Balancing Bike Sharing Systems through Customer Cooperation – A Case Study on London’s Barclays Cycle Hire

Philipp Aeschbach, Xiaojing Zhang, Angelos Georghiou, and John Lygeros

Abstract—A growing number of cities worldwide have been installing public bike sharing systems, offering citizens a flexible and “green” alternative of mobility. In most bike sharing systems, customers rent and return bikes at different stations, without prior notification of the system operator. As a consequence, bike systems often become unbalanced, leaving some stations either empty or full. In such a case, customers either cannot pick up or return their bikes, resulting in a low service level. Typically, system operators employ staff to manually relocate bikes using trucks, leading to considerable operational cost. In this paper, we describe various methods to balance bike sharing systems by actively engaging customers in the balancing process. In particular, we show that by appropriately sending “control signals” to customers requesting them to slightly change their intended journeys, bike sharing systems can be balanced without using staffed trucks. Through extensive simulations based on historical data from London’s Barclays Cycle Hire scheme, we show that simple control signals are sufficient to effectively balance the bike sharing system and offer service rates close to 100%.

I. INTRODUCTION

A public bike sharing system is a recent concept of public transportation that offers citizens a “green” and flexible transportation scheme in urban areas. For that purpose, a (large) number of bike stations are installed throughout the city, where customers are allowed to start and end their bike journeys at different docking stations. This flexibility has rendered bike sharing systems highly popular. Moreover, it is widely believed that for many cities such schemes are capable of covering a considerable part of citizens’ transportation needs [1], [2]. In fact, over 850 cities worldwide now offer such services, with an estimated total of 946’000 bikes. Unfortunately, the operation of bike sharing systems poses a number of new challenges. The biggest stem from the fact that bike sharing systems tend to become badly balanced, with stations becoming either empty or full. This phenomenon occurs due to an asymmetric flow of bikes from the residential areas to the city center. In such cases, customers are unable to pick up or return their bikes, resulting in a low service level. Indeed, the authors of [3] show theoretically that even in idealized networks, a loss in service level is inevitable without some kind of “control mechanism” that relocates bikes. The current practice to counteract the asymmetric flows is to employ staffed trucks for manual relocation of bikes to balance the difference between bike supply and demand at various stations. Unfortunately, as shown in [4], the optimal relocation of bikes is an NP-hard problem, so that most literature resort to some kind of heuristics, see e.g. [4]–[9] for different repositioning policies. This staffed relocation of bikes using trucks not only poses major operational costs to the operator, but also does not fit into the frame of a “green” transportation mode.

In this paper, we investigate the possibility of involving customers in the balancing process, where the main idea is to modify a user’s journey by slightly changing the pickup and return station. We are not the first to consider customers in the balancing process, and quite a literature already exists on this topic, see e.g. [10]–[14]. However, to the best of our knowledge, the aforementioned papers do not concentrate on the capabilities of customers to balance the system, but rather on how to incentivize and reward customers to participate in such balancing programs. Therefore, all the obtained service levels depend on the particular reward strategies. For example, when the authors in [10], [11] report service levels of around 77%, it is not clear if the (relatively low) service level is due to inherent inabilities of customers to balance the system, or if the proposed reward programs are not effective.

This paper decouples the problem of customer cooperation from reward provision by abstractly capturing the former through the customer cooperation factor, defined as the percentage of cooperative customers. We concentrate on the question whether in a truck-free bike system, customers are able to balance the system. We propose four different control strategies for balancing bike systems, which distinct themselves in the way they interact with customers. By treating the customer cooperation factor as a variable parameter, we evaluate the efficacy of the proposed strategies on a model based on historical data obtained from London’s Barclays Cycle Hire scheme. Our results form a useful benchmark for future work, since they provide insights on achievable service levels for a given customer cooperation factor. Likewise, our results can be used when designing appropriate reward programs since they indicate the fraction of customers that should be addressed for achieving a desired service level. As an additional point, we quantify the extra amount of distance cooperative customers will face in their journeys.

