Noname manuscript No.
(will be inserted by the editor)

Data-driven rebalancing methods for bike-share systems

Daniel Freund · Ashkan Norouzi-Fard · Alice Paul · Carter Wang · Shane G. Henderson · David B. Shmoys

Received: date / Accepted: date

Abstract As bike-share systems expand in urban areas, the wealth of publicly available data has drawn researchers to address the novel operational challenges" these systems face. One key challenge is to meet user demand for available bikes and docks by rebalancing the system. This chapter reports on a collaborative effort with Citi Bike to develop and implement real data-driven optimization to guide their rebalancing efforts. In particular, we provide new models to guide truck routing for overnight rebalancing and new optimization problems for other non-motorized rebalancing efforts during the day. Finally, we evaluate how our practical methods have impacted rebalancing in New York City.

This work was partially supported by National Science Foundation grants CCF-1526067, CMMI-1537394, CCF- 1522054, and CCF-1740822 as well as Army Research Office grant W911NF-17-1-0094.

Daniel Freund
Cornell University
E-mail: df365@cornell.edu

Ashkan Norouzi-Fard
EPFL
E-mail: ashkan.norouzifard@epfl.ch

Alice Paul
Brown University
E-mail: alice_paul@brown.edu

Carter Wang
Motivate International
E-mail: carterwang@motivateco.com

Shane G. Henderson
Cornell University
E-mail: sgh9@cornell.edu

David B. Shmoys
Cornell University
E-mail: dbs10@cornell.edu

Keywords First keyword · Second keyword · More

1 Introduction

Bike-share systems have become a fixture of urban transportation landscapes, offering sustainable alternatives for both tourists and everyday commuters. In providing connections between existing transportation modes, these systems enable users to change their entire commute to a more sustainable one. In addition, they provide access to transportation in neighborhoods that historically had none. The wealth of data these systems collect and the high predictability of aggregate user traffic allow for in-depth analysis of customer demand and system operations.

More so than with other modes of transportation, users directly affect the state of bike-share systems. This presents novel operational challenges. In particular, asymmetric demand creates empty and full stations across the city; while empty stations prevent customers from accessing the system, full stations, where every dock is taken, prevent them from leaving the bike there. Thus, one of the operator's key challenges is to meet demand by relocating bikes and temporarily increasing station capacity. This is referred to as rebalancing.

In the past few years, bike-share systems have developed different approaches to rebalancing. In the most common approach, trucks are used to move bikes to high-demand areas. This is particularly effective overnight, when both traffic and demand are low. During the day, vehicular traffic impairs these efforts, and instead operators use *trikes* or *corrals*. A trike is a trailer that typically holds at most five bikes and is towed by a cyclist to relocate bikes between a station A, with high supply, and a station B, with low supply (cf. Figure 1). A corral, on the other hand, artificially increases the capacity of a popular station by having an employee store bikes in between docks, thereby using all of the available space (cf. Figure 1).

We provide data-driven methods to help New York City Bikeshare (NYCBS) improve overall utilization of the Citi Bike system and reduce customer dissatisfaction. This chapter formulates and attacks the underlying optimization problems that arise in truck-based rebalancing overnight, trike-based rebalancing during the day, and the placement of a limited number of seasonal corrals. For each of these settings, we describe the methods developed and their impact for NYCBS.

First, we consider optimally routing trucks to relocate bikes overnight. This is called the *overnight rebalancing problem*. Our objective in this problem is to minimize expected customer dissatisfaction over the next day. We present an integer program (IP) that constructs routes for a given number of trucks and show how to find good solutions when given a fixed amount of computation time in practice. Next, we consider the *mid-rush rebalancing problem*, studying how to optimally assign trikes to cycle between pairs of stations. Here, we use a maximum-weight k-edge matching to assign trike routes and maximize the

Fig. 1 Pictures of a corral and a trike being used in NYC.

impact on customer satisfaction. Finally, we consider how to optimally place corrals at stations where our goal is to minimize the number of customers who cannot find an open dock within a quarter mile of their preferred destination. We model this question as a maximum coverage problem that we solve within seconds using a simple integer programming formulation.

Each of the methods in this paper are in different development stages for NYCBS. In particular, we have completed trial runs using our overnight rebalancing schedules, routing three to four of their trucks over an eight-hour period. In the trials we ran, our routes were able to reduce expected customer dissatisfaction an additional 12% on average when compared to their usual efforts. NYCBS has also used our proposed placement of corrals in 2016. Further ahead, we have implemented our method to match trikes to stations and are in discussions with NYCBS to have our model inform their operational schedule in future seasons. We are also in discussions with bike-share systems in other major American cities (e.g., Washington D.C., Boston, Bay Area) about how these methods can make local transportation more sustainable and user-friendly.

This chapter thus summarizes our methodological contributions and their impact on day-to-day operations in New York City.

2 Related Work

Our objective in overnight rebalancing is to minimize the *expected number of dissatisfied customers*, or the expected number of customers who are not able to access or leave the system. This objective (cf. Section 3) function, often referred to as the *user dissatisfaction function* (UDF), has first been defined in Raviv and Kolka [33]; different ways of computing it were suggested in Henderson et al. [16], Parikh and Ukkusuri [30], and Schuijbroek et al. [40].

The definition of the UDF triggered a line of work that used this objective to formulate a static rebalancing problem, wherein capacitated trucks are routed in a bike-sharing system over a finite time interval so as to minimize the subsequent number of dissatisfied users. Examples, solving larger and larger instances, include Forma et al. [12], Ho and Szeto [17], and Szeto et al. [44].

Our optimization model in Section 4 is similar to these, though it differs with respect to underlying constraints.

This approach is very much related to the so-called *Static Bicycle Rebalancing Problem*, which was first introduced in Benchimol et al. [1]. Here, each station has a target number of bikes, and the goal is to route finite capacity trucks to pick up and drop off bikes to meet these targets while minimizing the route length. Variations of this formulation have been studied in Erdoğan et al. [10], Erdoğan et al. [11], and Bulhoes et al. [2]. The last of these provides an IP formulation that allows vehicles to visit the same location repeatedly; Casazza et al. [4] shows that conditions exist that guarantee that multiple visits to the same location are unnecessary.

Whereas most of the previously mentioned approaches are based on integer programming techniques, Rainer-Harbach et al. [32] defines greedy heuristics to solve a similar formulation. In fact, this and subsequent works like Di Gaspero et al. [9] as well as Kloimullner et al. [23] and Papazek et al. [29] combine minimizing travel time and maximizing impact in their objectives. Kloimullner et al. [23] consider what they refer to as the *dynamic case*, where demand occurs both during and after the interval in which rebalancing happens. Fundamentally different methodological approaches to rebalancing have been taken by Liu et al. [25] whose rebalancing approach is based on a clustering of stations and Ghosh et al. [14], Ghosh et al. [15], and Lowalekar et al. [26] who develop a robust optimization framework.

