Abstract

As bike-share systems expand in urban areas, the wealth of publicly available data has drawn researchers to the novel operational challenges these systems face. The key challenge is to meet user demand for available bikes and docks by rebalancing the system. This paper addresses the optimization problems that arise from these efforts. In particular, we provide new models to guide truck routing for overnight rebalancing and new optimization problems for non-motorized rebalancing during the day. Finally, we report on the practical impact our methods have had for Citi Bike in New York City.

Introduction

Bike-share systems have become more prevalent in major cities, offering a new sustainable mode of transportation for both tourists and everyday commuters. In providing connections between existing transportation modes, these systems allow 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 pervasive analysis of customer demand and system operations.

More so than with other modes of transportation, users have the ability to 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 it. 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. Most commonly, 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 holds at most five bikes and is towed by a cyclist to relocate bikes between stations. 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.

Deciding when and how to implement these rebalancing mechanisms can help improve utilization of the bike-share system. This paper addresses the optimization problems that arise in rebalancing and details the impact of our proposed methods on the New York City bike-share system. We first 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 how to optimally assign trikes to circulate between pairs of stations. Here, we use a maximum weight $k$-edge matching to maximize the impact on customer satisfaction. Lastly, we consider how to optimally place corrals at stations. For this problem, our goal is to minimize the number of customers who cannot find an open dock within a quarter mile of their preferred destination.

All of the methods in this paper are in different development stages for New York City Bikeshare (NYCBS). In particular, we have completed a trial run using our overnight rebalancing schedules, routing three out of their five trucks over an eight hour period. Our routes are able to reduce customer dissatisfaction an additional 20% on average. As of Spring 2016, NYCBS has also been using our placement of corrals. Further ahead, we have implemented our matching of trikes to stations and are in discussions with NYCBS on how to better fit the modelling to their needs. We are also in discussions with bike-share systems in other major American cities (e.g. D.C., Boston, Bay Area) about how these methods can make local transportation more sustainable.

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 function has been introduced in (Raviv and Kolka 2013), (Henderson, O’Mahony, and Shmoys 2016), and (Schuijbroek, Hampshire, and van Hoeve to appear). Our work is most similar to (Raviv, Tzur, and Forma 2013), as we also use this objective function to optimize
truck routes over a given time interval. The main distinctions of our work lie in the IP formulation and the pre-processing techniques, which allow us to solve larger instances (i.e., over larger time horizons and with more trucks and stations).

An orthogonal approach is found in (Jian and Henderson 2015), in which the authors use simulation optimization to find the optimal allocation of bikes across the system without taking into account the routing of trucks to achieve that allocation. In follow-up work, (Jian et al. 2016), they give justification to the treatment of stations as independent when computing the expected number of dissatisfied customers, as we assume here.

The overnight rebalancing literature mainly focuses on the so-called Static Bicycle Rebalancing Problem, which treats time as an objective, rather than as a constraint. This problem was introduced in (Benchimol et al. 2011), and versions of it are studied in (Chemla, Meunier, and Calvo 2013), (Dell’Amico et al. 2014), (Rainer-Harbach et al. 2013), and (Nair et al. 2013). In this variant, each station has a target number of bikes, and the goal is to route finite capacity trucks to pick up/drop off bikes to meet these targets while minimizing the route length. In practice, these routes may not actually be feasible since there is a concrete time limit on rebalancing given by the employee schedule. In contrast, our model helps operators optimize their impact over the fixed time they are allotted.

In contrast to the extensive work on overnight rebalancing, very little research has been conducted on non-motorized rebalancing efforts, which form a crucial part of NYCBS’s operations. One such work is (Henderson, O’Mahony, and Shmoys 2016), which studies routing 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 authors aim 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 is to minimize the expected number of dissatisfied customers. Finally, our work distinguishes itself in that it was conducted in close cooperation with NYCBS and already influences their operations. This stands in contrast to the findings of (de Chardon, Caruso, and Thomas 2016), which conclude that very little of the existing work on rebalancing has had an impact in practice.

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 2015)), 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 2014)). While perfect customer satisfaction is not practical or sustainable, NYCBS aims to use rebalancing to minimize expected customer dissatisfaction.

