Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Minimizing Multimodular Functions and Allocating Capacity in Bike-Sharing Systems

Daniel Freund, Shane G. Henderson, David B. Shmoys

To cite this article:
Daniel Freund, Shane G. Henderson, David B. Shmoys (2022) Minimizing Multimodular Functions and Allocating Capacity in Bike-Sharing Systems. Operations Research 70(5):2715-2731. https://doi.org/10.1287/opre.2022.2320

https://pubsonline.informs.org/journal/opre

OPERATIONS RESEARCH
Vol. 70, No. 5, September–October 2022, pp. 2715–2731
ISSN 0030-364X (print), ISSN 1526-5463 (online)

Contextual Areas

Minimizing Multimodular Functions and Allocating Capacity in Bike-Sharing Systems

Daniel Freund,[a,*] Shane G. Henderson,[b] David B. Shmoys[b]

[a] Sloan School of Management, Massachusetts Institute of Technology, Cambridge, Massachusetts 02142; [b] School of Operations Research and Information Engineering, Cornell University, Ithaca, New York 14853
*Corresponding author

Contact: dfreund@mit.edu, https://orcid.org/0000-0001-8039-9805 (DF); sgh9@cornell.edu, https://orcid.org/0000-0003-1004-4034 (SGH); dbs10@cornell.edu, https://orcid.org/0000-0003-3882-901X (DBS)

Received: January 9, 2021
Revised: November 21, 2021; March 14, 2022
Accepted: May 2, 2022
Published Online in Articles in Advance: June 23, 2022

Area of Review: Transportation

https://doi.org/10.1287/opre.2022.2320

Abstract. The growing popularity of bike-sharing systems around the world has motivated recent attention to models and algorithms for their effective operation. Most of this literature focuses on their daily operation for managing asymmetric demand. In this work, we consider the more strategic question of how to (re)allocate dock-capacity in such systems. We develop mathematical formulations for variations of this problem (either for service performance over the course of one day or for a long-run-average) and exhibit discrete convex properties in associated optimization problems. This allows us to design a polynomial-time allocation algorithm to compute an optimal solution for this problem, which can also handle practically motivated constraints, such as a limit on the number of docks moved in the system. We apply our algorithm to data sets from Boston, New York City, and Chicago to investigate how different dock allocations can yield better service in these systems. Recommendations based on our analysis have led to changes in the system design in Chicago and New York City. Beyond optimizing for improved quality of service through better allocations, our results also provide a metric to compare the impact of strategically reallocating docks and the daily rebalancing of bikes.

Funding: This work was supported by the National Science Foundation through the Division of Civil, Mechanical and Manufacturing Innovation [Grants 1537394 and 2035086]; the Division of Computer and Network Systems [Grant 195206]; the Division of Mathematical Sciences [Grant 1839346]; and the Division of Computing and Communication Foundations [Grants 1522054, 1526067 and 1740822]. It was also supported by the Army Research Office [Grant W911NF-17-1-0094].
Supplemental Material: The online appendix is available at https://doi.org/10.1287/opre.2022.2320.

Keywords: discrete convexity • transportation • sharing economy

1. Introduction

As bike-sharing systems become an integral part of the urban landscape, novel lines of research seek to model and optimize their operations. In many systems, such as New York City's Citi Bike, users can rent and return bikes at any station within the city. This flexibility makes the system attractive for commuters and tourists alike. From an operational point of view, however, this flexibility leads to imbalances when demand is asymmetric, as is commonly the case. The main contributions of this paper are to identify key questions in the *design* of operationally efficient bike-sharing systems, to develop a polynomial-time algorithm for the associated discrete optimization problems, to apply this algorithm on real usage data, and to investigate the effect this optimization has in practice.

The largest bike-sharing systems in the United States are dock based, meaning that they consist of stations, spread across a city, each of which has a number of docks in which bikes can be locked. If a bike is present in a dock, users can rent it and return it at any other station with an open dock. However, system imbalance often causes some stations to have only empty (or *open*) docks and others to have only full docks (i.e., ones filled with bikes). In the former case, users need to find alternate modes of transportation, whereas in the latter they might not be able to end their trip at the intended destination. In many bike-sharing systems, this has been found to be a leading cause of customer dissatisfaction (Capital Bikeshare 2014).

To meet demand in the face of asymmetric traffic, bike-sharing system operators seek to *rebalance* the system by moving bikes from locations with too few open docks to locations with too few bikes. To facilitate these operations, a burst of recent research has investigated models and algorithms to increase their efficiency and increase customer satisfaction. Although similar in spirit to some of the literature on rebalancing, in this work we use a different control to increase

customer satisfaction. Specifically, we answer the question *how should bike-sharing systems allocate dock capacity to stations within the system to minimize the number of dissatisfied customers*?

A superficial analysis of usage data reveals that there may be potential in reallocating capacity: some stations have spare capacity that users never or rarely use (see Figure EC.1 in Online Appendix EC.1), whereas other stations have all of their capacity used on most days. We give a more theoretically grounded answer to this question by developing two optimization models, both based on the underlying metric that system performance is captured by the expected number of customers that do not receive service. In the first model, we focus on planning one day, say 6:00 a.m. to midnight, where for each station we determine its allocation of bikes and docks; this framework assumes that there is sufficient rebalancing capacity overnight to restore the desired bike allocation by 6:00 a.m. the next morning. As in practice this turns out to be quite difficult, the second model considers a setup induced by a long-run average that assumes that no rebalancing happens at all; in a sense, this exhibits the opposite regime. The theory developed in this paper enabled extensive computational experiments on real data sets; through these we found that there are dock allocations that simultaneously perform well with respect to both models, yielding improvements to both (in comparison with the current allocation) of up to 20%. These results were leveraged by system operators in Chicago and New York City and led to 100 (200) docks being moved in New York City (Chicago). Convinced by the impact analysis in these cities, operators of other major US bike-sharing systems, including Blue Bikes in Boston and Capital Bikeshare in Washington, DC, have run our analysis on their data to capture the potential of reallocated dock capacity as well.

1.1. Our Contribution

Raviv and Kolka (2013) defined a *user dissatisfaction function* (UDF) that measures the expected number of out-of-stock events at an individual bike-sharing station. To do so, they define a stochastic process on the possible number of bikes (between zero and the capacity of the station). The stochastic process observes attempted rentals and returns of bikes over time; this process is assumed to be exogenously given at each station and independent of our decisions/the availability of bikes and docks in other stations. Each arrival triggers a change in the state, either decreasing (rental) or increasing (return) the number of available bikes by one. When the number of bikes is zero and a rental is attempted, or when it equals the station capacity and a return is attempted, a customer experiences an out-of-stock event. Various follow-up papers (Parikh and Ukkusuri 2015, O'Mahony 2015, Schuijbroek et al. 2017) have suggested different ways to compute the expected number of out-of-stock events $c_i(d_i, b_i)$ that occur over the course of one day at each station i for a given allocation of b_i bikes and d_i empty docks (i.e., $d_i + b_i$ docks in total) at station i at the start of the day.

We use the same UDFs to model the question of how to allocate dock capacity within the system. Given $c_i(\cdot,\cdot)\ \forall i$, our goal is to find an allocation of bikes and docks in the system that minimizes the total expected number of out-of-stock events within a system of n stations, that is, $\sum_{i=1}^{n} c_i(d_i, b_i)$. Because the number of bikes and docks is limited, we need to accommodate a *budget constraint* B on the number of bikes in the system and another on the number of docks $D + B$ in the system. Other constraints are often important, such as lower and upper bounds on the capacity for a particular station; furthermore, through our collaboration with Citi Bike in NYC it also became apparent that *operational constraints* limit the number of docks moved from the current system configuration. Thus, we aim to minimize the objective among solutions that require at most some number of docks moved. Notice that D and B could either denote the inventory that is currently present in the system (in which case the question is how to reallocate it) or include new inventory (in which case the question is how to augment the current system design).

After formally defining this model and discussing its underlying assumptions in Section 2, we design in Section 3 a discrete gradient-descent algorithm that provably solves the minimization problem with $O(n + B + D)$ oracle calls to evaluate cost functions and an (in practice, vastly dominated) overhead of $O((n + B + D)\log(n))$ elementary list operations. In Section 4, we show that scaling techniques, together with a subtle extension of the analysis of the gradient-descent algorithm, improve the running-time to $O(n \log(B + D))$ oracle calls and $O(\log(B + D)(n \log(n)))$ elementary list operations for the setting without operational constraints; in Online Appendix EC.5 we include the proofs thereof and explanations of how operational constraints can be handled when aiming for running-time logarithmic in $B + D$. In Online Appendix EC.6, we include a computational study to complement this theoretical analysis of the efficiency of our algorithms.

The primary motivation of this analysis is to investigate whether the number of out-of-stock events in bike-sharing systems can be significantly reduced by a data-driven approach. In Section 5, we apply the algorithms to data sets from Boston, New York City, and Chicago to evaluate the impact on out-of-stock events. One shortcoming of that optimization problem is its assumption that we can perfectly restore the system to the desired initial bike allocation overnight. Through our collaboration with the operators of systems across the country, it has become evident that current

rebalancing efforts overnight are vastly insufficient to realize such an optimal (or even near-optimal) allocation of bikes for the current allocation of docks. Thus, we consider in Section 5.1 the opposite regime, in which no rebalancing occurs at all. To model this, we define an extension of the cost function under a long-run average regime. In this regime, the assumed allocation of bikes at each station is a function of only the number of docks and the estimated demand at that station. Interestingly, our empirical results reveal that operators of bike-sharing systems can *have their cake and eat it too*: optimizing dock allocations for one of the objectives (optimally rebalanced or long-run average) yields most of the obtainable improvement for the other.

Based on our recommendations the operators of Citi Bike in New York City agreed with the city's Department of Transportation to move 34 docks between six stations as part of a pilot program. We use these moves to evaluate the impact of reallocated capacity. Specifically, in Section 6, we prove that observing rentals and returns after capacity has been added provides a natural way to estimate the reduction in out-of-stock events (because of dock capacity added) that can be computed in a very simple manner. We apply this approach to the stations that were part of the pilot to derive estimates for the realized reduction in the number of stockouts at those stations.

1.2. Related Work

A recent line of work, including variations by Raviv et al. (2013), Forma et al. (2015), Kaspi et al. (2017), Ho and Szeto (2014), and Freund et al. (2020), considered static rebalancing problems, in which a capacitated truck (or a fleet of trucks) is routed over a limited time horizon. The truck may pick up and drop off bikes at each station to minimize the expected number of out-of-stock events that occur after the completion of the route. These are evaluated by the same objective function of Raviv and Kolka (2013) that we consider as well.

In contrast to this line of work, O'Mahony (2015) addressed the question of allocating both docks and bikes; he uses the UDFs (defined over a single interval with constant rental and return rates) to design a mixed integer program over the possible allocations of bikes and docks. Our work extends on this by providing a fast algorithm for generalizations of that same problem and extensions thereof. The optimal allocation of bikes has also been studied by Jian and Henderson (2015), Datner et al. (2019), and Jian et al. (2016), with the latter also considering the allocation of docks (in fact, the idea behind the algorithm considered by Jian et al. (2016) is based on an early draft of this paper). They each develop frameworks based on ideas from simulation optimization; whereas they also treat demand for bikes as being exogenous, their framework captures the downstream effects of changes in supply upstream. Jian et al. (2016) found that these effects are mostly captured by decensoring piecewise-constant demand estimates (see Section 2.1).