Beyond rebalancing, researchers have also looked at the question of what the optimal allocation of bikes should look like at any given time. This can be thought of as solving a rebalancing problem in which the time taken for rebalancing is neither bounded through constraints nor penalized in the objective. Methodologically, this has been approached through the UDFs and simulation optimization frameworks. Henderson et al. [16] and Freund et al. [13] apply UDFs; Jian and Henderson [19] and Jian et al. [20] develop a simulation optimization that aims to minimize the number of dissatisfied users, but takes into account (more explicitly) the interdependencies between stations. Datner et al. [7] also takes into account interdependencies between stations and additionally even considers customers' behavioral responses to empty/full stations with respect to transportation alternatives. Rather than minimizing the number of users that do not get served, Datner et al. [7] then minimize the total travel-times of all users, including ones that had to find an alternative to the bike-sharing system.

There are also flow formulations to capture the optimal allocations of bicycles rather than the routing problem: examples of such approaches include Shu et al. [41], Vogel et al. [45], and Contardo et al. [6]. These do not explicitly take into account routing, though they do contain some costs to account for rebalancing between stations.

Work on the optimal allocation of bikes at a given time has been complemented by studies of the system design itself. Such studies so far have mostly focused on the (re-)allocation of dock capacity within the system. Examples include the aforementioned Jian et al. [20] and, closely related, Saltzman and

Bradford [39] (simulation), Henderson et al. [16] and Freund et al. [13] (UDF). A somewhat orthogonal approach, using econometric techniques to infer the impact on demand from reallocated capacity, is found in Kabra et al. [21] and Zheng et al. [47].

In addition to the widespread study of optimization problems in the design and operation of bike-sharing systems, there are also a variety of forecasting-related problems that have been explored. In particular, there are several examples of demand forecasting, including Rudloff and Lackner [37], Li et al. [24], Salaken et al. [38], O'Mahony and Shmoys [28], Riquelme et al. [36] – in this chapter we apply the decensoring techniques suggested in O'Mahony and Shmoys [28]. Kaspi et al. [22] identified unusable bikes, given usage data, which has relevance for routing problems in maintenance as studied by Paul et al. [31]. Researchers have also focused on more specific demand problems: Hsu et al. [18] examines customer choices at empty or full stations, Zhang et al. [46] studies the likely destination of each customer conditioned on the customer's attributes (e.g., gender, age, origin), and Singhvi et al. [42] tries to predict demand in neighborhoods based on census data.

Although there has been extensive work on overnight, motorized rebalancing, comparably little research has been conducted on non-motorized rebalancing efforts, which form a crucial part of NYCBS's operations. One such work is Henderson et al. [16], which studies the routing of trikes. This paper partitions the set of stations into *producers*, which are stations likely to fill up, and *consumers*, likely to empty out. In investigating the problem of setting trike routes, the paper aims to minimize the distance of any consumer (producer) to another consumer (producer) that is rebalanced by one of the trikes. While our work on trikes is driven by the same application, our objective (cf. Section 5) is again to minimize the expected number of dissatisfied customers. A different kind of non-motorized rebalancing has been investigated by Singla et al. [43] and [5]: they each investigate operational questions related to an incentive program like NYCBS's *Bike Angels*, which crowdsources some of the rebalancing efforts to the user base.

Finally, our work distinguishes itself in that it was conducted in close cooperation with NYCBS and has had an impact on their day-to-day operations. This stands in contrast to the findings of de Chardon et al. [8], which conclude that very little of the existing work on rebalancing has had an impact in practice.

3 User dissatisfaction function

Prior work by Raviv and Kolka [33], O'Mahony [27], and Schuijbroek et al. [40] defines a *user dissatisfaction function* that uses demand information to map the number of bikes at a station at the beginning of a time interval to the expected number of customers that are unable to access/leave the system at that station over the course of the interval.

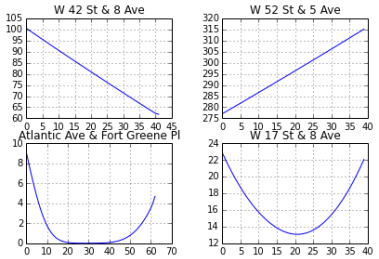

Fig. 2 User dissatisfaction function for four stations, with different demand patterns, in NYC for 7AM-10PM.

The user dissatisfaction function is based on an $M/M/1/\kappa$ queue at each station s, in which the state of the queue $X_s(t)$ at time t corresponds to the number of bikes in the station at that time. More precisely, there are two Poisson processes with rates μ and λ. Both rates can be functions of time, but they are assumed to be exogeneous and independent of the operator's actions. An arrival of the Poisson process with rate μ moves the state of the queue from i to $i-1$ if $i > 0$. If $i = 0$, the state of the queue does not change. This represents the arrival of a user who wants to pick up a bike but cannot due to a lack of available bikes. We assume that such a customer leaves the system dissatisfied. Similarly, an arrival of the Poisson process with rate λ moves the state of the queue from i to $i+1$ if $i < \kappa$. If $i = \kappa$, the state of the queue does not change; the user is also dissatisfied in this case as she cannot drop off her bike due to a lack of available docks. The UDF F_s uses these Poisson processes to map an initial number of bikes to the expected number of such dissatisfied users over a given finite time horizon (cf. Figure 2). In the prior work mentioned above it was shown that $F_s(\cdot)$ is convex and can be efficiently computed for time-invariant λ and μ. In later work (cf. Parikh and Ukkusuri [30]) a dynamic program was derived to extend those methods to the time-varying rates we have used in our work. To obtain values for λ and μ we use the decensoring method introduced in [28].

4 Overnight Rebalancing

New York City Bikeshare deploys between three and five box trucks every night to redistribute bikes. Based on an analysis of past usage data (cf. O'Mahony and Shmoys [28]), NYCBS aims to ensure that both bikes and open docks are available where needed. This is critical because customers' most cited complaint about bike sharing is the lack of available bikes or docks (cf. Capital Bikeshare [3]). Given the latent demand, perfect customer satisfaction, i.e. always having at least one bike and one dock available at every station, seems unattainable: our estimates for the user dissatisfaction function suggest that with the optimal distribution of bikes across the system and no rebalancing thereafter, there would still be more than 12000 dissatisfied over the course

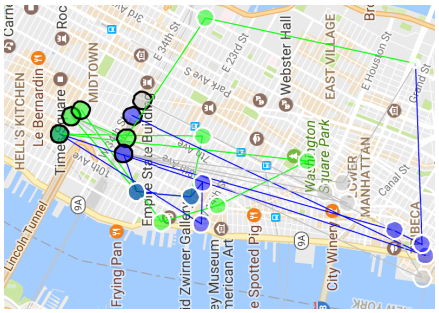

Fig. 3 Truck routes for three trucks on August 8, 2016. Each circle corresponds to a station at which at least one of the trucks stops. A white outer circle corresponds to a pick-up, a black outer circle to a drop-off. Initially, all trucks start at a NYCBS depot in the East Village. (Map data: Google)

of the day (cf. Figure 2). Thus, obtaining perfect availability would require the relocation of an estimated 6000 bikes (each relocated bike can help satisfy demand for 2 customers: one searching a dock at the station the bike is taken from, and one searching a bike at the station it is placed at), which is beyond the current scope of NYCBS's efforts. However, rebalancing can nevertheless yield improvements for the system when it decreases the (expected) number of dissatisfied customers. In this section we discuss our work with NYCBS to increase the efficiency of their overnight rebalancing towards that goal.

