUSED\_BIKES}(i) \qquad \forall i \in S \quad (43)$$

$$PU_i^+ \leq \text{DOCKS\_NEEDED}(i) \qquad \forall i \in S \quad (44)$$

$$PU_i = PU_i^+ + PU_i^o \qquad \forall i \in S \quad (45)$$

$$PU_i \leq M \cdot (1 - x_{ii}) \qquad \forall i \in S \quad (46)$$

$$DO_i \leq \text{UNUSED\_DOCKS}(i) \qquad \forall i \in S \quad (47)$$

$$DO_i^+ \leq \text{BIKES\_NEEDED}(i) \qquad \forall i \in S \quad (48)$$

$$DO_i = DO_i^+ + DO_i^o \qquad \forall i \in S \quad (49)$$

$$DO_i \leq M \cdot (1 - x_{ii}) \qquad \forall i \in S \quad (50)$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in S_0 \quad (51)$$

$$t_i, dur_i, b_i^T, PU_i, PU_i^+, PU_i^o, DO_i, DO_i^+, DO_i^o \in \mathbb{R}_+ \qquad \forall i \in S \quad (52)$$

Constraints (30) and (31) specify the truck routing. We impose that each node is visited exactly once, including the dummy node that corresponds to the truck. Nodes that are not visited by the truck correspond to self-loops in the solution. This idea is inspired by Yang et al. (2004).

Constraints (33) [for the first visit] and (32) [for subsequent visits] describe the arrival times of the truck at the stations which must be within the allowed horizon (34), and (35) allows for the necessary amount of time per station visit. (36) and (37) update the bikes in the truck after each visit, and (38) and (39) enforce that the truck contains its initial load of bikes at the beginning of its first visit. Constraints (40) ensure that the bikes in the truck do not exceed the truck capacity, while (41) and (42) are used to enforce load limits after the last truck visit. Due to (43) and (47), bikes are rebalanced only if they (or the docks) are unused at that moment to not interfere with customer flows. Each pick-up and drop-off can either be of positive or neutral value (45), (49), and they are nonzero only if the truck actually visits the station (46), (50). Finally, the positive value resources are bounded by the needed bikes and docks per station (44), (48). We can also add the following two constraints to obtain a tighter formulation:

$$\sum_{i \in S_0, j \in S} dist_{ij} x_{ij} + \sum_{i \in S} dur_i \leq \tau_{period} \tag{53}$$

$$\sum_{i \in S} \tau_{action} \cdot PU_i + \sum_{i \in S} \tau_{action} \cdot DO_i \leq \tau_{period} \tag{54}$$

Further details for model (DR) are provided in Appendix B.3.

5.3.1. Planning over Multiple Rebalancing Periods. Note that in (DR) we are planning over a single rebalancing period of length τ_{period}. In Appendix C, we show how Dynamic Rebalancing model (DR) can be generalized to Multiperiod Dynamic Rebalancing (MDR) to generate rebalancing plans considering jointly more than one rebalancing periods. The main ideas consist of introducing the constraints of (DR) for each rebalancing period, with additional constraints to ensure feasible transition of the truck from one period to the next, as well as overall feasible bike pick-ups and drop-offs if a station is visited at more than one period. The results of this approach are compared with the single-period rebalancing in the Computational Experiments Section 6.

5.4. Optimization with Station Groups

For each cluster, neighboring stations can sometimes be handled together by consecutive truck visits. The goal of this section is to group stations that are very close to each other, in order to further reduce the size of the network.

5.4.1. Leader Stations. We introduce a binary variable x_i for each station i and aim to allow only stations with $x_i = 1$ to potentially be visited by a truck. We name these stations *leader stations*. The goal is to minimize the number of leader stations, minimizing in that way the visit candidates of the truck. At the same time, we need to ensure that each station has at least one leader nearby, in other words, for each station the truck can visit and rebalance bikes in its neighborhood. Let S_R denote the stations that currently need rebalancing, and N_{il} the l-neighborhood of i, where distance l is a parameter of the model. The formulation follows.

$$\min \quad \sum_i x_i \tag{55}$$

$$\text{s.t.} \quad \sum_{j \in N_{il}} x_j \geq 1 \qquad \forall i \tag{56}$$

$$x_i \geq x_j \qquad \forall i \in S_R, j \notin S_R, j \in N_{il} \tag{57}$$

$$x_i \in \{0,1\} \qquad \forall i \tag{58}$$

This is a variation of the set covering problem: According to constraint (56) each station must be covered by/be in the neighborhood of at least one leader (the station itself could be that leader). Constraint (57) gives priority to stations that need rebalancing: If it is possible to select a leader between a station that needs rebalancing and one that does not, then the one that faces the problem will be prioritized. As a result, a station that does not currently require rebalancing can become a leader station only if none of its neighbors are in need of rebalancing.

After the leaders have been selected, each station is assigned to its closest leader, creating in that way mini-groups of stations, each of which is represented by its leader. The capacity and the rebalancing needs of the leader are then given by the total capacity and rebalancing needs of the stations in its mini-group. The advantage of this approach is that it allows to consider only a small subset of stations, facilitating in that way the optimization. At the same time, it does not ignore the remaining stations, since the problems of all stations are taken into account.

5.4.2. Rebalancing Only Leader Stations. Having identified a set of leader stations, a natural next step is to solve (DR) using only the leader stations. In this case, each leader station will have the total capacity, unused resources and rebalancing needs of its mini-group. An important drawback of this suggestion is that it implies that a visit to the leader station would momentarily solve the problem for many of the stations of its mini-group as well. This ignores the traveling time that is required to move from one station to the other, which might be close, but their distance is still not negligible. In the next section, we develop a method that can take information of each group into account in the model.

5.5. Optimization with Partial Group Information

Let us now propose an approach which will be demonstrated by considering one of the mini-groups as an example. First, we simplify our problem by assuming temporarily that all pairs of stations of the group require the same traveling time τ_{travel}. We can make this assumption, because all stations are close to each other, so their traveling times do not differ significantly. An important observation is that the work performed within a mini-group depends clearly on the time the truck spends in that mini-group. A short period of time might only be enough to visit the leader station, while longer visit durations allow serving more of the stations of the group. We will express now the performance of the truck as a function of the time spent inside the mini-group, with the help of a greedy routing algorithm.

5.5.1. Greedy Routing Algorithm. Consider first only the stations that require bikes to be picked up, i.e. they need extra empty docks to satisfy their incoming demand. The truck is currently at the leader station of the group. Aiming to maximize the number of picked-up bikes, the truck starts by picking up bikes, if any, from the leader station. Then, since all traveling times between stations are assumed to be the same and equal to τ_{travel}, the truck will greedily maximize its performance if it visits the station with the largest number of bikes to be picked up: by “paying” the price of τ_{travel} to reach a station, it gets the greatest possible “reward”, i.e. largest amount of bikes.

EXAMPLE 2. Consider the mini-group in Figure 6 which consists of five stations and let station 3 be the selected leader. Assume the truck is currently at station 3, there are no constraints on its capacity, the traveling time between any pair of stations is $\tau_{travel} = 3$ minutes, the overhead time for each station visit (parking, etc.) is $\tau_{station} = 2$ minutes, and picking up each bike requires $\tau_{action} = 1$ minute. Following the greedy routing algorithm described before, we get a solution where 5 bikes are picked-up from station 3, then 4 from station 5, and finally 3 from station 1.

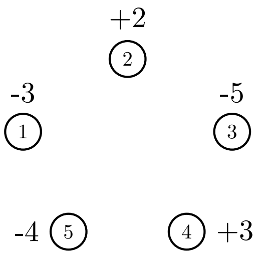

Figure 6 A mini-group of stations. The number by each station indicates the number of bikes that should be picked-up (negative sign) or dropped-off (positive sign).

PROPOSITION 2. *For a given rebalancing period length T, the greedy routing algorithm within a group of stations is optimal if we ignore the vehicle's load and capacity, the integrality of resources and assume equal traveling times between any two stations of the group.*

Proof of Proposition 2. In this case, the routing among a group of stations can be reduced to the fractional knapsack problem. The length of the rebalancing period T can be thought of as the capacity of the knapsack. Each station i is an item with "value" equal to the resources r_i (bikes or docks) that can be rebalanced and "weight" equal to the time required to travel to the station and perform the rebalancing $\tau_{travel} + \tau_{station} + r_i \cdot \tau_{action}$. The goal is to maximize the redistributed resources in the period length T. Hence, given that we allow fractional rebalancing of resources, the greedy algorithm selects the stations in decreasing order of value over weight, so the solution is optimal. $\square$

In the above, we have assumed that the value (amount of redistributed resources) is being obtained equally distributed over the total time for traveling and picking up or dropping off the bikes. In reality, no value is acquired during the traveling time among stations, but this is a simplifying assumption that allows us to model redistributed resources as piecewise linear concave functions of rebalancing time, as will be demonstrated in the following section.

5.5.2. Group Modeling Using Piecewise Linear Functions.

PROPOSITION 3. *The amount of redistributed resources in the greedy algorithm can be modeled as a piecewise linear concave function of the time the vehicle spends in the group of stations.*

Proof of Proposition 3. The greedy algorithm selects the stations in order of decreasing ratio $\frac{r_i}{\tau_i}$ of redistributed resources r_i over required time $\tau_i = \tau_{travel} + \tau_{station} + r_i \cdot \tau_{action}$. Since the ratio $\eta_i = \frac{r_i}{\tau_i}$ is decreasing in the selections of the algorithm, so are the slopes of r_i as a function of τ_i if we approximate them using functions $r_i(\tau_i) = \eta_i \tau_i$. Hence, the amount of redistributed resources is a concave piecewise linear function of the time the vehicle spends in the group. □

EXAMPLE 3. Using again the example of Figure 6, we have three stations that need bikes to picked-up, so the piecewise linear function that corresponds to the pick-ups will consist of three pieces (for stations 3, 5 and 1 respectively). The truck is already at station 3 and we need to pick up $r_3 = 5$ bikes, so the visit will last $\tau_3 = r_3 \cdot \tau_{action} = 5$ minutes. For station 5, $r_5 = 4$ bikes need to be picked up and the time required is $\tau_5 = \tau_{travel} + \tau_{station} + r_5 \cdot \tau_{action} = 9$. Finally, for station 1 and the pick-up of $r_1 = 3$ bikes: $\tau_1 = \tau_{travel} + \tau_{station} + r_1 \cdot \tau_{action} = 8$. As a result, the truck will visit station 3 for $t \in [0, 5]$, station 5 for $t \in [5, 14]$ and station 1 for $t \in [14, 22]$. The corresponding line slopes are given by $\frac{r_i}{\tau_i}$, which in this case are $\frac{5}{5} = 1$ for station 3, $\frac{4}{9} = 0.444$ for station 5, and $\frac{3}{8} = 0.375$ for station 1, which are decreasing and the function is concave, as it is also illustrated in Figure 7.