Prior work by (Schuijbroek, Hampshire, and van Hoeve to appear), (Raviv and Kolka 2013), and (O’Mahony 2015) defines a function that uses demand information to map an initial number of bikes at a station to the expected number of customers who will not be able to access/leave the system at that station. Based on these user dissatisfaction functions, the authors of (Raviv, Tzur, and Forma 2013) define a routing problem to optimize the truck routes relocating bikes in preparation for the morning rush hour. However, their solution methods are only applicable to at most three trucks and systems with at most 60 stations. In addition, their IP does not account for the time it takes to stop at each station. We expand upon this formulation and use pre-processing to handle Citi Bike’s rebalancing in Manhattan with 360 stations.

User dissatisfaction function

Our objective function is based on an \( \text{M/M}/1/\kappa \) 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. More precisely, there are two Poisson processes with rates \( \mu \) and \( \lambda \), \( \mu \) and \( \lambda \) 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 \( \mu \) (resp. \( \lambda \)) moves the state of the queue from \( i \) to \( i-1 \) (resp. \( i+1 \)) if \( i>0 \) (resp. \( i<\kappa \)). If \( i=0 \) (resp. \( \kappa \)), the state of the queue does not change. This represents a dissatisfied user who wants to pick up (drop off) a bike, but cannot due to a lack of available bikes (docks). The user dissatisfaction function \( F_s(\cdot) \) maps an initial number of bikes to the expected number of such dissatisfied users over a given finite time horizon. In prior work, (O’Mahony 2015), (Raviv and Kolka 2013), and (Schuijbroek, Hampshire, and van Hoeve to appear), mentioned above, it was shown that \( F_s(\cdot) \) is convex and can be efficiently computed. To obtain values for \( \lambda \) and \( \mu \) we use the decensoring method introduced in (O’Mahony and Shmoys 2015).

Integer Program Formulation

In this section, we introduce our integer program. The formulation will be 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 \( \ell \) time steps, we add a path of \( \ell - 1 \) stations between them. This allows us to move between these two stations in exactly \( \ell \) time steps while traversing one edge in each time step. This technique is the main difference between our IP and previous ones. The advantage of using this method is that it allows us to reduce the dimension of the IP.

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 from which a truck can move to \( s \) in a single time step.
- \( \gamma \) 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 1).
- \( S_+ \) is the set of stations \( s \) for which \( \text{start}(s) > \min(s) \).  
- \( S_- \) 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.

\[
\begin{align*}
\text{maximize} & \quad \sum_{s \in S, k \in [K]} (y_{stk} - y_{sTk}) c_s \\
\text{subject to:} & \quad x_{stk} \leq x_{s(t-1)k} + \sum_{s' \in N(s)} x_{s't(t-1)k} \quad \forall s, t, k \\
\sum_{s \in S} x_{stk} = 1 & \quad \forall t, k \\
\sum_{k \in [K]} y_{sk} = \text{start}(s) & \quad \forall s \\
\text{start}(s) \leq \sum_{k \in [K]} y_{stk} \leq \min(s) & \quad \forall s \in S_-, t \\
\min(s) \leq \sum_{k \in [K]} y_{stk} \leq \text{start}(s) & \quad \forall s \in S_+, t \\
\sum_{s \in S} y_{stk} + b_{tk} = \sum_{s \in S} y_{sk} + b_{1k} & \quad \forall t, k \\
|y_{stk} - y_{s(t-1)k}| + \gamma |x_{stk} - x_{s(t-1)k}| & \leq \gamma \quad \forall s, t, k
\end{align*}
\]

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.
- **Constraint (1)** allows each truck to only move to an adjacent station.
- **Constraint (2)** indicates that at each time step, each truck must be in exactly one station.
- **Constraint (3)** initiates the number of bikes at each station.
- **Constraints (4)** and (5) guarantee that the number of bikes in each station is between \( \min(s) \) and the minimizer \( \min(s) \). In other words, we enforce that moving a bike only improves the setup.
- **Constraint (6)** makes certain 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 \( \gamma \).
- **Constraint (8)** forces the truck to either move or pick/drop bikes in one time step. In most of the previous works, authors have omitted this constraint. This constraint makes the IP significantly harder to solve, but makes the resulting path more viable.

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. Due to the high dimensionality of this IP, we use several methods to make it solvable in a given time window.

