sensors

MDPI

Article

A Dynamic Approach to Rebalancing Bike-Sharing Systems

Federico Chiariotti [1], Chiara Pielli [1], Andrea Zanella [1,2,*] and Michele Zorzi [1,2]

[1] Department of Information Engineering, University of Padova, 35131 Padova PD, Italy; chiariot@dei.unipd.it (F.C.); piellich@dei.unipd.it (C.P.); zorzi@dei.unipd.it (M.Z.)
[2] Human Inspired Technologies (HIT) Research Center, University of Padova, 35131 Padova PD, Italy
* Correspondence: zanella@dei.unipd.it; Tel.: +39-049-827-7770

Received: 18 December 2017; Accepted: 30 January 2018; Published: 8 February 2018

Abstract: Bike-sharing services are flourishing in Smart Cities worldwide. They provide a low-cost and environment-friendly transportation alternative and help reduce traffic congestion. However, these new services are still under development, and several challenges need to be solved. A major problem is the management of rebalancing trucks in order to ensure that bikes and stalls in the docking stations are always available when needed, despite the fluctuations in the service demand. In this work, we propose a dynamic rebalancing strategy that exploits historical data to predict the network conditions and promptly act in case of necessity. We use Birth-Death Processes to model the stations' occupancy and decide when to redistribute bikes, and graph theory to select the rebalancing path and the stations involved. We validate the proposed framework on the data provided by New York City's bike-sharing system. The numerical simulations show that a dynamic strategy able to adapt to the fluctuating nature of the network outperforms rebalancing schemes based on a static schedule.

Keywords: bike sharing; Smart Cities; dynamic rebalancing

1. Introduction

One of the most promising applications of the emerging Internet of Things (IoT) paradigm is Smart Cities [1]: the myriad sensors and actuators deployed in a modern city can provide many novel services to citizens and institutions and make the existing ones more efficient and user-friendly.

Bike sharing is a perfect example of a Smart City-enabled service. Although bike-sharing schemes have existed since the 1960s [2], issues with theft, vandalism and wrongful usage prevented their widespread adoption [3] until the arrival of smart biking systems. With the possibility of electronically unlocking bicycles and identifying users, smart bike-sharing systems solved or mitigated these issues: by requesting users' credit card information before lending the bikes, bike-sharing companies can charge users for damages or theft, with a strong deterrent effect. The use of technology also allowed cities to provide a better service, embedding sensors to obtain real-time data, which can be used to plan and manage the bike-sharing docking stations and adapt to the needs of the users. Many services provide live maps of the available bikes, and ways to detect broken bikes remotely to quickly repair or substitute them are being studied [4]. Over the past few years, bike-sharing schemes were implemented in most major world cities, with almost universal success. Some of the biggest bike-sharing services in the world are in Barcelona, Paris, London, Hangzhou, Taiyuan and Shanghai in China, New York City, and Montreal. The usage data were easily recorded and used to improve the service and increase the number of users [5,6], reducing traffic and pollution. The abundance of data, sometimes including GPS tracking of the bikes' trajectories, has led to new opportunities for Smart City management, such as, for example, the planning of new bike lanes to cover the most common routes [7].

In May 2013, New York City launched CitiBike, the largest bike-sharing system in North America. The CitiBike bike-sharing network consists of more than 700 docking stations in Manhattan, Brooklyn and Jersey City, with tens of thousands of rides every day. Despite such large numbers, the service is not always satisfactory. For example, the massive use of the service by commuters results into the quick depletion of the stations in residential areas in the morning hours, and the rapid exhaustion of the available stalls in the docking stations in commercial areas, so that users cannot deposit more bikes. In addition, stations at the top of slopes are more likely to be empty, as users are less keen on cycling uphill. This disparity highlights the importance of finding smart ways to manage large bike-sharing systems, and is reflected by the high rate of negative reviews of the service on TripAdvisor (36% of users rate CitiBike as "average" or lower) and by competitors' recent experiments with "dockless" bikes that can be parked anywhere and unlocked with a smartphone app.

An effective way to improve bike-sharing systems consists of relocating bikes from overcrowded stations to those with a shortage of bikes. This technique is commonly known as *rebalancing*. Operators can, e.g., deploy a fleet of trucks to pick up and drop off bikes at different stations to balance the network. The trucks can all depart from the same location, or be distributed in different depots, depending on the size of the network and the operational costs of the service.

The design of an efficient rebalancing scheme, then, requires addressing two major challenges: predicting the demand for bikes and stalls in the docking stations in the different locations covered by the service, and selecting the most efficient way to relocate the bikes in order to best satisfy such a demand. Rather than performing bike rebalancing based on a static scheme, as is common in practical bikesharing services, in this study, we propose a *dynamic rebalancing* scheme that aims at increasing the users' satisfaction by minimizing the probability of *service failures*, i.e., the chance that a user experiences an unsatisfactory service because of the unavailability of bikes or stalls in the docking stations. We model the occupancy of each station as a Birth-Death Process (BDP), with time varying birth and death rates (i.e., arrivals and departures, respectively). In this way, we are able to estimate how long a station will be self-sufficient before running out of bikes or available stalls, knowing its initial state. Periodically, we use the knowledge about such expected survival times to decide whether rebalancing is necessary. Clearly, employing a fleet of vehicles to rebalance the system has a monetary cost that counterbalances the improvement in the network conditions by moving bikes among stations. This cost also depends on the covered distance, as this affects the fuel consumption and the time required for the rebalancing operations. These are important factors that need to be tuned carefully when designing a rebalancing strategy because they play a role in deciding when to perform the bike relocation.

To gauge the performance of our scheme, we use three different metrics: the fraction of time the system is out of service (empty or full stations), the number of daily rebalancing operations, and the daily distance covered by the rebalancing trucks. Our scheme is compared with the benchmark case in which no relocation is done, and a static rebalancing scheme from the literature that rebalances bikes twice a day at preset times [8]. All the numerical evaluations are based on the dataset of the CitiBike system. In particular, we use the data collected over the past four years, during which the bike-sharing system grew in both number of stations and available bikes. The observed trips are a lower bound for true demand, as empty or full stations prevent users from taking or returning bikes, respectively. In addition, there may be unexpected events such as broken or stolen bikes that affect the bike availability, or special events in the city that make the bike demand differ from its usual pattern. Nonetheless, although the numerical results may be altered by the censoring problem, they are still representative of the positive impact the rebalancing has on the bike-sharing system. Furthermore, the framework we propose is general and can be readily adapted to new data. It is also an effective tool to identify the critical stations and locations so as to improve the bike-sharing systems by adding bikes at selected stations and increasing the station density in the critical areas. For a deeper analysis of the dataset, with considerations on other patterns such as weather and distance from mass transit stops,

we refer the reader to our related work in [9], in which we compare the results from New York City with another dataset from a smaller system.

The rest of this work is organized as follows. Section 2 gives an outline of the state of the art on Smart Cities, bike sharing and rebalancing optimization. Section 3 presents our model of the bike-sharing problem, while Section 4 introduces the CitiBike dataset and the processing to derive the model parameters from it. Section 5 describes the performance of our dynamic rebalancing scheme. Finally, Section 6 contains our conclusions and some possible avenues of future research on the subject.

2. Related Work

Over the last few years, bike-sharing systems have gained a lot of momentum and become an active field of research, bolstered by the availability of real data from publicly open bike-sharing services in many cities. A comprehensive survey on the state of the art literature about bike-sharing systems can be found in [10], while [11] provides a history of bicycle sharing since 1965, when the city of Amsterdam launched the first bike-sharing program. In this section, we first introduce the possibilities of bike sharing as an integrated Smart City service. Then, we propose an overview of the two major branches of the scientific literature on bike-sharing system optimization, namely, the analysis of historical data to infer the demand, and rebalancing optimization techniques to limit system failures and user dissatisfaction.

2.1. Bike Sharing as a Smart Service

Bike sharing is a key component of future Smart Cities: bike-sharing services offer more flexibility than standard public transportation while also reducing both traffic and its impact on the environment [12] and improving public health [13]. According to an American Planning Association report [14], bike-sharing programs in North America significantly decreased subscribers' car use, and bike sharing in Hangzhou, China, reportedly decreased CO_2 emissions by 70 thousand tons a year. Like other mobility services, bike sharing provides a double benefit to Smart Cities: in addition to the direct improvements caused by the service, the data that the system gathers can be used to build models of citizens' behavior and make more informed policy decisions, as well as providing a direction in which the Smart City can grow organically. Since Smart Cities are "systems of systems" [15], mobility is a key component and the integration of bike sharing in a Smart Mobility framework [16] can help achieve the goals of both the transport system and the city as a whole. This integration between data, services and communications has been proposed as the key idea of the SymbioCity paradigm [17].

An integrated approach to public transport, with car- and bike-sharing schemes supplementing standard bus and light rail systems, can improve transit times and encourage the shift towards a more sustainable urban mobility [18]. A study on two American cities [19] has found that bike-sharing is an alternative to public transport in high-density areas where distances are short, while the independence of these systems from fixed schedules and the higher density of stations can complement the deficiencies of traditional mass transit systems in lower-density areas, creating the missing "last mile" connection. The possibility of using e-bikes to increase the viable range of the system is also being explored [20].

Bike sharing also has an impact on urban planning: the presence of bike lanes [21,22] and urban zoning [23] has a strong effect on ridership, and future urban planning in Smart Cities should take this into account, both using bike sharing as a sustainable and flexible way to connect citizens and encouraging the transition from a car-oriented perspective to a more integrated approach to transportation, zoning and urban design [24]. Although there are several trade-offs and political issues that need to be tackled to shift towards a smart and sustainable mobility [25], research is focusing on formulating principles to guide urban design and help in the transition [26]: a key component of future mobility is the physical separation of fast, high mass vehicles such as cars and trucks from bikes, pedestrians, and other slow, low mass vehicles, increasing both safety and efficiency.

2.2. Traffic Analysis and Rebalancing

Recently, researchers have put a lot of effort in analyzing bike-sharing usage data to build models and frameworks that help improve the system. In [27], the bike demand at each station is modeled as a queuing system with finite capacity, and the service level requirements are then found using the transient distribution of the bikes' availability and exploiting Kolmogorov forward equations [28]. This approach is similar to that proposed in this work. However, rather than determining a lower and upper bound for the target inventory level, we compute the occupancy state that maximizes the survival time of a station (i.e., the time before the stalls are either all empty or all full). Furthermore, our model is more flexible, being able to accommodate non stationary demand patterns.