Orthogonal approaches to the question of where to allocate docks have been taken by Kabra et al. (2020) and Wang et al. (2016). The former considers demand as endogenous and aims to identify the station density that maximizes sales, whereas we consider demand and station locations as exogenously given and aim to allocate docks and bikes to maximize the amount of demand that is being met. The latter aims to use techniques from retail location theory to find locations for stations to be added to an existing system.

Further related literature includes a line of work on rebalancing triggered by Chemla et al. (2013). Subsequent papers, for example, by Nair et al. (2013), Dell'Amico et al. (2014), Erdoğan et al. (2014), Erdoğan et al. (2015), Bruck et al. (2019), and Li et al. (2020), solve variants of a routing problem with fixed numbers of bikes that need to be picked up/dropped off at each station; de Chardon et al. (2016) extensively survey these papers. Before rebalancing bike-sharing systems became an object of academic study, the closely related traveling salesman problems with pickup and delivery had already been studied outside the bike-sharing domain since Hernández-Pérez and Salazar-González (2004). Other approaches to rebalancing include, for example, the papers of Liu et al. (2016), Ghosh et al. (2016), Rainer-Harbach et al. (2013), Shu et al. (2013), or more recently Brinkmann et al. (2019). We refer the readers to the surveys of Laporte et al. (2018), Freund et al. (2019), and Shui and Szeto (2020) for a wider overview of the rebalancing literature. Although all of these fall into the wide range of recent work on the operation of bike-sharing systems, they differ from our work in the controls and methodologies they use.

Finally, a great deal of work has been conducted in the context of predicting demand. In this work, we assume that the predicted demand is given, for example, using the methods of O'Mahony and Shmoys (2015) or Singhvi et al. (2015). Further methods to predict demand have been suggested by Li et al. (2015), Chen et al. (2016), and Zhang et al. (2016), among others. Our results can be combined with any approach that predicts demand at each station independently of all others.

1.2.1. Relation to Discrete Convexity.

Our algorithms and analyses crucially exploit the property that the UDFs $c_i(\cdot,\cdot)$ at each station are multimodular (see Definition 1). This provides an interesting connection to the literature on discrete convex analysis. Prior works connecting inventory management to discrete convexity include Lu and Song (2005), Zipkin (2008), and Li and Yu (2014), among others; we refer the reader to a recent survey by Chen and Li (2021) for an extensive overview. In concurrent work by Kaspi et al. (2017), it was shown that the number of out-of-stock events

$F(b, U-d-b)$ at a bike-sharing station with fixed capacity U, b bikes, and $U-d-b$ unusable bikes is $M^\natural$ (read M *natural*) convex in b and $U-d-b$; functions with such discrete convex properties, in particular M-convex and $M^\natural$ functions, were, respectively, introduced by Murota (1996; 1998) and Murota and Shioura (1999) (see the book by Murota (2003) for a complete overview of early results in discrete convexity). Unusable bikes effectively reduce the capacity at the station, because they are assumed to remain in the station over the entire time horizon. A station with capacity U, b bikes, and $U-b-d$ unusable bikes, must then have d empty docks; hence, $c(d,b) = F(b, U-d-b)$ for $d+b \le U$, which parallels our result that $c(\cdot,\cdot)$ is multimodular. Although this would suggest that algorithms to minimize $M^\natural$-convex functions could solve our problem optimally, one can show that $M^\natural$-convexity is not preserved, even in the version with only budget constraints: we provide in Online Appendix EC.8.1 an example that shows both that an $M^\natural$-convex function restricted to an $M^\natural$-convex set is not $M^\natural$-convex and that Murota's algorithm for $M^\natural$-convex function minimization can be suboptimal in our setting. In fact, when including the operational constraints even discrete midpoint convexity, a strict generalization of multimodularity studied for example by Fujishige and Murota (2000) and Moriguchi et al. (2020), which is in turn much weaker than $M^\natural$ convexity, breaks down. We provide an example for this in Online Appendix EC.8.2. Surprisingly, we are nevertheless able to design fast algorithms; these exploit not only the multimodularity of each individual c_i, but also the separability of the objective function, with respect to (w.r.t.) the stations, that is, the fact that each c_i is only a function of d_i and b_i. This not only extends ideas from the realm of unconstrained discrete convex minimization to the constrained setting, but also yields algorithms that (for our special case) have significantly faster running times than those that would usually arise in the context of multimodular function minimization. Since the conference version of this paper appeared, Shioura (2021) has taken our work as motivation to study M-convex function minimization under L1-distance constraints, a strict generalization of our objective. Finally, Shioura (private communication) pointed out an error in a preliminary version of this paper, and so, although all of the main elements of our proof of correctness of the discrete gradient-descent algorithm can be found in our preliminary version (Freund et al. 2016, 2017), the presentation here differs from that given earlier.

2. Model

The fundamental primitives of our model of a bike-sharing system are customers, bikes, docks, and stations. Here, we formally define these primitives and the optimization problem that is based on them.

2.1. Model Primitives

A bike-sharing system consists of n *stations*. Each station i is characterized by an exogenously given *demand profile* p_i, where p_i is a distribution over arrival sequences of *customers* at i over the course of a time horizon (e.g., 6:00 a.m. to 12:00 a.m.). Such arrival sequences are denoted $X = (X_1, X_2, \ldots, X_s) \in \{\pm 1\}^s$, where $X_t = -1$ corresponds to a customer arriving to rent a bike, and $X_t = 1$ corresponds to a customer arriving to return a bike. The ability of a customer arriving at a station to rent, respectively, return, a bike is dependent on the number of *bikes*, respectively, *empty docks*, available at the station at the time of arrival: if no bikes, respectively, empty docks, are available at the time of the customer's arrival, the customer is unable to rent, respectively, return, a bike and disappears with an out-of-stock event. If instead a customer arrives at a station to rent a bike and a bike is available, then the number of bikes at the station decreases by one, and the number of empty docks at the station increases by one. Similarly, if a customer arrives to return a bike, and an empty dock is available at that time, then the number of bikes at the station increases by one and the number of empty docks decreases by one. Throughout the time horizon, the total number of docks (empty and full) at station i remains the same. This is because the number of empty docks increases by one if and only if the number of full docks decreases by one (and vice versa). The respective number of bikes and empty docks at the station at the time of arrival of X_t is based only on (i) the initial allocation of bikes and empty docks at the beginning of the time horizon and (ii) the arrival sequence of customers up to X_t, which we denote $X(t-1) = (X_1, \ldots, X_{t-1})$.

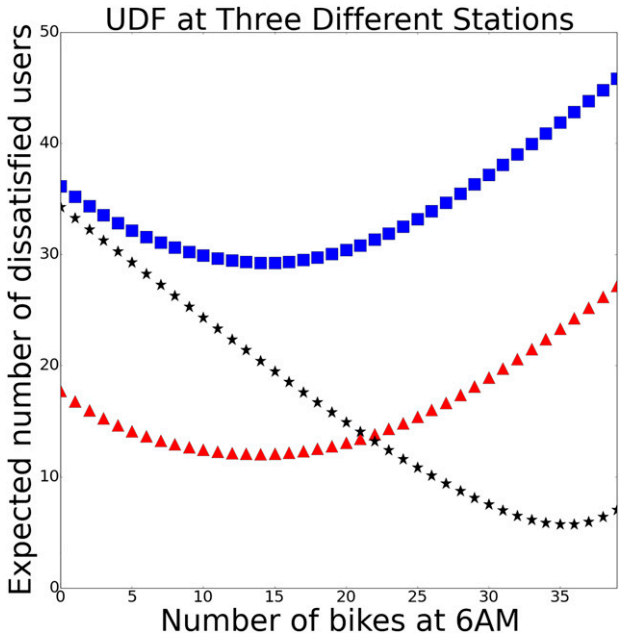
2.2. Initial Allocations of Bikes and Docks

The decision variables in our optimization model are the initial number of empty docks and bikes allocated to each station i at the beginning of the time horizon. We denote the initial number of empty docks at station i by d_i, and the initial number of bikes (full docks) by b_i; combining these two we find that a station i has an allocated capacity of $d_i + b_i$ docks in total.

2.3. User Dissatisfaction Function

The UDF $c^X(d,b)$ maps the initial number of empty docks and bikes to the number of customers, among the sequence $X = (X_1, \ldots, X_s)$, that experience out-of-stock events (Figure 1). Then, the UDF at station i is given by $c_i(d,b) = \mathbb{E}_{X\sim p_i}[c_i^X(d,b)]$. In an effort to keep notation concise in the main body of the text, we move a formal recursive definition of $c^X(\cdot,\cdot)$ to Online Appendix EC.3.1. UDFs are sometimes used with different weights for stock-outs depending on whether

Figure 1. (Color online) UDFs as a Function of Bikes for Three Different Stations with Capacity 39



they occur at empty or full stations; although we focus throughout on the unweighted case, in which $c^X(d,b)$ is just a count of the stockouts, our results extend to the weighted case (see Online Appendix EC.3.1).

Definition 1. A function $f:\mathbb{Z}^2 \to \mathbb{R}\cup\{\infty\}$ with

$$f(d+1,b+1)-f(d+1,b) \geq f(d,b+1)-f(d,b); \quad (1)$$
$$f(d-1,b+1)-f(d-1,b) \geq f(d,b)-f(d,b-1); \quad (2)$$
$$f(d+1,b-1)-f(d,b-1) \geq f(d,b)-f(d-1,b); \quad (3)$$

for all d, b is called *multimodular* (Hajek 1985, Altman et al. 2000, Murota 2003). For future reference, we also define the following implied additional inequalities ((6) and (1) are equivalent, (1) and (2) imply (5), and (3) and (6) imply (4)):

$$f(d+2,b)-f(d+1,b) \geq f(d+1,b)-f(d,b); \quad (4)$$
$$f(d,b+2)-f(d,b+1) \geq f(d,b+1)-f(d,b); \quad (5)$$
$$f(d+1,b+1)-f(d,b+1) \geq f(d+1,b)-f(d,b). \quad (6)$$

For f evaluating to infinite values, we assume the conventions that $\infty-\infty=\infty$, $\infty \geq x$, and $x \geq -\infty$ for every $x \in \mathbb{R}\cup\{-\infty,\infty\}$. We refer the reader to Figure EC.2 in Online Appendix EC.1 for a visual illustration of the diminishing return properties described by these inequalities.

2.4. System-Wide Objective

Our goal is to minimize the combined number of out-of-stock events across the system, that is, $\sum_i c_i(d_i,b_i)$. Writing $\vec{d}$ and $\vec{b}$ for the vectors that contain d_i and b_i in their ith position, we denote this sum by $c(\vec{d},\vec{b})$. We minimize $c(\vec{d},\vec{b})$ subject to four kinds of constraints that we introduce now.

