Bike Angels: An Analysis of Citi Bike's Incentive Program

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

ABSTRACT

Bike-sharing systems provide a sustainable and affordable transportation alternative in many American cities. However, they also face intricate challenges due to imbalance, caused by asymmetric traffic demand. That imbalance often-times leads to bike-sharing stations being empty (full), causing out-of-stock events for customers that want to rent (return) bikes at such stations. In recent years, the study of data-driven methods to help support the operation of such system, has developed as a popular research area.

In this paper, we study the impact of *Bike Angels*, an incentive program New York City's Citi Bike system set up in 2015 to crowdsource some of its operational challenges related to imbalance. We develop a performance metric for both online- and offline-policies to set incentives within the system; our results indicate that though Citi Bike's original offline policy performed well in a regime in which incentives given to customers are not associated to costs, there is ample space for improvement when the costs of the incentives are taken into consideration. Motivated by these findings, we develop several online- and offline- policies to investigate the trade-offs between real-time and offline decision-making; one of our online policies has since been adopted by Citi Bike.

Author Keywords

Sustainable Transportation, Crowdsourcing, Bikesharing, Incentives, Inventory Management

INTRODUCTION

Bike-sharing systems have become ubiquitous in many American cities. The largest of these systems consist of a number of stations placed densely within a city; each station consists of a number of docks that each either hold a bike (are full) or are empty. Users can rent a bike from any station with at least one bike and return bikes at any station with an empty dock. With more than 30 million annual rides nationwide (2017), about half of which occurred in New York City's Citi Bike system, these systems have become a fixture of urban life, providing commuters and tourists with a sustainable transportation option.

The major operational challenge faced by these systems is to avoid out-of-stock events: as each station has a finite number of docks (*capacity*) at which users can rent or return bikes, users are unable to enter the system when trying to rent bikes at stations at which all docks are empty, that is, when no dock holds a bike; arguably even worse, customers are not able to leave the system when they try to return bikes at stations at which all docks are full. In the past few years, researchers and system operators alike have developed tools to tackle such out-of-stock events and their underlying cause: *imbalance*. This is often referred to as *rebalancing*: moving bikes within the system to reduce the number of out-of-stock events. Rebalancing can be of various modes: for example, systems operate box trucks, pedicab trailers, and valet services to reduce out-of-stock events. In this paper, we describe our work with the operators of Citi Bike in setting up *Bike Angels*, a crowdsourcing approach to rebalancing, and analyze the relative effectiveness of a variety of natural incentive schemes for their system.

The Bike Angels incentive scheme was first set up in October 2015. In the initial set-up, we identified pairs of stations that (i) were close to each other, and (ii) had asymmetric demand patterns: we chose pairs of stations that were nearby to each other but over a certain time interval had the property that one station (A) overwhelmingly had bikes returned whereas the other (B) overwhelmingly had bikes rented. We then invited users that often used one of the two and asked them to switch to the other, i.e., we asked a small number of customers likely to return bikes at A in such intervals to become *Bike Angels* by instead returning bikes at B. Of course, in such a scheme, incentives can only be used for a tiny fraction of rides within the system.

Nevertheless, the pilot showed that customer behavior could be significantly affected through gamification [**?**], and also posed questions about how to optimally design such an incentive scheme, in particular with respect to when and where to incentivize. A second version of Bike Angels, at a much larger scale, statically incentivized a sizeable fraction of stations within the system either to enourage rentals or returns throughout each rush hour. This program was viewed as a great success, in spite of the fact that the pre-determined incentives made no use of the vast amounts of historic and real-time data available. At worst, this sometimes led to incentives encouraging customers to rent (return) bikes at empty (full) stations.

In this paper, we evaluate the efficiency of this incentive scheme with regard to its goal to reduce out-of-stock events.

CHI'16, May 07–12, 2016, San Jose, CA, USA

ISBN 123-4567-24-567/08/06...$15.00

DOI: http://dx.doi.org/10.475/123_4

We apply so-called *user dissatisfaction functions* to evaluate, for every trip rewarded by the program, an estimate of its reduction in future out-of-stock events. Further, we design a range of different policies, dictating the times at which each station is incentivized. The design of these policies involves trade-offs between their *efficiency* and *simplicity*. Below, we explain what we mean by those two terms:

Efficiency. The main goal of *Bike Angels* is to help rebalance the system, i.e., to reduce the number of out-of-stock events. We explain in subsequent sections how the user dissatisfaction functions introduced by Raviv and Kolka [?] can be used to evaluate the impact of each individual rental/return on the expected number of future out-of-stock events. Combining these with a cost for each point awarded in the incentive scheme, we obtain a score for each incentivized rental/return that can be interpreted as an offline evaluation of the efficiency of the incentive. In that sense, a perfectly efficient scheme would incentivize exactly those trips that net a positive score when accounting for both the impact on out-of-stock events and the cost of incentives.

Simplicity. Perfect efficiency, with respect to the score described above, is attainable in such an incentive scheme, but it requires the operator to decide in real-time whether or not to incentivize at each location. This is undesirable from the users' perspective as it reduces predictability on where incentives will be given in the future, i.e., when customers prepare their trip, they do not know whether or not their origin and, even more so, their destination has rentals and respectively, returns, incentivized. This raises the bar for participation. Further, from an operator's perspective, the IT set-up for a dynamic scheme requires more maintenance than, say, the completely static version.

We show that even though there exist natural trade-offs between the two, there are a variety of incentive policies that span the continuum from maximally efficient and maximally dynamic to less efficient and entirely static. Our results employ a data-driven methodology to design such policies, one of which is now in place at Citi Bike.