Another efficient tool to extrapolate valuable information from real datasets is represented by machine learning. For example, Ref. [29] uses a neural network to learn the bike traffic pattern directly from the observed data, and also includes information such as the weather conditions, which are often troublesome to consider in an analytical model. The main drawback of this approach is that it requires a customized neural network for each bike-sharing system, whereas parameterized analytical models can be readily adapted to different bike-sharing systems.

Traffic analysis is also useful to construct clusters of stations with similar demand patterns [30], which may help the rebalancing optimization.

Although the historical data analysis gives insights on the bike demand, the heterogeneous user behavior and unexpected events make it challenging to accurately predict the traffic pattern [31]. Thus, external intervention is required to keep the system operational and limit failures due to the unavailability of bikes or docks, as this would have a negative impact on the users. A possibility consists in developing pricing strategies to incentivize users to return bikes to the least loaded among the closest stations [32,33]. However, pricing mechanisms may not be sufficient to achieve self-rebalancing of the system, while being annoying for the users. Hence, the majority of the existing bike-sharing systems employ a fleet of vehicles that pick up and move bikes between stations.

The most common rebalancing approaches can be classified as *static*, where the rebalancing is performed according to a predetermined schedule, likely when the system use is minimum (e.g., at night); or *dynamic*, in which the rebalancing occurs during the daytime, when needed.

In the literature, static rebalancing problems are often tackled through Mixed Integer Programming (MIP) techniques. For example, Ref. [34] uses a branch-and-cut algorithm on a relaxed MIP model together with a taboo search to obtain upper bounds to the NP-hard problem of redistributing bikes among stations with the minimum traveling distance. A similar method is used by Ho and Szeto in [35] inside an iterative procedure, resulting in a better solution. Several MIP techniques are used in [36], where a convex penalty objective function aims at minimizing both the user dissatisfaction and the costs of moving the vehicle fleet. The numerical experiment shows significant service quality improvements for networks with up to 100 stations with two repositioning vehicles. The authors also take into account the time needed to load and unload bikes. Constraint programming is used in [37], where a large neighborhood search approach is used to tackle the problem of balancing the bike-sharing system. Ref. [38] proposes a three-step mathematical programming based heuristic, which consists of clustering stations based on geographical and inventory considerations, and then constructing an appropriate traversal route. Differently from other approaches, rebalancing is not limited within clusters, but the routing vehicles can travel through a sequence of clusters. Similarly, Ref. [27] proposes a static rebalancing scheme where each vehicle is assigned a 'self-sufficient' cluster of stations. Stations are in fact grouped in such a way that the target network configuration can be achieved by performing only within-cluster pickups and deliveries. Clusterization is useful to organize rebalancing paths more easily, but adopting such a technique in a dynamic rebalancing context is not trivial, since each rebalancing operation may involve different stations and clusters should continuously be identified. In [39], Dell'Amico et al. propose a metaheuristic that uses a destroy and repair approach to solve the static rebalancing problem, improving on the simple branch-and-cut approach that the same authors presented in [40], both in the quality of the solution and in the

convergence time. These works are particularly interesting, as they consider the variability of the demand for bikes during the day and the different features of the stations, distinguishing between three different types of stations with similar traffic patterns.

Static approaches, however, may not be sufficient to avoid network failures during the day. To overcome this issue, dynamic rebalancing aims at redistributing bikes throughout the day according to the current network state. Clearly, this is much more challenging, since it includes a scheduling component based on the users' activity during the rebalancing operation, and also a routing problem.

In [41], upper and lower bounds for the solution of a pickup-and-delivery problem are obtained by using Dantzig–Wolfe [42] and Benders [43] decompositions, respectively, but the study does not deal with a time-varying demand and, even in this simple scenario, the optimality gap of the approximation is large. An exact solution to the static rebalancing problem using Benders' cuts for given target levels is instead provided in [44].

In [45], the loading instructions for the rebalancing vehicles are derived from an optimization function that weighs the unfulfilled user demand with the target filling level. Furthermore, each truck is assigned a capacity so that there is a maximum number of bikes it can load. An accurate simulation model is proposed in [29]: it simulates the bike-sharing system in space and time to determine the optimal repositioning flows and time intervals between relocation operations, by explicitly considering the route choice for trucks among the stations. Some studies also determine the optimal size of the rebalancing fleet [46]. A Markov Decision Process (MDP) is used in [47] to solve a stochastic inventory routing problem for bike-sharing systems. The objective is to minimize the number of expected violations of *due dates*, where a due date is the deadline within which a station has to be served by a rebalancing vehicle in order to satisfy the requests and, hence, avoid failures. The paper shows the importance of using both long-term and short-term relocation strategies. The former tries to estimate the target level that can deal with predicted demand, while the latter assigns a priority level to each station based on its urgency of being rebalanced. The short-term approach is similar to the one proposed in this paper, since we dynamically serve only those stations that are going to experience a failure in a short time horizon. An interesting framework is presented in [8], where rebalancing comprises two different strategies to tackle the bike imbalance during night and day, respectively: (i) static rebalancing overnight to move the network to an optimal configuration that minimizes the probability of stations becoming either empty or full in the following day, and (ii) clustering optimization to handle rush-hour usage and ensure that users are never too far from an available bike or dock. The joint use of the two strategies brings a larger improvement than considering the nightly rebalancing only. This result encouraged us in choosing a dynamic approach.

For a more detailed discussion on the rebalancing problem and other open research issues in bike-sharing systems, we refer the reader to [10,48].

To the best of our knowledge, our work is the first to consider a dynamic rebalancing scheme that can minimize the time during which stations are unable to meet user demand while considering the full complexity of the time-varying demand. Most previous efforts in the literature concentrate on finding the optimal route to rebalance bikes at fixed times, while we consider both whether the system should be rebalanced and which stations need it most, avoiding unnecessary trips.

3. System Model

We define the network of bike-sharing stations as a fully connected directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}^2)$, where $\mathcal{V} \triangleq \{1, \ldots, V\}$ is the set of stations, and $\mathcal{E}^2 = (\mathcal{E}, \mathcal{E})$ with $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges between stations. Note that each link is counted twice to allow for different costs between two stations according to the direction of the path. Each node $v \in V$ is characterized by its capacity $M_v \in \mathbb{N}$, i.e., the total number of bike stalls at the station, and its current occupancy $m_v(t) \in \mathcal{M}_v \triangleq \{0, 1, \ldots, M_v\}$, i.e., the number of bikes available at time t. The edges are weighed by a distance metric between stations, denoted as $d(v_i, v_j)$. As the graph is fully connected, each station can be reached from any other station.

Our goal is to determine a dynamic rebalancing scheme that minimizes the user dissatisfaction while maintaining the rebalancing cost as low as possible. The dissatisfaction is related to the probability that users experience service failures, i.e., find stations either empty ($m_v(t) = 0$) or full ($m_v(t) = M_v$). (In this work, we do not distinguish between failures due to empty and full stations, though this aspect will be studied in future work.) We assume that a fleet of vehicles with given capacities is available for the rebalancing procedure and that the operating day is divided into discrete time frames of length T_r. The rebalancing fleet can be deployed at the beginning of each frame.

The rebalancing problem is made up of two main components.

1. At each time frame k, we need to determine the desired state $m_v^\star(kT_r)$ for each station $v \in \mathcal{V}$. Since we would like to maximize the time until a station becomes either empty or full, $m_v^\star(kT_r)$ is the inventory level that maximizes the *survival time* of the station. This is computed by analyzing the historical user data.
2. The real rebalancing problem, which consists of identifying which stations are either overcrowded or in shortage of bikes, according to their survival times, and designing the truck routes to perform the bikes pickups and deliveries.

In the following, we first describe the procedure to determine the survival times based on the historical data, and then the network-wide rebalancing optimization.

3.1. Survival Time

Modeling the network as a whole is not computationally tractable for a large number of stations. However, we can model each station's occupancy separately as a Markov-Modulated Poisson Process (MMPP) [49]: the occupancy follows a finite Markov BDP $m_v(t)$, where the birth and death processes are Poisson distributed with time-varying rates $\lambda_v(t)$ and $\mu_v(t)$, respectively. We can reasonably assume that the demand for bikes is not affected by the current occupancy of the station (unless it is either empty or completely full(The censoring problem will be described in Section 4), and therefore the birth and death rates are assumed to be independent of the current state. The BDP is limited by the station's capacity M_v.

In order to reduce the computational complexity [50,51] and accurately estimate $\lambda_v(t)$ and $\mu_v(t)$ from the historical data, we simplify the station behavior as a discrete-time BDP, with birth and death rates that are piecewise constant during time frames of duration T_r, which is an integer multiple of the BDP slot T_p. Basically, we map the continuos time t into its discrete counterpart

$$t \rightarrow nT_r + kT_p \quad \text{with} \quad n = \lfloor t/T_r \rfloor,\ k = \lfloor (t - nT_r)/T_p \rfloor, \tag{1}$$

where n is the index of the *frame* (which has duration T_r) and k is the index of the *slot* (which has duration T_p). The two assumptions we have made (piecewise constant birth and death rates, and discrete time) make it possible to have a simpler yet accurate model of the stations' dynamics, and also to derive the correct parameters of the arrival and departure processes for the available dataset. Once T_r was defined, we verified that the empirical distribution of the inter-arrival times in a single time frame T_r fits an exponential distribution, thus validating the short-term Poisson assumption.