2.5. Constraints

Our optimization problem involves two kinds of budget constraints. The first is on the total number of bikes allocated, that is, $\sum_i b_i$, bounded by B. The second is on the total number of docks allocated in the system, that is, $\sum_i d_i + b_i$, which is bounded by $D+B$. In addition, we have an *operational constraint* that bounds the number of docks that can be reallocated within the system. To formally state this constraint it is useful to define the following.

Definition 2. Consider two allocations $(\vec{d},\vec{b})$ and $(\vec{d}',\vec{b}')$ with $|\vec{d}+\vec{b}|_1 = |\vec{d}'+\vec{b}'|_1$, that is, the same number of docks allocated in total. The number of docks that need to be reallocated to get from $(\vec{d},\vec{b})$ to $(\vec{d}',\vec{b}')$ (ignoring the allocation of bikes) is $|\vec{d}'+\vec{b}'-\vec{d}-\vec{b}|_1/2$. For allocations $(\vec{d},\vec{b})$ and $(\vec{d}',\vec{b}')$ with the same total number of docks we define this as the *dock-move distance* between them.

Given an initial allocation $(\vec{\bar{d}},\vec{\bar{b}})$, for which we assume $\sum_i \bar{d}_i + \bar{b}_i \leq D+B$, the operational constraint is then of the form $|\vec{d}+\vec{b}-\vec{\bar{d}}-\vec{\bar{b}}|_1/2 \leq z$ for some z (where this constraint is well-defined even when $|\vec{d}+\vec{b}|_1 \neq |\vec{\bar{d}}+\vec{\bar{b}}|_1$). Finally, we have physical constraints that give lower and upper bounds on the number of docks allocated to each station i, where we assume that $0 \leq l_i \leq \bar{d}_i + \bar{b}_i \leq u_i$ for every i. The resulting optimization problem can be written as follows:

$$\begin{aligned}
\underset{(\vec{d},\vec{b})\in\mathbb{N}^n\times\mathbb{N}^n}{\texttt{minimize}}\ & c(\vec{d},\vec{b}) \\
\texttt{s.t.}\ & \sum_i d_i + b_i && \leq D+B, \\
& \sum_i b_i && \leq B, \\
& |\vec{d}+\vec{b}-\vec{\bar{d}}-\vec{\bar{b}}|_1/2 \leq z, \\
& l_i \leq d_i + b_i \leq u_i && \forall i. \quad \text{(P1)}
\end{aligned}$$

Following standard convention, we define $c_i(d_i,b_i) = \infty$ for $d_i + b_i > u_i$ and $d_i + b_i < l_i$, which allows us to drop the last row of inequalities in (P1). We also set $c_i(d,b)=\infty$ for $d<0$ or $b<0$. With these changes, $c_i(\cdot,\cdot)$ fulfills the inequalities in Definition 1.

Lemma 1. *The function $c_i(\cdot,\cdot)$ is multimodular.*

The proof of the lemma is based on a coupling argument and appears in Online Appendix EC.3.1. In addition, we may add a $(n+1)$ st dummy ("depot") station $\mathcal{D}$ to guarantee that the first two constraints hold with equality in an optimal solution. Thus, we can transform our original optimization problem into

one of the following form, which is our focus throughout the main body of the text:

$$\begin{aligned} &\underset{(\vec{d},\vec{b})\in\mathbb{Z}^{n+1}\times\mathbb{Z}^{n+1}}{\texttt{minimize}} \; c(\vec{d},\vec{b}) \\ &\texttt{s.t.} \sum_i d_i + b_i = D + B, \\ &\sum_i b_i = B, \\ &|\vec{d}+\vec{b}-\vec{d}-\vec{b}|_1/2 \le z, \text{ where } |\vec{d}+\vec{b}|_1 = D+B. \end{aligned} \tag{P2}$$

Specifically, the reduction from Problem (P1) to (P2) is based on the following: let $\bar{D} = D + B - \sum_i \bar{d}_i + \bar{b}_i$, that is, the number of docks that are not in the current allocation but can be added, $\bar{z} = z + \lfloor \frac{\bar{D}}{2} \rfloor$, and define station $\mathcal{D}$ with $l_{\mathcal{D}} = B, u_{\mathcal{D}} = 2B + D, \bar{d}_{\mathcal{D}} + \bar{b}_{\mathcal{D}} = B + \bar{D}$, and $c_{\mathcal{D}}(d,b) = d + b - B$ when $l_{\mathcal{D}} \le d + b \le u_{\mathcal{D}}$—observe that $c_{\mathcal{D}}$ fulfills the requirements of Definition 1. Furthermore, $c_{\mathcal{D}}$ has the property that its objective is increasing in the number of docks allocated to it, whereas the objective at all other stations is nonincreasing in the number of docks allocated. In the proof of the following proposition, this will be used to ensure that optimal solutions to (P2) fulfill $d_{\mathcal{D}} + b_{\mathcal{D}} = l_{\mathcal{D}}$. It is worth noting that our algorithm/analysis for (P2) does not rely on the c_i being nonincreasing in the number of docks allocated, that is, $c_{\mathcal{D}}$ being decreasing will not affect our analysis.

Proposition 1. *If* $(\vec{d},\vec{b})$ *is optimal for* (P2) *with stations* $[n]\cup\{\mathcal{D}\}$, *bike budget* B, *dock budget* $D+2B$, *and operational constraint* $\bar{z}$, *then restricting* $(\vec{d},\vec{b})$ *to* $[n]$ *is optimal for* (P1).

The proof of the proposition is in Online Appendix EC.3.2. There, we also show how to optimally solve an optimization problem that involves an additional tradeoff between the size of D and the size of z, that is, between the inventory cost of additional docks and the operational cost of reallocating docks. We are now ready to discuss the assumptions in our model before analyzing in Section 3 an algorithm to optimally solve (P2).

2.6. Discussion of Assumptions

Before describing and analyzing the algorithm we use to solve the optimization problem in Section 3, we discuss the assumptions and advantages that come along with them.

2.6.1. Seasonality and Frequency of Reallocations. In contrast to bike rebalancing, the reallocation of docks is a strategic question that involves docks being moved at most annually. As such, a concern is that the recommendations for a particular month might not yield improvement for other times of the year. One way to deal with this is to explicitly distinguish, in the demand profiles, between different seasons, that is, have k different distributions for k different types of days and then consider the expectation over these as the objective. Although the user dissatisfaction functions accommodate that approach, we find on real data (see Section 5.3) that this is not actually necessary: the reallocations that yield greatest impact for the summer months of one year also perform very well for the winter months of another. This even held true in New York City, where the system significantly expanded year-over-year: despite the number of stations in the system more than doubling and total ridership increasing by around 70% from 2015 to 2017, we find that the estimated improvement due to reallocated docks is surprisingly stable across these different months. In part this is because of the fact that the relative demand patterns at different stations strongly correlate between seasons, that is, the demand of each station in each interval in one month is well approximated by a constant multiple of demand in another. For example, the vectors of half-hourly demand estimates (either rentals or returns) for each New York City station in June and December 2018 have a Pearson correlation coefficient greater than 0.85. Although this does not formally imply that the improvement in the UDFs would correlate, it gives some explanation for why it might.

2.6.2. Cost of Reallocation. Rather than explicitly building in a cost for reallocations in our formulation, we instead bound the number of docks that are moved. This is mostly motivated by our industry partner's practical considerations: the cost of physically reallocating capacity from one location to another is negligible when compared with the administrative effort, a negotiation with city officials and other stakeholders, needed to reallocate capacity. As part of these negotiations the operator will request that a limited number of docks be moved from the current system configuration. While we solve the problem assuming that this number is a known constant, in practice it is part of the negotiations. To hold these negotiations it is of utmost importance for the operator to know the value of reallocating a given (fixed) number of docks; thus, our results were used to help prepare the operator for these negotiations, in particular, to answer for different values of z the crucial questions of *how much benefit would the system derive from moving* z *docks* and *which docks would be among those* z. This then also implies that tactical questions of how to carry out the reallocations is of minor importance in practice. Furthermore, the cost of reallocating docks can be compared with the cost of rebalancing bikes: while the (one-off) reallocation of a single dock is about an order of magnitude more expensive than that of a single bike, the reallocated dock has daily impact on improved service levels (in contrast to

the one-off impact of a rebalanced bike). Thus, the cost quickly amortizes, and Citi Bike estimates this will happen in as little as two weeks. Finally, the cost to acquire new docks is orders of magnitudes higher than all of the aforementioned costs, leading us to focus only on reallocated capacity in our analysis; nevertheless, we show in Online Appendix EC.3.3 that the algorithm also extends to capture the tradeoff between installing newly bought and reallocating existing docks.

2.6.3. Bike Rebalancing. The user dissatisfaction functions assume that no rebalancing takes place over the course of the planning horizon. System data indicate that this is close to reality at most stations; for example, in New York City, more than 60% of all rebalancing is concentrated at just 28 of 762 stations, which justifies the assumption for most stations. Now, consider the remaining few stations, at which almost all rebalancing is concentrated: perhaps unsurprisingly, we find that none of these stations are identified by the optimization as having their capacity reduced. In general, rebalancing can always limit the number of dissatisfied users to zero: consider a station with two docks that is stocked with one bike; as long as rebalancing adds/removes a bike after each pickup/dropoff, users will not experience stockouts. Thus, reducing the number of dissatisfied customers at a station with no rebalancing is somewhat analogous to reducing rebalancing needs at a station with rebalancing. For illustrative purposes, consider the following deterministic example: a station with 60 docks observes demand for 120 rentals in the morning and demand for 120 dropoffs in the afternoon. Suppose the station is full with bikes in the morning. Without rebalancing, the station observes $60 - x$ stockouts in the morning, and $60 - y$ stockouts in the afternoon, where $x, y \leq 60$ are respectively the number of bikes rebalancing drops off in the morning/picks up in the afternoon. With 15 docks added these quantities would turn into $45 - x$ and $45 - y$ for $x, y \leq 45$. Thus, the same amount of rebalancing, up to a smaller upper bound, would simply reduce the number of dissatisfied customers (by the amount captured by the UDFs); beyond that upper bound additional rebalancing is no longer needed. This example aligns with anecdotal experiences system operators have shared with us: stations that had dock capacity added to them subsequently required less rebalancing.

Although we assume that no rebalancing occurs over the course of the planning horizon, the optimization model assumes that the initial number of bikes at each station is optimally allocated. We relax this assumption in Section 5.1 when we consider a regime in which no rebalancing occurs at all. Despite the fact that the two regimes can be viewed as polar opposites (optimally rebalanced overnight and no rebalancing overnight), our results indicate that they yield very similar recommendations for the operators. Our motivation to focus on these opposite extremes is simple: modeling a modest amount of rebalancing poses significant challenges. For example, unlike the effect of daily usage patterns, overnight rebalancing is affected by greater variability from external factors, ranging from the number of trucks to the supply of just-repaired bikes.