In the previous example, the focus was placed only on bike pick-ups. However, it should be obvious that the same results hold for bike drop-offs.

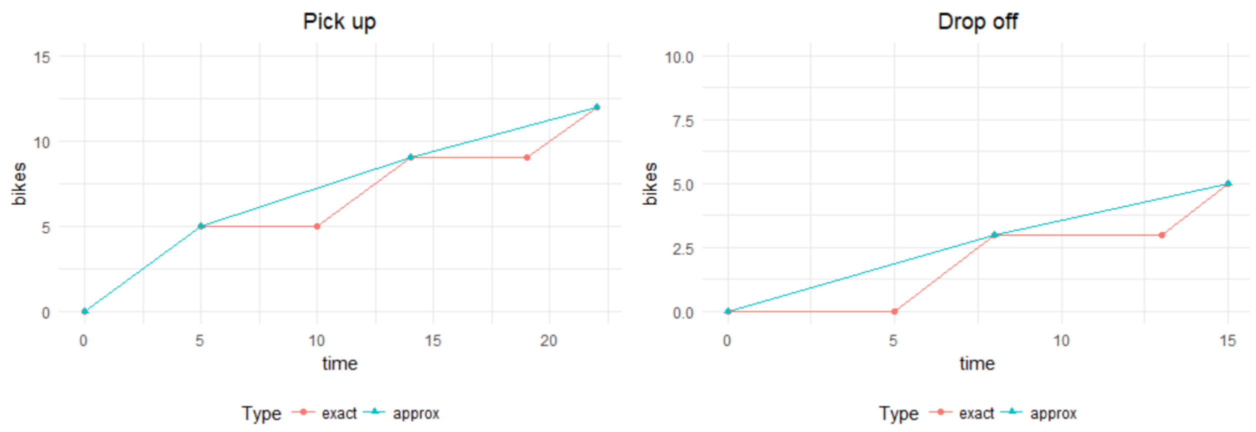

Figure 7 Redistributed resources as a function of time. With blue color the piecewise linear concave functions. With red the results taking into account that during traveling the number of resources remains constant (function parallel to x axis during traveling).

Corollary 1. *The amount of redistributed resources over time for each group can be formulated using linear constraints. If a_p and b_p denote the slope and intercept vector (with elements $a_{p,l}$ and $b_{p,l}$ for each line segment l) that correspond to the piecewise linear concave function for the pick-ups, and respectively a_d and b_d for the drop-offs, and, finally, $durPU_i$ is the duration of the visit in group i where the truck is picking up bikes and $durDO_i$ is the duration of the visit where the truck is dropping off bikes, then the number of bikes PU_i that can be picked up and DO_i that can be dropped off are the largest numbers that satisfy:*

$$PU_i \leq \min_l(\alpha^i_{p,l} \cdot durPU_i + \beta^i_{p,l}) \quad \forall i \tag{59}$$

$$DO_i \leq \min_l(\alpha^i_{d,l} \cdot durDO_i + \beta^i_{d,l}) \quad \forall i \tag{60}$$

5.5.3. Global Rebalancing. Having expressed the amount of redistributed resources for each group with respect to the duration of its visit, we can now introduce the rebalancing optimization using partial group information. The basis of the optimization is still model (DR) as described in Section 5.3, which is now applied only to the leader stations. The main difference is that we introduce for each group constraints (59) and (60) to model (DR), so it now performs dynamic rebalancing with partial information and is therefore denoted by (DRPI). Appendix B.1 presents in more details the changes in (DR) that lead to (DRPI).

The solution of (DRPI) produces a routing for each truck which specifies the order with which groups need to be visited. Moreover, it provides the number of bikes that need to be picked up and dropped off for each group, as well as an estimation of the time it will require. Note that picking up and dropping off the planned number of bikes is important, as these bikes might be accounted for in the resources needed for subsequent truck visits. Regarding the time that the actions per group actually require, there is slightly more flexibility: we have the possibility of delaying a group visit by a few minutes or performing it ahead of time. The next section concludes this method by describing how the exact routing within each group is obtained.

5.5.4. Final Retrieval of Within Group Routing. For the rebalancing within each group, the total number of bikes to be picked up and dropped off is provided by the solution of (DRPI) and now the exact routing needs to be retrieved. At this point, we remove the assumption that all pairs of stations in a group have the same distance, and consider their real distances. For the retrieval of the routing, one might use an adaptation of the greedy routing algorithm since the distances within each group are small and the routing is not very demanding in terms of time. The greedy routing algorithm selects the next station to be visited based on the largest ratio of bikes to be moved over the required time. This selection criterion can still be followed, but the load and the capacity of the truck need to be taken into account: if a truck is full, it cannot pick up more bikes, and similarly for dropping off when it is empty. Moreover, the total number of bikes to be picked up and dropped off per group is determined by the solution of (DRPI). Therefore, we adapt the selection step by considering the ratio of bikes moved over time but only for the number of bikes that is allowed based on the current load, the total capacity limits of the truck, and the (DRPI) solution. In case the final schedule includes visits that exceed the rebalancing period duration, we remove all visits with arrival time after the period end.

6. Computational Experiments

The results of this section are based on real-world data for Capital Bikeshare, Washington DC, for the time period May to September 2017. For each past trip, we have details about its origin, its destination, as well as its starting and ending times. We also have information about station outages, i.e. the starting and ending time, as well as the station where it was observed, and its status - full or empty. Finally, the design details of the system are also known: the location and the capacity for each station, and the total number of bikes in the system. The data for past trips and system information (Capital Bikeshare (2017)), as well as the data for outages history (Capital Bikeshare Tracker (2017)) are publicly available.

6.1. Demand Estimation

6.1.1. Lost Demand. In order to evaluate the methods suggested in Section 3.2, we identify all periods for which each station is not empty, and for each such period we assume that the demand is unknown and estimate it as the convex combination of the demand obtained by the average station behavior and the daily trends.

$$ld^t_{out,i} = \lambda \cdot \text{ AVG\_}ld^t_{out,i} + (1-\lambda) \cdot \text{ TREND\_}ld^t_{out,i} \tag{61}$$

Since the station is not actually empty at that time, we know the exact demand that was observed. So we can compare our estimation results with the observed demand of the day using the mean square error (MSE) as the evaluation metric.

Computation of parameter λ. The optimal value of the parameter λ that minimizes the MSE can be computed in closed form. Note that $d^t_{out,i}$ appears instead of $ld^t_{out,i}$ below, since in this setting only stations with known demand participate.

$$\lambda = \frac{\sum_{i,t}(d^t_{out,i} - \text{ TREND\_}ld^t_{out,i}) \cdot (\text{ AVG\_}ld^t_{out,i} - \text{ TREND\_}ld^t_{out,i})}{\sum_{i,t}(\text{ AVG\_}ld^t_{out,i} - \text{ TREND\_}ld^t_{out,i})^2} \tag{62}$$

The optimal value of λ is calculated using half of the dataset as the training set, and then the MSE with the selected λ is evaluated on the remaining dataset. We ran 100 experiments with different random splits for the training and the testing set, and the values are consistent. The average optimal λ is 75.56% with a standard deviation of 1.45%.

The average MSE results on the testing sets across all experiments are presented in Table 2. In this table, weekends and vacations are excluded and only the 80% busiest days have been considered

Table 2 Out-of-sample demand estimation MSE during rush hours and for the whole day.

Time	Minimum trips	Average behavior	Average & Trends
Morning rush hour	20	2.081	**2.049**
(7:30am-9:30am)	50	2.994	**2.939**
	100	5.278	**5.155**
Evening rush hour	20	4.273	**4.101**
(5pm-7pm)	50	7.190	**6.875**
	100	13.797	**13.118**
All day	20	2.272	**2.215**
(6am-11pm)	50	3.695	**3.581**
	100	7.566	**7.219**

The minimum trips column shows the least number of daily trips for a station to be considered, which allows to evaluate separately the results for the busiest stations of the system. The combination of the average demand and the daily trends outperforms considering just the average which is a common approach.

to avoid very slow days and outliers where external factors may lower bike usage, so as to have more accurate comparisons with O'Mahony (2015), who remove days with rain and snow from their estimation based on average station behavior. We show the results for the whole day as well as for the morning and the evening rush hour. Similarly, we present separately results for busy stations (the ones that have a minimum of 100 or 50 trips per day). In all instances, using a combination of the average station behavior and the daily trends outperforms taking simply the average demand.

Having evaluated the methods on intervals where the demand is known, we will now illustrate some of the results on lost demand estimation. The data has already been preprocessed to identify intervals of lost demand, and the size of this lost demand is being estimated according to Section 3.2. In the following Figures 8 and 9, we present the results on the lost demand for each of the five months, May to September 2017, and compare it with the corresponding observed demand for the same period.

Figure 10 shows how the lost demand is distributed in the duration of the day. We can observe that there are two peaks both in the lost outgoing and the lost incoming demand. Assuming the morning rush hour is from 7:30am to 9:30am and the evening from 5pm to 7pm, we can see that these peaks emerge towards the end and after each rush hour of the day. This is not an unexpected behavior, since the large masses of customers that move during the rush hours often leave big parts of the network unbalanced, and many outages may follow raising the amount of lost demand.

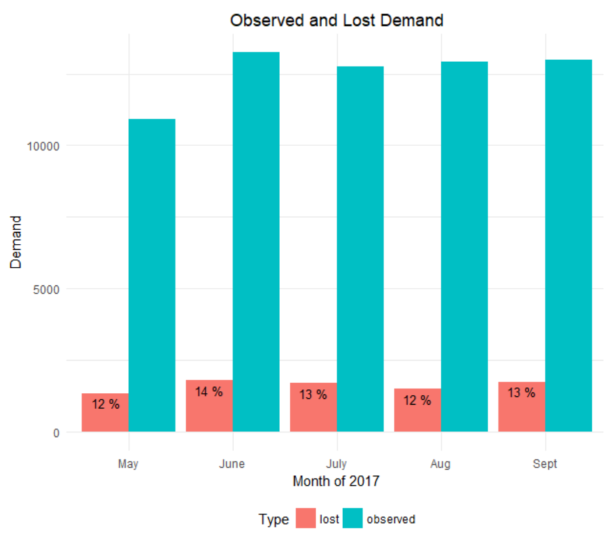

Figure 8 Daily average observed (blue) and estimated lost (red) outgoing demand from May to September 2017. The numbers in the bars show the percentage of the lost demand with respect to the corresponding observed one.