Related Work

Our analysis is a novel application of the user dissatisfaction functions defined by [?] which provide a mathematical framework to characterize the expected number of out-of-stock events in the system over a specified planning period. Different ways to compute these user dissastisfaction functions were subsequently suggested [?, ?, ?]. Each of these relies on demand estimates at each station, which we obtain through the decensoring method suggested in [?]. Within the literature on operations in bikesharing systems, these user dissatisfaction functions have usually been used in routing problems, see [?, ?, ?, ?, ?] among others. Additionally, they have also been successfully applied in the context of system design, that is, to analyse the number of docks that should be placed at every location within a system [?, ?].

An incentive scheme like the one we study here was first suggested for bikehsaring schemes by [?]; in other transportation settings, a similar scheme was succesfully implemented by [?]. In contrast, while the work by [?] also studies an incentive scheme for bikesharing, that work focuses on how much to incentivize to ensure people act upon it, rather than on the effects of incentivized trips on the balance of the system.

In the queueing literature, on-demand transportation systems have been investigated under steady-state regimes with approaches to incentives being studied by [?, ?]. This also has relations to pricing questions in ridesharing systems as studied by [?, ?, ?]. The steady-state assumption, however, is difficult to justify in bikesharing systems, given that (i) most of the traffic occurs during rush hours and (ii) the configuration of bikes within the system vastly differs at the beginning and end of each rush hour. Despite the steady-state assumption being hard to justify, it is worth highlighting a parallel to [?] who study the trade-offs between dynamic and static pricing policies in ride-sharing systems. Similar such trade-offs were more recently studied by [?]. Though these works consider very different models, they consider the same trade-offs as we do in the context of modern transportation systems.

The Incentive Scheme

Before outlining the structure of our exposition, it is worthwhile to give a high-level description of the *Bike Angel* program.

Bike angels accrue points by doing rides that benefit the balance of the system. While the way in which stations are chosen for incentives has changed over time, the following accounting has been the guiding principle since the original pilot ended: the program awards 1 point for a trip from an incentivized rental station to a neutral station or from a neutral station to an incentivized return station and 2 points for a trip from an incentivized rental to an incentivized return station. The exact reward structure of the incentive scheme has also changed over time, yet the basic idea that more points translate into higher rewards has remained the same. For example, in February 2018, the program rewarded users as follows [?]:

- For 10 points, bike angels receive a 24-hour day pass
- For every 20 points, up to 80, bike angels receive a membership extension of 1 week
- For every 10 points above 80, bike angels receive $1 as a gift card
- The 5 angels with the most points receive gift cards worth $100, $75, $50, $25, and $25 respectively.

It is worthwhile to mention that bike angels occasionally get bonuses for beneficial trips on days with special conditions (e.g., due to weather); however, since our analysis mostly focuses on the choice of stations/times to incentivize, we ignore such bonuses.

We present our results as follows: in the next section we describe the data-sets used in our analysis and formally define the models developed to evaluate the incentive scheme. Thereafter, we define the various policies before giving a detailed comparison of the different policies, analyzing the inherent trade-offs between simplicity and efficiency. Finally, we conclude by reporting on the changes Citi Bike implemented for the Bike Angel program based on our analysis.

DATA ANALYSIS AND DEFINITIONS

In this section, we describe the data-sets at our disposal, the system parameters we derive from them, and the models by which we evaluate the different policies for the incentive scheme.

Data-Sets

Our analysis relies on three different data-sets. First, we have a list of all trips in the system with origin location, destination location, and respective times of the start and the end of the trip [?]. Second, we have a list that indicates which of the above trips were rewarded with points by the existing offline incentive policy (as well as whether the point was awarded for the rental, the return, or both) [?]. Last, for each minute and each station in the system, we have the number of bikes reported to be at the station at that time as well as the total number of docks available [?].

We use data collected over three different time periods in 2016 in our analysis: 04/01–05/13 (base), 10/03–10/30 (training) and 10/31–12/14 (testing). The base period includes the most recent 6-weeks of data when no incentive program was in use, and represents the system before incentives were introduced. The training period involves the first 20 days of the winter data after incentives were introduced, and the remainder is used as the testing period. The base and training periods are used to help calculate the system parameters, such as the rental/return rates and the stochastic interpretation of the incentives' effect explained later in this section, and also serve as the training period for some of our incentive policy algorithms. The testing period is used to evaluate and compare our suite of policies.

Data Censoring

In our data, censoring proves to be a challenge as the data does not accurately reflect the true demand within the system. For example, a rider wanting to rent or return a bike to a station, with all docks empty or full respectively, cannot do so; thus, the corresponding demand would not be reflected in our data sets. To account for this problem of demand-censoring, when estimating average rental/return demand, we follow the methodology of [?] and examine station behavior on a minute-by-minute basis, only considering "active rental/return minutes", that is, minutes when a station has non-empty or non-full docks, respectively.

Arrival and Departure Rates

Our analysis relies on three types of arrival and departure rates: regular-angel, incentive-angel, and non-angel rates. Regular-angel rates are the natural rental/return rates into a station for all angel riders combined, when no incentive is provided. The incentive-angel rates are the rates for the same set of customers when incentives are given. The non-angel rates are the rates for all other customers and are assumed to be unaffected by the incentive program. In our calculation of rates, we follow the work by [?] in assuming that the rates are independent across stations and calculate rates for every half-hour interval of the day starting from 12AM in the units bikes per minute.

To calculate the incentive-angel and non-angel rates, we take the data for a station's previous q weekdays, and divide the total number of all arrival/departure trips by the specified riders by the total "active rental/return minutes" for each half-hour interval. In the results we present, q is set to 20.