As mentioned, a service failure occurs whenever a station is in state 0 or in state M_v: if the stalls are all empty, bikes cannot be rented, while if they are all full, users cannot deposit more bikes. In order to compute the first passage time to either of the two failure states, we can consider an equivalent process in which 0 and M_v are absorbing states. Accordingly, we formally define the survival time $S_v(t, m)$ of node v as the smallest time τ after which the probability $P_{m,a}(\tau; t)$ of reaching state $a \in \{0, M_v\}$, starting from state $m \in \mathcal{M}_v$ at time t, exceeds a predefined threshold p_{th}, i.e.:

$$S_v(t, m) = \inf \{\tau : P_{m,0}(\tau; t) + P_{m,M_v}(\tau; t) \geq p_{\text{th}}\}. \tag{2}$$

Note that $S_v(t, m)$ is time varying, reflecting the fluctuations of the arrival and departure rates. For ease of writing, however, the parameter t will be omitted in the following, whenever possible. The threshold p_{th} makes it possible to tune the reactivity of the rebalancing system: lower values of p_{th} will yield shorter survival times and, in turn, more frequent rebalancing, which will reduce the service failure risk but increase the operational costs of the service.

Since the birth and death rates vary from frame to frame, the transition probability $P_{i,j}(t)$ needs to be calculated by considering the state distribution at the end of the previous frame. Applying the Markov property, we get:

$$P_{i,j}(t) = P_{i,j}(nT_r + kT_p) = \sum_{l \in \mathcal{M}_v} P_{i,l}(nT_r)\, P_{l,j}(kT_p), \qquad i, j \in \mathcal{M}_v\,, \tag{3}$$

where n and k are given by the time relation (1). The transition matrix after $k < T_r/T_p$ steps is the $(k+1)$-th power of the one-step transition matrix (note that the time slot index k starts from 0 according to the time relation (1)), and then we use the Markov property as in Equation (3) to derive the transition probabilities when the system is in frame n and slot k.

To compute $P_{m,0}(t)$ and $P_{m,M_v}(t)$ for a given station v, we need to evaluate the transition probabilities starting from state m at every time step. In the following, we thus focus only on a single slot of length T_p, with birth and death rates denoted as μ_v and λ_v, respectively. Let D_v and A_v be the random variables that describe the numbers of departures and arrivals at station v in the considered slot, respectively. We model the demand for bikes (i.e., potential departures) and empty stalls (i.e., potential arrivals) as Poisson variables, with mean $\mu_v T_p$ and $\lambda_v T_p$, respectively. We assume that a sequence of events results in the same state regardless of the order of occurrence, so that the next state of the station is given only by the total number of departures and arrivals in a time slot. This is not strictly correct, as different orderings of the same events might bring the system in an absorbing state. In general, however, the probability of this event is sufficiently low and does not significantly affect the accuracy of the simplified model. The probability $P_{i,j}$ of going from state i to state j in one time step T_p is

$$\begin{aligned} P_{i,j} &= \Pr[A_v - D_v = j - i] = \sum_{\ell=-\infty}^{\infty} \Pr[A_v = j - i + \ell] \Pr[D_v = \ell] \\ &= \sum_{\ell=\max\{0, j-i\}}^{+\infty} \frac{(\lambda_v T_p)^{-(j-i+\ell)}\, (\mu_v T_p)^{-\ell}}{\ell!\, (j-i+\ell)!}\,, \end{aligned} \tag{4}$$

where the last equality follows from the Poisson distribution of A_v and D_v. The result of the sum in Equation (4) follows a truncated Skellam distribution [52]:

$$p_{\text{Sk}}(\ell; \mu_1, \mu_2) = e^{-(\mu_1+\mu_2)} \left(\frac{\mu_1}{\mu_2}\right)^{l/2} I_{|\ell|}(2\sqrt{\mu_1 \mu_2})\,, \tag{5}$$

with $\ell = j - i$, $\mu_1 = \lambda_v T_p$ (the average of A_v), and $\mu_2 = \mu_v(t) T_p$ (the average of D_v). $I_k(z)$ is the modified Bessel function of the first kind and order k [53]:

$$I_\alpha(x) = \lim_{N \to \infty} \sum_{n=0}^{N} \frac{1}{n! \Gamma(n + \alpha + 1)} \left(\frac{x}{2}\right)^{2n+\alpha}\,, \tag{6}$$

$$\Gamma(z) = \int_0^\infty x^{z-1} e^{-x} dx\,. \tag{7}$$

The mass probability function given in Equation (5) is for the standard Skellam distribution, while we need to truncate it over the finite range $\{0, \ldots, M_v\}$. Thus, the transitions to the two absorbing states 0 and M_v have the following probabilities:

$$P_{i,0} = \sum_{k=-\infty}^{0} p_{\mathrm{Sk}}(k-i;\mu_1,\mu_2)\,, \qquad i \in \mathcal{M}_v \backslash \{0, M_v\}\,, \tag{8}$$

$$P_{i,M_v} = \sum_{k=M_v}^{+\infty} p_{\mathrm{Sk}}(k-i;\mu_1,\mu_2)\,, \qquad i \in \mathcal{M}_v \backslash \{0, M_v\}\,. \tag{9}$$

Since 0 and M_v are absorbing states, $P_{0,0} = P_{M_v,M_v} = 1$, while $P_{0,j} = 0\,\forall j \neq 0$ and $P_{M_v,j} = 0\,\forall j \neq M_v$.

In practice, the transition probabilities at time instant t are computed in three steps with a time hierarchical interrelation, which exploits the time equivalence of Equation (1):

1. We use the Markov property to separate the problem at a frame level; to calculate the transition probabilities in frame $n > 0$, it is necessary to compute those at the end of each frame $n' < n$.
2. Within a single frame n, the transition probabilities at slot $k > 0$ are derived by elevating the single-step transition matrix of frame n to the power of $k + 1$ (since slot indices start from 0).
3. The transition probabilities at the beginning of a frame (i.e., for slot index $k = 0$) are then computed using Equations (4), (8) and (9).

Hence, for each time instant t and starting state m, we are able to compute the probability of being in one of the failure states, 0 or M_v, and, consequently, the survival time $S_v(t, m)$ of station v. Intuitively, $S_v(t, m)$ is first increasing and then decreasing in m (it is 0 for both $m \equiv 0$ and $m \equiv M_v$), and there exists a state $m_v^\star(t)$ that leads to the maximum survival time $S_v^\star(t)$:

$$m_v^\star(t) = \underset{m \in \mathcal{M}_v}{\operatorname{argmax}}\, S_v(t, m) \quad \text{and} \quad S_v^\star(t) = S_v(t, m_v^\star(t)), \qquad v \in \mathcal{V}. \tag{10}$$

Ideally, at time t, we would like to be in state $m_v^\star(t)$ as it corresponds to the longest time for which station v is self-sufficient (i.e., it does not need rebalancing). Next, we describe how to plan the rebalancing operation by exploiting the knowledge of the survival times.

3.2. Network-Wide Optimization

We model the dynamic rebalancing problem as an optimization problem on the bike-sharing network graph $\mathcal{G}$. The number of deployed trucks is predefined and denoted as X, and we assume that the subsets of stations visited by each vehicle are disjoint. Each truck has a given capacity, which defines the maximum number of bikes it can transport concurrently. Then, we define a circular graph $\mathcal{H} = (\mathcal{V}', \mathcal{F})$, where $\mathcal{V}' \subseteq \mathcal{V} \cup \{0\}$ is the union of the set of stations that the fleet of trucks rebalances ($\subseteq \mathcal{V}$) and the truck depot ($\{0\}$), while $\mathcal{F}$ is the set of edges that solves the Vehicle Routing Problem (VRP) [54] on $(\mathcal{V}', \mathcal{E}^2)$.

We denote the station occupancy vector before rebalancing as $\mathbf{m}^{(\varnothing)}(t) \triangleq [m_1^{(\varnothing)}(t), \ldots, m_{|\mathcal{V}|}^{(\varnothing)}(t)]$, while the occupancy vector after rebalancing is $\mathbf{m}^{(\mathcal{H})}(t)$.The obtained reward is measured as the time gained before a system failure occurs, i.e., the difference between the minimum survival times among all the stations after and before the rebalancing operation. Each rebalancing operation also implies a fixed cost for each vehicle in the fleet, plus an additional cost that depends on the total length $D_x(\mathcal{H})$ of the route of each vehicle $x \in [1, \ldots, X]$. The rebalancing optimization function $\tilde{f}(\mathcal{H}, t)$ can thus be modeled as

$$\tilde{f}(\mathcal{H}, t) = \left(\min_{v \in \mathcal{V}} S_v(t, m_v^{(\mathcal{H})}(t)) - \min_{v \in \mathcal{V}} S_v(t, m_v^{(\varnothing)}(t)) \right) - \alpha X - \beta \sum_{x=1}^{X} D_x(\mathcal{H})\,, \tag{11}$$

where and $\alpha \geq 0$ and $\beta \geq 0$ are two weights (measured in time units and time units per kilometer, respectively), and the survival times $S_v(\cdot, \cdot)$ are computed as described in Section 3.1.

We introduce a further parameter γ in the optimization function, which indicates the clipping threshold for the survival time: this has the dual purpose of disregarding longer survival times, which might have a larger estimation error, and avoiding useless rebalancing operations when the minimum survival time of the system is already very long. The optimization function then becomes

$$f(\mathcal{H},t) = \left(\min\left\{\min_{v\in\mathcal{V}} S_v(t, m_v^{(\mathcal{H})}(t)), \gamma\right\} - \min\left\{\min_{v\in\mathcal{V}} S_v(t, m_v^{(\varnothing)}(t)), \gamma\right\}\right) - \alpha X - \beta \sum_{x=1}^{X} D_x(\mathcal{H}). \quad (12)$$

As discussed in the introduction of Section 3, the operating day is divided into discrete frames of length T_r. The rebalancing problem is solved periodically, every T_r, and consists of defining $\mathcal{V}'$ and $\mathcal{F}$, which means i) identifying which stations require rebalancing, and ii) determining the path for the rebalancing truck. Note that, if at time t the optimization function $f(\mathcal{H},t)$ is negative for any $\mathcal{V}' \neq 0$, the cost of moving the vehicle fleet is larger than the reward obtained for reallocating the bikes, and thus no rebalancing is performed. In this case, $\mathcal{V}' = \{0\}$.

Since the optimization function also depends on the distance covered by the rebalancing truck, the set of stations to visit, $\mathcal{V}'$, and the vehicle route, $\mathcal{F}$, should be jointly determined. For simplicity, we first assume to know the minimum path length of the rebalancing route for a given set of nodes, so that we can derive $\mathcal{V}'$ and then describe how to compute such distance. The computation is simplified by the following theorem, which states that $\mathcal{V}'$ contains the nodes with the smallest survival times, so that all subsets that do not satisfy the theorem can be safely discarded as suboptimal.