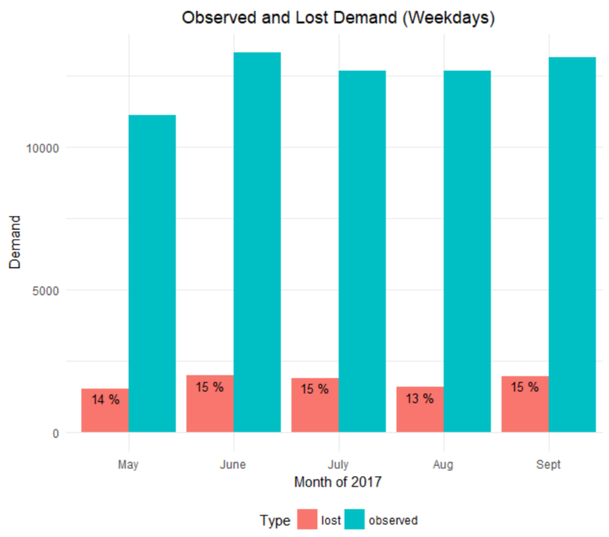

Figure 9 Daily average observed (blue) and estimated lost (red) outgoing demand from May to September 2017 considering only the weekdays. The numbers in the bars again show the percentage of the lost demand with respect to the observed one.

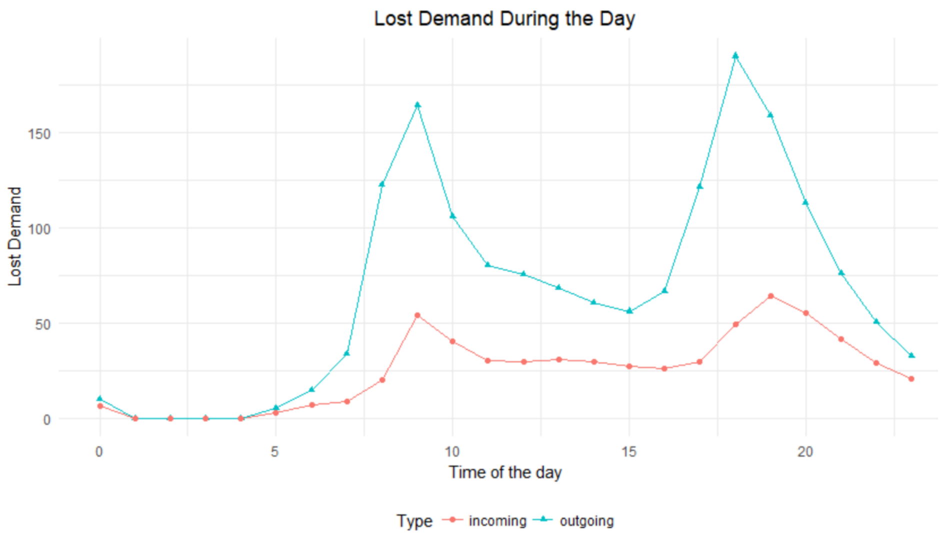

Figure 10 Estimated lost demand in the duration of a day. This is the average daily lost demand over the period May-September 2017. The two peaks visible in both incoming and outgoing lost demand correspond to the morning and evening rush hours.

6.1.2. Shifted Demand. Having calculated the lost demand, both the observed and the lost demand are aggregated in periods of thirty minutes, and we use the model of Section 3.3 to calculate the probability p with which users choose to walk if there is a station with available bikes nearby.

The maximum distance l_{max} that people are willing to walk is a parameter of the model. We will now experiment with various values of l_{max} and show how that influences the value of p. We assume that if people have more than one station they can walk to in the l_{max}-radius, then the probability of going to each station is inversely proportional to its distance, as explained in Section 3.3, making it more likely to visit closer stations than further ones. We also experiment with various values for the shifting distance multipliers α. The results are illustrated in Figure 11.

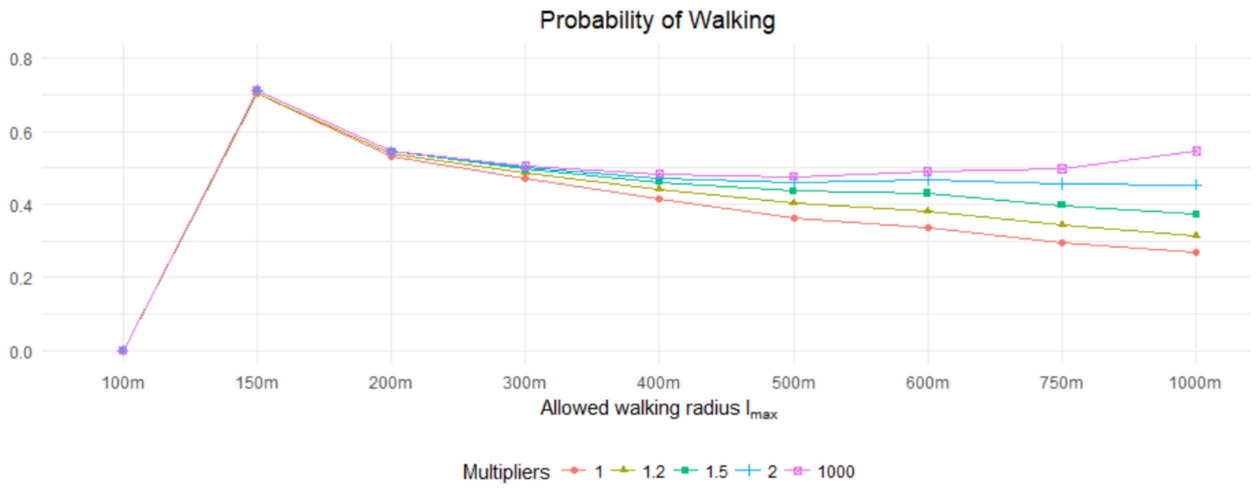

Figure 11 Probability of walking p as the allowed walking radius l_{max} and the shifting distance multiplier α vary.

In order to evaluate the results of the model, we want to check its consistency across the network. So, we randomly split the set of 485 stations into two separate groups A and B of approximately the same size. The model first runs on the stations of group A and provides a walking probability value p_A, and then runs on group B and calculates the value of p_B. Every time we run the model on a set of stations, the shifted demand that originates from stations of the other group is still taken into account. The reason for this is that at the end it was served by the stations under consideration, so it consists part of their observed demand. The difference of the values across the two groups is then given by $|p_A - p_B|$.

Table 3 Consistency results for various parameter values.

$l_{max} \setminus \alpha$	1	1.2	1.5
$400m$	0.0228	0.0272	0.0319
$500m$	0.0181	0.0235	0.0249
$600m$	0.0165	0.0204	0.0236

These values for the mean absolute difference dp were obtained for 100 iterations and various values of maximum radius l_{max} and multiplier α.

Since a unique split is not sufficient to evaluate the performance of the model, we run 100 iterations, each with a different random split of the stations into the two groups. Let A^{it} and B^{it} denote the two groups that were obtained during iteration it. We then use p_A^{it} (resp. p_B^{it}) for the value of the probability of walking that was obtained by running the model on group A^{it} (resp. B^{it}). We can now compute dp, which is the average absolute difference of the values p_A^{it} and p_B^{it} across the iterations. For a number of I iterations, we have:

$$dp = \frac{1}{I}\sum_{it=1}^{I} |p_A^{it} - p_B^{it}| \tag{63}$$

Table 3 summarizes the results for 400m, 500m and 600m of maximum walking distance l_{max}, and values 1, 1.2 and 1.5 for shifting distance multiplier α. All combinations for l_{max} and α that we tested lead to a small average difference between the p values of the station groups. Since the number of iterations is large, this suggests that in the average case the p values of the groups do not differ significantly. Similar values of p indicate that this method is consistent throughout the network.

Having evaluated the consistency of the model, at this point we need to make a selection regarding the value of l_{max} and α that will be used to calculate the shifted demand. We cannot base our selection solely on the objective value of (SD), since large values for l_{max} are not representative of the human behavior in the real world: someone would rarely walk one kilometer in order to find an available bike. For this reason, we decided to use $l_{max} = 500m$, which is a reasonable choice for a walking distance. Similarly, we selected $\alpha = 1.5$, which assumes people might go to other stations besides their closest one as long as they are not very far from it. Indeed, often the closest station might be on the opposite direction of the desired destination, so someone might decide to walk slightly further but simultaneously cover part of the distance towards their final destination. These values for l_{max} and α correspond to a walking probability $p = 43.59\%$.

6.2. System Rebalancing

In this section, we present results of the proposed system rebalancing methods using both synthetic and real-world datasets. Varying the structure on synthetic data can offer good intuition on factors that influence algorithms' performance. On the other hand, using real-world datasets allows us to evaluate the methods in an actual bike sharing system. For the implementations, we use Julia, a high-performance language for numerical computing (Bezanson et al. (2017)), and JuMP, a modeling language for mathematical optimization (Dunning et al. (2016)).

6.2.1. Experiments on Synthetic Data. All experiments are run on grid-structured instances of two types (see Figures 12 and 13). In both types, each group of stations occupies a single cell of the grid, but Type I instances consist of a sparser grid with larger distance among groups, while Type II instances are denser. We vary the sizes of the grid, the number of stations per group, and the rebalancing requirement levels. In these experiments, we consider a single rebalancing period, since it is sufficient to demonstrate the differences among the various methods. We report the results and compare three methods: using the MIP formulation (DR) directly on the set of stations (denoted as "S"), grouping using only the leaders' information in (DR) (denoted as "G/L"), and grouping with partial information using (DRPI) (denoted as "G/PI").

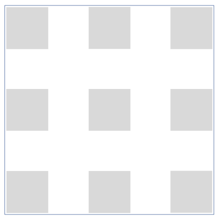

Figure 12 Type I: Sparse grid 3×3. Each grey region corresponds to a group of stations.

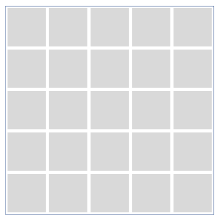

Figure 13 Type II: Dense grid 5×5. Each grey region corresponds to a group of stations.