As was done in the earlier work on overnight rebalancing, we use an IP formulation to model this optimization problem. The main distinguishing feature of our approach is that we greedily partition the problem into subproblems, thereby controlling the size of the IP. In particular, we find routes for subintervals of the entire time horizon, subsets of trucks, and subsets of stations. Our choice of stations is based on carefully weighing the potential benefits of rebalancing at each station and the distance of the station from the location of the truck at the beginning of the interval.

4.1 Integer Programming Formulation

We begin by introducing our integer program. The formulation is time-indexed, and we assume the given time for overnight rebalancing has been broken into T identical time steps. In each time step, a truck will either pick up/drop off bikes or move to an adjacent station. In that way, the edges between the stations are unweighted and traversing any edge takes one time step. In order to make this assumption reasonable, we add dummy stations to break long distances into individual time steps. More precisely, if the distance between two stations s_1 and s_2 is ℓ time steps, we add a path of $\ell - 1$ stations between them, that allows us to move between such two stations in exactly ℓ time steps by traversing one edge in each step. This technique significantly reduces the dimension of the IP and distinguishes our formulation from previous ones.

Notation.

After adding all dummy stations, let S be the set of stations, T be the number of time steps, and K be the number of trucks. For $s \in S$, $t \in [T]$, and $k \in [K]$, the variable x_{stk} represents whether or not truck k is at station s at time t. Similarly, the variable y_{stk} represents the number of bikes at station s at time t to which truck k has access. This prevents multiple trucks from moving the same bikes. Lastly, the variable b_{tk} represents the number of bikes in truck k at time t.

We use the following notation:

- $N(s)$ denotes the neighborhood of s, that is, the stations to which a truck can move from s in a single time step.
- γ is the number of bikes that can be picked up or dropped off in one time step.
- $\text{start}(s)$ is the number of bikes in station s at time $t = 1$.
- $\min(s)$ is the minimizer of the user dissatisfaction function at station s. That is, the number of bikes at station s that minimizes the expected number of dissatisfied customers.
- $c_s = \frac{F_s(\text{start}(s)) - F_s(\min(s))}{|\text{start}(s) - \min(s)|}$ is a linear approximation of the slope of F_s and gives the improvement per bike moved at s (see Figure 4).
- S_+ is the set of stations s for which $\text{start}(s) > \min(s)$. For example, the station in Figure 4 is in S_+.
- S_- is the set of stations s for which $\text{start}(s) \leq \min(s)$.

For ease of presentation, we state here only the main constraints of the IP before we explain the effect of each. The reader is advised to read the IP in parallel with the explanations below.

$$\text{maximize}_{\ x,y,b} \sum_{s\in S, k\in[K]} (y_{s1k} - y_{sTk})c_s$$

subject to:

$$x_{stk} \le x_{s(t-1)k} + \sum_{s':s\in N(s')} x_{s'(t-1)k}, \qquad \forall s,t,k; \qquad (1)$$

$$\sum_{s\in S} x_{stk} = 1, \qquad \forall t,k; \qquad (2)$$

$$\sum_{k\in[K]} y_{s1k} = \text{start}(s), \qquad \forall s; \qquad (3)$$

$$\text{start}(s) \le \sum_{k\in[K]} y_{stk} \le \min(s), \qquad \forall s \in S_-, t; \qquad (4)$$

$$\min(s) \le \sum_{k\in[K]} y_{stk} \le \text{start}(s), \qquad \forall s \in S_+, t; \qquad (5)$$

$$\sum_{s\in S} y_{stk} + b_{tk} = \sum_{s\in S} y_{s1k} + b_{1k}, \qquad \forall t,k; \qquad (6)$$

$$|y_{stk} - y_{s(t-1)k}| \le \gamma x_{stk}, \qquad \forall s,t,k; \qquad (7)$$

$$|y_{stk} - y_{s(t-1)k}| + \gamma|x_{stk} - x_{s(t-1)k}| \le \gamma, \qquad \forall s,t,k. \qquad (8)$$

Below we explain the function of each part of the IP.

- The objective function is the summation of changes in the linearized user dissatisfaction functions at each station, i.e. the reduction in expected number of dissatisfied customers due to the relocation of bikes.
- **Constraint (1)** allows each truck to only move to a station adjacent to the one it is currently at.
- **Constraint (2)** indicates that at each time step, each truck must be in exactly one station.
- **Constraint (3)** initiates the number of bikes at every station. Notice that at this point already the bikes are distributed among the K trucks.
- **Constraints (4)** and **(5)** guarantee that the number of bikes in each station s remains between start(s) and the minimizer $\min(s)$. In other words, we enforce that moving a bike only improves the setup (cf. *Pareto Constraints and Optimal Fleet Size*).
- **Constraint (6)** enforces that the total number of bikes in the system does not change over time.
- **Constraint (7)** makes sure that we pick/drop bikes at a station from a truck only if the truck is at that station and that the number of bikes moved is bounded by γ, the number of bikes rebalancers are able to move within one period.
- **Constraint (8)** ensures the truck either moves or picks/drops bikes in one time step but not both. In most of the previous works, researchers have omitted this constraint. This constraint makes the IP significantly harder

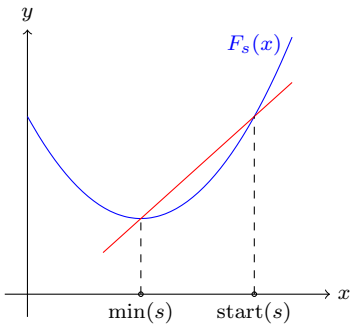

Fig. 4 Linearization of F_s.

to solve but makes the resulting path viable in practice. Notice that the absolute values in the constraints can be linearized.

Moreover, we add capacities to the truck by bounding b_{tk}. In practice, we extend this IP to fix the starting/finishing stations for each truck, as well as the number of bikes in each truck at the beginning of the night.