Formally, for each day d, station s, and time index t, let ($m^-_{d,s,t}$, $m^+_{d,s,t}$) denote the active rental/return minutes, ($r^{a,+}_{d,s,t}$,$r^{a,-}_{d,s,t}$), ($r^{n,+}_{d,s,t}$, $r^{n,-}_{d,s,t}$) the number of arrival/departure trips for angel and non-angel riders respectively, and $D(d)$ be the set of the preceding q weekdays before day d. Then the incentive-angel arrival rates $\lambda^{i,+}_{d,s,t}$ for day d, station s and time index t, where $t \in \{0,1,2,...,47\}$ for the 48 half-hour intervals of the day (e.g. t=12 is 6 a.m.) are calculated as the rate per minute normalized to account for only active minutes, as follows (similar for departure rates $\lambda^{i,-}_{d,s,t}$ and non-angel arrival/departure rates $\lambda^{n,+}_{d,s,t}$ and $\lambda^{n,-}_{d,s,t}$):

$$\lambda^{i,+}_{d,s,t} = \frac{\sum_{d' \in D(d)} r^{a,+}_{d',s,t}}{\sum_{d' \in D(d)} m^+_{d',s,t}} \tag{1}$$

To estimate the regular-angel rates we used data from the base period, when no incentive program was offered. The regular-angel rates are assumed to be identical across days, and are calculated by dividing the angel rider's total number of all arrival/departure trips by the total "active minutes" in the base period for each half-hour interval, and applying a correction factor. The correction factor, U, accounts for any general macro-change in system usage from the base time period to our testing time period. The correction factor is calculated by dividing the average daily total number of trips by non-angel riders during the training period by their average daily total number of trips during the base time period. Finally, because the correction factor is an estimate of the change, we capped the regular-angel rates to be at most the incentive-angel rates to reflect our prior beliefs that incentivization actually helps our system. If we define D to be the set of weekdays in the base time period and U to be the correction factor, the regular-angel arrival rates are calculated as follows (similarly for departure rates):

$$\lambda^{r,+}_{d,s,t} = min(U \cdot \frac{\sum_{d' \in D} r^{a,+}_{d',s,t}}{\sum_{d' \in D} m^+_{d',s,t}}, \lambda^{i,+}_{d,s,t}) \tag{2}$$

Stochastic View of Incentives

Our analysis also relies on the probability that an incentivized rental/return would not have occurred if it had not been for the incentive; in particular, the rebalancing due to a rental/return that happens regardless of the incentives should not be attributed to the incentive scheme. We compute these probabilities for each day, station and half-hour interval by dividing the difference of the incentive-angel and regular-angel rates by the incentive-angel rate. Formally, with $p^+_{d,s,t}$ and $p^-_{d,s,t}$ denoting the probabilities that a rental/return was triggered by the incentives, we compute these probabilities as

$$p^+_{d,s,t} = \frac{\lambda^{i,+}_{d,s,t} - \lambda^{r,+}_{d,s,t}}{\lambda^{i,+}_{d,s,t}} \tag{3}$$

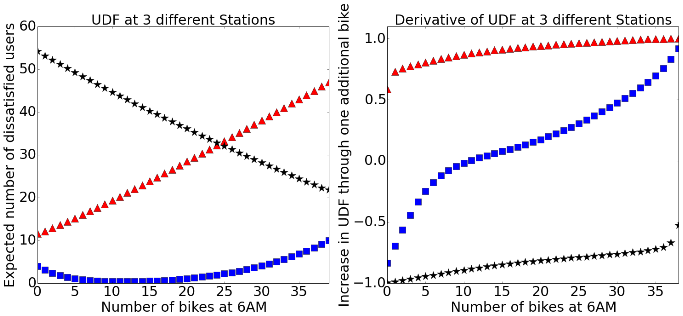

Figure 1: User dissatisfaction function $\phi_{s,t}(\cdot)$ and its discrete derivative for 3 different stations s with the same capacity.

User Dissatisfaction Function

The user dissatisfaction functions defined by [**?**] map the number of bikes at a station at a given time to the expected number of out-of-stock events over the course of a subsequent finite time-horizon. Formally, the functions assume that two exogeneous stochastic processes create arrivals of customers intending to return and rent bikes at the station. In our implementation, the stochastic processes are given by time-varying Poisson processes with rates being the non-angel rates. Crucially, the processes are assumed to be independent of the initial condition, that is, of the number of bikes at the station at the beginning of the time horizon. Then, each intended rental decreases the number of bikes in the station by 1 if a bike is available, i.e., if the number of bikes is at least 1. Intended returns increase the number of bikes by 1 if an empty dock is available at the station, i.e., if the number of bikes is less than the capacity of the station.

Customers experience out-of-stock events if they intend a rental and the station is empty or if they intend a return and the station is full. Given the non-angel rental and return rates, one can then compute for a given station s with ℓ bikes at time-index t, the expected number of out-of-stock events over a time-horizon beginning at time-index t, i.e., the expected number of intended returns when the station is at capacity plus the expected number of intended rentals when the station is empty. We denote this expectation by $\phi_{s,t}(\ell)$.

Performance Metrics and Policies

For each incentivized return, we can estimate the reduction in out-of-stock events by computing the discrete derivative of the user dissatisfaction function with respect to one additional bike, evaluated as a function of the number of bikes present in the station before the arrival. Similarly, for incentivized rentals, the estimated reduction in out-of-stock events equals the difference in the value of the user dissatisfaction function evaluated with one bike less and the actual number of bikes in the station. In Figure **??**, we display the respective user dissatisfaction functions and the value of the derivatives for three different stations at 6AM. Evaluating the user dissatisfaction functions for every half-hour interval, we estimate the impact of each rental/return on future out-of-stock events by evaluating the derivative of the user dissatisfaction function for that time-interval at the number of bikes present at the time of the return (and similarly for rentals). If we let δ_r denote the estimated reduction on future out-of-stock events for trip r, let ℓ denote the bike-level at the station before the trip completed, and let s and t denote the station and time-index associated with the trip, we find that we may compute δ_r for a return as follows (similarly for rentals):

$$\delta_r = \phi_{s,t}(\ell) - \phi_{s,t}(\ell+1) \tag{4}$$

Though δ_r captures the impact of a return (rental) on the estimated number of out-of-stock events, it lacks two important aspects for our analysis. First, it lacks the cost associated with the incentive itself. To capture the operator's cost of the incentive scheme in the form of electronic gift cards, membership extensions, and other rewards (cf. [**?**]), we include a constant cost parameter β for every point awarded by an incentive scheme. Second, for every incentivized rental (return), we can incorporate the probability $p_{d,s,t}$ that the rental (return) would not have occurred without an incentive given; here, d,s,t is the date, station and time index respectively in which the rental (return) occurred. This gives a stochastic perspective of the causal effects of incentives, in contrast to the deterministic assumption that incentivized rentals (returns) are always caused by the incentive. Using the above, we can define for every single incentivized return (rental) the performance of incentivizing it with both the deterministic and the stochastic perspective as

$$\Delta_r = \begin{cases} \delta_r - \beta & \text{if deterministic} \\ \delta_r \cdot p_{d,s,t} - \beta & \text{if stochastic} \end{cases} \tag{5}$$