**Solution Methods** The following heuristic methods help decrease the computation time to solve our 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.

**Greedily selecting stations.** To further reduce the size of the IP, we remove 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 bikes, and run the IP with between 30 and 40 stations.

**Dividing \( T \) into smaller time intervals.** At the start of rebalancing, the operator may not have much time between receiving the current states of each station and when trucks must begin their routes. To gain computation time, we break \( T \) into smaller intervals and only route trucks for the first interval in the beginning. While this part of the route is being implemented, 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.
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, etc. 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.

Dynamically updating stations. After solving the IP for a single truck and time interval, we update the number of bikes available at each station and accordingly update the stations we consider using the greedy methods above. This ensures that we will focus on the most useful stations in the next interval, while keeping the dimensionality of the IP low.

The routes we construct for each truck and time interval will be compatible in that they can be pieced together to form one coherent route.

Fairness Constraints and Optimal Fleet Size. The fairness constraints (4) and (5) should be interpreted as a commandment that rebalancing should never increase the number of dissatisfied customers at a station. We enforced this constraint to avoid station bias. Unlike (Raviv, Tzur, and Forma 2013), we find that at many stations the user dissatisfaction functions are far from flat around the minimum. As such, the fairness constraints are needed to avoid one station being sacrificed for another. In a system in which the number of bikes in the system is equal to station demand, i.e. $\sum_s \min(s) = \sum_s \text{start}(s)$, the fairness constraints would not heavily affect the solution. However, in New York City, we found that often $\sum_s \text{start}(s) << \sum_s \min(s)$. In some cases, this led constraint (5), rather than the time constraint, to be the most limiting constraint. Noticeably, these cases also had smaller integrality gaps. That is, the solutions produced are close in value to the upper bound given by a linear relaxation of our IP.

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. Second, we compare our solutions to the current routes used by NYCBS. On average, our solutions reduce customer dissatisfaction by 20% compared to a previous computationally informed approach.

<table>
<thead>
<tr>
<th># of Trucks</th>
<th>Avg Objective Function</th>
<th>Avg Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>148</td>
<td>28.1%</td>
</tr>
<tr>
<td>2</td>
<td>232</td>
<td>22.1%</td>
</tr>
<tr>
<td>3</td>
<td>301</td>
<td>12.3%</td>
</tr>
<tr>
<td>4</td>
<td>330</td>
<td>9.5%</td>
</tr>
</tbody>
</table>

Figure 2: Results for Manhattan

IP gap. 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 has around 360 stations. As inputs for $\text{start}(s)$, we used the number of bikes at each station at midnight over the course of a week. In Figure 2, 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.

The average integrality gap decreases as the number of trucks grows. This seems to be a consequence of the fairness constraints, as there are fewer bikes available to be moved (per truck). With more trucks, it is easier to achieve the best possible solution.

While similar IPs have been solved in (Raviv, Tzur, and Forma 2013), (Ho and Szeto 2014), and (Forma, Raviv, and Tzur 2015) – including some with smaller reported integrality gaps – these formulations make simplifying assumptions about the time for picking up/dropping off bikes and are tested on smaller instances. Thus, it is difficult to compare the results directly.

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 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 created by the dispatchers at NYCBS. On average our results showed objectives about 20% higher. Together with the NYCBS management, we reviewed our proposed routes and are now running pilots to route between 2

Figure 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.
and 5 of their trucks overnight.

**Midrush Rebalancing**

**Trikes**

In this section, we formally define our model for measuring the impact of adding $m$ trikes running between stations on the expected number of dissatisfied users. Assuming that every station may have only one trike route incident to it, the problem of finding the optimal $m$ trike routes can be formulated as a bipartite maximum $m$-edge matching. The weight of an edge between two stations will be the reduction in user dissatisfaction at those stations when a trike route is added between the two.