Pareto Constraints and Optimal Fleet Size

Notice that the extension of the linearization of F_s beyond the point $\min(s)$ does not capture the actual behavior of the UDF. In particular, at that point the UDF sees more dissatisfied customers while the linearization sees fewer. This, however, is not the reason we impose the *Pareto constraints* (4) and (5), since optimizing over the linear envelope of F_s is not significantly harder than over the linearization; instead, the constraints are imposed to ensure that service for customers at one station is not sacrificed for improved service for customers elsewhere. This is particularly true when, as is the case in NYC, $\sum_s \text{start}(s) << \sum_s \min(s)$ (at the time we started the pilots, the bike fleet size was about 7.000 whereas the minimizers summed to about 10.000). In such a setting, not having fairness constraints can yield undesirable outcomes. For instance, without fairness constraints we might find solutions in which bikes can be picked up from a station far away that has more bikes than needed but are instead picked up from a station nearby that has fewer than needed. While this may lead to fewer dissatisfied customers in total, NYCBS aims for rebalancing to be a Pareto improvement to the system (i.e., no station is worse off, some are improved); thus, we include constraints (4) and (5).

4.2 Solution Methods

As presented, state-of-the-art IP solvers are too slow to solve this formulation within the limited time window between when the operator receives the data and when the trucks must start their routes. The following heuristic methods

help decrease the computation time to solve this IP and improve the quality of the solution returned.

Reducing the number of edges. To avoid adding too many dummy stations and inflating the size of the IP, we choose a threshold d on the distance between two stations and only add a path between stations s_1 and s_2 if they are at most d time steps apart.

Dividing T into smaller time intervals. At the start of the rebalancing period, there is little time (typically about 20 minutes) between the time all the data (state of the system, number of bikes on each truck, etc.) becomes available and the time when trucks are meant to begin their routes. To gain computation time, we break T into smaller intervals and only route trucks for the first few hours of the route. While this part of the route is being executed, we then use the time to solve for the next interval. This segmentation of the computation time greatly improved the quality of our overall routes.

Greedily selecting stations. For each time interval, we further reduce the size of the IP by removing stations where the room for improvement is low. We rank stations based on a combination of c_s, the potential benefit of each bike picked or dropped, and $|\min(s) - \text{start}(s)|$, the number of available/required bikes, and run the IP with roughly 40 stations, further refined by excluding stations too far away from the starting point of the truck.

Splitting trucks. Instead of solving one IP for all K trucks, we break the computation time into K equal pieces and solve for the route of the first truck, then the second truck, and so forth. For example, if we have two trucks and two hours of computation time, we would solve for the route of each truck in one hour.

To decrease the size of the IP, we only add stations in reasonable proximity to the current position of the truck. We then compute the path over one time interval and update the set of stations. On the one hand, our IP can find very good solutions for the smaller instances on which we solve. On the other hand, by picking stations in the described greedy fashion, we ensure that the combination of the solutions to the small instances has objective close to the global optimum. The routes we construct for each truck and time interval are compatible in that they can be pieced together to form one coherent route. We show in our results that these heuristics still yield good solutions to the original IP, that is, we compare our objectives to the LP relaxation of the full IP.

4.3 Results

In this section, we summarize our results. First, we report the gap between the solution we return using the techniques above and an optimal solution of a valid LP relaxation. Second, we compare our solutions to the current routes employed by NYCBS, as guided by tools based on our earlier work. On

average, our solutions reduce customer dissatisfaction by 20% compared to this previous computationally informed approach. All results were produced using Gurobi v6.5 on a machine with 8GB RAM and an Intel i7-2600 processor with 3.4GHZ.

IP gap

360 Stations		
# of Trucks	Avg Objective Function	Avg Gap
1	148	28.1%
2	232	22.1%
3	301	12.3%
4	330	9.5%

Fig. 5 A posteriori optimization for overnight truck rebalancing in Manhattan.

When solving the IP, we assume the number of time steps is 60, each time step corresponds to 6 minutes, and that workers can load/unload up to 7 bikes in each time step. These numbers are based on discussions with NYCBS. To evaluate the performance of our IP, we used our IP to route various numbers of trucks in Manhattan, which had around 360 stations at the time of our study. As inputs for start(s), we used the number of bikes at each station at midnight over the course of a week. In Figure 5, we report the average impact on the number of dissatisfied customers and the average integrality gap over the week. The worst integrality gap occurred Sunday night when the distribution of bikes was furthest from the commuter demand during the week. We emphasize that this gap is computed with respect to the LP-relaxation on all 360 stations, all trucks, and all time-steps, i.e. the LP-relaxation of the provided integer program; hence, it includes both the integrality gap we find for the smaller problem instances (on 30-40 stations) and the gap between the optima of these smaller instances and the global optimum.

The average integrality gap decreases as the number of trucks grows. This seems to be a consequence of the Pareto constraints, since there are fewer bikes available to be moved (per truck). With more trucks, it is easier to achieve the best possible solution. We remark that a much smaller integrality gap can be obtained without constraints (7) and (8) as has been done in Raviv et al. [34], Ho and Szeto [17], and Forma et al. [12]. The solutions obtained this way, however, allow for as many stops (with loading/unloading of few bikes at each) as there are time periods; in discussions with NYCBS we found that in many cases the time required to find a place to park the truck is not sufficiently dominated by driving/(un)loading time that it could be ignored. We thus had to add these constraints, even though it makes the IP much harder to solve.

Practical Results

Dispatchers at NYCBS currently use a myopic decision aid to route trucks. This aid is based on user dissatisfaction functions and was developed in close cooperation with our group. While this decision aid shows dispatchers the optimal fill level at stations and indicates stations where rebalancing could yield large improvements, it does not provide optimized routes. In contrast, our IP solutions look to globally optimize routes. We formulated our model with feedback from NYCBS; their expertise led, for example, to the refinement that in each time step a truck can either move or pick/drop bikes but not both. Over the course of a week in July 2016, we then compared our proposed routes with the manual routes executed by the dispatchers at NYCBS. On average our results showed objectives about 20% higher. Together with the NYCBS management, we reviewed our proposed routes before running pilots to route between 2 and 5 of their trucks overnight.

Pilot Experiences

Despite the improvements, the pilots conducted with NYCBS proved to be more difficult than expected, as we quickly describe here.

Unknown constraints. Our first attempt to pilot our method had to be aborted because we routed a truck to a station that was in a narrow street. Indeed, the street was so narrow that the driver did not feel comfortable driving the truck to the station. Since we did not know how to handle the missed stop (and the resulting difference in number of bikes aboard the truck) at subsequent stops, we aborted that first attempt as a pilot. While events like this are rare, they do happen and when they do, they significantly complicate the routing problem. As an heuristic to overcome such problems, our subsequent pilots included alternative stations (greedily selected after solving the IP) at which each truck could stop without interfering with the routes of the other trucks.

Cost of solving offline. Solving the system offline can have detrimental consequences when unexpected demand occurs late at night. While the system significantly slows down at night, we have encountered cases where our route dictated picking up some number of bikes from a station and by the time the truck had arrived there, fewer than that were left. Similarly, it may occur that the Pareto constraint is violated in the execution of the route because the number of bikes at a station changes between the time we solve and the time the truck arrives. To avoid this occurring regularly, we excluded certain areas that tend to slow down later than others (e.g., East Village) in the first part of the route.