Theorem 1. *If* $v \in \mathcal{V}'$, *then* $S_v(t, m_v^{(\varnothing)}(t)) < S_u(t, m_u^{(\varnothing)}(t)) \; \forall u \in \mathcal{V} \setminus \mathcal{V}'$

Proof. Assuming the truck follows the optimal route $\mathcal{F}$, adding a node to $\mathcal{H}$ cannot decrease the distance $D_x(\mathcal{H})$ because of the triangle inequality. It follows that, in order for v to be part of the rebalancing set $\mathcal{V}'$, its presence needs to increase the first term of the function in (12), i.e., $S_v(t, m_v^{(\varnothing)}(t)) < \min_{u\in(\mathcal{V}\setminus\mathcal{V}')} S_u(t, m_u^{(\varnothing)}(t))$. □

The procedure to determine $\mathcal{V}'$ is hence described in Algorithm 1. Intially, $\mathcal{V}'$ only contains the deposit node 0 (Line 1); then, at each iteration i, we identify the node $v(i) \in \mathcal{V}\setminus\mathcal{V}'$ with the minimum survival time (Line 9). We compute the reward obtained by adding $v(i)$ to $\mathcal{V}'$, as if its survival time were optimal, $S_{v(i)}(t) \equiv S^{\star}_{v(i)}(t)$ (Line 10), and the cost to add it to the truck route (Line 14). In particular, the reward is computed as the difference between the smallest survival time in set $\mathcal{V}'$ and the smallest survival time before rebalancing (Lines 4 and 11), while the cost depends on the new route length (see Equation (12)). If the optimization function increases (Line 15), node $v(i)$ is indeed added to $\mathcal{V}'$. The algorithm stops if either all nodes of $\mathcal{V}$ have been included in $\mathcal{V}'$, or the optimization function can no longer be increased because the minimum of the optimal survival times among all visited nodes $w \in \mathcal{V}'$ after rebalancing is smaller than the current survival time of any of the unvisited nodes (those in $\mathcal{V}\setminus\mathcal{V}'$) (Line 18).

The solution of the dynamic rebalancing problem has two distinct parts: (i) determining the route for the set of stations that need to be rebalanced (Line 12 in Algorithm 1) including possible additional constraints given by vehicle capacity and travel times, and (ii) deciding whether to start the rebalancing operation and, possibly, which stations to visit. The literature on the first part of the problem is already extremely vast and well-developed. In particular, any of the solutions discussed in Section 2.2 can be used to determine the rebalancing path. Instead, there are very few studies on the second problem, i.e., deciding which station to rebalance and when, which is the focus of our model. We remark that the path-planning algorithm has an impact on the rebalancing strategy provided by our approach, since the cost computed in Line 14 includes the covered distance that, in turn, depends on the chosen path. However, our framework is general and can accomodate any path-planning strategy.

Algorithm 1 Rebalancing strategy

Initialize $i = 0$, $\mathcal{V}' = \{0\}$, $f(\mathcal{H}, t) = 0$, done $= 0$
Set parameters α, β, γ, X
Set $\mathbf{S}(t) = [S_1(t, m_1^{(\varnothing)}), \ldots, S_{|\mathcal{V}|}(t, m_{|\mathcal{V}||}^{(\varnothing)})]$ ▷ Vector with survival times before rebalancing
Set $\sigma = \min_{w \in \mathcal{V}} \mathbf{S}(t)$ ▷ Smallest survival time before rebalancing
$\sigma = \min\{\sigma, \gamma\}$ ▷ Set the survival time threshold
Set $\mathbf{S}^\star(t) = [S_1^\star(t), \ldots, S_{|\mathcal{V}|}^\star(t)]$ ▷ Vector with optimal survival times for each station $v \in \mathcal{V}$
while $(i < V)$ and (done $==$ 0) **do** ▷ Until there are nodes to visit that can improve the net reward
    $i \leftarrow i + 1$
    $v(i) \leftarrow \text{argmin}_{w \in \mathcal{V} \setminus \mathcal{V}'} \mathbf{S}(t)$ ▷ Choose unvisited node with smallest survival time
    $[\mathbf{S}(t)]_{v(i)} \leftarrow [\mathbf{S}^\star]_{v(i)}(t)$ ▷ Update survival time of node $v(i)$
    reward $\leftarrow \min\{\min \mathbf{S}(t), \gamma\} - \sigma$ ▷ Update reward
    Determine rebalancing path $\mathcal{F}$ over $\mathcal{V}' \cup v(i)$
    $D \leftarrow$ compute path length of $\mathcal{F}$ ▷ Compute distance to cover for rebalancing
    cost $\leftarrow \alpha X + \beta D$ ▷ Update cost
    if (reward $-$ cost) $> f(\mathcal{H}, t)$ **then** ▷ It is worth to include node $v(i)$ in the rebalancing
        $\mathcal{V}' \leftarrow \mathcal{V}' \cup v(i)$
        $f(\mathcal{H}, t) \leftarrow$ reward $-$ cost
    if $\min_{w \in \mathcal{V}'}\{[\mathbf{S}^\star(t)]_w\} < \min_{w \in \mathcal{V} \setminus \mathcal{V}'}\{[\mathbf{S}(t)]_w\}$ **then**
        done $\leftarrow 1$ ▷ The optimization function $f(\mathcal{H}, t)$ can no longer be increased

The path-planning problem can be *optimally* solved by the well-known VRP, which yields the lowest possible cost. However, the VRP is an NP-hard problem, so that several efficient optimization algorithms have been defined in the literature for different scenarios and sets of assumptions, in order to reduce the computational complexity. In our simulation, we made the following assumptions to simplify the problem:

1. We use the Euclidean distance between stations as distance metric for the set of edges $\mathcal{E}^2$. This implies that $d(v_i, v_j) \equiv d(v_j, v_i)$ for any two stations i and j.
2. We consider a single rebalancing vehicle, i.e., $X = 1$, with infinite capacity. Notice that this scenario reduces the VRP to the Traveling Salesman problem.

The Traveling Salesman problem is still NP-hard. Since the focus of this work is on the dynamic algorithm and not on the solution to the routing problem, we opted for a simple heuristic that greedily chooses the closest unvisited node at each step. Let $\mathcal{U}_i \subseteq \mathcal{V}'$ be the set of nodes visited up to iteration i. Since the truck deposit is the first node to be visited, we set $\mathcal{U}_0 = u(0) \equiv \{0\}$, while for $i \in \{1, \ldots, |\mathcal{V}'| - 1\}$ (Note that the last iteration is $i = |\mathcal{V}'| - 1$ because $\mathcal{V}'$ includes also the deposit node $\{0\}$.), we have $\mathcal{U}_i = \mathcal{U}_{i-1} \cup \{u_i\}$, where u_i is the node visited at iteration i, such that

$$u_i = \underset{w \in \mathcal{V}' \setminus \mathcal{U}_{i-1}}{\text{argmin}} \; d(w, u_{i-1}), \tag{13}$$

where, as mentioned, $d(x, y)$ is the Euclidean distance between nodes x and y. This rule implements the well-known *nearest-neighbor* heuristic, which has often been used as a benchmark for Traveling Salesman heuristic solutions [55]. The total path distance covered by the rebalancing truck is

$$D(\mathcal{H}) = \sum_{i=1}^{|\mathcal{V}'|-1} d(u_i, u_{i-1}) + d(u_{|\mathcal{U}|-1}, 0), \tag{14}$$

where the last term is the distance the vehicle covers to go back to the deposit after the rebalancing operations. This heuristic can be computed in $\mathcal{O}(|\mathcal{V}'|^2)$ time. The total running time of the heuristic algorithm is $\mathcal{O}(|\mathcal{V}|^3)$, since $|\mathcal{V}'| \leq |\mathcal{V}|$. Although using a greedy heuristic may not give the optimal routing solution, solving an NP-hard problem might not be practical for large instances.

4. Data Analysis

In this work, we used the publicly available dataset (https://www.citibikenyc.com/system-data) from New York City's CitiBike network, which we briefly described in Section 1. A map of the city, with the positions of the docking stations, is shown in Figure 1.

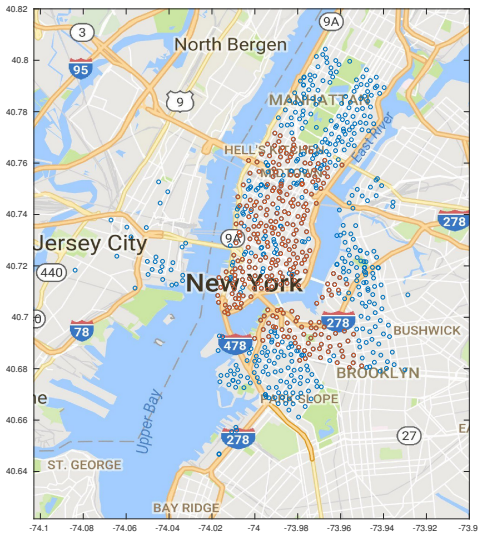

Figure 1. Map of the bike-sharing stations: red points identify the stations considered in our study.

The service publishes monthly reports with all the recorded rides, the start and destination times and stations, the unique identifiers of the bikes, and some records about the service subscribers, such as year of birth and gender. In this study, we use the data from July 2013 (the earliest available period) to July 2017. Since several stations were added during the considered time frame, we limit ourselves to the 280 stations that were present for a sufficiently long time to extract the demand data. (By discarding some of the stations, the system is not closed; bikes enter and exit it as they travel to other stations. We clarify how we deal with this issue in Section 5.)

Traffic Analysis and Simulation

The traffic pattern intuitively depends on several factors, such as the time of the year (fewer people tend to use bikes in winter than in summer), whether it is a weekday or weekend (people may tend to use bikes at more regular times on weekdays and for shorter periods), and the position of the considered station (and the facilities nearby), as shown in Figure 2, which reports the traffic pattern for a station close to CUNY Baruch College during the month of July 2015. On weekdays, the rush hours around 8 a.m. and 5 p.m. can be identified by the spikes in the demand for bikes; since the station is probably used by many students and workers, there are more arrivals than departures in the morning and more departures than arrivals in the evening. On Fridays, there is a more homogeneous pattern in the afternoon, likely because many people leave work early, and the demand is much lower during the weekend. These patterns highlight that both the day of the week and the hour need to be considered when calculating the arrival and departure rates. Another important factor is the weather: the summer months have the highest demand for bikes, while the service is used far less during colder months.