Given Δ_r as a performance measure for incentivizing a single return (rental) r, we define the performance of a policy as the sum of all Δ_r corresponding to incentivized rentals (returns) r. Formally, denoting by τ the set of all rentals (returns) in the testing period, we define a policy P as a function $P : \tau \to \{0,1\}$ identifying for each rental (return) r whether or not P incentivizes it. Then, to compute the overall performance $H(P)$ of a policy P we compute

$$H(P) = \sum_{r \in \tau} P(r) \cdot \Delta_r \tag{6}$$

POLICIES

In this section, we define a number of incentive policies and identify the fundamental differences between offline and online policies. During the time periods from which our data is drawn, two 6-hour time periods were incentivized in the Citi Bike system: 6AM-12PM and 4PM-10PM – we refer to them as the AM and PM periods, respectively. Since all of the incentivized trip data comes from only these two periods (cf. Figure **??**), we test/apply our policies only for these periods. Furthermore, each policy treats the two time periods independently, and calculates an incentive scheme for each time period and station. From the figure, we see a generally equal distribution of rental/returns in the PM period, but a return-skewed distribution in the AM period.

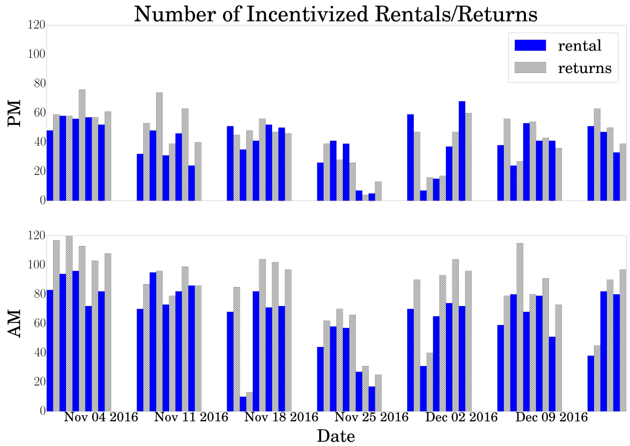

Figure 2: Total number of incentivized rentals/returns in the test period.

Offline Policies

We first describe the offline policies. *Offline* policies are characterized by the fact that the decision, for each AM/PM period, as to which interval will be incentivized for each station is made (irrevocably) at the beginning of that time interval (6AM and 4PM for AM and PM, respectively); in doing so, these policies have access to past data and the number of bikes at that time, i.e., 6AM and 4PM, respectively. (Note that this use of term offline is different from its use in other contexts, where offline often means that the algorithm has access to *all* of the data for the input to be optimized before needing to commit to any decision.)

Static

The Static model is the simplest offline policy and was also the policy employed in Citi Bike at the time that this research project initiated. This policy determines a subset of stations to incentivize for which the entire AM period is incentivized on every day, and similar subset is determined for the PM period. This model serves as a baseline against which all other policies are compared.

Static Hindsight

The Static Hindsight model is an offline policy that chooses, for each station, the optimal continuous incentive period when looking back in hindsight for the past q' weekdays. More precisely, the model assumes an incentive interval can start/stop every 30-minutes, starting from the beginning of the time period (e.g., 06:00 – 07:30 or 07:00 – 12:00 for the AM period), and chooses the incentive interval that achieves the highest performance when constrained on the past q' weekdays of trip data. Throughout, we will use the term incentive interval to refer to such a continuous time period with these discrete start/end times. In our results, q' is set to be 10. Thus, we let $D(d)$ denote the set of dates which are q' days in hindsight; we let $\tau(D(d),s)$ denote the possible morning incentive trips belonging to those dates and station s, and let $I_{d,s}$ to be the incentive interval for station s on day d. Furthermore, let us say trip $r \in I$ if and only if the trip occurred during the interval I. The interval $I^\star_{d,s}$ chosen by Static Hindsight for the morning period is

$$\arg\max_{I} \sum_{r\in\tau(D(d),s)} \Delta_r \cdot 1_{r\in I}, \qquad (7)$$

where the notation $1_{r\in I}$ means the indicator function that is equal to 1 if $r \in I$, and is 0, otherwise.

Cluster Hindsight

This offline policy uses a clustering of the stations to help make incentive decisions. As we describe in more detail below, the model first groups the stations by station dock-capacity, clusters each group by bike-level behavior, and finally chooses the optimal incentive interval for each cluster when looking back in hindsight (similar to Static Hindsight).

We compute groups of stations by sorting them from smallest to largest capacity, and dividing them into g roughly equal-size groups. The intuition for the grouping is our prior belief that station activity level is highly correlated with station capacity size and it is unwise, for example, to compare stations with bike-capacity 5 with stations of bike-capacity 60. The actual data-points used to cluster are 12-dimensional vectors consisting of the 12 half-hour interval bike-level percentages (bike-level divided by station dock capacity) for a station and date. The data points were obtained from our training data period, and bike-level percentages, rather than absolute values, were used to normalize the data points. To cluster each group of stations, we run the k-means clustering algorithm (k-centroids) with the objective of minimizing the distortion, which is defined to be the sum of the squared Euclidean distances between each vector and its centroid. Finally, our model uses a hard labeling system to associate each station with a single cluster (since for each station we have multiple points corresponding to distinct dates), and labels a station as belonging to the cluster for which its data points are most prevalent.