Evaluating results. In evaluating our results, we found that certain system conditions greatly influence the efficiency of rebalancing. On nights when the system is in great imbalance, it is not uncommon for each truck to move enough bikes to reduce the expected number of dissatisfied customers by

> 100 users. In contrast, on nights were the system is already reasonably balanced, it is often not feasible for a truck to move bikes to reduce the objective by > 50.

While the final point indicates that evaluating results is difficult, the improvements through rebalancing on the nights during which we ran this pilot were on average 12% higher than in the rest of that month.

5 Trikes

In this section, we study the impact of trikes on the expected number of dissatisfied users and solve a corresponding optimization problem. Recall that a trike is a trailer towed by a cyclist that holds at most 5 bikes at a time (cf. Figure 1). NYCBS uses trikes between fixed pairs of stations to move bikes from high-demand, low-supply stations to low-demand, high-supply stations. NYCBS considers trikes a preferred choice of rebalancing during rush hour when trucks are slowed down by traffic. We first assume that every station may have only one trike route incident to it. In this regime, the problem of finding the optimal m trike routes can be formulated as a bipartite maximum m-edge matching, where the weight of an edge between two stations corresponds to the reduction in user dissatisfaction with a trike added between them. Next, we generalize these ideas to the case where a station can utilize multiple trikes. With a slight variation, we show that this as well as can be formulated as a matching problem and efficiently solved.

5.1 Model

Similar to the user dissatisfaction function defined in Section 3, we model the *user dissatisfaction function with a trike* between stations A and B as follows: Poisson processes with rates $\lambda_A, \mu_A, \lambda_B, \mu_B$ correspond to arrivals and departures of users at station A and B, respectively. We use the random variable $X_s(t) \in \{0, \ldots, k_s\}$ to represent the number of bikes at station $s \in \{A, B\}$ at time t, where k_s is the capacity of station $s \in \{A, B\}$. As before, a dissatisfied customer at s corresponds to an arrival (resp. departure) when s is full (resp. empty), i.e., $X_s(t) = k_s$ (resp. $X_s(t) = 0$).

In addition to arrivals and departures of users, we also have a trike with capacity k_R that moves as many bikes as possible from A to B. We assume that at times $t_1, \ldots, t_r$ the trike stops at one of the stations. Without loss of generality, the stop at t_i is at A if i is odd and at B if it is even. In other words, the trike cycles back and forth between A and B. When the trike arrives at station A it picks up as many bikes as possible given the number of bikes at A and the number of bikes already in the trike; similarly, when the trike arrives at station B it drops off as many bikes as possible given the number of available docks at B and the number of bikes already in the trike. We use

the random variable $X_R(t) \in \{0, \dots, k_R\}$ to represent the number of bikes in the trike at time t.

We are interested in the expected number of dissatisfied users at A and B over the time horizon from t_0 to t_{r+1}. We can write the expected number of dissatisfied users at stations A and B as follows:

$$F_{A\to B} = \sum_{i=0}^{r}\sum_{j=0}^{k_A} \Pr(X_A(t_i^+) = j) F_A^i(j) + \sum_{i=0}^{r}\sum_{j=0}^{k_B} \Pr(X_B(t_i^+) = j) F_B^i(j)$$

The two terms correspond to the expected number of dissatisfied users at each station; to obtain them, notice that at the beginning of time-interval i with probability $\Pr(X_s(t_i^+) = j)$ there are j bikes at station s and in that case an expected $F_s^i(j)$ users will be dissatisfied in that interval. Thus, we need to find $\Pr(X_s(t_i^+) = j)\ \forall s, i, j$. For ease of notation, we denote $\Pr(X_A(t_i^+) = \alpha, X_B(t_i^+) = \beta, X_R(t_i^+) = \rho)$ as $\pi^i(\alpha, \beta, \rho)$ and remark that by setting $X_R(t_0^+) = 0$ with probability one and assuming that $X_A(t_0), X_B(t_0)$ are independent we obtain:

$$\pi^0(\alpha, \beta, \rho) = \begin{cases} 0, \text{ if } \rho > 0 \\ \Pr(X_A(t_0^+) = \alpha) \cdot \Pr(X_B(t_0^+) = \beta), \text{ else.} \end{cases}$$

This will represent the base case to recursively compute $F_{A\to B}$.

To calculate $\pi^i(\alpha, \beta, \rho)$ in general we need to analyze the system changes in each time interval. Specifically, given the Poisson process rates and the current number of bikes x at a station $s \in \{A, B\}$, we can compute the expected number of dissatisfied customers $F_s^i(\cdot)$ in an interval (t_i, t_{i+1}). In doing so, we also obtain the probability that there are y bikes in station s at the start of the next time interval. More precisely, $\forall s \in \{A, B\}, x, y \in \{0, \dots, k_s\}$ we let

$$P_{x,y}^{s,i} := \Pr(X_s(t_{i+1}^-) = y | X_s(t_i^+) = x),$$

where $X_s(t^+) := \lim_{\epsilon\to 0^+} X_s(t+\epsilon)$, i.e., $X_s(t_i^+)$ is the number of bikes at s just after the trailer has stopped at the station and $X_s(t^-) := \lim_{\epsilon\to 0^+} X_s(t-\epsilon)$ is the number of bikes at s just before the trailer has stopped at the station.

Thus, for even i, we have $\pi^{i+1}(\alpha, \beta, \rho) = 0$ if $\alpha > 0$ and $\rho < k_R$, as otherwise the trike would pick up more bikes. Otherwise, we obtain

$$\pi^{i+1}(\alpha, \beta, \rho) = \sum_{x=0}^{k_A}\sum_{y=0}^{k_B}\sum_{z=0}^{\rho} \pi^i(x, y, z) P_{x,\alpha+\rho-z}^{A,i} P_{y,\beta}^{B,i},$$

where we define $P_{x,y}^{s,i} = 0$ for $y \notin \{0, \dots, k_s\}$. This is because, with $X_R(t_i^+) = z$, the event that $\{X_A(t_{i+1}^+) = \alpha$ and $X_R(t_{i+1}^+) = \rho\}$ happens if and only if the number of bikes at A before the pick-up at t_{i+1} is $\alpha + \rho - z$.

Similarly, for odd i, we have $\pi^{i+1}(\alpha, \beta, \rho) = 0$ if $\beta < k_B$ and $\rho > 0$. Otherwise,

$$\pi^{i+1}(\alpha, \beta, \rho) = \sum_{x=0}^{k_A} \sum_{y=0}^{k_B} \sum_{z=\rho}^{k_R} \pi^i(x, y, z) P^{A,i}_{x,\alpha} P^{B,i}_{y,\beta+\rho-z}.$$

Recognizing that

$$\Pr(X_A(t_i^+) = j) = \textstyle\sum_{y=0}^{k_B} \sum_{z=0}^{k_R} \pi^i(j, y, z) \quad \text{and}$$
$$\Pr(X_B(t_i^+) = j) = \textstyle\sum_{x=0}^{k_A} \sum_{z=0}^{k_R} \pi^i(x, j, z),$$

we can now compute all $\Pr(X_s(t_i^+) = j)$ and $\pi^i(\alpha, \beta, \rho)$ recursively starting with the base cases for $i = 0$ and incrementing i. Thus, we can efficiently compute $F_{A\to B}$.