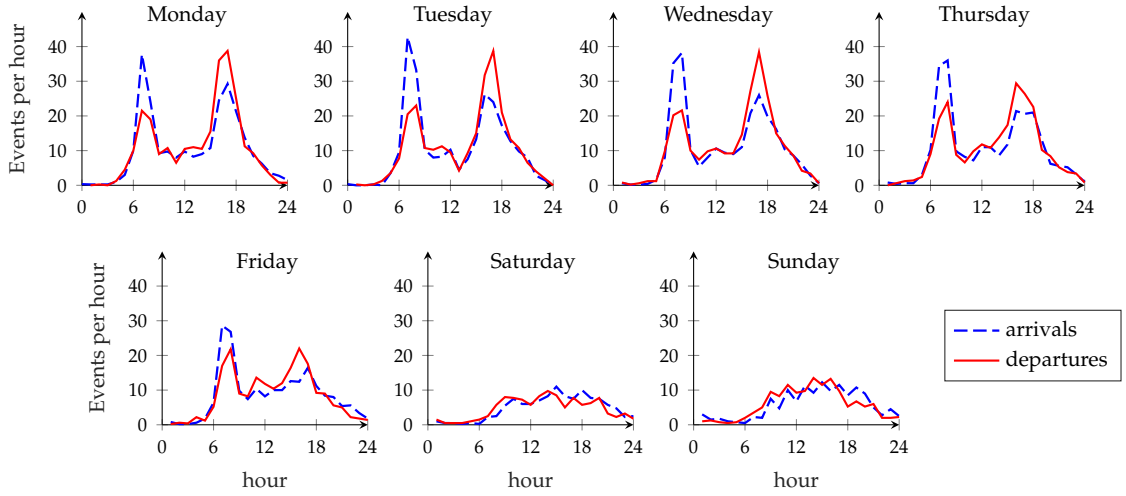

Figure 2. Traffic patterns for the month of July 2015 at station 537 (Lexington Ave. and East 24th St.)

This fits with the simplifying assumptions in Section 3. We assume that the Poisson rates $\lambda_v(\cdot)$ and $\mu_v(\cdot)$ for the arrival and departure process, respectively, are piecewise constant at steps of duration T_r equal to one hour. Hence, we can extract the value of the arrival and departure rates for each period Ω, defined as the hour, day, and month of the year, for all the stations in the network. Since the dataset contains four years of data, we calculate $\lambda_v(\Omega)$ and $\mu_v(\Omega)$, $v \in \mathcal{V}$ as the mean arrival rate in the first three years, from July 2013 to June 2016. The test set on which we perform the optimization is the last year of data, from July 2016 to June 2017. This provides a sizeable amount of data, though the increasing of the number of stations and subscribers affects the accuracy of the estimates, which is unavoidable when using past data for future optimization.

There are some other issues that need to be mentioned. Stations that are empty or full prevent customers from taking or returning bikes, so that the observed trip data may actually differ from the true demand and rather represent a lower bound for it. In addition, the bike demand may increase thanks to the rebalancing operations, as people would see that the system is becoming more efficient. Unfortunately, we have no way to gauge the real demand from the available data, and the effect of the current rebalancing operations prevents the extrapolation of the real state of the system at any given time. However, the framework we propose is general and can be easily adapted to different datasets. Thus, although the numerical evaluation could be affected by the censoring issue, the model and the proposed solution have general validity.

5. Results

In order to validate our optimization scheme, we compare its performance with two other scenarios: a system with no rebalancing and one with static rebalancing. In the first case, we just let the bike-sharing system evolve without intervening. In the second one, we bring the whole network to the optimal state twice a day, at 3 a.m. and 3 p.m., as in [8]. In our simulations, we make the following assumptions.

- The CitiBike data accurately represent the demand for bikes. As discussed in Section 4, this may not be strictly true because of the censoring effect, so that the demand patterns in the data represent a lower bound for the real demand.
- Bikes coming from or going to stations that are not considered in the optimization framework because of lack of data are assumed to be new bikes entering the system, or broken bikes exiting it, respectively.
- If a user does not find a bike at their desired station, they simply exit the system and the trip does not happen. If they find a bike at their departure station but do not find a free spot at the arrival station, the bike exits the system.

- At the beginning of every month, the state of each station is reset to the optimal value. Since the dataset does not provide the state of each station, this assumption was necessary to define the initial state of the system.

Naturally, the optimization schemes themselves are not affected by these assumptions, but the numerical results of the simulation may change when choosing different assumptions. The simulations were run for a whole year of data, from July 2016 to June 2017, and the results are presented on a month-by-month basis. We set the cost of taking a rebalancing trip, α, to 2700 s (45 min); since the duration of a slot T_p is 15 min, this means that the minimum survival time must improve by at least an hour for the system to consider a rebalancing operation. The parameter β, which represents the cost of moving the rebalancing truck, was set to four different values, namely 0.02, 0.04, 0.08, and 0.12 s/m.

We used the data without any optimization scheme to test the assumptions of our model. First, we verified the accuracy of the Poisson model for the interarrival times by considering the data of several randomly chosen stations, which exhibited an extremely good fit with the exponential distribution, despite the relatively small sample.

The second test was intended to check the validity of the survival time formula, for which we run the prediction without implementing any rebalancing strategy, in order not to perturb the system evolution. This test was performed for two randomly chosen weeks in September 2016 and January 2017, and the results were extremely good, since the empirical distribution of the survival time obtained from the dataset was almost overlapping with our model, with a relative error on the survival time lower than 1.5%. The slight discrepancy can be explained by time-dependent factors such as the yearly increase in the demand for bikes, which causes the model to slightly underestimate the demand in late 2016 and early 2017 since we computed the survival time using the data from July 2013 to June 2016.

Service improvement. Figure 3 shows the fraction of time that the bike-sharing system spends in a failure state, i.e., either completely full or empty, averaged for each station and for each month of the year. We used this metric instead of the number of service failures because it is less affected by the censoring problem we described in Section 4, as requests that could not be fulfilled are not shown and would not be counted in the results.

The unoptimized system has a failure probability of about 14%, which means that, without any rebalancing intervention, stations cannot satisfy the users' needs for about 14% of the time. The static rebalancing strategy from [8], which optimizes the whole system twice a day at preset times, can improve this to about 11% during the summer and 6% during the winter. This discrepancy is due to the far lower demand during colder months, which makes rebalancing actions more effective. The dynamic system can improve the service even further: with low values of β, the failure time can be as low as 3% during the summer and 0.4% during the winter. This obviously comes at a cost, as an aggressive rebalancing strategy will result in more daily trips and a higher cost to move the trucks. However, the framework we propose makes it possible to tune the dynamic scheme to the specific requirements of the system, in order to balance the operational costs and quality of the service as desired. For example, setting $\beta = 0.12$, we can approximately replicate the performance of the static scheme, as shown in Figure 3.

In order to have a full picture of the effectiveness of the scheme, we need to consider not only the total failure time but also the kinds of failures, as the two failure modes have a different impact on the behavior of the users and on the future planning of the system. Figure 4 shows the ratio of the time the stations spend completely full over the total failure time. The unoptimized system has a very skewed failure mode, with almost 90% of failures due to empty stations. The static rebalancing scheme has a full station ratio of about 35%, which is almost constant throughout the year. The dynamic rebalancing schemes show a higher ratio of full stations during the winter months, mirroring the overall pattern from Figure 3: since most of the failures during the summer months are due to empty stations, the improvement during the winter comes from the stations being empty less often, while the time they spend full does not decrease as much. This effect is particularly significant for $\beta = 0.12$, which shows

a variation from 15% in summer to over 30% in winter, while the other settings of the dynamic scheme are more similar to the static case.

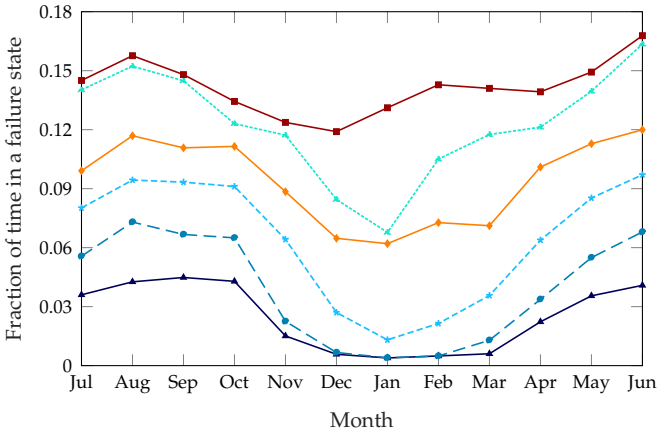

Figure 3. Monthly average probability of being in a failure state, over a year.

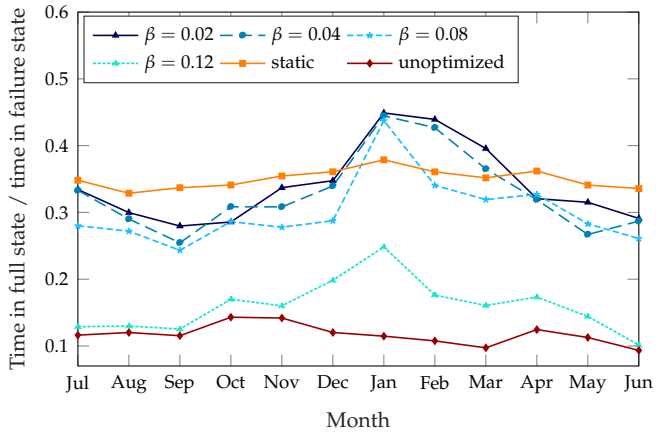

Figure 4. Fraction of failures due to full stations, over a year.

Rebalancing costs. The improvement in the service obviously comes at a cost: the fleet of rebalancing trucks must be deployed several times a day, and a more aggressive rebalancing strategy increases both the number of trips and the total distance that needs to be covered, with higher fuel and personnel costs.