Table 4 Served users in the scenario of homogeneous rebalancing needs.

Grid size	Stations per group	Grid type	S [5 min]	(Opt. Gap)	S [20 min]	(Opt.Gap)	G/L	G/PI
4×4	5	I (sparse)	80	(43.4%)	85	(32.6%)	86.31	81.80
4×4	5	II (dense)	90	(28.7%)	95	(21.3%)	91.86	92.37
4×4	10	I (sparse)	86.09	(46.7%)	95	(31.5%)	100	98.27
4×4	10	II (dense)	82.86	(56%)	100	(25.1%)	101.16	101.45
5×5	5	I (sparse)	0	–	80	(52%)	87.08	89.11
5×5	5	II (dense)	57.03	(117%)	92.37	(31.6%)	93.17	92.61
5×5	10	I (sparse)	89.45	(48.1%)	91.22	(41.8%)	95	100
5×5	10	II (dense)	85	(55.7%)	95	(36%)	97.69	95.97

Each station requires the same number of bikes to be picked up or dropped off. Three methods are compared: using the (DR) formulation directly on the set of stations (denoted as "S"), grouping using only the leaders' information in (DR) (denoted as "G/L"), and grouping with partial information with (DRPI) (denoted as "G/PI"). G/PI and G/L get within seconds solutions that are better or comparable to solutions that S obtains after 20 minutes.

We consider instances with grid sizes 4×4 and 5×5, and each station group has 5 or 10 stations. Regarding the rebalancing needs, we consider one scenario where all stations need the same amount of bikes to be picked up or dropped off (homogeneous needs), and a scenario where each group has two dominant stations with much larger needs than the rest (heterogeneous needs). In both cases, each station is being labelled independently as pick-up station or drop-off station with probability one half.

Parameter Selection. In the experiments that follow, all stations have capacity 20 and are initially half filled, the rebalancing period length is 100, and the truck has a capacity of 25 bikes. Each side of the grid cells is 1000 meters, and in Type I (sparse) instances the vertical and horizontal distance between cells is 1000 meters, while in Type II (dense) 200 meters. We assume the overhead time for each station visit (for parking, etc.) is $\tau_{station} = 1$ minute, the time for each bike pick-up or drop-off is $\tau_{action} = \tau_{pickup} = \tau_{dropoff} = 0.5$ minute, and the average traveling time between stations of the same group is $\tau_{travel} = 1$ minute since they are close by. In the scenario with homogeneous rebalancing needs, each station needs five bikes to be picked up or dropped off, and in the one with heterogeneous needs, two stations of each group need ten bikes to be picked up or dropped off and the rest just one. In the case where the MIP is applied directly on the stations without grouping (S), we report the results (including the optimality gap) after 5 minutes and 20 minutes of running time, in order to compare the tradeoff of the solution quality and the required time with the results of our methods.

Table 5 Served users in the scenario of heterogeneous rebalancing needs.

Grid size	Stations per group	Grid type	S [5 min]	(Opt. Gap)	S [20 min]	(Opt. Gap)	G/L	G/PI
4×4	5	I (sparse)	86	(41.2%)	87	(34.4%)	78	94.45
4×4	5	II (dense)	106.63	(18.3%)	108	(15.2%)	86.81	105.22
4×4	10	I (sparse)	79	(71.5%)	84	(57.8%)	65	80.67
4×4	10	II (dense)	0	–	91	(45.7%)	65	110.27
5×5	5	I (sparse)	0	–	86	(49.4%)	85.18	101.26
5×5	5	II (dense)	0	–	107.17	(26.6%)	80	112.68
5×5	10	I (sparse)	0	–	0	–	71.71	100.4
5×5	10	II (dense)	10	(1301%)	10	(1295%)	72.72	108.89

Each group has two stations with much larger rebalancing needs than the rest. Three methods are compared: using the (DR) formulation directly on the set of stations (denoted as "S"), grouping using only the leaders' information in (DR) (denoted as "G/L"), and grouping with partial information with (DRPI) (denoted as "G/PI"). Incorporating partial group information leads to G/PI significantly outperforming G/L and getting better solutions than S within seconds of running time.

Results. Tables 4 and 5 present the number of bikes that was redistributed successfully (or equivalently the number of additional served users) in each method within the rebalancing period length. We observe that grouping can in general achieve comparable or better solutions in faster running times; in particular, G/PI provides within seconds solutions that are better or comparable to solutions obtained after 20 minutes of running the MIP in S. The difference in performance becomes more significant as instances become larger, and manifests in a larger degree in Type I (sparse) instances. In these instances, the groups of stations are not adjacent, but there is some distance among them, and the traveling times among groups are larger. This seems to complicate the solution for the MIP in S, which at times needed more than five or twenty minutes to find a feasible solution, while G/PI had already found a good quality solution within seconds.

Comparison between grouping where only leader information is taken into account (G/L) and grouping with partial group information (G/PI) shows that solutions are comparable when all stations have equal rebalancing needs. On the other hand, the difference is significant when there are stations with much larger needs (dominant stations) than the rest. G/L cannot differentiate between this and the previous case; since within group routing and time per visit are not taken into account, only the aggregate number of bikes needed influences the MIP results. However, in G/PI, the piecewise linear functions that model the redistribution operations within each group guide the MIP solution to visit mostly the dominant stations. The reason is that the value of these visits (bikes redistributed over required time) is larger than visiting the remaining stations, where the travel time is not worth the amount of bikes to be picked up or dropped off in most groups. Overall, the extra information provided in G/PI produces solutions that outperform G/L, and also compare favorably with MIP results without grouping (S) while requiring significantly less running time.

6.2.2. Experiments on Real-World Data. In this section, we will use the past trips dataset provided by Capital Bikeshare, Washington DC, to evaluate our proposed methods. In the sections that follow, we compare multiple aspects that influence rebalancing, and more detailed results can be found in Appendix D. Unless noted otherwise, we consider the rebalancing that takes place each day until noon, which includes the preparation for the morning rush hour, the rush hour itself, as well as the reorganization that is required when it is over as a result of the increased user traffic.

Parameter Selection. In the following experiments, each rebalancing period has length of 30 minutes, the overhead time for each station visit is $\tau_{station} = 1$ minute and the time for each bike pick-up or drop-off is $\tau_{action} = \tau_{pickup} = \tau_{dropoff} = 0.5$ minute. In the leader selection, we set the maximum distance between stations of the same neighborhood to be 1000 meters, and the travel time between stations of the same group to be $\tau_{travel} = 1.14$ minutes, as this is the average value we obtained experimentally. In order to determine the rebalancing needs and the unused resources of each station, which then guide the actions of the trucks, we consider the user demand from the current period and up to five periods in the future, as described in Section 5.1. For these experiments there are 8 trucks, each of capacity 25 bikes, the stations start the day half full, and the MIP solver time limit for (DR) and (DRPI) is 1 minute per cluster and per time period.

Method Comparisons. Similarly to the synthetic instances, let us begin with comparing the three different methods: using (DR) on the set of stations without grouping (denoted as "S"), using (DR) to the set of leaders without incorporating partial information (denoted as "G/L"), and using (DRPI) which considers the set of leaders with partial information (denoted as "G/PI"). Results for the rebalancing gain for the three methods are presented in Figure 14, and more detailed results are available in Appendix D.

The rebalancing gain refers to the additional number of users that the system can serve compared to the scenario where no rebalancing takes place. We observe that in all instances G/L and G/PI outperform S, and, with a few exceptions, G/PI outperforms G/L. In particular, G/L achieves an 6.78% increase of the rebalancing gain compared to S (averaged over all instances), and G/PI an 11.36% increase in average. Moreover, despite obtaining better results, G/L and G/PI require 5 to 8 times less total running time compared to S.

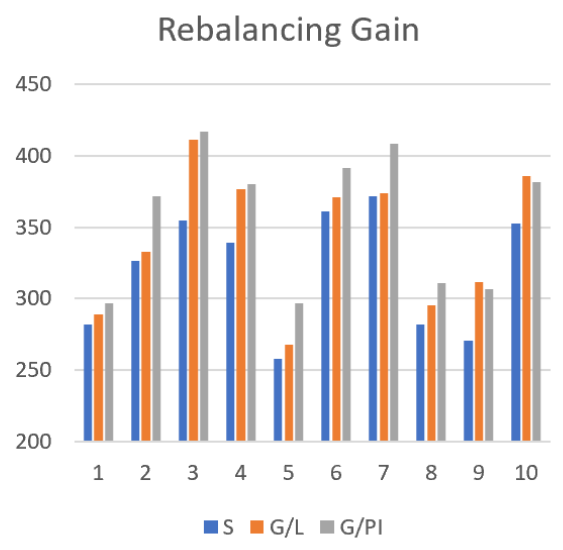

Figure 14 Rebalancing gain for the (DR) formulation applied directly on the set of stations (denoted as “S”), (DR) applied on the group leaders (denoted as “G/L”), and (DRPI) on group leaders with partial information (denoted as “G/PI”), over 10 instances. G/PI obtains the largest rebalancing gain among the three methods.

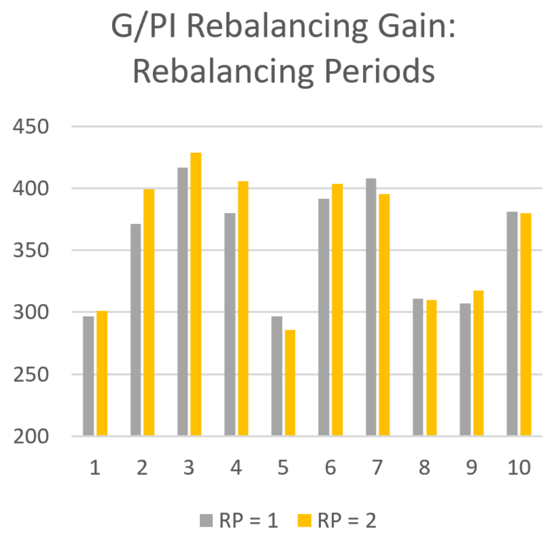

Figure 15 Rebalancing gain for G/PI when the planning horizon consists of a single rebalancing period ($RP = 1$) and (DRPI) is applied, or two ($RP = 2$) rebalancing periods, in which case its multi-period version (MDRPI) is used. The multi-period planning horizon leads to a small improvement of the average gain.