Given $F_{A\to B}$ for each pair of stations, we use these values to assign the trike routes. To formulate the problem as a maximum m-edge matching problem, we create a bipartite graph $G = (X \cup Y, E)$ where $X = Y$ is the set of stations. We set the weight of edge (A, B) equal to the difference of the expected number of dissatisfied users at stations A and B without the trike, given by

$$\sum_{j=0}^{k_A} \Pr(X_A(t_0) = j) F_A(j) + \sum_{j=0}^{k_B} \Pr(X_B(t_0) = j) F_B(j),$$

and the expected number with the trike, $F_{A\to B}$. Thus, the weight of any matching is equal to the reduction in the expected number of dissatisfied customers from the corresponding trike routes and vice versa. Further, there is a well-known and efficient algorithm for finding the maximum m-edge matching for any bipartite graph.

Relaxed Assumptions

If we allowed more than one trike incident to a particular station, the dimensionality of the dynamic program explodes, yielding it infeasible. A slight variation of the model, however, allows us to model this problem as a maximum-weight matching as well. Instead of coupling the random variables of numerous stations and trikes, we estimate for each station how many bikes trikes can pick up/drop off bikes over the time horizon in which they operate. To find a solution for k trikes we then create for each station s nodes $s_1, \ldots s_k$ and create an edge (A_i, B_j) with weight set to be the improvement at station A through the ith additional trike and at B through the jth additional trike. Notice that, in contrast to the earlier formulation, this does not incorporate the chance that a trike incident to station A does not have an effect because there are no bikes to pick up at the adjacent station B.

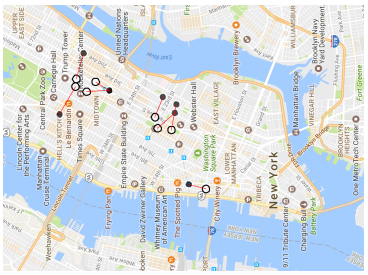

Fig. 6 Trike routes identified by the maximum-weight matching formulation. Red lines indicate trikes that pick up bikes at white circles and drop them off at black ones. (Map data: Google)

5.2 Results

Setting $k_R = 5$ and eight trike stops between 7:30 AM and 9:15 AM (thus moving a total of at most 20 bikes per station pair), we use the above dynamic program to compute the weight of the maximum matchings of various sizes; in Figure 6 we display the maximum matching of size 8, in Figure 7 the weight of all maximum matchings of size 1 through 20. Notice that for up to 5 trailers, there is an improvement of about 20 (fewer dissatisfied customers). While these numbers can be viewed in comparison to the numbers in Figure 5, the costs of operating a truck are significantly higher than those of a trailer. Decisions about which to operate thus also depend on the relative costs of trucks and trailers.

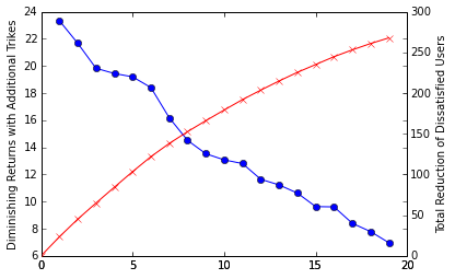

Fig. 7 Total improvement (red) of trikes and corresponding diminishing returns (blue).

6 Corrals

Our work on corrals is motivated by studies on the correlation between distance to transportation modes and willingness to use these modes. A study in Regional Plan Association [35], for example, claims that commuters are much more likely to use public transportation when living within a quarter mile of a

station than when further away. Furthermore, it was shown in Kabra et al. [21] that shorter distances to available stations correlates with increased demand for bike-share systems.

6.1 Model

Based on the findings described before, we say a station has a *shortage* if no station within a quarter mile has at least 15% of its docks available. In other words, a station is in shortage if a user intending to end their trip at that station will likely have to search more than a quarter mile away to find an available dock. Thus, stations in shortage significantly impact customer utilization.

We define a shortage measure for the system given by the total time stations are in shortage. Formally, let $N(s) \subseteq S$ be the set of neighboring stations within a quarter mile of s, including s itself, and let $A_{s,t}$ the indicator of the event that station s has at least 15% of its docks available at time t. For a set of stations S and a set of points in time $\mathcal{T}$, we define the shortage measure as

$$w(S, \mathcal{T}) = \sum_{s \in S} \sum_{t \in \mathcal{T}} \max\{0, 1 - \sum_{j \in N(s)} A_{j,t}\}.$$

It is possible to extend this measure to a weighted version that associates a time-dependent coefficient based on dock demand to each station. Since the unweighted version was the one that informed decision-making for NYCBS, we restrict ourselves to that.

By placing a corral at a station s, the amount of time that stations in $N(s)$ are in shortage is significantly reduced. Given a budget B, the goal is to place at most B corrals to minimize the shortage measure. For a set of past time points in $\mathcal{T}'$, let

$$w_s = \sum_{t \in \mathcal{T}'} \max\{0, 1 - \sum_{j \in N(s)} A_{j,t}\}.$$

This represents the amount of time that station s is in shortage and is equivalent to the reduction in the shortage measure for s if a station in $N(s)$ is assigned a corral. It is then natural to model corral placement as a maximum coverage problem via the following IP:

$$\begin{aligned} \text{maximize}_{\ x,y} & \sum_{s \in S} w_s x_s \\ \text{subject to: } & x_s \leq \sum_{s' \in N(s)} y_{s'} \quad \forall s \\ & \sum_{s \in S} y_s \leq B \\ & x_s, y_s \in \{0, 1\} \quad \forall s. \end{aligned}$$

In the given IP, the variable y_s represents whether or not a corral is placed at station s and the variable x_s represents whether or not there is a corral placed within a quarter mile from station s. The first constraint sets x_s to be 0 if no corral is assigned in $N(s)$, and the second constraint corresponds to the constraint that we can assign at most B corrals.

6.2 Results

The evaluation of our work in this section is two-fold. Given the real implementation of our suggested corrals by NYCBS, we are able to observe the change in shortages from 2015 (without corrals) to 2016 (with corrals). However, we cannot be certain this corresponds 1-to-1 with a decrease in unsatisfied customers. Since the presence/absence of these is in general difficult to measure, we also use a discrete-event simulation to estimate the reduction in dissatisfied customers through our choice of corrals.