2.6.4. Exogenous Rentals and Returns. The demand profiles assume that the sequences of arrivals are exogenous; that is, there is a fixed distribution that defines the sequence of rentals and returns at each station. Before justifying this assumption, it is worth considering a setting in which it fails spectacularly: consider an allocation of bikes and docks that allocates no bikes at all. With no bikes, no attempted rental is ever successful and therefore no returns ever occur. As such, the sequence of arrivals of returns at one station are not independent of the allocations elsewhere.

Another extreme arises where the stations never run out of bikes, and there is always capacity available to receive bike returns. In this case, bike rentals and returns proceed smoothly independent of allocations and therefore can be viewed as exogenously given. This ideal case is the one to which we strive in our reallocation efforts. Of course, the assumption is never realized exactly in practice.

At this point, it is helpful to discuss the stochastic model for rentals and returns that we use in our calculations. Suppose that at each station, potential bikers arrive according to a Poisson process that is independent of that at all other stations. Such a model is plausible because of the Palm-Khintchine theorem that states, roughly speaking, that the superposition of the bike rentals of a large number of independent users is well modeled by a Poisson process (Cinlar 1972; Karlin and Taylor 1975, p. 221; Nelson 2013, p. 107). Also, suppose that users select their destinations according to an origin-destination routing matrix, thereby splitting the Poisson incoming flows into independent biker flows. Assuming biking times between any fixed pair of stations are identically distributed, and are independent across all bikers and station pairs, it follows that the process of returning bikes at a destination station from a fixed origin station, being a delayed Poisson process, is again a Poisson process. However, then, the overall bike-return process at the destination station, being a superposition of such flows from all origin stations, is again a Poisson process. Moreover, because of the splitting property of Poisson processes, the rental-return processes at each destination station are mutually independent. Thus, at each station, it is reasonable to model the returns and

rentals of bikes as Poisson processes, justifying the exogenous arrivals assumption. This modeling structure is approximate for several reasons: (i) the Poisson flows entering a destination station are interrupted if an upstream station runs out of bikes, (ii) a destination station may observe additional returns due to nearby stations being full, and (iii) a station may observe additional demand because of nearby stations being empty. As mentioned earlier, we attempt to minimize such shortages, so that we strive for conditions under which the approximation is close to reality, although it is still an approximation. Under this Poisson model, the rental and return processes at different stations are not independent. For example, a surge in rentals at one station may result in a surge of returns at a "downstream" station. Fortunately, our objective function is additively separable in stations, so independence at different stations is not required to compute the objective function; the "marginal" property that flows are Poisson at each station considered individually suffices.

Perhaps an even stronger justification for the exogenous-arrivals assumption comes from work by Jian et al. (2016) and Datner et al. (2019), who both use simulation optimization approaches. Datner et al. (2019) use their simulation optimization approach to identify only the optimal allocation of bikes. They endogenize (i) to (iii) and compare their results to optimizing with the UDF. Although they focus on a slightly different objective (*total user travel time*, where stockouts may lead to pickups/dropoffs at other stations or to users walking), they also report the fraction of rides affected by stockouts, which is what we/the UDFs aim to minimize; for this objective, their solutions improve on the UDFs by only 1.2% on average (across six scenarios). Similarly to us, Jian et al. (2016) aim to find the configuration of bikes and docks across the system that minimizes the number of out-of-stock events over the course of the day. In contrast to the user dissatisfaction functions, decensoring the demand data for their simulation required additional modeling decisions that allow them to endogenize (i) and (ii). Although this simulation approach still assumed that demand for rentals was exogenous, it endogenized returns, excluding (at least) the example suggested previously. However, it causes the resulting simulation optimization problem to be nonconvex in an unbounded fashion. Indeed, for any bound L, one can construct highly contrived examples in which there exists an initial allocation $(\vec{d}, \vec{b})$ and stations i and j such that when starting at allocation $(\vec{d}, \vec{b})$ it is the case that (a) moving two bikes from i to j improves the objective by at least L and (b) moving one bike from i to j gives a solution that is worse than $(\vec{d}, \vec{b})$. Such examples not only show that the objective function in that model is nonconvex, they also show that solutions from such a framework are harder to interpret. Jian et al. (2016) proposed a range of different gradient-descent algorithms as heuristics to find good solutions, including adaptations of the algorithms we present and analyze here. Despite the simulation adding key complexities to the system, the heuristics gave only limited improvements, approximately 3%, when given the solution found by our algorithms as a starting point. Thus, there exists substantial data-driven evidence to justify the use of UDFs.

Finally, the assumptions that rentals and returns are exogenous, and the objective is separable across stations, are quite common in the rebalancing literature. This includes, for example, Raviv and Kolka (2013), Raviv et al. (2013), Di Gaspero et al. (2013), Rainer-Harbach et al. (2013), Raidl et al. (2013), Ho and Szeto (2014), Kloimüllner et al. (2014), Kaspi et al. (2017), Forma et al. (2015), Alvarez-Valdes et al. (2016), and Schuijbroek et al. (2017), most of whom make the assumption implicitly.

2.6.5. Out-of-Stock Events and Demand Profiles. In practice, we cannot observe attempted rentals at empty stations nor can we observe attempted returns at full stations. Worse still, given that most bike-sharing systems have mobile apps that allow customers to see real-time information about the current number of bikes and empty docks at each station, there may be customers who want to rent a bike at a station, see on the app that the station has few bikes available presently, and decide against going to the station out of concern that by the time they arrive, the remaining bikes will already have been taken by someone else. Should such a case be considered an out-of-stock event (respectively, an attempted rental)? The user dissatisfaction functions assume that such events do not occur as the definition relies on out-of-stock events occurring only when stations are either entirely empty or entirely full.

Furthermore, to compute the user dissatisfaction functions, we need to be able to estimate the demand profiles: using only observed rentals and returns is insufficient as it ignores latent demand at empty/full stations. To get around this, we mostly apply a combination of approaches by O'Mahony and Shmoys (2015), O'Mahony et al. (2016), and Parikh and Ukkusuri (2015): we estimate Poisson arrival rates (independently for rentals and returns) for each 30-minute interval and use a formula developed by O'Mahony et al. (2016) to compute, for any initial condition (in number of bikes and empty docks) the expected number of out-of-stock events over the course of the interval. We plug these into a stochastic recursion suggested by Parikh and Ukkusuri (2015) to obtain the expected number of out-of-stock events over the course of a day as a function of the number of bikes and empty docks at 6:00 a.m. This is far from being

the only approach to compute user dissatisfaction functions; for example, in Section 6 we explicitly combine empirically observed arrivals with estimated rates for times when rentals/returns are censored at empty/full stations.

2.6.6. Advantages of User Dissatisfaction Functions.

The user dissatisfaction functions yield several advantages over a more complicated model such as the simulation. First, they provide a computable metric that can be used for several different operations: in Section 3, we show how to optimize over them for reallocated capacity, and in Section 6, we use them to evaluate the improvement from already reallocated capacity. Chung et al. (2018) used them to study an incentive program operated by Citi Bike in New York City, and they have been used extensively for motorized rebalancing (see Section 1.2). As such, the user dissatisfaction functions provide a single metric on which to evaluate different operational efforts to improve service quality, which adds value in itself. Second, for the particular example of reallocating dock capacity that we study here, they yield a tractable optimization problem, which we prove in Section 3. Third, for the reallocation of dock capacity, the discrete convexity properties we prove imply that a partial implementation of the changes suggested by the optimization (see Section 5) is still guaranteed to yield improvement. Finally, given a solution to the optimization problem, it is easy to track the partial contribution to the objective from changed capacity at each station, making solutions interpretable.

3. Discrete Gradient-Descent Algorithm

We begin this section by examining the mathematical structure of Problem (P2) that allows us to develop efficient algorithms. In Section 3.1, we define a natural neighborhood structure on the set of feasible allocations and define a discrete gradient-descent algorithm on this neighborhood structure. We prove in Section 3.2 that for the problem without operational constraints ((P2) with $z = \infty$), solutions that are locally optimal with respect to the neighborhood structure are also globally optimal; since our algorithm continues to make local improvements until it finds a local optimum, this proves that the solution returned by our algorithm must be globally optimal. Finally, in Section 3.3, we prove that the algorithm takes at most z iterations to find the best allocation obtainable by moving at most z docks within the system (see Definition 3 for a formal definition of a move of a dock); this not only proves that the gradient-descent algorithm optimally solves the minimization problem when including operational constraints, but also guarantees that doing so requires at most $D + B$ iterations.

3.1. Algorithm

We now present our algorithm before analyzing it for settings without the operational constraints. Intuitively, in each iteration our discrete gradient-descent algorithm picks one dock and at most one bike within the system and moves them from one station to another. It chooses the dock and the bike to maximize the reduction in objective value; that is, in a discrete sense, it executes a gradient-descent step. To formalize this notion, we define the *movement of a dock* via the following transformations. Denote by $e_i = (0, \ldots, 1, \ldots, 0)$ the canonical unit vector that has a one in its ith position and zeros elsewhere.

Definition 3. A *dock-move* from i to j corresponds to one of the following transformations of feasible solutions:

1. Moving one (empty) dock from i to j:

$$o_{ij}(\vec{d}, \vec{b}) = (\vec{d} - e_i + e_j, \vec{b})$$

2. Moving one dock and one bike from i to j, i.e., one full dock:

$$e_{ij}(\vec{d}, \vec{b}) = (\vec{d}, \vec{b} - e_i + e_j)$$

3. Moving one dock from i to j and one bike from h to j:

$$E_{ijh}(\vec{d}, \vec{b}) = (\vec{d} - e_i + e_h, \vec{b} + e_j - e_h)$$

4. Moving one bike from i to h and one dock from i to j:

$$O_{ijh}(\vec{d}, \vec{b}) = (\vec{d} + e_j - e_h, \vec{b} - e_i + e_h)$$

We often refer to the first kind as *moving an empty dock* from i to j and to the second kind as *moving a full dock* from i to j to indicate that the dock is moved by itself (empty) or with a bike (full). Without qualification, the movement of a dock can refer to any of the above. Furthermore, we define the *neighborhood* $N(\vec{d}, \vec{b})$ as the set of allocations that are one dock-move away from $(\vec{d}, \vec{b})$:

$$N(\vec{d}, \vec{b}) := \{o_{ij}(\vec{d}, \vec{b}), e_{ij}(\vec{d}, \vec{b}), E_{ijh}(\vec{d}, \vec{b}), O_{ijh}(\vec{d}, \vec{b}) : i, j, h \in [n]\}.$$

Notice that $(\vec{d}, \vec{b}) \in N(\vec{d}', \vec{b}')$ implies that $|\vec{d} + \vec{b} - \vec{d}' - \vec{b}'|_1 = 2$, that is, a dock-move distance of one; the converse, however, does not hold true as the dock-move distance between two allocations does not take into account their allocation of bikes.

Throughout the paper we also sometimes refer to the move of a bike from i to j, by which we mean a transformation from $(\vec{d}, \vec{b})$ to $(\vec{d} + e_i - e_j, \vec{b} - e_i + e_j)$. This changes the allocations of bikes to stations while keeping the number of docks at each station constant.