Figure 5 shows the number of trips the rebalancing truck has to take every day, averaged over each month; the unoptimized system is not shown in this figure, since no trips are taken at all. The static scheme has a constant number of trips per day, since it explicitly requires the truck to take two trips at predetermined times. As for the dynamic schemes, the large difference between the summer and winter months might be counterintuitive, as one would expect a lower demand to correspond to a lower number of trips. However, when demand is lower survival times are larger in general; stations that are not in their optimal state might therefore benefit more from a rebalancing, increasing the potential reward. Since the cost of rebalancing the system is constant, rebalancing the system is often more rewarding during the winter, resulting in a higher number of trips and a better service, as Figures 3 and 5 show. This issue is exacerbated by time-dependent factors: as described in Section 5, most of the anomaly disappears if we use the real demand data instead of the estimate based on historical patterns, and this can be explained by weather patterns, as 2017 was a particularly warm and dry winter.

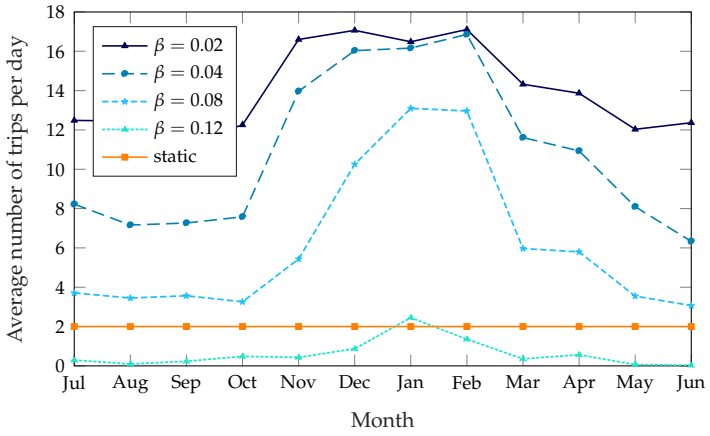

Figure 5. Average number of rebalancing trips per day, over a year.

The dynamic schemes need a higher number of trips to get the performance improvement, but the distance the trucks cover on each trip is far lower: while the static scheme rebalances the whole system and covers every station that needs rebalancing, the dynamic schemes only target the stations that need it most. As Figure 6 shows, the dynamic schemes need between 30 and 40 km per trip, while the static scheme can need more than 90 km during the summer. Figure 7 shows the average distance the truck covers during a day: even if the dynamic scheme with $\beta = 0.04$ takes more than twice as many trips as the static scheme, with a corresponding performance benefit, the total distance it covers is similar during the summer. The scheme with $\beta = 0.12$ reaches a service quality similar to the static scheme's while taking fewer trips in the summer and covering a smaller distance during most of the year. The parameter α, which we kept at a constant value of 2500 s throughout these simulations, can be used to change the preference of the system from frequent short trips to fewer and longer ones.

In general, the dynamic schemes can take shorter trips, potentially deploying a smaller fleet. (We remark that we only consider one rebalancing truck in this work; future developments are going to address this issue, as the model and optimization function we propose are fully general.) The dynamic nature of our solution allows us to tune the rebalancing to the needs of the system, giving system controllers the freedom to choose the optimal point in the trade-off between service quality and rebalancing costs by changing α and β.

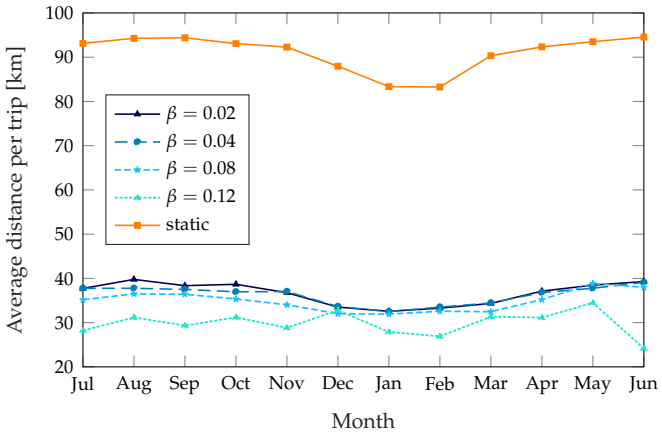

Figure 6. Average distance per rebalancing trip, plotted over 12 months.

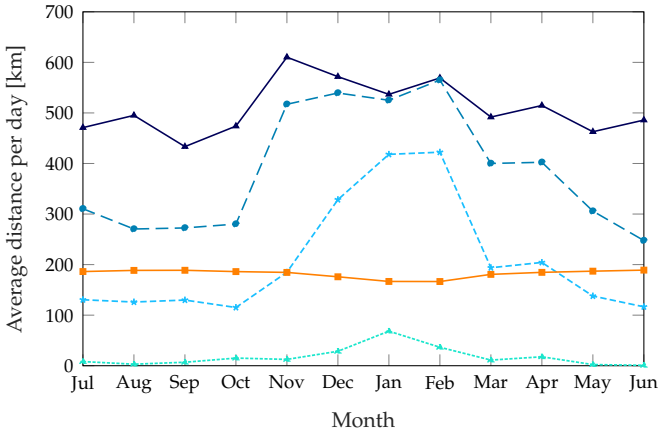

Figure 7. Average distance covered by rebalancing trucks per day, plotted over 12 months.

Figures 8 and 9 show the average failure rate and average daily distance covered by the rebalancing trucks for different values of α and β, considering the full year. The plots clearly show the trade-off between rebalancing costs and service performance: using lower values of β results in a lower failure rate, but the rebalancing trucks need to be deployed more often and, consequently, cover far more distance every day. The dynamic scheme with $\beta = 0.12$ results in a slightly higher failure rate than the static scheme, but requires less than half the rebalancing distance. The parameter β is the key to choose the operating point in the trade-off, as it has a significant impact on the performance of the scheme. The value of α has a limited effect over the considered range, so it can be used for fine-tuning the system after β has been selected.

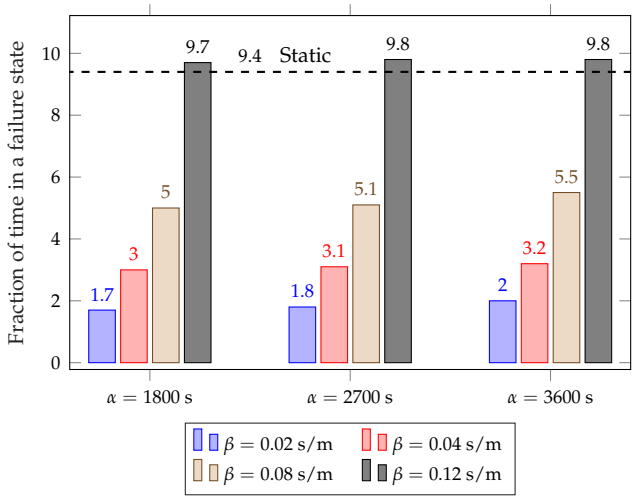

Figure 8. Average fraction of time in a failure state (completely empty or full station) for different values of α and β.

Evolution of the system over time. Figure 10 shows the average fraction of time that a station spends in a failure state for 13 September (Tuesday) and 17 September (Saturday) 2016. As shown in Figure 2, the difference between a weekday and a holiday is striking: in the first plot, the two intense demand periods starting around 7 a.m. and 5 p.m. are evident from the steep increase of the failure rate, which is almost entirely absent in the other one. The limitations of the static approach to rebalancing can be clearly seen in the first plot: while the rebalancing operation at 3 p.m. brings the failure rate very close to 0 in the following hour, the system is left to itself during rush hour, and the failure rate quickly rises to match that of the unoptimized system. Even though the dynamic approach also has a peak failure rate at rush hour, it quickly recovers and keeps the system in a better state throughout

the day; the bike-sharing system is probably underdimensioned for the peak demand, so it is hard for any rebalancing scheme to avoid failure during rush hour, but the dynamic system still yields a significant improvement. The weekend plot shows a similar pattern for the static approach, with perfect functioning right after rebalancing and a steadily increasing failure rate afterwards. However, the lower and more homogeneous demand makes rebalancing less urgent and decreases the gain of using a dynamic approach.

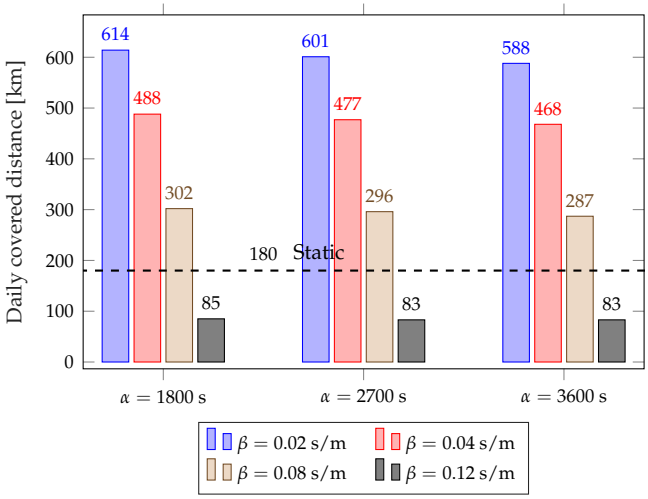

Figure 9. Average daily distance covered by rebalancing trucks for different values of α and β.

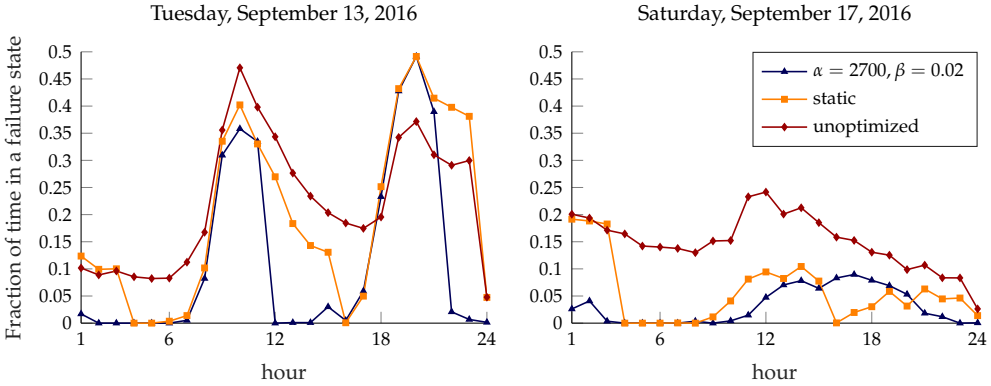

Figure 10. Average fraction of time in a failure state, plotted over the course of the day.