Varying the Rebalancing Planning Horizon. As explained in Section 5.3.1, we can generalize (DR) and (DRPI) into multiperiod versions (MDR) and (MDRPI), where we jointly plan over more than one rebalancing periods. Figure 15 illustrates the results of (DRPI), where we consider a single rebalancing period ($RP = 1$), and (MDRPI) that considers two rebalancing periods ($RP = 2$). Introducing a second rebalancing period in the action planning leads to a further improvement of 1.78% compared to G/PI with $RP = 1$, which brings the average improvement over S (from Figure 14) to 13.36%.

Benefits of Neutral Value Rebalancing Actions. In both models (DR) and (DRPI), we allow for rebalancing actions of neutral value PU_i^o and DO_i^o, where essentially we can pick up and drop off bikes from stations that do not currently need rebalancing in order to better serve other stations that need the resources. We now evaluate the effect of these neutral value rebalancing actions, by comparing the rebalancing gain with the case where neutral value actions are not allowed. In this scenario, trucks can only transfer bikes between stations with imminent rebalancing needs, and this is achieved by enforcing $PU_i^o = 0$ and $DO_i^o = 0$, $\forall i \in S$ in the model. The results are presented in Figure 16. Prohibiting neutral value rebalancing actions leads to an average decrease of 55.3% of the rebalancing gain compared with G/PI with RP = 1. The main reason is that at each cluster at any time the number of bikes that need to be picked up might differ significantly from the number of bikes that need to be dropped off. As a result, auxiliary visits to neighboring stations that do not currently need rebalancing help the truck acquire the necessary resources to satisfy more rebalancing needs.

Comparison with Static Rebalancing. We conclude the experimental analysis by also considering the case of static rebalancing, where redistribution operations take place overnight to prepare the stations for the demand of the following day. In order to determine the initial state of the stations at the beginning of the day, we use our model for user trips (UT) from Section 4. Without specifying the initial state for each station, (UT) provides the initial number of bikes that optimizes the objective function. We can think of this as the ideal static rebalancing that delays the most the time when the first rebalancing need appears. We do not consider the specifics of how this static rebalancing is performed, but wish to evaluate the extent to which our dynamic rebalancing can improve further the service level of the system. Since static rebalancing benefits

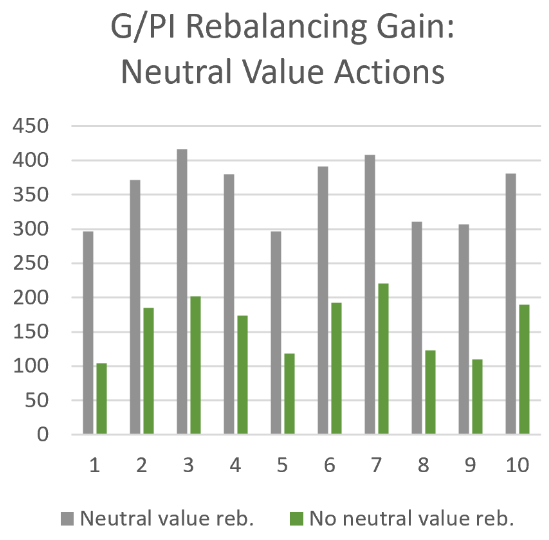

Figure 16 Rebalancing gain for G/PI (model (DRPI)) compared to (DRPI) without neutral value rebalancing actions. Forbidding neutral value rebalancing actions leads to a significant decrease of the rebalancing gain.

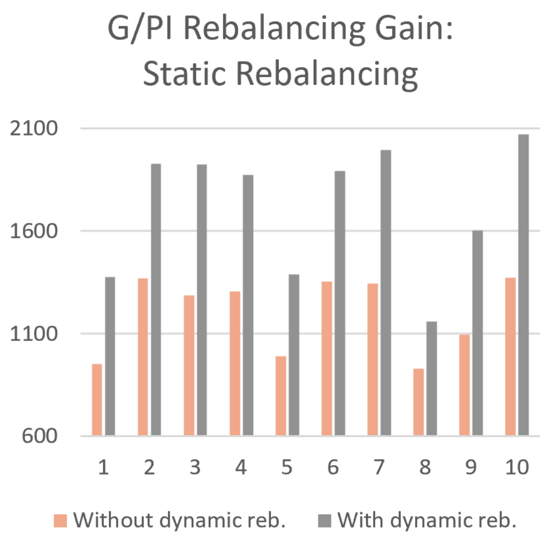

Figure 17 Rebalancing gain for ideal initial state of stations without and with dynamic rebalancing. Dynamic rebalancing improves significantly the rebalancing gain, even when the stations start the day with their ideal bike inventory.

the morning periods (by setting each station to its ideal state) significantly more than the evening ones, in order to properly evaluate its effects, we need to consider the rebalancing gain throughout the whole day for the experiments that follow. The results are presented in Figure 17, which shows the rebalancing gain (over the case with no rebalancing and initially half-full stations) when there is only static rebalancing, and when dynamic rebalancing is performed as well. Performing dynamic rebalancing increases the rebalancing gain by 42.88% in average over all instances.

7. Conclusion

In this work, we focus on demand estimation and dynamic redistribution with an application in bike sharing systems. First, we proposed a data-driven method for the estimation of lost user demand and showed it achieved lower errors with experiments on data from Capital Bikeshare, the bike-sharing system of Washington DC. We addressed the issue of demand relocation to neighboring stations (shifted demand) caused by lack of resources, and developed a linear programming formulation to estimate the probability with which people choose to walk to nearby stations. Computational experiments validate this approach by showing its consistency across the network, as well as demonstrate the effects that the selected parameter values have on the walking probability.

Since dynamic redistribution takes place simultaneously with user trips, we proposed a linear programming formulation that captures the user trips that are performed successfully in the system. With a novel mixed integer programming formulation and the development of a decomposition and optimization algorithm with partial group information, we were able to solve large real-world instances of the dynamic rebalancing problem in running times that were fast enough to allow real-time information to be taken into account. Evaluation of this approach on both synthetic and real-world data revealed the benefits of this method as well as the value of incorporating partial group information, especially when the stations have heterogeneous rebalancing needs.

Our proposed algorithm is not specific to bike sharing systems and can be generalized to other application domains. More specifically, the optimization with partial group information can be applied to the solution of any pick-up and delivery vehicle routing problem. Furthermore, its fast running times extent its applicability to online versions of these problems as well: with our method, online requests and real-time information can be easily taken into account with quick re-optimization of relevant aspects of the current plans. Evaluating the benefits of incorporating partial group information in more general pick-up and delivery applications is of particular interest and the focus of our future work in this area.

Acknowledgments

The research described in this paper was funded in part by the Singapore National Research Foundation (NRF) through the Singapore-MIT Alliance for Research and Technology (SMART) Center for Future Mobility (FM).

References

Arya V, Garg N, Khandekar R, Meyerson A, Munagala K, Pandit V (2014) Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing* 33(3):544–562.

Bezanson J, Edelman A, Karpinski S, Shah VB (2017) Julia: A Fresh Approach to Numerical Computing. *SIAM Review* 59:65–98. doi: 10.1137/141000671. url: http://julialang.org/publications/julia-fresh-approach-BEKS.pdf.

Borgnat P, Abry P, Flandrin P, Robardet C, Rouquier JB, Fleury E (2011) Shared Bicycles in a City: A Signal Processing and Data Analysis Perspective. *Advances in Complex Systems (ACS)* 14(3):415–438.

Caggiani L, Ottomanelli M (2013) A dynamic simulation based model for optimal fleet repositioning in bike-sharing systems. *Procedia - Social and Behavioral Sciences* 87:203–210.

Capital Bikeshare System Data (2017) www.capitalbikeshare.com/system-data.

Capital Bikeshare Tracker (2017) www.cabitracker.com.

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

Contardo C, Morency C, Rousseau LM (2012) Balancing a dynamic public bike-sharing system. *CIRRELT* 4.

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

DeMaio P (2009) Bike-sharing: History, Impacts, Models of Provision, and Future. *Journal of Public Transportation* 12(4):41–56.

Di Gaspero L, Rendl A, Urli T (2013) A Hybrid ACO+CP for Balancing Bicycle Sharing Systems. *Blesa M.J., Blum C., Festa P., Roli A., Sampels M. (eds) Hybrid Metaheuristics. HM 2013. Lecture Notes in Computer Science, vol 7919. Springer, Berlin, Heidelberg.*

Dunning I, Huchette J, Lubin M (2016) JuMP: A Modeling Language for Mathematical Optimization. *In arXiv:1508.01982v3 [math.OC].*

Fishman E (2015) Bikeshare: A Review of Recent Literature. *Transport Reviews* 36(1):92–113

Freund D, Henderson SG, Shmoys DB (2017) Minimizing multimodular functions and allocating capacity in bike-sharing systems. *International Conference on Integer Programming and Combinatorial Optimization.* Springer, Lecture Notes in Computer Science, 10328:186–198.

Forma IA, Raviv T, Tzur M (2015) A 3-step math heuristic for the static repositioning problem in bike-sharing systems. *Transportation Research Part B: Methodological* 71:230–247.

Ghosh S, Varakantham P, Adulyasak Y, Jaillet P (2017) Dynamic Redeployment to Reduce Lost Demand in Bike Sharing Systems. *Journal of Artificial Intelligence Research* 58:387–430.

Goh CY, Yan C, Jaillet P (January 6, 2019) Estimating Primary Demand in Bike-sharing Systems. Available at SSRN: https://ssrn.com/abstract=3311371 or http://dx.doi.org/10.2139/ssrn.3311371.

Ho SC, Szeto WY (2014) Solving a static repositioning problem in bike-sharing systems using iterated tabu search. *Transportation Research Part E: Logistics and Transportation Review* 69:180–198.

Kanungo T, Mount DM, Netanyahu NS, Piatko CD, Silverman R, Wu AY (2014) A local search approximation algorithm for k-means clustering. *Computational Geometry* 28(2-3):89–112.

Lin JR, Yang TH (2011) Strategic design of public bicycle sharing systems with service level constraints. *Transportation Research Part E: Logistics and Transportation Review* 47(2):284–294.

Martinez LM, Caetano L, Eiró T, Cruz F (2012) An optimisation algorithm to establish the location of stations of a mixed fleet biking system: an application to the city of Lisbon. *Procedia - Social and Behavioral Sciences* 54:513–524.

Nair R, Miller-Hooks E (2011) Fleet Management for Vehicle Sharing Operations. *Transportation Science* 45(4):524–540.

Nair R, Miller-Hooks E (2014) Equilibrium network design of shared-vehicle systems. *European Journal of Operational Research* 235(1):47–61.