The previously defined neighborhood structure gives rise to a very simple algorithm (see Algorithm 3 in Online Appendix EC.2): we first find an optimal allocation of bikes for the current allocation of docks, that is, when each station i is restricted to have $\bar{d}_i + \bar{b}_i$

docks allocated to it (see Algorithm 1). The convexity of each c_i in the number of bikes (see Fact 1 in Online Appendix EC.4.1), with fixed number of docks, implies that this can be done greedily by taking out all the bikes and then adding them one by one (see the first page of Hochbaum 1994). Denote this allocation by $(\vec{d}^0, \vec{b}^0)$. Then, iterate over $r \in \{1, \ldots, z\}$, that is, through z periods, by either setting $(\vec{d}^r, \vec{b}^r)$ to be the best allocation in the neighborhood of $(\vec{d}^{r-1}, \vec{b}^{r-1})$, or if that allocation is no better than $(\vec{d}^{r-1}, \vec{b}^{r-1})$, returning $(\vec{d}^{r-1}, \vec{b}^{r-1})$ (see Algorithm 2). After iteration z, the algorithm returns $(\vec{d}^z, \vec{b}^z)$.

3.2. Optimality Without Operational Constraints

We first prove that Algorithm 3 returns an optimal solution to the problem without operational constraints. Specifically, we analyze the following:

$$\begin{aligned} \underset{(\vec{d},\vec{b})}{\texttt{minimize}}\ & c(\vec{d}, \vec{b}) \\ \texttt{s.t.}\ & |\vec{d}\ + \vec{b}|_1 = D + B, \\ & |\vec{b}|_1 = B. \end{aligned} \tag{P3}$$

We show that an allocation $(\vec{d}, \vec{b})$ that is locally optimal with respect to $N(\cdot, \cdot)$ must also be globally optimal with respect to (P3). Thus, if Algorithm 3, initialized with $z = \infty$, finds a solution for which there is no better solution in the neighborhood, then it returns an optimal solution to (P3). However, as that solution may have dock-move distance greater z to $(\vec{d}, \vec{b})$, global optimality of the algorithm only follows for (P3) and not for (P2). Before we prove Lemma 3 to establish this, we first define an allocation of bikes and docks as *bike-optimal* if it minimizes the objective among allocations with the same number of docks at each station.

Definition 4. An allocation $(\vec{d}, \vec{b})$ is *bike-optimal* if

$$(\vec{d}, \vec{b}) \in \operatorname{argmin}_{(\vec{d}', \vec{b}'): \forall i, d_i + b_i = d_i' + b_i', |\vec{b}'|_1 = B} \{c(\vec{d}', \vec{b}')\}.$$

The following lemma ensures that our analysis can, for the most part, focus only on bike-optimal solutions.

Lemma 2. *Suppose $(\vec{d}, \vec{b})$ is bike-optimal. Then given any i and j, the allocation resulting from the best dock-move from i to j is bike-optimal.*

The proof of the lemma is in Online Appendix EC.4.1. We highlight three implications of the lemma:

1. Because Algorithm 3 finds a bike-optimal solution initially and picks the best dock-move in each iteration, bike-optimality is an invariant of the algorithm, that is, $(\vec{d}^0, \vec{b}^0), (\vec{d}^1, \vec{b}^1), (\vec{d}^2, \vec{b}^2), \ldots$ are all bike-optimal.

2. Consider a bike-optimal allocation $(\vec{d}, \vec{b})$ and an allocation $(\vec{d}', \vec{b}')$ at dock-move distance 1 that is not in $N(\vec{d}, \vec{b})$; then there exists a dock-move from $(\vec{d}, \vec{b})$ that creates an allocation with (i) the same allocation of docks at each station as $(\vec{d}', \vec{b}')$, that is, $d_i' + b_i'$ at each station i, and (ii) an objective no worse than $(\vec{d}', \vec{b}')$.

3. To prove optimality of Algorithm 3 for (P3) it suffices to prove that bike-optimal solutions that are locally optimal w.r.t. our neighborhood structure are also globally optimal.

We formalize the last of these in Lemma 3, the proof of which we defer to Online Appendix EC.4.2. That appendix also contains the proof of Lemma 4, of which Lemma 3 is a corollary.

Lemma 3. *Suppose $(\vec{d}, \vec{b})$ is bike-optimal, but not optimal for either (P2) or (P3). Let $(\vec{d}^\star, \vec{b}^\star)$ denote a better solution. Then there exists $(\vec{d}', \vec{b}') \in N(\vec{d}, \vec{b})$ such that $(\vec{d}', \vec{b}')$ has both a lower objective value and a smaller dock-move distance to $(\vec{d}^\star, \vec{b}^\star)$ than $(\vec{d}, \vec{b})$ does.*

In the statement of Lemma 3, the allocation $(\vec{d}', \vec{b}')$ is not restricted to fulfill the operational constraints; that is, it may not be feasible for Problem (P3) (else, this would already imply that Algorithm 3 always, eventually, terminates with an optimal solution). Thus, optimality only follows for (P3). The allocations identified in the next lemma, $(\vec{d}', \vec{b}')$ and $(\vec{d}^{\star\star}, \vec{b}^{\star\star})$, also need not satisfy the operational constraints.

Lemma 4. *Consider any bike-optimal solution $(\vec{d}, \vec{b})$ and a better allocation $(\vec{d}^\star, \vec{b}^\star)$; let j and k denote stations with $d_j + b_j < d_j^\star + b_j^\star$ and $d_k + b_k > d_k^\star + b_k^\star$. Then either there exist $(\vec{d}', \vec{b}') \in N(\vec{d}, \vec{b})$ with $d_j' + b_j' = d_j + b_j + 1$ and $(\vec{d}^{\star\star}, \vec{b}^{\star\star}) \in N(\vec{d}^\star, \vec{b}^\star)$ with $d_j^{\star\star} + b_j^{\star\star} = d_j^\star + b_j^\star - 1$ or there exist $(\vec{d}', \vec{b}') \in N(\vec{d}, \vec{b})$ with $d_k' + b_k' = d_k + b_k - 1$ and $(\vec{d}^{\star\star}, \vec{b}^{\star\star}) \in N(\vec{d}^\star, \vec{b}^\star)$ with $d_k^{\star\star} + b_k^{\star\star} = d_k^\star + b_k^\star + 1$ such that*

1. *We have $c(\vec{d}, \vec{b}) - c(\vec{d}', \vec{b}') \geq c(\vec{d}^{\star\star}, \vec{b}^{\star\star}) - c(\vec{d}^\star, \vec{b}^\star)$*
2. *The dock-move distance from $(\vec{d}', \vec{b}')$ to $(\vec{d}^\star, \vec{b}^\star)$ is less than from $(\vec{d}, \vec{b})$ and the dock-move distance from $(\vec{d}^{\star\star}, \vec{b}^{\star\star})$ to $(\vec{d}, \vec{b})$ is less than from $(\vec{d}^\star, \vec{b}^\star)$*
3. *The dock-move from $(\vec{d}, \vec{b})$ to $(\vec{d}', \vec{b}')$ yields $(\vec{d}^\star, \vec{b}^\star)$ when applied to $(\vec{d}^{\star\star}, \vec{b}^{\star\star})$ (so, e.g., if $e_{kj}(\vec{d}, \vec{b}) = (\vec{d}', \vec{b}')$, then $e_{kj}(\vec{d}^{\star\star}, \vec{b}^{\star\star}) = (\vec{d}^\star, \vec{b}^\star)$, or equivalently $e_{jk}(\vec{d}^\star, \vec{b}^\star) = (\vec{d}^{\star\star}, \vec{b}^{\star\star})$).*

Remark 1. In the discrete convexity literature, a rewriting of the objective allows this to be interpreted as the exchange property of $M^\natural$-convex functions; this connection has been explored in the follow-up work of Shioura (2021).

3.3. Operational Constraints and Running Time

In this section, we show that Algorithm 3 is optimal for (P2) by proving that, for any r, in r iterations it

finds the best allocation obtainable by moving at most r docks. We thereby also provide an upper bound on the running-time of the algorithm, since an optimal solution can be at most $\min\{D+B, z\}$ dock-moves apart from $(\vec{d}^0, \vec{b}^0)$.

Our proof works inductively. We begin by showing (Lemma 5) that, assuming that $(\vec{d}^r, \vec{b}^r)$ minimizes the objective among solutions at dock-move distance at most r to $(\vec{d}^0, \vec{b}^0)$, $(\vec{d}^{r+1}, \vec{b}^{r+1})$ must be a local optimum among solutions at dock-move distance at most $r+1$ to $(\vec{d}^0, \vec{b}^0)$. This local optimality in the $(r+1)$ st iteration guarantees a particular structural property (see Lemma EC.1 in Online Appendix EC.4.4 for details). In Theorem 1, we use this structural property, together with the optimality of the solution in the rth iteration and the gradient-descent step, to show that $(\vec{d}^{r+1}, \vec{b}^{r+1})$ is globally optimal among solutions with dock-move distance at most $r+1$ to $(\vec{d}^0, \vec{b}^0)$. The proofs of Lemma 5, Lemma EC.1, and Theorem 1 can be found in Online Appendix EC.4.

Lemma 5. *Suppose $(\vec{d}^r, \vec{b}^r)$ minimizes the objective among solutions with $|\vec{d}+\vec{b}-\vec{d}^r-\vec{b}^r|_1 \le 2r$ and let $(\vec{d}^{r+1}, \vec{b}^{r+1})$ denote the next choice of the gradient-descent algorithm, that is, an allocation in the neighborhood of $(\vec{d}^r, \vec{b}^r)$ that minimizes the objective. Then $(\vec{d}^{r+1}, \vec{b}^{r+1})$ is a local optimum among solutions with $|\vec{d}+\vec{b}-\vec{d}^r-\vec{b}^r|_1 \le 2(r+1)$; that is, there is no solution in $N(\vec{d}^{r+1}, \vec{b}^{r+1})$ that is at dock-move distance at most $r+1$ to $(\vec{d}, \vec{b})$ and has a lower objective than $(\vec{d}^{r+1}, \vec{b}^{r+1})$.*

Theorem 1. *Initialized at $(\vec{d}, \vec{b})$, Algorithm 3 finds in iteration r an allocation that minimizes the objective among those at dock-move distance at most equal to r.*

Remark 2. In an earlier manuscript, as well as the proceedings version of the paper, we had falsely stated that local optima are globally optimal among allocations with dock-move distance at most r to $(\vec{d}, \vec{b})$, that is, that Lemma 3 continues to hold in the setting where allocations at dock-move distance $r+1$ are infeasible, and immediately derived optimality from that and Lemma 5. An example by Shioura (2021) shows that this is false: there exist solutions that are locally optimal with respect to our neighborhood structure and at dock-move distance at most r to $(\vec{d}, \vec{b})$, despite not being globally optimal among the solutions at dock-move distance at most r. Shioura (2021) also provides an alternative proof of correctness for our algorithm.

4. Scaling Algorithm

We now extend our analysis in Section 3 to adapt our algorithm to a scaling algorithm that provably finds an optimal allocation of bikes and docks for Problem (P3), that is, the setting without *operational* constraints, in $O(n\log(B+D))$ iterations.