To calculate the actual incentive intervals for each cluster, the model finds the optimal single, continuous incentive interval when looking back in hindsight of q' weekdays for the stations in the cluster. Then let us define $C(s)$ to be the cluster of stations which s belongs to. The interval that the model chooses is as follows:

$$I^\star_{d,s} = \arg\max_{I} \sum_{s'\in C(s)} \sum_{r\in\tau(D(d),s')} \Delta_r \cdot 1_{r\in I} \qquad (8)$$

The number of groups and the number of clusters are hyperparameters of the model. To fit these parameters and avoid high bias/variance problems, k-fold cross-validation was used with our training data, and the parameters with the best average scores were chosen. Due to the temporal aspect of the data (e.g., weather plays a big role in system behavior) the k-folds were also divided to maintain the temporal order. The final parameters used in our results are $g = 10$, $k = 3$ and $q' = 10$.

Fluid Model

A natural way to use the incentive-angel and non-angel rates to define an offline policy is by defining a so-called fluid model, in which it is assumed that exactly the expected number of rentals and returns occurs continuously per unit of time.

Within such a model, it is easy to find the interval during which incentivizing minimizes the (fluid) number of out-of-stock events. In fact, one can show that under mild assumptions on the data, it is guaranteed that incentivizing over a single interval is optimal in such a model.

Though fluid models have successfully been applied to operational questions in bike-sharing, e.g., in [**?**], we found in our analysis that the fluid model was vastly dominated by all other models; thus, we omit both its formal definition here and its performance in the results section.

Offline Benchmark

Offline policies are constrained in two ways: first, by the fact that we allow them to incentivize only during subintervals, as opposed to incentivizing, say, from 6-8AM and then again from 9-10AM but not in between; second, since the decision is based solely on the number of bikes at each station at 6AM, they lack information about demand later in the course of the day. To distinguish between the lack of information and the constraint of incentivizing only during a subinterval, we define the following *static optimal* benchmark that has access to full information, but is constrained to incentivize only during a subinterval.

Static Optimal

The static optimal model is not a feasible policy since it assumes future knowledge that is not given in practice; instead, it is a model built in hindsight to benchmark offline policies. More specifically, for each week, station and AM/PM period, the Static Optimal model considers in hindsight, with complete knowledge, the best possible single continuous incentive period in which to incentivize.

Online Policies

In contrast to the offline policies described above, online policies gain information over the course of the day and thus are able to adopt whether or not to incentivize at a given station at a given time. We define only a very simple set of online policies.

Dynamic

The Dynamic model is a completely online policy that chooses in real-time for each trip whether or not it is incentivized to guarantee perfect efficiency. Formally, the model incentivizes all trips with $\Delta_r > 0$ (cf. Equation 5), i.e., the performance is exactly the sum of all trips that contribute positively to the objective. This policy serves as an upper-bound on what any policy can possibly achieve within our analysis.

Dynamic CC (X)

Beyond the fully dynamic model, we define a parameterized family of policies, wherein each policy breaks up the incentive period into intervals and the parameter X dictates the size of the intervals. For a given X, Dynamic CC starts at the beginning of each AM/PM period and decides every X minutes whether or not to incentivize trips for the next X minutes. To decide whether to incentivize the next X minutes, the model simulates the occurrence of one incentivized trip with the current number of bikes and empty docks at the station. If the simulated value of Δ_r is positive, Dynamic CC chooses to incentivize trips at the station for the next X minutes. For example, the Dynamic CC 15 model at 6 AM will check the station's current number of available bikes and docks, and simulate an incentivized trip for that user dissatisfaction function and that number of bikes and docks available. If the Δ_r value for this trip is positive, it will choose to incentivize the station from 06:00–06:15. Then, at 06:15 it simulates a new trip, based on updated information, to determine whether or not to incentivize from 06:15–06:30.

Table 1: Policy Performance Scores in Deterministic Regime.

Time	Policies	Cost Parameter				
		0.0	0.1	0.2	0.3	0.4
AM	Dynamic	1.0	1.0	1.0	1.0	1.0
	Stat. Opt.	0.985	0.981	0.976	0.97	0.961
	Static	0.939	0.911	0.870	0.809	0.715
	Stat. HS	0.960	0.947	0.929	0.905	0.873
	Clus. HS	0.961	0.944	0.918	0.887	0.845
	Dyn 15	1.0	1.0	1.0	1.0	1.0
	Dyn 30	0.995	0.994	0.993	0.992	0.990
	Dyn 60	0.989	0.984	0.980	0.977	0.969
	Dyn 120	0.967	0.953	0.944	0.940	0.924
PM	Dynamic	1.0	1.0	1.0	1.0	1.0
	Stat. Opt.	0.980	0.977	0.974	0.972	0.967
	Static	0.844	0.777	0.680	0.538	0.318
	Stat. HS	0.943	0.934	0.917	0.891	0.858
	Clus. HS	0.932	0.917	0.897	0.866	0.830
	Dyn 15	1.0	1.0	1.0	1.0	1.0
	Dyn 30	0.986	0.984	0.982	0.983	0.980
	Dyn 60	0.975	0.965	0.956	0.958	0.951
	Dyn 120	0.949	0.926	0.909	0.919	0.904

Relative performances of each policy during AM and PM periods compared to the completely online policy under the deterministic performance evaluation.

We consider the Dynamic CC policy for $X \in \{15, 30, 60, 120\}$. Furthermore, one could also view the Dynamic model as Dynamic CC with $X = 0$.