Our discrete-event simulation is based on Poisson arrivals with fixed destinations at each station. In contrast to the user dissatisfaction function described in Section 3, the discrete-event simulation captures interdependencies between stations. For instance, customers who do not find a dock at a particular station, roam around to nearby stations until eventually they do. For details of the implementation, we refer the reader to Jian et al. [20] and O'Mahony [27]; we use the same simulation with the addition of allowing for some stations to have temporarily increased capacity between 7.30AM and 5.30PM.

Simulated Impact on Unsatisfied Customers

To compare the performance of our coverage formulation, we consider several allocations of corrals (cf. Figure 8):

1. *coverage '15/'16* are the allocations of corrals obtained by solving the coverage problem using data from 10 days in June 2015, and June 2016 respectively.
2. *k-median* is based only on the biking distance between separate stations. Using the Google Maps API, we obtained pairwise distances and then solved the resulting k-median instance.
3. *max minutes* relies only on the number of minutes a station had no docks available in the month of May 2016.
4. *None* allocates no corrals at all.

Our results are visualized in Figure 8. Based on data from June 2016, we simulated 40 realizations of demand and computed, using common random numbers, for each realization and each allocation of corrals, the number of unsatisfied customers. Notice first that adding corrals, usually, only ever improves the objective (though corrals may also have second-order effects that increase the expected number of dissatisfied customers, which the plot shows). Moreover, the solutions obtained from the coverage formulation (run both

with data from 2015 and with more current data from 2016) outperforms the solutions that are based only on geography (*k-median*) or only on number of full minutes (*max minutes*): on average the two coverage solutions observe 304 and 162 fewer unsatisfied customers, compared to just 4 fewer for the k-median solution; the max minutes solutions ranks in the middle with an average reduction of 77. The results thus show the necessity of taking both usage data and geography into account.

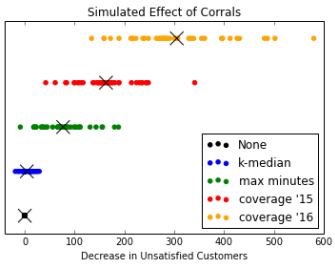

Fig. 8 Results of 40 simulated days with different sets of corrals; × denotes the average performance.

Observed Real Impact

Six corrals identified by the maximum coverage formulation (see Figure 9) with $B = 6$ have been in place for most of Summer 2016. To evaluate the benefit of these corrals, we first consider the value of the shortage measure in July 2016 (with the corrals) and in July 2015 (without). As the size of the system has expanded significantly, we restricted the shortage measure to only include stations S in Manhattan that were in use for most of July 2015 and July 2016. For reference, $|S| = 277$. We find a 31.6% reduction in the time that stations were in shortage. Further, to show that the majority of the improvement is indeed due to the effects of the corrals, we also calculate the shortage measure in July 2015 and July 2016 for $\bar{S}$ which excludes the stations with corrals from S. This should give an idea as to how much the shortage measure has decreased through other rebalancing efforts. Here, we only find a 14.6% improvement which shows that the majority of the improvement was due to the added corrals. Results of these analyses are summarized in Figure 10.

7 Conclusion

We provide models and solutions for three different optimization problems that arise in rebalancing bike-share systems. For overnight rebalancing and trailer routing, we focus on minimizing the expected number of dissatisfied users.

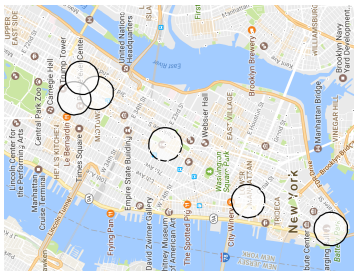

Fig. 9 Corral stations in NYC with $\frac{1}{4}$-mile radius. (Map data: Google)

	S	$\bar{S}$
July 2015	49435	56989
July 2016	33804	48657
Percent Reduction	31.6%	14.6%

Fig. 10 Shortage measure for July 2015 and July 2016.

First, we present our IP for the overnight rebalancing problem. This formulation, along with our pre-processing techniques, allows us to route trucks for New York City's large and complicated bike-share system. Here, we are able to improve on current rebalancing efforts by 20% on average. Next, we consider how to optimally add trikes between stations. We formulate this problem as a maximum-weight matching. In focusing on the same objective for these two problems, the expected number of dissatisfied users, we provide the operator with a quantitative comparison between two rebalancing methods. Lastly, we consider where to add corrals. Our analysis shows that a small number of corrals significantly improves user access to the system. While our work considered the effect of these rebalancing efforts separately, future work could consider how to combine these efforts. For example, it would be interesting to consider how adding an additional trike affects overnight rebalancing and analyzing the cost-effectiveness of these efforts. Since combining these efforts would likely lead to intractable optimization problems, a promising direction is to incorporate rebalancing into simulation frameworks.

References

1. M. Benchimol, P. Benchimol, B. Chappert, A. De La Taille, F. Laroche, F. Meunier, and L. Robinet. Balancing the stations of a self service bike hire system. *RAIRO-Operations Research*, 45(1):37–61, 2011.
2. T. Bulhoes, A. Subramanian, G. Erdoğan, and G. Laporte. The static bike relocation problem with multiple vehicles and visits. *European Journal of Operational Research*, 264:508–523, 2018.
3. Capital Bikeshare. 2014 capital bikeshare member survey report, 2014.

4. M. Casazza, A. Ceselli, and R. W. Calvo. Inventory rebalancing in bike-sharing systems. In *15th Cologne-Twente Workshop on Graphs and Combinatorial Optimization*, page 35, 2017.
5. H. Chung, D. Freund, and D. B. Shmoys. Bike angels: An analysis of citi bike's incentive program. In *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, page 5. ACM, 2018.
6. C. Contardo, C. Morency, and L.-M. Rousseau. Balancing a dynamic public bike-sharing system. *Cirrelt Montreal*, 4, 2012.
7. S. Datner, T. Raviv, M. Tzur, and D. Chemla. Setting inventory levels in a bike sharing network. *Transportation Science*, 0(0):null, 2017. doi: 10.1287/trsc.2017.0790. URL `https://doi.org/10.1287/trsc.2017.0790`.
8. C. M. de Chardon, G. Caruso, and I. Thomas. Bike-share rebalancing strategies, patterns, and purpose. *Journal of Transport Geography*, 55: 22–39, 2016.
9. L. Di Gaspero, A. Rendl, and T. Urli. Constraint-based approaches for balancing bike sharing systems. In *International Conference on Principles and Practice of Constraint Programming*, pages 758–773. Springer, 2013.
10. G. Erdoğan, G. Laporte, and R. W. Calvo. The static bicycle relocation problem with demand intervals. *European Journal of Operational Research*, 238:451–457, 2014.
11. G. Erdoğan, M. Battara, and R. W. Calvo. An exact algorithm for the static rebalancing problem arising in bicycle sharing systems. *European Journal of Operational Research*, 245:667–679, 2015.
12. I. A. Forma, T. Raviv, and M. Tzur. A 3-step math heuristic for the static repositioning problem in bike-sharing systems. *Transportation research part B: methodological*, 71:230–247, 2015.
13. D. Freund, S. G. Henderson, and D. B. Shmoys. Minimizing multimodular functions and allocating capacity in bike-sharing systems. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 186–198. Springer, 2017.
14. S. Ghosh, M. Trick, and P. Varakantham. Robust repositioning to counter unpredictable demand in bike sharing systems. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 3096–3102. AAAI Press, 2016.
15. S. Ghosh, P. Varakantham, Y. Adulyasak, and P. Jaillet. Dynamic repositioning to reduce lost demand in bike sharing systems. *Journal of Artificial Intelligence Research*, 58:387–430, 2017.
16. S. G. Henderson, E. O'Mahony, and D. B. Shmoys. (citi)bike sharing. Submitted, 2016.
17. S. C. Ho and W. Szeto. Solving a static repositioning problem in bike-sharing systems using iterated tabu search. *Transportation Research Part E: Logistics and Transportation Review*, 69:180–198, 2014.
18. Y.-T. Hsu, L. Kang, and Y.-H. Wu. User behavior of bikesharing systems under demand–supply imbalance. *Transportation Research Record: Journal of the Transportation Research Board*, pages 117–124, 2016.