Effects of the demand estimation error. Figure 11 shows the effects of the estimation error in the demand data: we ran the simulations again, giving the system a perfect estimate of the demand for each hour, and used the performance as a benchmark. During the winter months (from November to February), the estimation error leads the system to take far more rebalancing trips (the distance covered on each trip was similar); a quick analysis of historical weather data suggests that the 2016–2017 winter was both considerably warmer and less snowy than previous years; this is the most plausible cause for the difference between the actual demand patterns and the estimates, which relied on data from previous years.

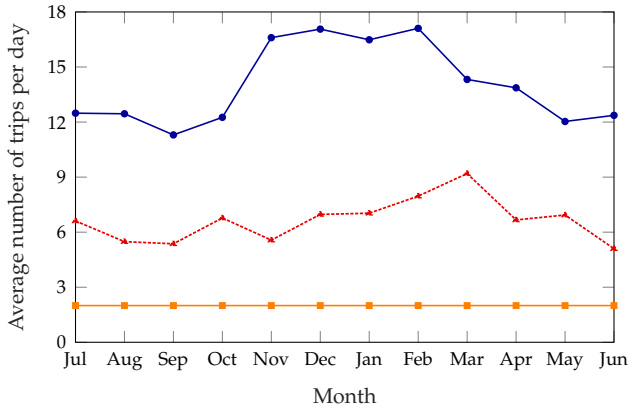

Figure 11. Average number of rebalancing trips per day with perfect and imperfect demand information, plotted over 12 months.

Figure 12 shows the effect that estimation errors have on the service quality. The system with perfect knowledge tends to rebalance far less because of its better knowledge of the demand; consequently, its failure rate is slightly higher (although still significantly lower than that of the static approach), but it needs half as many trips. We remind the reader that the trips taken by the dynamic systems, which do not rebalance the whole bike-sharing system at once, are far shorter than those in the static system; as such, the system with perfect knowledge gets a performance improvement over the static system while keeping the same rebalancing costs. A more aggressive setting of the α and β parameters would bring the failure rate closer to that of the imperfect information system, while presumably still limiting the rebalancing costs. An adaptive system that actively corrects the estimates based on a combination of current usage patterns, weather predictions and extraordinary events instead of relying on data from previous years would definitely improve the performance of the system. This is perfectly in line with the Smart City paradigm, which advocates a tighter integration of city services to exploit correlations in the data they generate. The remaining seasonal variations can either be accepted or smoothed out by adapting the parameters α and β to the seasonal behavior patterns of the system.

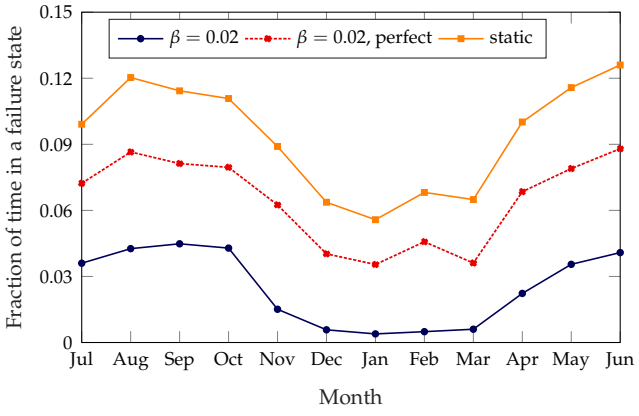

Figure 12. Average fraction of time in a failure state with perfect and imperfect demand information, plotted over 12 months.

Critical stations and consequences for system planning. Figure 13 shows a map of the optimized stations, color-coded based on their failure rate. Green stations have a lower than average failure rate, while the orange ones have an extremely high failure rate. The only red station is the one with the highest failure rate, almost four times the average. The map clearly shows two clusters of congested

stations in the Midtown and East Village neighborhoods; although the map was based on the results of the dynamic algorithm with $\beta = 0.04$, the same pattern can be seen across all the considered schemes.

Although CitiBike is mostly focusing on expanding throughout Upper Manhattan and to the other boroughs, increasing either the number of stations or the capacity of the existing ones in these two congested areas would greatly increase the quality of the offered service. The presence of stations that are often empty or completely full also contributes to further skew the data from the real demand, exacerbating the censoring problem we discussed in Section 4. A more thorough analysis of stations' criticality, including time-dependent patterns and correlations with public transport schedules, would be extremely useful to improve both the bike-sharing and mass transit systems.

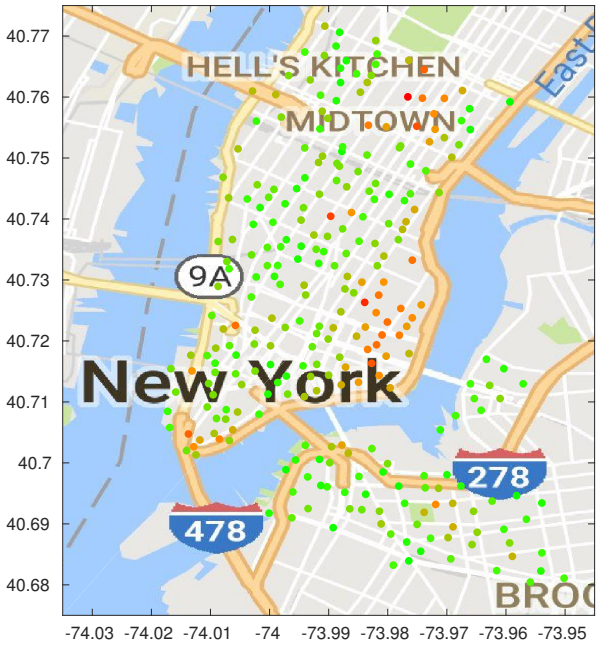

Figure 13. Map of the bike-sharing stations, color-coded from green (zero failure rate) to red (maximum failure rate).

6. Conclusions

In this paper, we proposed a general framework for the dynamic rebalancing of a bike-sharing system, in order to improve the availability of the service, i.e., to prevent stations from becoming either completely full or completely empty. A periodical scan of the network can identify overcrowded or almost empty stations, which will not be able to support the expected bike demand without human intervention to balance the system. Bike-sharing usage is estimated by analyzing the historical data, which makes it possible to identify traffic patterns over time. In particular, we extrapolated bike arrival and departure rates for each station that depend on the hour, the day of the week, and the month. This information is used to model each bike station through a BDP. In this way, we are able to estimate the time until the station becomes either empty or full with a high probability. Rebalancing is performed if the gain obtained by reallocating bikes exceeds the cost of moving the rebalancing truck. We identify the stations to visit and a greedy rebalancing path.

The results we obtained show that the model we defined is realistic and flexible, and that a dynamic approach to rebalancing can outperform static ones and allow system controllers to decide whether to prioritize maintenance costs or service quality. The dynamic approach can also reduce the rush hour problem during the weekdays, keeping the quality of the service more stable and avoiding high failure rates during times of peak demand. The data from the system can also be used for future planning, as critical stations and areas are easy to identify. Finally, the results show that some of the issues of rebalancing systems are due to an inaccurate estimation of the demand patterns; in order to

be more effective, the system should not just take into account historical data, but also current trends and weather data.

We would like to stress that, although we made some simplifications for the sake of mathematical tractability, the model we proposed is of general validity. In particular, the rebalancing optimization function of Section 3.2 can readily accommodate different costs and rewards by appropriately tuning its parameters. Furthermore, although we specifically provided a simple heuristic to compute the rebalancing path for a single vehicle, the rebalancing algorithm (1) considers a fleet with an arbitrary number of trucks. Similarly, we focused on a scenario where the cost between two stations is independent of the link direction and is only defined by the Euclidean distance between the nodes, but our model envisages a directed graph with different costs, so as to model the presence of one-way streets or other differences in the two directed links between two stations. The simplified framework we proposed already improves the bike-sharing system, and this encourages us to relax some assumptions and adopt a more accurate model to obtain further performance enhancements. We plan to consider multiple vehicles in charge of rebalancing bikes, since in big cities it is unrealistic and probably inefficient to use a single vehicle; the trip time and the limited capacity of rebalancing trucks should also be taken into account in the optimization model. The extension to a fleet of vehicles requires that stations be grouped into almost independent clusters; then, each vehicle will perform rebalancing only within its associated cluster. This would also allow us to consider the time it takes to perform rebalancing, resulting in a more accurate modeling of the rebalancing costs. Finally, it would be interesting to model the user satisfaction and use more complex behavior patterns, e.g., users taking a bike from a nearby station if the one they choose is empty.