O'Mahony E (2015) Smarter Tools for (Citi)Bike Sharing. *Doctoral dissertation*, Cornell University.

O'Mahony E, Shmoys DB (2015) Data Analysis and Optimization for (Citi)Bike Sharing. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* 687–694.

Rainer-Harbach M, Papazek P, Hu B, Raidl GR (2013) Balancing Bicycle Sharing Systems: A Variable Neighborhood Search Approach. *Middendorf M., Blum C. (eds) Evolutionary Computation in Combinatorial Optimization. EvoCOP. Lecture Notes in Computer Science, vol 7832. Springer, Berlin, Heidelberg.*

Raviv T, Kolka O (2013) Optimal inventory management of a bike-sharing station. *IIE Transactions* 45(10):1077–1093.

Raviv T, Tzur M, Forma IA (2014) Static repositioning in a bike-sharing system: Models and solution approaches. *EURO Journal on Transportation and Logistics* 2(3):187–229.

Schuijbroek J, Hampshire R, van Hoeve WJ (2017) Inventory Rebalancing and Vehicle Routing in Bike Sharing Systems. *European Journal of Operational Research* 257(3):992–1004.

Shaheen SA, Guzman S, Zhang H (2010) Bikesharing in Europe, the Americas, and Asia: Past, Present, and Future. *Transportation Research Record: Journal of the Transportation Research Board* 2143:159–167.

Shaheen S, Zhang H, Martin E, Guzman S (2011) Hangzhou Public Bicycle: Understanding Early Adoption and Behavioral Response to Bikesharing in Hangzhou, China. *Transportation Research Record* 2247:34–41.

Shu J, Chou MC, Liu Q, Teo CP, Wang IL (2013) Models for Effective Deployment and Redistribution of Bicycles within Public Bicycle-Sharing Systems. *Operations Research* 61(6):1346–1359.

Singhvi D, Singhvi S, Frazier PI, Henderson SG, O'Mahony E, Shmoys DB, Woodard DB (2015) Predicting bike usage for New York City's bike sharing system. *AAAI 2015 Workshop on Computational Sustainability.*

Vogel P, Greiser T, Mattfeld DC (2011) Understanding Bike-Sharing Systems using Data Mining: Exploring Activity Patterns. *Procedia Social and Behavioral Sciences* 20:514–523.

Vogel P, Mattfeld DC (2010) Modeling of repositioning activities in bike-sharing systems. *Proceeding of the 12th world conference on transport research, 1115 July 2010, Lisbon.*

Tao S, Pender S (2017) A Stochastic Analysis of Bike Sharing Systems. *In arXiv:1708.08052v1 [math.PR].*

Yang J, Jaillet P, Mahmassani H (2004) Real-Time Multivehicle Truckload Pickup and Delivery Problems. *Transportation Science* 38(2):135–148. https://doi.org/10.1287/trsc.1030.0068.

Appendix A: Modeling User Trips

A.1. Extension to Multiple Resource Uses per Time Period

Model (UT) in Section 4 with AVAILABLE_BIKES$(i,t) = b_i^t$ and AVAILABLE_DOCKS$(i,t) = C_i - b_i^t$, assumes that each bike and dock can be used only once per time period. This assumption runs throughout this paper including our computational experiments, but since this approach might be considered conservative, we present a way we can relax this by considering the expected number of arrivals until a station becomes empty or full.

Consider the number of bikes at a station i during period t. Initially, it is equal to b_i^t. For each customer arrival, this number increases by one if the customer is returning a bike, and decreases by one if the customer is borrowing a bike. Given that we have $d_{in,i}^t$ customers that want to return a bike, and $d_{out,i}^t$ that want to take one, we can define the probabilities $p_{+1}^{t,i}$ and $p_{-1}^{t,i}$ for a customer returning and taking a bike respectively.

$$p_{+1}^{t,i} = \frac{d_{in,i}^t}{d_{in,i}^t + d_{out,i}^t} \tag{64}$$

$$p_{-1}^{t,i} = \frac{d_{out,i}^t}{d_{in,i}^t + d_{out,i}^t} \tag{65}$$

These two probabilities that sum to 1 for each i and t define a random walk. Each customer arrival is assumed to be an independent random variable that is equal to $+1$ (increases the bikes by 1) with probability $p_{+1}^{t,i}$ and equal to -1 (decreases them by 1) with probability $p_{-1}^{t,i}$. We can now use known results from the theory of random walks to estimate when the station will run out of bikes or docks for the first time.

Let $E_0^{t,i}$ be the expected number of customers for the station i to become empty during period t, and $E_C^{t,i}$ the expected number of customers for the station to become full (reach full capacity C_i). Then, we have:

$$E_0^{t,i} = \begin{cases} +\infty & \text{if } p_{-1}^{t,i} \le p_{+1}^{t,i} \\ \dfrac{b_i^t}{p_{-1}^{t,i} - p_{+1}^{t,i}} & \text{if } p_{-1}^{t,i} > p_{+1}^{t,i} \end{cases} \tag{66}$$

and, similarly:

$$E_C^{t,i} = \begin{cases} +\infty & \text{if } p_{+1}^{t,i} \le p_{-1}^{t,i} \\ \dfrac{C_i - b_i^t}{p_{+1}^{t,i} - p_{-1}^{t,i}} & \text{if } p_{+1}^{t,i} > p_{-1}^{t,i} \end{cases} \tag{67}$$

By replacing $p_{+1}^{t,i}$ and $p_{-1}^{t,i}$ these can equivalently be written as follows.

$$E_0^{t,i} = \begin{cases} +\infty & \text{if } d_{out,i}^t \le d_{in,i}^t \\ b_i^t \cdot \dfrac{d_{in,i}^t + d_{out,i}^t}{d_{out,i}^t - d_{in,i}^t} & \text{if } d_{out,i}^t > d_{in,i}^t \end{cases} \tag{68}$$

and, similarly:

$$E_C^{t,i} = \begin{cases} +\infty & \text{if } d_{in,i}^t \le d_{out,i}^t \\ (C_i - b_i^t) \cdot \dfrac{d_{in,i}^t + d_{out,i}^t}{d_{in,i}^t - d_{out,i}^t} & \text{if } d_{in,i}^t > d_{out,i}^t \end{cases} \tag{69}$$

$E_0^{t,i}$ and $E_C^{t,i}$ are the expected number of customers that can be served, before a lack of resources emerges. From these, the $p_{+1}^{t,i}$ fraction of them will be incoming and the rest outgoing. AVAILABLE_BIKES and AVAILABLE_DOCKS represent the number of outgoing and incoming users that can be served, so they can be easily computed as follows.

$$\text{AVAILABLE\_BIKES}(i,t) = p_{-1}^{t,i} \cdot E_0^{t,i} = \begin{cases} +\infty & \text{if } d_{out,i}^t \le d_{in,i}^t \\ b_i^t \cdot \dfrac{d_{out,i}^t}{d_{out,i}^t - d_{in,i}^t} & \text{if } d_{out,i}^t > d_{in,i}^t \end{cases} \tag{70}$$

$$\text{AVAILABLE\_DOCKS}(i,t) = p_{+1}^{t,i} \cdot E_C^{t,i} = \begin{cases} +\infty & \text{if } d_{in,i}^t \le d_{out,i}^t \\ (C_i - b_i^t) \cdot \dfrac{d_{in,i}^t}{d_{in,i}^t - d_{out,i}^t} & \text{if } d_{in,i}^t > d_{out,i}^t \end{cases} \tag{71}$$

Notice that the above expressions are still linear and, thus, can be incorporated in the linear model (UT) that models the user flows. However, the objective function selected in the Section 4 is no more adequate to satisfy the FCFS principle, since it assumes that each bike and dock is used once per period. Hence, a larger weight needs to be selected that takes into account how many times each bike and dock might be used per time period.

Appendix B: Omitted Details for Models (DR) and (DRPI)

B.1. Obtaining (DRPI) from (DR)

In this section, we present the changes that are required to obtain (DRPI) from (DR). We first rewrite constraint (35) of (DR), which is repeated here for convenience:

$$dur_i \ge \tau_{station} \cdot (1 - x_{ii}) + \tau_{action} \cdot PU_i + \tau_{action} \cdot DO_i \quad \forall i \in S \tag{72}$$

by using auxiliary variables $durPU_i^+$, $durPU_i^o$, $durDO_i^+$, $durDO_i^o \in \mathbb{R}_+$, $\forall i$, which denote the duration of the visit at i spent picking up bikes with positive rebalancing values and neutral rebalancing values, and similarly dropping off bikes with positive and neutral rebalancing values. The above constraint can then be written equivalently as:

$$dur_i \ge \tau_{station} \cdot (1 - x_{ii}) + durPU_i^+ + durPU_i^o + durDO_i^+ + durDO_i^o \quad \forall i \in S \tag{73}$$

$$durPU_i^+ = \tau_{action} \cdot PU_i^+ \quad \forall i \in S \tag{74}$$

$$durPU_i^o = \tau_{action} \cdot PU_i^o \quad \forall i \in S \tag{75}$$

$$durDO_i^+ = \tau_{action} \cdot DO_i^+ \quad \forall i \in S \tag{76}$$

$$durDO_i^o = \tau_{action} \cdot DO_i^o \quad \forall i \in S \tag{77}$$

Now, we are ready to introduce partial information for the actions of each group, using the results of Corollary 1. We keep constraints (73), but replace (74)-(77) with (78)-(81).

$$PU_i^+ \le \min_l(\alpha_{p,l}^{+,i} \cdot durPU_i^+ + \beta_{p,l}^{+,i}) \quad \forall i \in S \tag{78}$$

$$PU_i^o \le \min_l(\alpha_{p,l}^{o,i} \cdot durPU_i^o + \beta_{p,l}^{o,i}) \quad \forall i \in S \tag{79}$$

$$DO_i^+ \le \min_l(\alpha_{d,l}^{+,i} \cdot durDO_i^+ + \beta_{d,l}^{+,i}) \quad \forall i \in S \tag{80}$$

$$DO_i^o \le \min_l(\alpha_{d,l}^{o,i} \cdot durDO_i^o + \beta_{d,l}^{o,i}) \quad \forall i \in S \tag{81}$$

In the following section, we describe how we can obtain the coefficients α and β that are used in (78)-(81).