The idea underlying the scaling algorithm (see Algorithm 4 in Online Appendix EC.2) is to proceed in $\lfloor \log_2(B+D) \rfloor + 1$ phases, where in the kth phase, each move involves $\alpha_k = 2^{\lfloor \log_2(B+D) \rfloor + 1 - k}$ bikes/docks rather than just one. The kth phase is prefaced by finding the bike-optimal allocation of bikes (given the constraints of only moving α_k bikes at a time), and terminates when no move of α_k docks yields improvement. We first observe that the multimodularity of $c(d, b)$ implies multimodularity of $c(\alpha_k d, \alpha_k b)$ for all k (table 1.3 in Murota 2018). Thus, our analysis in the last section implies that in the kth phase, the scaling algorithm finds an optimal allocation among all that differ in a multiple of α_k in each coordinate from $(\vec{d}, \vec{b})$. Furthermore, as $\alpha_{\lfloor \log_2(B+D) \rfloor + 1} = 1$, it finds the globally optimal allocation in phase $\lfloor \log_2(B+D) \rfloor + 1$.

Theorem 2. Algorithm 4 *finds an optimal allocation for* (P3) *in* $O(n\log(B+D))$ *iterations.*

Theorem 2 can provide a nontrivial speedup, relative to $O(n+D+B)$ for the gradient-descent algorithm, when B and D are large relative to n, that is, when stations have many docks on average. Otherwise it may create unnecessary overhead; in Online Appendix EC.6 we observe this on real data when comparing the different algorithms on data sets from different cities. Motivated by this insight, we then also define a hybrid algorithm (see Algorithm 5) that proceeds like the scaling algorithm but skips some of the phases. The proof of Theorem 2, included in Online Appendix EC.5, relies on a proximity result that bounds the distance (in dock-move distance) between optimal solutions in consecutive phases; although we prove such a result for Problem (P3) (Lemma EC.3), the running time bound in Theorem 2 relies on a bound from Shioura (2021) that is smaller but less general (as it applies only to (P3) but not to (P2)). We then extend our scaling algorithm (see Algorithm 6) to also work for Problem (P2), and use our own proximity bound to prove the following theorem.

Theorem 3. Algorithm 6 *runs in time polynomial in n and* $\log(B+D)$.

We remark that a significantly faster scaling algorithm for (P2) has been developed in the concurrent work of Shioura (2021).

5. Case Studies

In this section, we present the results of case studies based on data from three different bike-sharing systems: Citi

Bike in New York City, Blue Bikes in Boston, and Divvy in Chicago. Some of our results are based on an extension of the user dissatisfaction function which we first define in Section 5.1. Thereafter, in Section 5.2 we describe the data sets underlying our computation. Finally, in Section 5.3, we describe the insights obtained from our analysis. Although some of the results presented in this section are based on proprietary data, our discussion in Online Appendix EC.7 shows what can be reproduced using a data set and the source code we are making public.

5.1. Long-Run-Average Cost

A topic that has come up repeatedly in discussions with operators of bike-sharing systems is the fact that their means to rebalance overnight do not usually suffice to begin the day with the bike-optimal allocation. In some cities, like Boston, no rebalancing at all happens overnight. As such, it is desirable to optimize for reallocations that are robust with respect to the amount of overnight rebalancing. To capture such an objective, we define the long-run average of the user dissatisfaction function. Rather than mapping an initial condition in bikes and empty docks to the expected number of out-of-stock events over the course of one day, the long-run average maps to the average number of out-of-stock events over the course of infinitely many days. Notice that (under a weak ergodicity assumption discussed later) in this model, the initial allocation of bikes is irrelevant, and so is the total number of bikes allocated, as the long-run distribution of bikes present is determined solely by the distribution of arrival sequences. Formally, denoting by $X \oplus Y$ the concatenation of arrival sequences X and Y, that is, $(X_1, \dots, X_t, Y_1, \dots, Y_s)$, we define the long-run average of a station i with demand profile p_i as follows.

Definition 5. The long-run-average of the user dissatisfaction function at station i with demand profile p_i is

$$c_i^{\pi}(d,b) = \lim_{T\to\infty} \frac{\mathbb{E}_{Y_j \sim p_i (i.i.d.)}[c^{Y_1 \oplus Y_2 \oplus \cdots \oplus Y_T}(d,b)]}{T}.$$

We can compute $c_i^{\pi}(d,b)$ by computing for a given demand profile p_i the transition probabilities $\rho_{xy} := \sum_X p_i(X) \mathbf{1}_{\delta_X(d_i+b_i-x,x)=y}$, that is, the probability of station i having y bikes at the end of a day, given that it had x at the beginning, and given that each sequence of arrivals X occurs with probability $p_i(X)$. Given the resulting transition probabilities, we define a discrete Markov chain on $\{0, \dots, d_i + b_i\}$ and denote by $\pi_{p_i}^{d_i+b_i}$ its stationary distribution. This permits us to compute $c_i^{\pi}(d,b) = \sum_{k=0}^{d+b} \pi_{p_i}^{d+b}(k) c_i(d+b-k,k)$. Furthermore, from the definition of $c_i^{\pi}(\cdot,\cdot)$, it is immediately clear that $c_i^{\pi}(\cdot,\cdot)$ is also multimodular; as such, all results proven in the previous sections about $c_i(\cdot,\cdot)$ also extend to $c_i^{\pi}(\cdot,\cdot)$. In addition, we observe that, as long as the discrete Markov chain with transition probabilities ρ_{xy} is ergodic (e.g., with demand based on nonzero Poisson rates for both bike rentals and returns), $c_i^{\pi}(\cdot,\cdot)$ depends only on the sum of its two arguments but not on the value of each (as the initial number of bikes does not influence the steady-state number of bikes). Before comparing the results of optimizing over $c_i^{\pi}(\cdot,\cdot)$ and over $c_i(\cdot,\cdot)$, we now give some intuition for why the long-run average provides a contrasting regime.

5.1.1. Intuition for the Long-Run Average.

It is instructive to consider an example to illustrate where optimizing over the long-run average deviates from optimizing over a single day. Consider two stations i and j: at station i demand consists, deterministically, of k rentals every day; at station j, with probability $P \ll \frac{1}{2}$, there are k rentals followed by k returns, and with probability $1-P$ there are no rentals at all. At station i the user dissatisfaction function decreases by one for each of the first k full docks added; however, its long-run average objective remains constant at k: No matter how many docks and bikes are added, in the long-run the station is empty at the beginning of the day and therefore all k customers experience out-of-stock events. At station j, the first k full docks added only decrease the user dissatisfaction function by $2P \ll 1$ each, but the long-run average is also decreased by $2P$ for each dock added. Thus, optimally placing k docks and bikes at the two stations yields fundamentally different solutions depending on whether we optimize for one objective or the other. Furthermore, optimizing for the long-run average only gives a fraction $2P$ of the optimal improvement for a single day, whereas optimizing for a single day gives no improvement at all for the long-run average objective. Two lessons can be derived from this example. First, optimizing over one regime can, in theory, return solutions that are very bad in the other. Second, stations at which demand is antipodal (rentals in the morning, returns in the afternoon or vice versa) make better use of additional capacity in the long-run average regime.

5.2. Data Sets

We use data sets from the bike-sharing systems of three major American cities to investigate the effect different allocations of docks might have in each city. The three cities, New York City, Boston, and Chicago, vary widely in the sizes of their systems. When the data were collected from each system's open data feed (summer 2016), Boston had 1,300 bikes and 2,805 docks across 158 stations, Chicago had 4,700 bikes and 9,987 docks across 581 stations, and New York City had 6,750 bikes and 15,274 docks across 455 stations (given that the feeds only provide the number of bikes in each station, they do not necessarily capture the entire fleet size, for example, in New York City, a

significant number of bikes are kept in depots overnight).

For each station (in each system), we compute piecewise constant Poisson arrival rates to inform our demand profiles. To be precise, we take all weekday rentals/returns in the month of June 2016, bucket them in the 30-minute interval of the day in which they occur, and divide the number of rentals/returns at each station within each half-hour interval by the number of minutes during which the station was nonempty/nonfull. We compute the user dissatisfaction functions assuming that the demand profiles stem from these Poisson arrivals (Parikh and Ukkusuri 2015, O'Mahony et al. 2016). Some of our results in this section rely on the same procedure with data collected from other months.

Given that (in practice) we do not usually know the lower and upper bounds on the size of each station, we set the lower bound to be the current minimum capacity within the system and the upper bound to be the maximum one. Furthermore, we assume that $D + B$ is equal to the current allocated capacity in the system; that is, we only reallocate existing docks.

5.3. Impact on Objective

We summarize our results in Table 1. The columns Present, OPT, and 150-moved compare the objective with (i) the allocation before any docks are moved, (ii) the optimal allocation of bikes and docks, and (iii) the best allocation of bikes and docks that can be achieved by moving at most 150 docks from the current allocation. The columns headed c contain the bike-optimal objective for a given allocation of docks; the columns headed c^π are the long-run-average objective (for the same dock allocation). Two interesting observations can be made. First, although the optimizations are done over bike-optimal allocations without regard to the long-run average, the latter improves greatly in all cases. Second, in each of the cities, moving 150 docks yields a large portion of the total possible improvement. This stands in contrast to the large number of moves needed to find the actual optimum (displayed in the column Moves to OPT) and is due to diminishing returns of the moves.

A more complete picture of these insights is given in Figure EC.3 in Online Appendix EC.1. The x axis shows the number of docks moved starting from the present allocation, and the y axis shows the cumulative *improvement in objective*, that is, the difference between the initial objective and the objective after moving x docks. Each of the solid lines corresponds to different demand estimates being used to evaluate the same allocation of docks. The dotted lines (in the same colors) represent the maximum improvement, for each of the demand estimates, that can be achieved by reallocating docks; whereas these are not achieved through the dock moves suggested by the estimates based on June 2016 data, significant improvement is made toward them in every case. In particular, the initial moves yield approximately the same improvement for the different objectives/demand estimates. Thereafter, the various improvements diverge, especially for the New York City data from August 2016. This may be partially due to the system expansion in New York City that occurred in the summer of 2016 but does not contradict that all allocations corresponding to values on the x axis are optimal in the sense of Theorem 1.

5.3.1. Seasonal Effects.

As we mentioned in Section 2, we also consider the impact of seasonal effects. In Table 2, we show the improvement in objective when optimizing the movement of 200 docks in New York City based on demand estimates in June 2016 and evaluate the objective with the long-run average based on demand estimates based on March and November 2017. The estimated improvements suggest that optimizing with respect to June yields notable improvement with respect to any other.