19. N. Jian and S. G. Henderson. An introduction to simulation optimization. In *Proceedings of the 2015 Winter Simulation Conference*, pages 1780–1794. IEEE Press, 2015.
20. N. Jian, D. Freund, H. M. Wiberg, and S. G. Henderson. Simulation optimization for a large-scale bike-sharing system. In *Proceedings of the 2016 Winter Simulation Conference*, pages 602–613. IEEE Press, 2016.
21. A. Kabra, E. Belavina, and K. Girotra. Bike-share systems: Accessibility and availability. 2016.
22. M. Kaspi, T. Raviv, and M. Tzur. Detection of unusable bicycles in bike-sharing systems. *Omega*, 65:10–16, 2016.
23. C. Kloimullner, P. Papazek, B. Hu, and G. R. Raidl. Balancing bicycle sharing systems: an approach for the dynamic case. *European Conference on Evolutionary Computation in Combinatorial Optimization*, pages 73–64, 2014.
24. Y. Li, Y. Zheng, H. Zhang, and L. Chen. Traffic prediction in a bike-sharing system. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 33. ACM, 2015.
25. J. Liu, L. Sun, W. Chen, and H. Xiong. Rebalancing bike sharing systems: A multi-source data smart optimization. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1005–1014. ACM, 2016.
26. M. Lowalekar, P. Varakantham, S. Ghosh, S. D. JENA, and P. Jaillet. Online repositioning in bike sharing systems. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 2017.
27. E. O'Mahony. *Smarter Tools For (Citi) Bike Sharing*. PhD thesis, Cornell University, 2015.
28. E. O'Mahony and D. B. Shmoys. Data analysis and optimization for (citi) bike sharing. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
29. P. Papazek, C. Kloimüllner, B. Hu, and G. R. Raidl. Balancing bicycle sharing systems: an analysis of path relinking and recombination within a grasp hybrid. In *International Conference on Parallel Problem Solving from Nature*, pages 792–801. Springer, 2014.
30. P. Parikh and S. V. Ukkusuri. Estimation of optimal inventory levels at stations of a bicycle sharing system. 2014.
31. A. Paul, D. Freund, A. Ferber, D. B. Shmoys, and D. P. Williamson. Prize-collecting tsp with a budget constraint. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 87. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.
32. M. Rainer-Harbach, P. Papazek, B. Hu, and G. R. Raidl. Balancing bicycle sharing systems: A variable neighborhood search approach. In *European Conference on Evolutionary Computation in Combinatorial Optimization*, pages 121–132. Springer, 2013.

33. T. Raviv and O. Kolka. Optimal inventory management of a bike-sharing station. *IIE Transactions*, 45(10):1077–1093, 2013.
34. T. Raviv, M. Tzur, and I. A. Forma. Static repositioning in a bike-sharing system: models and solution approaches. *EURO Journal on Transportation and Logistics*, 2(3):187–229, 2013.
35. Regional Plan Association. Building transit-friendly communities a design and development strategy for the tri-state metropolitan region, 1997.
36. C. Riquelme, R. Johari, and B. Zhang. Online active linear regression via thresholding. In *AAAI*, pages 2506–2512, 2017.
37. C. Rudloff and B. Lackner. Modeling demand for bikesharing systems: neighboring stations as source for demand and reason for structural breaks. *Transportation Research Record: Journal of the Transportation Research Board*, pages 1–11, 2014.
38. S. M. Salaken, M. A. Hosen, A. Khosravi, and S. Nahavandi. Forecasting bike sharing demand using fuzzy inference mechanism. In *ICONIP 2015: Proceedings of the 22nd International Conference on Neural Information Processing*, pages 567–574. Springer, 2015.
39. R. M. Saltzman and R. M. Bradford. Simulating a more efficient bike sharing system. *Journal of Supply Chain and Operations Management*, 14 (2):36, 2016.
40. J. Schuijbroek, R. Hampshire, and W.-J. van Hoeve. Inventory rebalancing and vehicle routing in bike sharing systems. *Euorpean Journal of Operations Research*, to appear.
41. J. Shu, M. C. Chou, Q. Liu, C.-P. Teo, and I.-L. Wang. Models for effective deployment and redistribution of bicycles within public bicycle-sharing systems. *Operations Research*, 61(6):1346–1359, 2013.
42. D. Singhvi, S. Singhvi, P. I. Frazier, S. G. Henderson, E. O'Mahony, D. B. Shmoys, and D. B. Woodard. Predicting Bike Usage for New York City's Bike Sharing System. In *AAAI Workshop: Computational Sustainability*, 2015.
43. A. Singla, M. Santoni, G. Bartók, P. Mukerji, M. Meenen, and A. Krause. Incentivizing users for balancing bike sharing systems. In *AAAI*, pages 723–729, 2015.
44. W. Szeto, Y. Liu, and S. C. Ho. Chemical reaction optimization for solving a static bike repositioning problem. *Transportation research part D: transport and environment*, 47:104–135, 2016.
45. P. Vogel, B. A. N. Saavedra, and M. D. C. A hybrid metaheuristic to solve the resource allocation problem in bike sharing systems. *International Workshop on Hybrid Metaheuristics*, pages 16–29, 2014.
46. J. Zhang, X. Pan, M. Li, and S. Y. Philip. Bicycle-sharing system analysis and trip prediction. In *Mobile Data Management (MDM), 2016 17th IEEE International Conference on*, volume 1, pages 174–179. IEEE, 2016.
47. F. Zheng, P. He, E. Belavina, and K. Girotra. Customer preference and station network in the london bike share system. 2018.