Author Contributions: The mathematical model has developed by F. Chiariotti, C. Pielli, and A. Zanella, and validated by M. Zorzi. The performance analysis has been carried out by F. Chiariotti and, C. Pielli. The paper has been written by F. Chiariotti and C. Pielli, and revised by A. Zanella and M. Zorzi.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zanella, A.; Bui, N.; Castellani, A.; Vangelista, L.; Zorzi, M. Internet of Things for smart cities. *IEEE Int. Things J.* **2014**, *1*, 22–32.
2. Shaheen, S.; Guzman, S.; Zhang, H. Bikesharing in Europe, the Americas, and Asia: Past, present, and future. *Transp. Res. Record J. Transp. Res. Board* **2010**, *2143*, 159–167.
3. Midgley, P. The role of smart bike-sharing systems in urban mobility. *Journeys* **2009**, *2*, 23–31.
4. Kaspi, M.; Raviv, T.; Tzur, M. Detection of unusable bicycles in bike-sharing systems. *Omega* **2016**, *65*, 10–16.
5. Ahillen, M.; Mateo-Babiano, D.; Corcoran, J. Dynamics of bike sharing in Washington, DC and Brisbane, Australia: Implications for policy and planning. *Int. J. Sustain. Transp.* **2016**, *10*, 441–454.
6. Jiménez, P.; Nogal, M.; Caulfield, B.; Pilla, F. Perceptually important points of mobility patterns to characterise bike sharing systems: The Dublin case. *J. Transp. Geogr.* **2016**, *54*, 228–239.
7. Bao, J.; He, T.; Ruan, S.; Li, Y.; Zheng, Y. Planning Bike Lanes based on Sharing-Bikes' Trajectories. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; pp. 1377–1386.
8. O'Mahony, E.; Shmoys, D.B. Data Analysis and Optimization for (Citi) Bike Sharing. In Proceedings of the 29th AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; pp. 687–694.
9. Chiariotti, F.; Pielli, C.; Cenedese, A.; Zanella, A.; Zorzi, M. Bike Sharing as a Key Smart City Service. In Proceedings of the International Conference on Modern Circuits and System Technologies (MOCAST), Thessaloniki, Greece, 7–9 May 2018.
10. Laporte, G.; Meunier, F.; Calvo, R.W. Shared mobility systems. *Q. J. Oper. Res. (4OR)* **2015**, *13*, 341–360.
11. DeMaio, P. Bike-sharing: History, impacts, models of provision, and future. *J. Public Transp.* **2009**, *12*, 41–56.
12. Fishman, E.; Washington, S.; Haworth, N. Bike share's impact on car use: Evidence from the United States, Great Britain, and Australia. *Transp. Res. Part D* **2014**, *31*, 13–20.

13. Rojas-Rueda, D.; de Nazelle, A.; Tainio, M.; Nieuwenhuijsen, M.J. The health risks and benefits of cycling in urban environments compared with car use: Health impact assessment study. *Br. Med. J.* **2011**, *343*, d4521.
14. Cohen, A.; Shaheen, S. *Planning for Shared Mobility*; American Planning Association: Chicago, IL, USA, 2016.
15. Townsend, A.M. *Smart Cities: Big Data, Civic Hackers, and the Quest for a New Utopia*; WW Norton & Company: New York, NY, USA, 2013.
16. Benevolo, C.; Dameri, R.P.; D'Auria, B. Smart mobility in smart city. In *Empowering Organizations*; Springer: Berlin, Germany, 2016; pp. 13–28.
17. Chiariotti, F.; Condoluci, M.; Mahmoodi, T.; Zanella, A. SymbioCity: Smart cities for smarter networks. *Trans. Emerg. Telecommun. Technol.* **2018**, *29*, e3206.
18. Jäppinen, S.; Toivonen, T.; Salonen, M. Modelling the potential effect of shared bicycles on public transport travel times in Greater Helsinki: An open data approach. *Appl. Geogr.* **2013**, *43*, 13–24.
19. Martin, E.W.; Shaheen, S.A. Evaluating public transit modal shift dynamics in response to bikesharing: A tale of two US cities. *J. Transp. Geogr.* **2014**, *41*, 315–324.
20. Boshoff, D.G. Sustainable Transport Calls for Sustainable Infrastructure. In *QUAESTI-Virtual Multidisciplinary Conference*, EDIS-Publishing Institution of the University of Zilina: Zilina, Slovakia, 2016; Volume 1, pp. 153–159.
21. Faghih-Imani, A.; Eluru, N.; El-Geneidy, A.M.; Rabbat, M.; Haq, U. How land-use and urban form impact bicycle flows: Evidence from the bicycle-sharing system (BIXI) in Montreal. *J. Transp. Geogr.* **2014**, *41*, 306–314.
22. Buck, D.; Buehler, R. Bike lanes and other determinants of Capital bikeshare trips. In Proceedings of the 91st Transportation research board annual meeting, Washington, DC, USA, 22–26 January 2012; Volume 12, pp. 3539–3550.
23. Mateo-Babiano, I.; Bean, R.; Corcoran, J.; Pojani, D. How does our natural and built environment affect the use of bicycle sharing? *Transp. Res. Part A* **2016**, *94*, 295–307.
24. Kumar, A.; Teo, K.M. A Systems Perspective to Commuter Cycling in Urban Mobility. In *INCOSE International Symposium*; Wiley Online Library: Hoboken, NJ, USA, 2013; Volume 23, pp. 1282–1294.
25. Johnson, B.J.; White, S.S. Promoting sustainability through transportation infrastructure? Innovation and inertia in the Kansas City metropolitan area. *J. Urban Plan. Dev.* **2010**, *136*, 303–313.
26. Delucchi, M.; Kurani, K.S. How to have sustainable transportation without making people drive less or give up suburban living. *J. Urban Plan. Dev.* **2013**, *140*, 04014008.
27. Schuijbroek, J.; Hampshire, R.C.; Van Hoeve, W.J. Inventory rebalancing and vehicle routing in bike sharing systems. *Eur. J. Oper. Res.* **2017**, *257*, 992–1004.
28. Morse, P.M. *Queues, Inventories and Maintenance: The Analysis of Operational Systems with Variable Demand and Supply*; Courier Corporation: North Chelmsford, MA, USA, 2004.
29. Caggiani, L.; Ottomanelli, M. A dynamic simulation based model for optimal fleet repositioning in bike-sharing systems. *Procedia-Soc. Behav. Sci.* **2013**, *87*, 203–210.
30. Vogel, P.; Greiser, T.; Mattfeld, D.C. Understanding bike-sharing systems using data mining: Exploring activity patterns. *Procedia-Soc. Behav. Sci.* **2011**, *20*, 514–523.
31. Faghih-Imani, A.; Eluru, N. Incorporating the impact of spatio-temporal interactions on bicycle sharing system demand: A case study of New York CitiBike system. *J. Transp. Geogr.* **2016**, *54*, 218–227.
32. Fricker, C.; Gast, N. Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity. *EURO J. Transp. Logist.* **2016**, *5*, 261–291.
33. Waserhole, A.; Jost, V. Pricing in vehicle sharing systems: Optimization in queuing networks with product forms. *EURO J. Transp. Logist.* **2016**, *5*, 293–320.
34. Chemla, D.; Meunier, F.; Calvo, R.W. Bike sharing systems: Solving the static rebalancing problem. *Discret. Optim.* **2013**, *10*, 120–146.
35. Ho, S.C.; Szeto, W. Solving a static repositioning problem in bike-sharing systems using iterated tabu search. *Transp. Res. Part E* **2014**, *69*, 180–198.
36. Raviv, T.; Tzur, M.; Forma, I.A. Static repositioning in a bike-sharing system: Models and solution approaches. *EURO J. Transp. Logist.* **2013**, *2*, 187–229.
37. Di Gaspero, L.; Rendl, A.; Urli, T. Balancing bike sharing systems with constraint programming. *Constraints* **2016**, *21*, 318–348.

38. Forma, I.A.; Raviv, T.; Tzur, M. A 3-step math heuristic for the static repositioning problem in bike-sharing systems. *Transp. Res. Part B* **2015**, *71*, 230–247.
39. Dell'Amico, M.; Iori, M.; Novellani, S.; Stützle, T. A destroy and repair algorithm for the bike sharing rebalancing problem. *Comput. Oper. Res.* **2016**, *71*, 149–162.
40. Dell'Amico, M.; Hadjicostantinou, E.; Iori, M.; Novellani, S. The bike sharing rebalancing problem: Mathematical formulations and benchmark instances. *Omega* **2014**, *45*, 7–19.
41. Contardo, C.; Morency, C.; Rousseau, L.M. *Balancing a Dynamic Public Bike-Sharing System*; CIRRELT: Montreal, QC, Canada, 2012; Volume 4.
42. Dantzig, G.B.; Wolfe, P. Decomposition principle for linear programs. *Oper. Res.* **1960**, *8*, 101–111.
43. Benders, J.F. Partitioning procedures for solving mixed-variables programming problems. *Numer. Math.* **1962**, *4*, 238–252.
44. Erdoğan, G.; Battarra, M.; Calvo, R.W. An exact algorithm for the static rebalancing problem arising in bicycle sharing systems. *Eur. J. Oper. Res.* **2015**, *245*, 667–679.
45. Kloimüllner, C.; Papazek, P.; Hu, B.; Raidl, G.R. Balancing bicycle sharing systems: An approach for the dynamic case. In Proceedings of the European Conference on Evolutionary Computation in Combinatorial Optimization, Granada, Spain, 23–25 April 2014; pp. 73–84.
46. Lu, C.C. Robust multi-period fleet allocation models for bike-sharing systems. *Netw. Spat. Econ.* **2016**, *16*, 61–82.
47. Brinkmann, J.; Ulmer, M.W.; Mattfeld, D.C. Short-term strategies for stochastic inventory routing in bike sharing systems. *Transp. Res. Procedia* **2015**, *10*, 364–373.
48. Espegren, H.M.; Kristianslund, J.; Andersson, H.; Fagerholt, K. The Static Bicycle Repositioning Problem-Literature Survey and New Formulation. In Proceedings of the International Conference on Computational Logistics, Lisbon, Portugal, 7–9 September 2016; pp. 337–351.
49. Fischer, W.; Meier-Hellstern, K. The Markov-modulated Poisson process (MMPP) cookbook. *Perform. Eval.* **1993**, *18*, 149–171.
50. Krinik, A.; Mortensen, C. Transient probability functions of finite birth-death processes with catastrophes. *J. Stat. Plan. Inference* **2007**, *137*, 1530–1543.
51. Crawford, F.W.; Suchard, M.A. Transition probabilities for general birth-death processes with applications in ecology, genetics, and evolution. *J. Math. Biol.* **2012**, *65*, 553–580.
52. Skellam, J.G. The frequency distribution of the difference between two Poisson variates belonging to different populations. *J. R. Stat. Soc. Ser. A* **1946**, *109*, 296.
53. Abramowitz, M.; Stegun, I.A. *Handbook of Mathematical Functions*; Applied Mathematics Series; U.S. Government Printing Office: Washington, DC, USA, 1966; Volume 55.
54. Laporte, G. The vehicle routing problem: An overview of exact and approximate algorithms. *Eur. J. Oper. Res.* **1992**, *59*, 345–358.
55. Bellmore, M.; Nemhauser, G.L. The traveling salesman problem: A survey. *Oper. Res.* **1968**, *16*, 538–558.