RESULTS

In this section, we present the performance of a number of policies when run with varying cost-parameter for both the AM and PM periods. The data-set for the test period, on which our analysis relies, consists of a total of 4944 trips in the AM and 2800 trips in the PM, across 147 stations. We focus our analysis on the deterministic regime in which we assume that an incentivized rental/return is always triggered by the incentive given and would not have occurred otherwise. Thereafter, we contrast those results with the ones in the stochastic regime. The scores for the former are displayed in Table **??** and for the latter in Table **??**. All scores are given as the fraction of improvement of the optimal dynamic policy (i.e., each policy's absolute score is divided by the dynamic policy's absolute score). We begin by giving a high-level summary of our most interesting findings before providing more details for each of them.

Table 2: Policy Performance Scores in Probabilistic Regime.

Period	Policies	Cost Parameter			
		0.0	0.01	0.1	0.2
AM	Dynamic	1.0	1.0	1.0	1.0
	Stat. Opt.	0.979	0.973	0.945	0.912
	Static	0.928	0.916	0.771	0.479
	Static HS	0.957	0.944	0.878	0.796
	Clus. HS	0.958	0.931	0.831	0.699
	Dyn 15	1.0	1.0	1.0	1.0
	Dyn 30	0.995	0.995	0.993	0.991
	Dyn 60	0.985	0.870	0.832	0.773
	Dyn 120	0.966	0.68	0.612	0.514
PM	Dynamic	1.0	1.0	1.0	1.0
	Stat. Opt.	0.978	0.971	0.943	0.917
	Static	0.923	0.903	0.646	0.15
	Stat. HS	0.967	0.948	0.883	0.784
	Clus. HS	0.958	0.935	0.839	0.682
	Dyn 15	1.0	1.0	1.0	1.0
	Dyn 30	0.992	0.990	0.992	0.99
	Dyn 60	0.982	0.889	0.850	0.807
	Dyn 120	0.960	0.823	0.756	0.702

Relative performance of each policy during AM and PM periods compared to the completely online policy under the probabilistic performance evaluation.

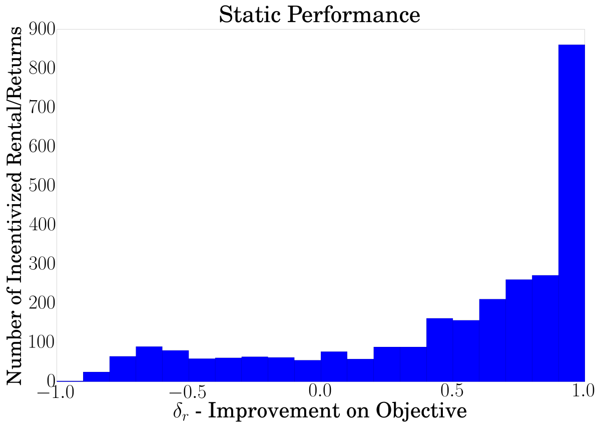

Figure 3: Static Policy's total number of incentivized trips, grouped by impact on objective (δ_r) for the PM period.

Key Observations of Deterministic Results

Considering the rows indexed by Static for both AM and PM periods and both tables, it is noticeable that the static policy performs quite well, especially in the regime with low cost parameters, which is strong evidence of the operator's domain expertise in the initial choice of incentivized stations. Next, we observe the near-optimal performance of the Static Optimal benchmark across all cost parameters; this supports our decision to restrict the offline policies to incentivize only over sub-intervals. For the dynamic policies, we observe a smooth decrease in performance as we transition from the maximally dynamic to more static policies, which highlights the trade-offs between efficiency and simplicity in the policies. Noticeably, none of the offline policies handle high cost-parameters well. Last, we observe a general decrease in performance from the AM to PM period, indicating that the system behavior is more erratic in the afternoon.

Static Performance

At first glance, the strong performance of the Static policy in Table **??** (0.939 in the AM without costs) may seem surprising. It is explained, however, by the operator's domain expertise when deciding which stations to include in the incentive scheme. Its degrading performance with increased costs is explained by the fact that the policy does not adapt to the shrinking set of trips with positive impact as costs increase.

Despite its reasonable performance in the AM in the regime without costs, the Static policy is still dominated by all policies across all cost parameters. This is especially prominent in the PM period, where differences range from 9% (Stat. HS) up to 15% (Dyn 15) even without cost parameters; the differences are even higher when accounting for costs. All of the incentivized trips in the PM, grouped by their improvement on the objective, δ_r, are visualized in Figure **??**. Though the vast majority of incentivized trips have positive-impact, the other policies are able to accurately exclude those trips that do not, thus achieving higher performance. These results exemplify the importance of a data-driven approach to improve the Bike Angels program. For example, even a rather simple policy with minimal overhead, such as Static Hindsight, significantly improves the efficiency of the incentives.

Static Optimal Benchmark Performance

The online policies have two advantages over the offline policies: the flexibility to adapt decisions with updated information and the flexibility to incentivize over periods that are not sub-intervals. Comparing the performance of the Static Optimal to the Dynamic policy helps us distinguish between these two effects. More specifically, the benchmark operates with perfect information but is constrained to only incentivize over sub-intervals.

In Table **??** we do indeed observe Static Opt achieving a near optimal score of 0.98, which only slightly worsens with increased costs. This supports our assumption that it suffices to incentivize over a single sub-interval. However, Static Opt assumes perfect knowledge and still only matches the performance of Dynamic CC (60). In that sense, it also demonstrates the limitations of the best feasible offline policies.

Online to Offline Policies

Unsurprisingly, we find that the online policies outperform offline policies. As we transition from the maximally dynamic and online policy to the entirely static and offline policy, the decrease in performance occurs in a somewhat smooth way.

An interesting result in this context is the performance of the Dynamic CC (15) policy in Tables **??** and **??**, as it achieves optimal scores for all sets of parameters; this demonstrates that making decisions completely online/in real time is not necessary to obtain perfect efficiency.