**Model** Similar to the user dissatisfaction function, 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 $X_s(t)$ to represent the number of bikes at station $s$ at time $t$, where $k_s$ is the capacity of station $s$. As before, a dissatisfied customer at station $s$ 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, we also have a trike with capacity $k_R$ that moves bikes 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 $t_1$ is at $A$ if $i$ is odd and at $B$ otherwise. When the trike arrives at station $A$ ($B$) it picks up (drops off) as many bikes as possible given the number of bikes at $A$ (available docks at $B$) and the number of bikes already in the trike. We use the random variable $X_{R}(t) \in \{0, \ldots, k_R\}$ 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 $t_{r+1}$. Given the Poisson process rates, we can compute the expected number of dissatisfied customers $F_s(\cdot)$ for $s \in \{A, B\}$ in each 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, for all $s \in \{A,B\}$, for $x, y \in \{0, \ldots, k_s\}$ we let

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

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 started at the station.

We can write the expected number of dissatisfied users at stations $A$ and $B$ as follows:

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

Thus, we need to find $\Pr(X_s(t_i^+) = j)$ for all $s, i, j$. 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 $X_R(t_i^+) = 0$ with probability one and assuming that $X_A(t_0), X_B(t_0)$ are independent we obtain:

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

Then, 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

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

where we define $P_{x,y}^{i,j} = 0$ for $y \notin \{0, \ldots, k_A\}$. This is because, with $X_R(t_i^+) = z$, the event that $\{X_A(t_i^+) = \alpha \text{ and } X_B(t_i^+) = \beta\}$ 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,

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

This allows us to compute $F_{A \rightarrow B}$ since

$$\Pr(X_A(t_i^+) = j) = \sum_{y=0}^{k_B} \sum_{z=0}^{\rho} \pi^i(j, y, z),$$

$$\Pr(X_B(t_i^+) = j) = \sum_{x=0}^{k_A} \sum_{z=0}^{\rho} \pi^i(x, j, z).$$

**Results** 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 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, $F_{A \rightarrow B}$. 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), the weight of the maximum matching $M^*$ is given in Figure 4 for various sizes of $M^*$. The maximum matching with eight trikes is shown in Figure 5.

| $|M^*|$ | $w(M^*)$ |
|-------|-----------|
| 1     | 23.8      |
| 5     | 109.5     |
| 8     | 165.5     |
| 10    | 196.7     |
| 20    | 321.8     |

Figure 4: Objective with $|M^*|$ trikes.

**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 1997), 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, (Kabra, Belavina, and Girotra 2015) shows that shorter distances to available stations correlates with increased demand for bike-share systems. Based on these findings, we say a station has a shortage if no station within a quarter mile has at least 15% of its docks available. Our shortage measure for the system will then be 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 \( s \), 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 \( T \), we define the shortage measure as

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

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 station \( s \), the amount of time stations in \( N(s) \) are in shortage is significantly reduced. Given a budget \( B \), we thus aim to place \( B \) corrals to minimize the shortage measure. For a set of past time points in \( T' \), let

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

It is then natural to model corral placement as a maximum coverage problem via the following IP:

\[
\begin{align*}
\text{maximize} & \quad \sum_{s \in S} w_s x_s \\
\text{subject to:} & \quad x_s \leq \sum_{s' \in N(s)} y_{s'} \quad \forall s \\
& \quad \sum_{s \in S} y_s \leq B \\
& \quad x_s, y_s \in \{0, 1\} \quad \forall s.
\end{align*}
\]

In the IP, \( y_s \) is one if a corral is placed at station \( s \) and \( x_s \) is one if there is a corral within a quarter mile from station \( s \).

**Results**

Six corrals identified by the maximum coverage formulation (see Figure 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) and find a 31.6% reduction in the shortage measure. Further, to show that the improvement is indeed due to the effects of the corrals, we repeat this analysis excluding the stations with corrals from \( S \). Here, we only find a 14.6% improvement. Results of these analyses are in Figure 7, where \(|S| = 277\) and \(|T| = 12000\).

**Conclusion**

We provide models for three different problems that arise in rebalancing bike-share systems. For overnight rebalancing and trailer routing, we focus on minimizing the expected number of dissatisfied users. First, we present our IP for the overnight rebalancing problem. This formulation, along with our pre-processing techniques, allows us to model New York City’s 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, the expected number of dissatisfied users, for both problems, 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. Since combining these results would likely lead to intractable optimization problems, a promising direction is to incorporate rebalancing into simulation frameworks.
References


Raviv, T.; Tzur, M.; and Forma, I. A. 2013. Static repositioning in a bike-sharing system: models and solution ap-