The remainder of this paper is organized as follows. Section II explains how we derived a model of London’s Barclays Cycle Hire from historical data. In Section III, we propose different control strategies to balance the bike system. Simulation results are presented in Section IV. Section V discusses practical aspects for real-world implementation, and Section VI concludes this paper.

The authors are with the Automatic Control Laboratory, Department of Electrical Engineering and Information Technology, ETH Zurich, 8092 Zurich, Switzerland. Contact: philipae@student.ethz.ch, {xiaozhan,angelosg,lygeros}@control.ee.ethz.ch
II. System Model

In this section, we describe a generic simulation environment for bike sharing systems based on historical data, which we use to simulate Barclays Cycle Hire in London. The simulation follows the work of [10], and requires data about the size and location of all stations, as well as a record of all conducted journeys within a certain timespan. In the case of Barclays Cycle Hire, the official Transport for London government body makes such information freely available [15]. We evaluate information of 5’110’650 bike rides between 5. January – 19. July 2014, conducted among a total of 745 stations. These stations contain altogether 18912 slots with overall 9109 bikes.

A. Infrastructure

We define a set $S$ containing all stations $s \in S$. The capacity of each station $s$ is denoted by $c_s$, while $b_s(t)$ stands for the number of bikes at station $s$ at time $t$. For simplicity, we introduce a normalized variable called fill level, defined as $f_s(t) := b_s(t)/c_s$. Finally, we divide a day into 72 time slices, each consisting of a 20 minute interval, so that $t \in T := \{0, 1, \ldots, 71\}$.

B. Journey Generation

A journey is defined by three parameters: (a) The start and end station, which we denote by $s_i, s_j \in S$, respectively; (b) the starting time, $t$, and (c) the time duration. All three parameters are estimated from historical data. In particular, we assume that the number of journeys going from station $s_i$ to station $s_j$, starting at time $t$, follow a Poisson distribution with mean $\lambda_{i,j}(t)$. We empirically determine the value of $\lambda_{i,j}(t)$ by computing the average number of journeys in the historical data set belonging to the triplet $(s_i, s_j, t)$. In our simulation, the duration of journeys is determined by uniformly sampling from a set $D_{i,j}$, where $D_{i,j}$ contains all journey durations from station $s_i$ to $s_j$ in the historical data set. We note that the data is processed separately for workdays and weekends. This is done since the traffic load of the system is significantly different, with workdays providing more than one no-service event, the service level defined in (1) does not, in general, capture the percentage of satisfied customers. As we show in Section III-A, in an unbalanced system customers regularly generate multiple no-service events. However, in a reasonably well-balanced system a single customer almost never causes more than one no-service event.

The current practice to achieve high service levels is to redistribute bikes by employing staffed trucks. Below, we demonstrate that, by actively involving customers in the relocation process, public bike sharing systems can remain well-balanced even without manual relocation.

III. Control Strategies

In this section, we introduce four different control strategies. These strategies differ in the extent to which they intervene in customers’ journeys and in the way they interact with customers. We start with the simplest strategy and gradually increase its complexity. The efficacy of each strategy is evaluated in our simulation of the Barclay Cycle Hire in London, where no trucks are used to relocate bikes.

A. No Control (NC)

In order to establish a base performance level of a bike sharing system without trucks, the simulation is run without interfering in journeys of customers. To do that, we first define the customer behavior when they experience no-service events. In case of a station-empty event the customer walks to the nearest station and tries to pick up a bike there. If that station is also empty, the customer leaves the system. Customers experiencing a station-full event ride to the nearest station and try to return their bike there. They do so until they find an empty slot. Note that a customer never visits the same station twice, preventing customers from commuting between two neighboring full stations. This behavior captures a natural customer reaction when they are given no further directions about which station to visit next.