5.3.2. Operational Considerations.

It is worth comparing the estimated improvement realized through reallocating docks to the estimated improvement realized through current rebalancing efforts. According to its monthly report (NYCBS 2016), Citi Bike rebalanced an average 3,452 bikes per day in June 2016: this number counts the average number of *rebalancing actions*, meaning that each pickup/dropoff counts as one bike rebalanced. A simple coupling argument implies that a single pickup/dropoff yields at most a change of one in the user dissatisfaction function (Figure 1); thus,

Table 1. Main Computational Results

	Present		OPT		150-moved		
City	c	c^π	c	c^π	c	c^π	Moves to OPT
Boston	831	1,118	607	945	672	985	412
Chicago	1,462	2,340	763	1,847	1,224	2,123	1,556
New York City	8,251	10,937	6,499	9,232	7,954	10,643	2,821

Notes. c, bike-optimal; c^π, long-run-average cost. Obtained from Algorithm 3 that was applied to data sets from June 2016.

Table 2. Improvement of 200 Docks Moved Based on the Long-Run Average Objective c^{π} Evaluated with Demand Estimates from June 2016, Evaluated with the Long-Run Average Objective c^{π} and Demand Estimates from 2017

	June 2016	March 2017	November 2017
New York City	358.7	260.3	294.6

rebalancing reduced out-of-stock events by *at most* 3,452 per day (assuming that each rebalanced bike actually has that much impact is extremely optimistic). Contrasting that to the estimated impact of strategically moving, for example, 500 docks diminishes the estimated number of out-of-stock events *by more than a fifth* of Citi Bike's (daily) rebalancing efforts.

Second, discussions with operators uncovered an additional operational constraint that can arise because of the physical design of the docks. Because these usually come in triples or quadruples, the exact moves suggested may not be feasible; for example, it may be necessary to move docks in multiples of four. By running the scaling algorithm (see Algorithm 4 in Online Appendix EC.2) only with $k \geq 2$, we can find an allocation in which docks are only moved in multiples of four. With that allocation, the objective of the bike-optimal allocation is 640, 848, and 6,573 in Boston, Chicago, and New York City, respectively, suggesting that, despite this additional constraint, compared with the column headed by OPT and c, almost all the improvements can be realized.

6. A Posteriori Evaluation of Impact

In this section, we show how one can use the user dissatisfaction functions to estimate (after the fact) the impact of reallocated capacity, and apply this approach to the 6 stations that were part of the pilot program mentioned in Section 1.1. One way to evaluate the impact would be to estimate new demand rates after docks have been reallocated, compute new user dissatisfaction functions for stations with added (decreased) capacity, and evaluate for those stations and the new demand rates the decrease (increase) between the old and the new number of docks. A drawback of such an approach is the heavy reliance on the assumed underlying stochastic process. Instead, we present here a data-driven approach with only little reliance on estimated underlying demand profiles.

Throughout this section, we denote by d and b the number of empty docks and bikes at a station after docks were reallocated, whereas d' and b' denote the respective numbers before docks were reallocated. Although $d + b$ and $d' + b'$ are known (capacity before and after docks were moved) and b can be found on any given morning (number of bikes in the station at 6:00 a.m.), we rely on some assumed value for b'—for that, in our implementation, we picked both $\min\{d' + b', b\}$ and $b \times \frac{(d'+b')}{d+b}$, that is, either the same number of bikes (unless that would be larger than the old capacity before docks were added) or the same proportion of docks filled with bikes.

6.1. Arrivals at Stations with Increased Capacity

In earlier sections, we assumed a known distribution for the sequence of arrivals based on which we compute the user dissatisfaction functions. In contrast, in this section we rely exclusively on observed arrivals (without any assumed knowledge of the underlying stochastic process) to analyze stations with increased capacity. This is motivated by a coupling argument to justify that censoring need not be taken care of explicitly in this case. To formalize our argument, we need to introduce some additional notation for the arrival sequences. Recall from Section 2 that we denoted by $X = (X_1, \ldots, X_S)$ a sequence of customers arriving at a bike-sharing station to either rent or return a bike and that X included failed rentals and returns, which in practice would not be observed because they are censored. Which X_i are censored depends on the (initial) number of bikes and docks at the station. Let us denote by $X(d, b)$ the subsequence of X that only includes those customers whose rentals/returns are successful (not censored) at a station that is initialized with d empty docks and b bikes, that is, the ones who do not experience out-of-stock events. Given the notation $c^X(\cdot,\cdot)$ used in Section 2 for a particular sequence of arrivals, we can then compute $c^{X(d,b)}(\cdot,\cdot)$. In particular, denoting by d', b' the number of empty docks and bikes without the added capacity, we may compute $c^{X(d,b)}(d', b')$. The following proposition then motivates the notion that censoring may be ignored at stations with added capacity.

Proposition 2. *For any X, $d' \leq d$ and $b' \leq b$, we have*

$$c^X(d', b') - c^X(d, b) = c^{X(d,b)}(d', b') - c^{X(d,b)}(d, b) = c^{X(d,b)}(d', b').$$

Proof. The proof of the second equality follows immediately from $X(d, b)$ including exactly those customers among X that are not censored, when a station is initialized with d empty docks and b bikes, so $c^{X(d,b)}(d, b) = 0$. Now, on the left-hand side, we can inductively go through all customers among X that are out-of-stock events when the station is initialized with d empty docks and b bikes. Since $d \geq d'$ and $b \geq b'$, each one of those increases both terms in the difference by one. Thus, taking them out of X does not

affect the value of the difference. However, then, we are left with only $X(d, b)$.

6.1.1. Extension to Stations with Decreased Capacity.

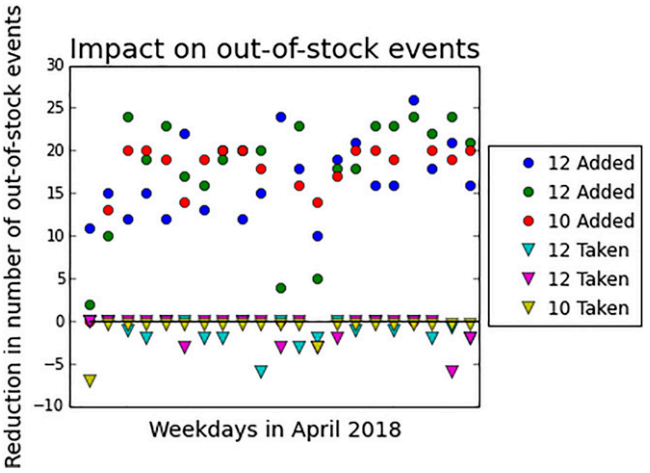
Proposition 2 does not apply to stations with decreased capacity: suppose $d < d'$ and $b = b'$; once the station (initialized with d empty docks and b bikes) becomes full, $X(d, b)$ observes no further returns even though these would be part of $X(d', b')$. To account for out-of-stock events occurring in that way, we fill in the censored periods with demand estimates. This does not usually require knowledge of the full demand-profile; for example, for a station that is nonempty and nonfull over the course of the day, no estimates are needed at all. Furthermore, for periods of time in which the station is full, we only need to estimate the number of intended returns; rentals over that period of time would not be censored.

6.1.2. Extension to Rebalancing.

Based on our reasoning in Section 2.6, our analysis of the user dissatisfaction functions and the resulting optimization problem (see Sections 2 and 3) thus far did not consider the rebalancing of bikes. In contrast, in the a posteriori analysis, we are able to take rebalancing into account.

To simplify the exposition, we restrict ourselves here to rebalancing that adds bikes to a station, though the reasoning extends to rebalancing that removes bikes. The simplest approach to treat bikes added through rebalancing is to just treat them as returns and thus include them (as virtual customers) in the sequence of arrivals X. However, this may cause an unreasonable increase to the value of $c^{X(d,b)}(d', b')$ (when the number of bikes added is greater than the number of empty docks would have been at that point in time if the station had initially had d' empty docks). In that case, the virtual customers (corresponding to rebalanced bikes) would incur out-of-stock events and thereby increase the value of the user dissatisfaction function. A more optimistic method that also treats rebalanced bikes as virtual customers is to redefine the user dissatisfaction function in such a way that out-of-stock events are only incurred by returns that correspond to nonrebalanced bikes. This, in essence, decouples the user dissatisfaction functions into subsequences, each of which are evaluated independently. Our analysis applied the latter, more optimistic method.

Figure 2. (Color online) Evaluation of Impact at Stations with Increased and Decreased Capacity



6.2. Impact of Initial Dock Moves in New York City

We consider three stations at which capacity was increased and three stations at which it was decreased based on our recommendations. For two of the stations at which capacity was increased, 12 docks were added; for one of them, the capacity was increased by 10; the decreases were by the same amounts, so in total this involved reallocating 34 docks. In Figure 2, we present the estimated impact for each weekday in April 2018 (without the extension to rebalancing). For stations with added capacity, we set d and b according to the number of bikes at 6:00 a.m. We evaluated $c^{X(d,b)}(d', b')$ for stations with docks added (see Proposition 2) using the observed arrivals $X(d, b)$ for each day. For the stations with docks taken away we estimated X by assuming a Poisson number of rentals (returns) whenever the station was empty (full), where the rate is based on decensored estimated demand from the same month. We use that to compute $c^{X(d,b)}(d', b') - c^{X(d,b)}(d, b)$ for these stations. The resulting values for different implementations are summarized in Table 3; aggregated over the entire month, the estimated net reduction in out-of-stock events varies

Table 3. Estimated Cumulative Changes at Stations Affected by Dock Reallocations Based on the Different Evaluations Described in Section 6.1

	No rebalancing		Rebalancing	
	$\min\{b, d' + b'\}$	$b \times \left(\frac{d'+b'}{d+b}\right)$	$\min\{b, d' + b'\}$	$b \times \left(\frac{d'+b'}{d+b}\right)$
Decrease where capacity was added	831.0	1,121.0	882.0	1,027.0
Increase where capacity was taken	0	58.7	0	59.7
Net reduction	831.0	1,062.3	882.0	967.3

between 831 and 1,121, which is about 1.2–1.6 fewer dissatisfied users per day and dock moved. Translating this into reduced rebalancing costs and comparing it to the cost of reallocating docks, strategically reallocating docks amortizes (depending on some system idiosyncrasies) in 2–5 weeks.

7. Conclusion

We considered several models that capture central questions in the design of dock-based bike-sharing systems, as are currently prevalent in North America. These models gave rise to new algorithmic discrete optimization questions, and we have demonstrated that they have sufficient mathematical structure to permit their efficient solution, thereby also extending existing theory in discrete convexity. We have focused on the (re)allocation of docks throughout the footprint of a bike-sharing system, capturing aspects of both better positioning of existing docks, and the optimal augmentation of an existing system with additional docks. These algorithms and models have been employed by systems within the United States with the desired effect of improving their day-to-day performance.

An alternative to optimizing dock allocations is to abandon the need to do so at all, by means of adopting a so-called dockless system. This approach has become prevalent in China and is gradually being implemented in North America on a much smaller scale (both in comparison with the systems in China and to the dock-based systems in North America); the management of these systems has its own challenges, and it remains to be seen whether these challenges can be overcome. Hybrid systems in which differential pricing enables centralized docking/parking areas that work in concert with dockless bikes may provide another path forward, as is done, for example, in Portland's Biketown system. Extensions of the methods we developed here will likely see continued use in this new setting as well.