B.2. Additional Details for Coefficients α and β for (DRPI)

As illustrated in constraints (78)-(81), for each group i, we need four sets of coefficients: $(\alpha_{p,l}^{+,i}, \beta_{p,l}^{+,i})$ for PU_i^+, $(\alpha_{p,l}^{o,i}, \beta_{p,l}^{o,i})$ for PU_i^o, $(\alpha_{d,l}^{+,i}, \beta_{d,l}^{+,i})$ for DO_i^+, and $(\alpha_{d,l}^{o,i}, \beta_{d,l}^{o,i})$ for DO_i^o. Sections 5.5.1 and 5.5.2 outline how these coefficients can be obtained based on the pick-up and drop-off needs of the stations.

Indeed, pick-up and drop-off needs are used to obtain the coefficients for the positive value rebalancing actions PU_i^+ and DO_i^+. In the case of neutral value rebalancing actions, we need to consider the remaining resources for each station, i.e. resources that are both not used by customers and not offering positive rebalancing value. Then, a process similar to Example 3 of Section 5.5.2 can be followed to obtain the line slopes and intercepts.

Since a station might be able to offer resources of both positive and neutral rebalancing value, in order to avoid counting the travel time τ_{travel} twice, for these stations we take τ_{travel} into account only in the line derivation for the positive actions and we skip it for the neutral ones, i.e. the slope is equal to 1 for these stations in the latter case.

B.3. Overhead Time per Visit $\tau_{station}$

In both (DR) and (DRPI), we assume that each visit requires an overhead time $\tau_{station}$ that corresponds to parking, etc. This parameter appears in constraint (35) or equivalently constraint (73), and this overhead time is being enforced on every truck visit. However, there are cases where the truck starts its route being already at the station that constitutes its first visit. In this case, we wish to avoid having to pay the $\tau_{station}$ value, and a way to do that is using the following constraints instead for the station i that is the initial truck location i_0:

$$dur_i \ge \tau_{station} \cdot (1 - x_{ii} - x_{0i}) + \tau_{action} \cdot PU_i + \tau_{action} \cdot DO_i \quad \text{if } i = i_0 \tag{82}$$

$$dur_i \ge \tau_{station} \cdot (1 - x_{ii} - x_{0i}) + durPU_i^+ + durPU_i^o + durDO_i^+ + durDO_i^o \quad \text{if } i = i_0 \tag{83}$$

Variables x_{0i} are used for the dummy node that corresponds to the truck: $x_{0i}=1$, if the truck travels from its initial location directly to i, and 0 otherwise. The use of x_{0i} in the above constraints results in not paying $\tau_{station}$ in the cases where the truck is already at the location of its first visit.

Appendix C: Planning over Multiple Rebalancing Periods

C.1. Model Formulation

Both models (DR) from Section 5.3 and its extension with partial information (DRPI) consider rebalancing actions over a single rebalancing period of length τ_{period}. The generalization for more rebalancing periods, each of length τ_{period}, is provided below.

Each rebalancing period is indexed by r, which takes values between 1 and RP (the number of rebalancing periods for which we are planning, with $RP \geq 2$). For this multi-period version of the formulation, we introduce the constraints of (DR) for each period, using the extra index r for all variables in order to specify the corresponding rebalancing period. For instance, binary variable x_{ij}^r equals 1 if the truck travels directly from i to j during rebalancing period r. We also introduce new binary variables y_i^r to facilitate the transition between consecutive rebalancing periods. In particular, y_i^r equals 1 if the transition from period r to $r+1$ takes place at station i.

(MDR):

$$\max \quad \sum_{i\in S,r}(PU_i^{+r}+DO_i^{+r})-\gamma\sum_{\substack{i\in S_0\\ j\in S,r}} dist_{ij}x_{ij}^r-\delta\sum_{i\in S,r}(PU_i^r+DO_i^r) \tag{84}$$

$$\text{s.t.} \quad \sum_{j\in S_0} x_{ij}^r + y_i^r = 1 \qquad \forall i\in S, r \tag{85}$$

$$\sum_{j\in S_0} x_{ji}^r = 1 \qquad \forall i\in S, r=1 \tag{86}$$

$$\sum_{j\in S_0} x_{ji}^r + y_i^{r-1} = 1 \qquad \forall i\in S, r>1 \tag{87}$$

$$\sum_{i\in S_0} x_{0i}^r = 1, \quad \sum_{i\in S_0} x_{i0}^r = 0 \qquad r=1 \tag{88}$$

$$\sum_{i\in S_0} x_{0i}^r = 0, \quad \sum_{i\in S_0} x_{i0}^r = 1 \qquad r=RP \tag{89}$$

$$x_{00}^r = 1, \quad x_{0i}^r = 0, \quad x_{i0}^r = 0 \qquad \forall i\in S, 1<r<RP \tag{90}$$

$$t_j^r \geq t_i^r + dur_i^r + dist_{ij}\cdot x_{ij}^r - M\cdot(1-x_{ij}^r) \qquad \forall i,j\in S, i\neq j, r \tag{91}$$

$$t_i^r \geq dist_{0i}\cdot x_{0i}^r + \tau_0 \qquad \forall i\in S, r=1 \tag{92}$$

$$t_i^r \leq r\cdot\tau_{period} \qquad \forall i\in S, r \tag{93}$$

$$dur_i^r \geq \tau_{station}\cdot(1-x_{ii}^r) + \tau_{action}\cdot PU_i^r + \tau_{action}\cdot DO_i^r \qquad \forall i\in S, r \tag{94}$$

$$b_j^{Tr} \geq b_i^{Tr} + PU_i^r - DO_i^r - M\cdot(1-x_{ij}^r) \qquad \forall i,j\in S, i\neq j, r \tag{95}$$

$$b_j^{Tr} \leq b_i^{Tr} + PU_i^r - DO_i^r + M\cdot(1-x_{ij}^r) \qquad \forall i,j\in S, i\neq j, r \tag{96}$$

$$b_i^{Tr} \geq \zeta_0 \cdot x_{0i}^r \qquad \forall i \in S, r = 1 \quad (97)$$

$$b_i^{Tr} \leq C^T - (C^T - \zeta_0) \cdot x_{0i}^r \qquad \forall i \in S, r = 1 \quad (98)$$

$$b_i^{Tr} \leq C^T \qquad \forall i \in S, r \quad (99)$$

$$b_i^{Tr} + PU_i^r - DO_i^r \leq C^T \qquad \forall i \in S, r \quad (100)$$

$$b_i^{Tr} + PU_i^r - DO_i^r \geq 0 \qquad \forall i \in S, r \quad (101)$$

$$PU_i^r \leq \text{UNUSED\_BIKES}(r,i) \qquad \forall i \in S, r \quad (102)$$

$$PU_i^{+r} \leq \text{DOCKS\_NEEDED}(r,i) \qquad \forall i \in S, r \quad (103)$$

$$PU_i^r = PU_i^{+r} + PU_i^{or} \qquad \forall i \in S, r \quad (104)$$

$$PU_i^r \leq M \cdot (1 - x_{ii}^r) \qquad \forall i \in S, r \quad (105)$$

$$DO_i^r \leq \text{UNUSED\_DOCKS}(r,i) \qquad \forall i \in S, r \quad (106)$$

$$DO_i^{+r} \leq \text{BIKES\_NEEDED}(r,i) \qquad \forall i \in S, r \quad (107)$$

$$DO_i^r = DO_i^{+r} + DO_i^{or} \qquad \forall i \in S, r \quad (108)$$

$$DO_i^r \leq M \cdot (1 - x_{ii}^r) \qquad \forall i \in S, r \quad (109)$$

$$t_i^r + dur_i^r \leq t_i^{r+1} + M \cdot (1 - y_i^r) \qquad \forall i \in S, r < RP \quad (110)$$

$$t_i^r + dur_i^r \geq t_i^{r+1} - M \cdot (1 - y_i^r) \qquad \forall i \in S, r < RP \quad (111)$$

$$b_i^{Tr} + PU_i^r - DO_i^r \leq b_i^{Tr+1} + M \cdot (1 - y_i^r) \qquad \forall i \in S, r < RP \quad (112)$$

$$b_i^{Tr} + PU_i^r - DO_i^r \geq b_i^{Tr+1} - M \cdot (1 - y_i^r) \qquad \forall i \in S, r < RP \quad (113)$$

$$\sum_{r'=r_1}^{r_2} PU_i^{r'} \leq \text{ALL\_UNUSED\_BIKES}(r_1, r_2, i) \qquad \forall i \in S, r_1 < RP, r_2 > r_1 \quad (114)$$

$$\sum_{r'=r_1}^{r_2} PU_i^{+r'} \leq \text{ALL\_DOCKS\_NEEDED}(r_1, r_2, i) \qquad \forall i \in S, r_1 < RP, r_2 > r_1 \quad (115)$$

$$\sum_{r'=r_1}^{r_2} DO_i^{r'} \leq \text{ALL\_UNUSED\_DOCKS}(r_1, r_2, i) \qquad \forall i \in S, r_1 < RP, r_2 > r_1 \quad (116)$$

$$\sum_{r'=r_1}^{r_2} DO_i^{+r'} \leq \text{ALL\_BIKES\_NEEDED}(r_1, r_2, i) \qquad \forall i \in S, r_1 < RP, r_2 > r_1 \quad (117)$$

$$\sum_{i \in S_0, j \in S} dist_{ij} x_{ij}^r + \sum_{i \in S} dur_i^r \leq \tau_{period} \qquad \forall r \quad (118)$$

$$\sum_{i \in S} \tau_{action} \cdot PU_i^r + \sum_{i \in S} \tau_{action} \cdot DO_i^r \leq \tau_{period} \qquad \forall r \quad (119)$$

$$x_{ij}^r \in \{0,1\} \qquad \forall i,j \in S_0, r \quad (120)$$

$$y_i^r \in \{0,1\} \qquad \forall i \in S, r \quad (121)$$

$$t_i^r, dur_i^r, b_i^{Tr}, PU_i^r, PU_i^{+r}, PU_i^{or}, DO_i^r, DO_i^{+r}, DO_i^{or} \in \mathbb{R}_+ \qquad \forall i \in S, r \quad (122)$$

Constraints (85), (86) and (87) specify the truck routing between the stations: the truck can travel from station to station or transition to the next rebalancing period while remaining at the same station. We have one dummy node for the truck per period, which has an outgoing arc at period 1 (88) that denotes the start of the route, an incoming arc at period RP (89) denoting the end of the route, and corresponds to self-loops in all intermediate periods (90). Constraints (91) to (109) and (118) to (119) are directly derived from (DR) (Section 5.3) by introducing one such constraint per rebalancing period r. Constraints (110) to (113) ensure that the state of the truck (load and arrival time) remains consistent between consecutive rebalancing periods. Finally, constraints (114) to (117) impose limits on the resources: a bike could be available for pick-up/drop-off during more than one rebalancing periods, but if this action takes place during one of the periods, it shouldn't take place again in any other period. We introduce such constraints for any pair r_1, r_2 with $1 \leq r_1 < r_2 \leq RP$, but since RP is small in our application this does not lead to a large number of constraints.