On the other end of the spectrum from online to offline, the results of Table **??** demonstrate that the offline policies Static Hindsight and Cluster Hindsight perform almost on par with the online Dynamic CC (120) policy, especially in the regime with low cost parameters; this points to the limited advantage of the simple online policies as the cost parameter increases.

The Cost Parameter

In considering the relative performance change of policies with increasing cost parameters, we find that the offline policies are comparatively much worse at handling high cost parameters than the online policies. Intuitively, it might seem that the interval incentive restrictions of offline policies leads to this result. For example, imagine taking the original incentive interval with no cost parameter, and changing some of the positive-impact trips within the interval to become negative-impact trips (due to increase in cost). Then unless these trips all exist at the outer limits of the interval, these trips will "shatter" the incentive interval: either the offline policies give up on the positive-impact trips at the beginning or they give up on the positive-impact trips at the end or they include the negative-impact trips in the middle. Online policies on the other end can avoid this conundrum as they are not restricted to a single sub-interval.

However, the performance of the Static Optimal benchmark in Table **??** does not significantly degrade with high cost parameters. Thus, despite the interval restriction, the offline policies still have room to have better predictions yield improved performance.

In contrast, Figure **??** displays for the Dynamic CC (60) all incentivized trips with their time of day, their expected impact on future out-of-stock events, and for each one, the decision whether or not it is incentivized when run with 2 different cost parameters.The cost parameter is specified in each plot by the y-value of the horizontal line dividing the positive-impact trips (above the line) and negative-impact trips (below the line). As the cost parameter increases from 0.0 to 0.3, the policy excludes most trips having a δ_r value between that range, that it had previously included. Thereby, it manages to retain its near-optimal performance.

AM and PM Periods

Comparing the policy performances from the AM to PM period, we find there is a significant performance difference (i.e., all policies perform worse in the PM). However, in light of the results in Table 1, we see the relative performance order of the policies with each other is consistent. This indicates that the system behavior of Citi Bike is fundamentally more erratic in the afternoon, leading to all policies having a harder time predicting when to incentivize.

Stochastic Evaluation

In this section we highlight four noticeable differences between the results in Tables **??** and **??**.

Limiting Cost Parameters

When evaluating Stochastic performances we limit the cost parameters to only go as high as 0.2. This is because when incorporating the likelihood of trips into the δ_r calculations,

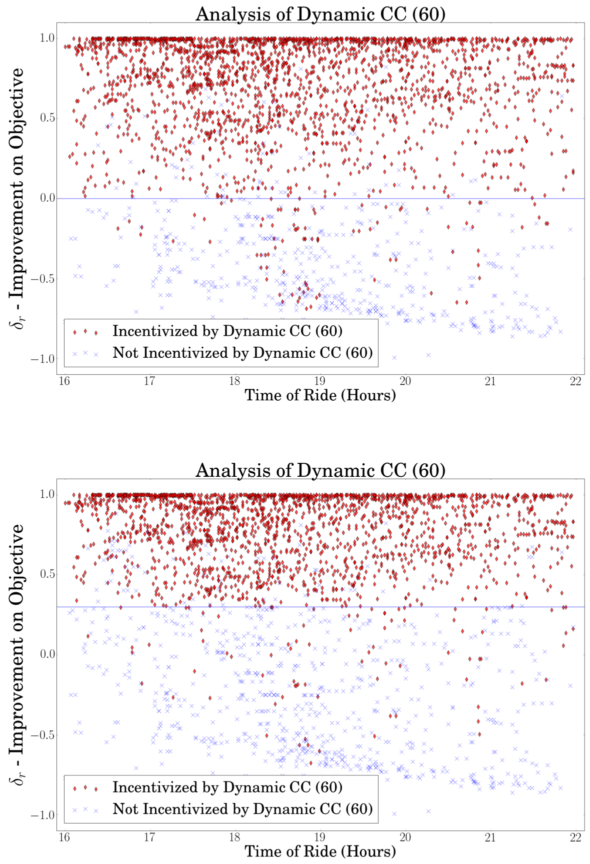

Figure 4: Scatter plots of incentivized trips indicating which trips are included/excluded in Dynamic CC (60) incentivization policy, when cost parameter is 0.0 (top) and 0.3 (bottom).

the expected impact of a trip before subtracting the cost is much lower; in particular, it is easy to show that the impact of a single incentivized rental/return is at best a reduction of 1 in the number of out-of-stock events. But then, if for example the probability of a particular rental having been triggered by an incentive was 0.5, then even with an impact of 0.8, there would be no improvement with cost parameter 0.4.

Relative Order

The results in Table **??** show that the same general trends in relative performance order (most dynamic to most static) that hold for the deterministic results, also hold for the stochastic results. Likewise, the results indicate that the Static Optimal benchmark still performs near optimally in the stochastic setting.

A key difference between the deterministic and stochastic results is the sensitivity to cost parameters. In particular, except for the Dynamic CC policies parameterized with 15 and 30, all policies decrease significantly in performance with even small increases in the cost parameter. Intuitively, due to the newly introduced stochasticity, the expected impacts of all rides are reduced, and small changes in cost are still large, relative to the reduced impacts.

Advantage of Hindsight Policies

Another interesting difference found between the deterministic and stochastic results is the relative performances of the Dynamic CC 60 and 120 policies compared to the Static Hindsight and Cluster Hindsight policies. Without stochasticity, the online policies dominated the offline policies in performance. However, when introducing stochasticity and for higher cost parameter regimes, the opposite seems to be true. Intuitively this makes sense. The Dynamic CC X policies only consider the status of the station at the beginning of each X-minute interval, which includes the probability of an incentivized rental/return having occurred due to the incentive at that time (thus ignoring the differing probabilities existing within the X-minute interval). In contrast, the Static Hindsight and Cluster Hindsight policies actually incorporate all probabilities when computing the optimal incentive intervals in hindsight.

Static Policy

Finally, from the stochastic results we see an even greater difference in performance between the baseline Static policy and all other policies (especially with increasing costs). The large differences again underline the performance improvements that can be obtained by transitioning from the Static policy to a data-driven policy.