Acknowledgments

The authors gratefully acknowledge the collaboration with Motivate (the company operating Citi Bike) and the comments of A. Shioura, who identified a false claim in an earlier version of the manuscript. The authors also thank anonymous reviewers and editors for suggestions that improved the paper throughout several revisions. A preliminary version of this paper, including parts of the first three sections, appeared in the proceedings of the 19th Conference on Integer Programming and Combinatorial Optimization (IPCO 2017).

References

Altman E, Gaujal B, Hordijk A (2000) Multimodularity, convexity, and optimization properties. *Math. Oper. Res.* 25(2):324–347.

Alvarez-Valdes R, Belenguer JM, Benavent E, Bermudez JD, Muñoz F, Vercher E, Verdejo F (2016) Optimizing the level of service quality of a bike-sharing system. *Omega* 62:163–175.

Brinkmann J, Ulmer MW, Mattfeld DC (2019) Dynamic lookahead policies for stochastic-dynamic inventory routing in bike sharing systems. *Comput. Oper. Res.* 106:260–279.

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

Capital Bikeshare (2014) Capital Bikeshare member survey report. https://d21xlh2maitm24.cloudfront.net/wdc/cabi-2014surveyreport.pdf?mtime=20161206135936.

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

Chen L, Zhang D, Wang L, Yang D, Ma X, Li S, Wu Z, et al. (2016) Dynamic cluster-based over-demand prediction in bike sharing systems. *Proc. ACM Internat. Joint Conf. on Pervasive and Ubiquitous Comput.* (ACM, New York), 841–852.

Chen X, Li M (2021) Discrete convex analysis and its applications in operations: A survey. *Production Oper. Management* 30(6):1904–1926.

Chung H, Freund D, Shmoys DB (2018) Bike angels: An analysis of citi bike's incentive program. *Proc. 1st ACM SIGCAS Conf. on Comput. and Sustainable Societies* (ACM, New York).

Cinlar E (1972) Superposition of point processes. Lewis PAW, ed. *Stochastic Point Processes: Statistical Analysis, Theory, and Applications* (Wiley Interscience, New York), 549–606.

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

de Chardon CM, Caruso G, Thomas I (2016) Bike-share rebalancing strategies, patterns, and purpose. *J. Transportation Geography* 55: 22–39.

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

Di Gaspero L, Rendl A, Urli T (2013) A hybrid ACO+CP for balancing bicycle sharing systems. *Proc. Internat. Workshop on Hybrid Metaheuristics* (Springer, Berlin), 198–212.

Erdoğan G, Battarra M, Calvo RW (2015) An exact algorithm for the static rebalancing problem arising in bicycle sharing systems. *Eur. J. Oper. Res.* 245(3):667–679.

Erdoğan G, Laporte G, Calvo RW (2014) The static bicycle relocation problem with demand intervals. *Eur. J. Oper. Res.* 238(2):451–457.

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

Freund D, Henderson SG, Shmoys DB (2016) Minimizing multimodular functions and allocating capacity in bike-sharing systems. Preprint, submitted November 28, https://arxiv.org/abs/1611.09304.

Freund D, Henderson SG, Shmoys DB (2017) Minimizing multimodular functions and allocating capacity in bike-sharing systems. *Proc. Internat. Conf. on Integer Programming and Combinatorial Optimization* (Springer, Berlin), 186–198.

Freund D, Henderson SG, Shmoys DB (2019) Bike sharing. *Sharing Economy* (Springer, Berlin), 435–459.

Freund D, Norouzi-Fard A, Paul A, Henderson SG, Shmoys DB (2020) Data-driven rebalancing methods for bike-share systems. *Analytics for the Sharing Economy: Mathematics, Engineering and Business Perspectives* (Springer, Cham), 255–278.

Fujishige S, Murota K (2000) Notes on l-/m-convex functions and the separation theorems. *Math. Programming* 88(1):129–146.

Ghosh S, Trick M, Varakantham P (2016) Robust repositioning to counter unpredictable demand in bike sharing systems. *Proc. 25th Internat. Joint Conf. on Artificial Intelligence* (AAAI Press, Palo Alto, CA), 3096–3102.

Hajek B (1985) Extremal splittings of point processes. *Math. Oper. Res.* 10(4):543–556.

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

Ho SC, Szeto W (2014) Solving a static repositioning problem in bike-sharing systems using iterated tabu search. *Transportation Res., Part E Logist. Transportation Rev.* 69:180–198.

Hochbaum DS (1994) Lower and upper bounds for the allocation problem and other nonlinear optimization problems. *Math. Oper. Res.* 19(2):390–409.

Jian N, Henderson SG (2015) An introduction to simulation optimization. *Proc. Winter Simulation Conf.* (IEEE, New York), 1780–1794.

Jian N, Freund D, Wiberg HM, Henderson SG (2016) Simulation optimization for a large-scale bike-sharing system. *Proc. Winter Simulation Conf.* (IEEE, New York), 602–613.

Kabra A, Belavina E, Girotra K (2020) Bike-share systems: Accessibility and availability. *Management Sci.* 66(9):3803–3824.

Karlin S, Taylor HM (1975) *A First Course in Stochastic Processes*, 2nd ed. (Academic Press, Boston).

Kaspi M, Raviv T, Tzur M (2017) Bike-sharing systems: User dissatisfaction in the presence of unusable bicycles. *IISE Transactions.* 49(2):144–158.

Kloimüllner C, Papazek P, Hu B, Raidl GR (2014) Balancing bicycle sharing systems: An approach for the dynamic case. *Proc. Eur. Conf. on Evolutionary Comput. in Combinatorial Optim.* (Springer, Berlin), 73–84.

Laporte G, Meunier F, Calvo RW (2018) Shared mobility systems: An updated survey. *Ann. Oper. Res.* 271(1):105–126.

Li J, Qin H, Baldacci R, Zhu W (2020) Branch-and-price-and-cut for the synchronized vehicle routing problem with split delivery, proportional service time and multiple time windows. *Transportation Res., Part E Logist. Transportation Rev.* 140:101955.

Li Q, Yu P (2014) Multimodularity and its applications in three stochastic dynamic inventory problems. *Manufacturing Service Oper. Management* 16(3):455–463.

Li Y, Zheng Y, Zhang H, Chen L (2015) Traffic prediction in a bike-sharing system. *Proc. 23rd SIGSPATIAL Internat. Conf. on Advances in Geographic Inform. Systems* (ACM, New York), 33.

Liu J, Sun L, Chen W, Xiong H (2016) Rebalancing bike sharing systems: A multi-source data smart optimization. *Proc. 22nd ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining* (ACM, New York), 1005–1014.

Lu Y, Song J-S (2005) Order-based cost optimization in assemble-to-order systems. *Oper. Res.* 53(1):151–169.

Moriguchi S, Murota K, Tamura A, Tardella F (2020). Discrete midpoint convexity. *Math. Oper. Res.* 45(1):99–128.

Murota K (1996) Convexity and Steinitz's exchange property. *Adv. Math.* 124(2):272–310.

Murota K (1998) Discrete convex analysis. *Math. Programming* 83(1): 313–371.

Murota K (2003) *Discrete Convex Analysis* (SIAM, Philadelphia).

Murota K (2018) Main features of discrete convex analysis. Accessed February 21, 2019, http://www.comp.tmu.ac.jp/kzmurota/paper/rimsCOSS2018/COSS2018chap1.pdf.

Murota K, Shioura A (1999) M-convex function on generalized polymatroid. *Math. Oper. Res.* 24(1):95–105.

Nair R, Miller-Hooks E, Hampshire RC, Bušić A (2013) Large-scale vehicle sharing systems: Analysis of vélib'. *Internat. J. Sustainable Transportation* 7(1):85–106.

Nelson BL (2013) *Foundations and Methods of Stochastic Simulation*, vol. 187 of *International Series in Operations Research & Management Science* (Springer, New York).

NYCBS (2016) June 2016 monthly report.

O'Mahony E (2015) Smarter tools for (Citi) bike sharing. PhD thesis, Cornell University, Ithaca, NY.

O'Mahony E, Henderson SG, Shmoys DB (2016) (Citi)Bike sharing. Working paper, Cornell University, Ithaca, NY.

O'Mahony E, Shmoys DB (2015) Data analysis and optimization for (Citi) bike sharing. *Proc. 29th AAAI Conf. on Artificial Intelligence* (AAAI, Palo Alto, CA), 687–694.

Parikh P, Ukkusuri S (2015) Estimation of optimal inventory levels at stations of a bicycle sharing system. In *Transportation Research Board 94th Annual Meeting Compendium of Papers. Transportation Research Board*, 1–14.

Raidl GR, Hu B, Rainer-Harbach M, Papazek P (2013) Balancing bicycle sharing systems: Improving a VNS by efficiently determining optimal loading operations. *Proc. Internat. Workshop on Hybrid Metaheuristics* (Springer, Berlin), 130–143.

Rainer-Harbach M, Papazek P, Hu B, Raidl GR (2013) Balancing bicycle sharing systems: A variable neighborhood search approach. *Proc. Eur. Conf. on Evolutionary Comput. in Combinatorial Optim.* (Springer, Berlin), 121–132.

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

Raviv T, Tzur M, Forma IA (2013) Static repositioning in a bike-sharing system: Models and solution approaches. *EURO J. Transportation Logist.* 2(3):187–229.

Schuijbroek J, Hampshire R, van Hoeve W-J (2017) Inventory rebalancing and vehicle routing in bike sharing systems. *Eur. J. Oper. Res.* 257(3):992–1004.

Shioura A (2021) M-convex function minimization under l1-distance constraint and its application to dock reallocation in bike-sharing system. *Math. Oper. Res.* 47(2):1566–1611.

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

Shui C, Szeto W (2020) A review of bicycle-sharing service planning problems. *Transportation Res., Part C Emerging Tech.* 117:102648.

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. *Proc. AAAI Workshop: Computat. Sustainability*. https://dblp.org/rec/conf/aaai/SinghviSFHOSW15.html?view=bibtex.

Wang J, Tsai C-H, Lin P-C (2016) Applying spatial-temporal analysis and retail location theory to public bikes site selection in Taipei. *Transportation Res. Part A Policy Practice* 94:45–61.

Zhang J, Pan X, Li M, Yu PS (2016) Bicycle-sharing system analysis and trip prediction. Preprint, submitted April 3, https://arxiv.org/abs/1604.00664.

Zipkin P (2008) On the structure of lost-sales inventory models. *Oper. Res.* 56(4):937–944.

Daniel Freund is the Class of 1947 development professor and an assistant professor of operations management at the MIT Sloan School of Management. His research applies optimization and stochastic modeling to improve the design of large-scale systems, especially platforms in the transportation space and in the gig economy.

Shane Henderson holds the Charles W. Lake, Jr. chair in productivity in the School of Operations Research and Information Engineering at Cornell University. He works in simulation and applied probability with applications in a range of areas, emphasizing emergency services and transportation.

David Shmoys is the Laibe/Acheson professor in the School of Operations Research and Information Engineering and the Department of Computer Science, as well as the founding director of the Center for Data Science for Enterprise & Society at Cornell University. His research interests are in the design and analysis of optimization models and algorithms for a broad cross section of decision-making settings.