Overhead Time per Visit $\tau_{station}$. Similar to Appendix B.3, we can make sure we do not consider the overhead time $\tau_{station}$ multiple times for the same visit, by further specifying (94) to take into account the truck transitioning to the next period, since it remains at the same location:

$$dur_i^r \geq \tau_{station} \cdot (1 - x_{ii}^r - y_i^{r-1}) + \tau_{action} \cdot PU_i^r + \tau_{action} \cdot DO_i^r \quad \forall i \in S, r > 1 \tag{123}$$

as well as the station i that consists the initial location i_0 of the truck during the first rebalancing period:

$$dur_i^r \geq \tau_{station} \cdot (1 - x_{ii}^r - x_{0i}^r) + \tau_{action} \cdot PU_i^r + \tau_{action} \cdot DO_i^r \qquad \text{if } i = i_0, r = 1 \tag{124}$$

For $r = 1$ and $i \neq i_0$, (94) remains as is.

(MDR) with Partial Information. The incorporation of partial information to (MDR) can take place in a similar way as in Appendix B.1, and this generates the Multiperiod Dynamic Rebalancing with Partial Information model (MDRPI).

Appendix D: Computational Results

This section includes detailed statistics of the solutions in the Computational Experiments Section 6.2.2. Unless noted otherwise, they refer to the rebalancing actions that take place until noon in each instance. The rebalancing gain is the extra number of customers that can be served compared to the case where no rebalancing is performed and the stations start the day half full. The travel distance is in kilometers, the travel and service times in minutes, and the running time in seconds. All quantities are aggregated for all trucks and time periods.

Table 6 Rebalancing results for method comparison: S [(DR) on the set of stations], G/L [(DR) on the set of leaders], G/PI [(DRPI)].

Instance	Method	Served users	Rebalancing gain	Bikes rebalanced	Travel distance	Travel time	Service time	Run time
1	S	3207.48	281.89	326.29	263.96	791.88	532.29	3550.88
	G/L	3214.63	289.04	377.78	297.37	892.11	606.79	548.83
	G/PI	3222.22	296.62	363.20	286.17	858.51	572.20	561.01
2	S	3527.41	326.09	426.67	275.68	827.05	648.67	4195.29
	G/L	3534.31	333.00	485.98	309.07	927.21	764.98	752.01
	G/PI	3572.71	371.40	479.71	320.06	960.18	741.72	722.60
3	S	3637.30	354.91	446.95	323.98	971.94	675.95	4223.83
	G/L	3693.63	411.24	498.66	347.47	1042.41	778.66	543.12
	G/PI	3699.15	416.76	487.59	356.84	1070.51	751.59	617.78
4	S	3467.68	339.37	475.45	276.75	830.26	711.45	3571.24
	G/L	3504.70	376.39	546.24	289.54	868.63	828.24	534.52
	G/PI	3508.42	380.11	533.85	297.38	892.13	799.86	588.18
5	S	3123.00	257.61	332.24	262.39	787.16	543.24	3589.69
	G/L	3133.53	268.14	379.32	278.71	836.12	615.32	524.66
	G/PI	3162.12	296.73	376.89	289.88	869.65	616.90	589.37
6	S	3328.63	361.15	437.51	277.07	831.21	659.51	3533.19
	G/L	3338.14	370.65	510.62	304.90	914.69	786.62	560.72
	G/PI	3358.92	391.43	515.45	298.99	896.98	778.46	609.03
7	S	3406.20	371.73	440.75	291.81	875.42	663.75	3903.83
	G/L	3408.13	373.67	498.18	320.32	960.97	762.18	547.78
	G/PI	3442.52	408.05	494.34	323.42	970.26	749.34	602.76
8	S	3114.34	281.96	384.51	241.41	724.23	603.51	3465.21
	G/L	3127.75	295.37	444.36	256.81	770.44	712.36	482.50
	G/PI	3143.35	310.96	423.33	264.32	792.96	659.34	535.93
9	S	3167.63	270.72	339.25	262.23	786.68	550.25	4111.02
	G/L	3208.47	311.57	411.87	287.54	862.61	649.87	532.29
	G/PI	3203.84	306.94	396.49	298.57	895.71	640.49	591.51
10	S	3565.88	352.60	466.47	326.99	980.96	717.47	3908.96
	G/L	3598.69	385.41	501.53	340.61	1021.82	801.53	553.99
	G/PI	3594.39	381.12	509.23	334.67	1004.01	776.23	669.61

Table 7 Rebalancing results for single-period rebalancing (DRPI) ($RP = 1$) and multi-period rebalancing (MDRPI) with $RP = 2$.

Instance	Rebalancing periods (RP)	Served users	Rebalancing gain	Bikes rebalanced	Travel distance	Travel time	Service time	Run time
1	1	3222.22	296.62	363.20	286.17	858.51	572.20	561.01
	2	3226.47	300.87	384.86	215.49	646.48	580.86	745.87
2	1	3572.71	371.40	479.71	320.06	960.18	741.72	722.60
	2	3600.80	399.49	539.31	275.64	826.93	788.32	1362.67
3	1	3699.15	416.76	487.59	356.84	1070.51	751.59	617.78
	2	3711.22	428.82	504.44	293.67	881.01	750.44	1124.86
4	1	3508.42	380.11	533.85	297.38	892.13	799.86	588.18
	2	3534.33	406.02	567.77	254.68	764.04	811.77	1136.69
5	1	3162.12	296.73	376.89	289.88	869.65	616.90	589.37
	2	3151.01	285.62	395.23	265.06	795.19	611.23	906.79
6	1	3358.92	391.43	515.45	298.99	896.98	778.46	609.03
	2	3370.82	403.34	516.02	265.66	796.97	759.02	1245.97
7	1	3442.52	408.05	494.34	323.42	970.26	749.34	602.76
	2	3429.93	395.46	527.92	265.59	796.76	762.92	918.46
8	1	3143.35	310.96	423.33	264.32	792.96	659.34	535.93
	2	3142.24	309.86	469.28	225.94	677.83	692.29	746.82
9	1	3203.84	306.94	396.49	298.57	895.71	640.49	591.51
	2	3214.38	317.48	411.88	230.51	691.54	617.88	918.41
10	1	3594.39	381.12	509.23	334.67	1004.01	776.23	669.61
	2	3593.51	380.24	530.23	294.75	884.25	792.24	1045.79

Table 8 Rebalancing results comparing the benefit of neutral value rebalancing actions.

Instance	Neutral value actions allowed	Served users	Rebalancing gain	Bikes rebalanced	Travel distance	Travel time	Service time	Run time
1	True	3222.22	296.62	363.20	286.17	858.51	572.20	561.01
	False	3029.70	104.11	98.14	187.21	561.63	198.14	623.45
2	True	3572.71	371.40	479.71	320.06	960.18	741.72	722.60
	False	3386.46	185.15	160.93	271.20	813.59	309.94	704.32
3	True	3699.15	416.76	487.59	356.84	1070.51	751.59	617.78
	False	3484.33	201.94	148.10	268.87	806.61	285.10	644.80
4	True	3508.42	380.11	533.85	297.38	892.13	799.86	588.18
	False	3301.77	173.46	168.96	212.69	638.08	301.96	601.16
5	True	3162.12	296.73	376.89	289.88	869.65	616.90	589.37
	False	2983.64	118.25	90.76	171.82	515.46	173.77	752.86
6	True	3358.92	391.43	515.45	298.99	896.98	778.46	609.03
	False	3159.55	192.06	166.19	244.33	732.98	304.20	597.84
7	True	3442.52	408.05	494.34	323.42	970.26	749.34	602.76
	False	3254.62	220.16	178.90	234.12	702.36	305.91	592.47
8	True	3143.35	310.96	423.33	264.32	792.96	659.34	535.93
	False	2955.20	122.81	111.28	156.89	470.67	216.28	600.55
9	True	3203.84	306.94	396.49	298.57	895.71	640.49	591.51
	False	3007.05	110.14	89.08	180.56	541.68	180.08	682.53
10	True	3594.39	381.12	509.23	334.67	1004.01	776.23	669.61
	False	3402.69	189.41	177.70	237.45	712.34	321.71	583.63

Table 9 Results when dynamic rebalancing is combined with ideal static rebalancing.

Instance	Served users	Rebalancing gain	Bikes rebalanced	Travel distance	Travel time	Service time	Run time
1	10749.45	1375.52	455.87	445.89	1337.68	834.87	771.17
2	12858.70	1929.27	681.68	592.85	1778.54	1183.69	854.30
3	12532.99	1925.33	691.02	667.61	2002.83	1250.02	841.53
4	11453.28	1872.85	651.61	588.83	1766.48	1130.61	864.52
5	11115.80	1389.29	453.77	457.79	1373.36	841.77	834.03
6	11986.09	1892.76	600.09	623.04	1869.12	1137.09	846.21
7	13033.88	1995.86	727.58	569.19	1707.58	1224.58	854.15
8	7099.40	1158.66	297.04	308.61	925.82	578.04	749.17
9	12100.82	1602.95	646.40	566.54	1699.62	1145.40	848.98
10	13142.20	2072.26	760.87	566.25	1698.76	1269.87	877.45

These results refer to the duration of the whole day.

Table 10 Served users without dynamic rebalancing.

	Half day		Whole day	
Instance	Half-full	Ideal	Half-full	Ideal
1	2925.59	3466.75	9373.94	10326.48
2	3201.31	4057.04	10929.42	12298.92
3	3282.39	4169.91	10607.66	11894.27
4	3128.31	3994.23	9580.42	10886.20
5	2865.38	3415.82	9726.51	10718.06
6	2967.49	3822.17	10093.33	11446.87
7	3034.47	3867.97	11038.02	12382.42
8	2832.38	3458.28	5940.74	6870.32
9	2896.90	3499.44	10497.87	11593.02
10	3213.28	4084.85	11069.94	12441.93

Number of served customers for half day (until noon) and the whole day when stations' initial state is half-full or at its ideal level (obtained by (UT) and representing the results of an ideal static redistribution in the system).