CONCLUSION

We have proposed a number of data-driven policies to guide incentives for rebalancing in bike-sharing systems via crowdsourcing. While our analysis clearly displays the performance differences between these policies, the superior performance of the more dynamic policies comes with the cost of greater complexity for users and operators alike.

There are other important considerations beyond their simplicity and performance. For example, when comparing the performance of the Static and Cluster Hindsight policies, it seems unclear at first glance what additional value the Cluster Hindsight policy provides, given that it relies on heavier machinery – after all, they perform very similarly. However, the Static Hindsight policy can only be defined for stations for which the Static policy had been in place, whereas the Cluster Hindsight policy can be defined for other stations as well. Thus, in a way, each of the policies presented has its own advantage.

Most importantly, our analysis shows that slightly limiting the online fashion of decision-making only causes limited decreases in performance. On the academic side, this adds a data-driven analysis to a recent stream of literature in operations management that compares dynamic and static decision-making in similar applications. On the practical side, our analysis led to Citi Bike adopting a version of the Dynamic CC (30) policy in 2017.

REFERENCES

1. Siddhartha Banerjee, Daniel Freund, and Thodoris Lykouris. 2017. Pricing and Optimization in Shared Vehicle Systems: An Approximation Framework. In *Proceedings of the 2017 ACM Conference on Economics and Computation*. ACM, 517–517.
2. Siddhartha Banerjee, Ramesh Johari, and Carlos Riquelme. 2015. Pricing in ride-sharing platforms: A queueing-theoretic approach. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*. ACM, 639–639.
3. Yiwei Chen and Ming Hu. 2017. Pricing and matching with forward-looking buyers and sellers. (2017).
4. Iris A Forma, Tal Raviv, and Michal Tzur. 2015. A 3-step math heuristic for the static repositioning problem in bike-sharing systems. *Transportation Research Part B: Methodological* 71 (2015), 230–247.
5. Daniel Freund, Shane G. Henderson, and David B. Shmoys. 2017. Minimizing multimodular functions and allocating capacity in bike-sharing systems. In *Integer Programming and Combinatorial Optimization Proceedings (Lecture Notes in Computer Science)*, F. Eisenbrand and J. Koenemann (Eds.), Vol. 10328. Springer, 186–198. arXiv preprint arXiv:1611.09304.
6. Daniel Freund, Ashkan Norouzi-Fard, Alice Paul, Shane G. Henderson, and David B. Shmoys. 2016. Data-Driven Rebalancing Methods for Bike-Share Systems. (2016). Working paper.
7. Christine Fricker and Nicolas Gast. 2012. Incentives and regulations in bike-sharing systems with stations of finite capacity. *arXiv preprint arXiv:1201.1178* (2012), 2.
8. Christine Fricker and Nicolas Gast. 2016. Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity. *Euro journal on transportation and logistics* 5, 3 (2016), 261–291.
9. David K George. 2012. *Stochastic Modeling and Decentralized Control Policies for Large-Scale Vehicle Sharing Systems via Closed Queueing Networks*. Ph.D. Dissertation. The Ohio State University.

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

11. Nanjing Jian, Daniel Freund, Holly M Wiberg, and Shane G Henderson. 2016. Simulation optimization for a large-scale bike-sharing system. In *Proceedings of the 2016 Winter Simulation Conference*. IEEE Press, 602–613.

12. Deepak Merugu, Balaji S Prabhakar, and N Rama. 2009. An incentive mechanism for decongesting the roads: A pilot program in bangalore. In *Proc. of ACM NetEcon Workshop*.

13. NYCBS. 2016. Private Communication. (2016).

14. NYCBS. 2017a. Bike Angel Program. (2017). `https://bikeangels.citibikenyc.com`

15. NYCBS. 2017b. Citi Bike JSON Feed. (2017). `https://gbfs.citibikenyc.com/gbfs/en/station_status.json`

16. NYCBS. 2017c. Citi Bike System Data. (2017). `https://www.citibikenyc.com/system-data`

17. Eoin O'Mahony, Shane G. Henderson, and David B. Shmoys. 2016. (Citi)Bike sharing. (2016). working paper.

18. Eoin O'Mahony and David B Shmoys. 2015. Data Analysis and Optimization for (Citi) Bike Sharing. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*. 687–694.

19. Pulkit Parikh and Satish Ukkusuri. 2014. Estimation of Optimal Inventory Levels at Stations of a Bicycle Sharing System. *TRB 94th Annual Meeting* (2014).

20. Tal Raviv and Ofer Kolka. 2013. Optimal inventory management of a bike-sharing station. *IIE Transactions* 45, 10 (2013), 1077–1093.

21. Tal Raviv, Michal Tzur, and Iris A Forma. 2013. Static repositioning in a bike-sharing system: models and solution approaches. *EURO Journal on Transportation and Logistics* 2, 3 (2013), 187–229.

22. J. Schuijbroek, R.C. Hampshire, and W.-J. van Hoeve. 2017. Inventory rebalancing and vehicle routing in bike sharing systems. *European Journal of Operational Research* 257, 3 (2017), 992 – 1004. DOI: `http://dx.doi.org/https://doi.org/10.1016/j.ejor.2016.08.029`

23. Adish Singla, Marco Santoni, Gábor Bartók, Pratik Mukerji, Moritz Meenen, and Andreas Krause. 2015. Incentivizing Users for Balancing Bike Sharing Systems.. In *AAAI*. 723–729.

24. WY Szeto, Ying Liu, and Sin C Ho. 2016. Chemical reaction optimization for solving a static bike repositioning problem. *Transportation research part D: transport and environment* 47 (2016), 104–135.

25. Ariel Waserhole and Vincent Jost. 2014. Pricing in vehicle sharing systems: optimization in queuing networks with product forms. *EURO Journal on Transportation and Logistics* (2014), 1–28.