We can expect a low service level in this No Control case, since there is no mechanism that balances the system. Indeed, the service level in such a setting, averaged over 100 simulations, is 0.06, with a sample standard deviation of 0.055. The main reason for this extremely low service level is that certain customers cause multiple no-service events in a row. To better understand this phenomenon, Fig. 1a depicts a snapshot of the station fill levels at 4 pm on a workday. Since no re-balancing mechanism is in place, we observe clusters of completely full and empty stations. Moreover, in Fig. 1b, we see the average fill level of a group

We will refer to either station-empty or station-full events as no-service events. This definition allows us to define a simple measure of service quality provided by the bike sharing system, also used in [10]–[12]:

$$\text{service level} = \frac{\# \text{customers} - \# \text{no-service events}}{\# \text{customers}}. \quad (1)$$

The motivation for using (1) is to calculate the fraction of satisfied customers who do not experience no-service events. However, since one particular customer may cause more than one no-service event, the service level defined in (1) does not, in general, capture the percentage of satisfied customers. As we show in Section III-A, in an unbalanced system customers regularly generate multiple no-service events. However, in a reasonably well-balanced system a single customer almost never causes more than one no-service event.

The current practice to achieve high service levels is to redistribute bikes by employing staffed trucks. Below, we demonstrate that, by actively involving customers in the relocation process, public bike sharing systems can remain well-balanced even without manual relocation.
of neighbouring stations from the city center and a group from the outer region. It clearly showcases the common travel pattern of people going to work in the city center in the morning and driving back home later in the day. This opens the following question: Can we improve the service level by helping customers avoid multiple no-service events?

B. Minimal Intervention Control (MIC)

Minimal Intervention Control (MIC) only intervenes when customers experience no-service events. The main idea is to divert customers to a nearby station with many full/empty slots for the start/finish of the journey, respectively, through an interface installed at each station.

To properly define the term nearby, we introduce the concept of a neighbourhood. For each station $s \in S$ we define a set of neighbours $N(s) := \{ n \in S \setminus \{ s \} : d(n, s) \leq r \}$, where $d(\cdot, \cdot)$ denotes the euclidean distance between two stations, and $r$ is the so-called neighbourhood radius, which is an adjustable control parameter. Whenever a customer experiences a no-service event at station $s$, MIC diverts the customer to a station $\tilde{s} \in N(s)$ that has the most full/empty slots so that she can rent/return her bike. In case the station has no neighbours ($r$ was chosen too small) or all neighbours are empty/full, the customer is sent to the closest station, regardless of its fill level.

By construction, the MIC strategy helps avoiding multiple no-service events. Nevertheless, it does not prevent stations from becoming empty or full in the first place, but only reacts to no-service events. This problem is tackled by the next control strategy. We note that all subsequent strategies use the MIC strategy whenever no-service events occur.

C. Preemptive Control (PC)

The idea behind the Preemptive Control (PC) strategy is to preemptively change journeys before a no-service event occurs. To do this, we assume that cooperative customers will use the communication interface installed at each station before they rent or return a bike. Through the interface, they are told which neighbouring station they must use to rent or return their bikes. The PC controller selects the neighbouring station $\tilde{s} \in N(s)$ with the most bikes or empty slots, within a given radius $r$, unless the current station has more bikes/empty slots than the best neighbouring station.

One downside of this control strategy from the cooperative customer’s perspective is that after arriving at the start/destination station, the customer is potentially sent to a neighbouring station, even if she would be able to rent/return a bike at the current station. Most customers may find this rather inconvenient. However, if customers knew a priori the changes in their itinerary, they could directly head for the correct station. In addition, such a priori information benefits the controller, as it can use this extra knowledge for future decisions. This shortcoming is addressed by the next controller.

D. Preemptive Control via Mobile App (PC-MA)

Instead of communicating with cooperative customers at the stations, we propose that the PC strategy is announced to customers through a mobile application. This allows customers to use the app shortly before undertaking their journeys and therefore walk (ride) directly to the start (end) station selected by the controller. For further convenience, the selection of the destination station happens beforehand. This way all changes in journeys are decided before customers embark on them. There are two positive effects to this: (i) The customers know a priori the journey route which will not include any detours; (ii) This information can be exploited by the controller, since it can explicitly account for these journey arrangements, and therefore better estimate the short-term station fill levels. Our simulations suggest that a drawback of this strategy is that almost all app users will have to change their journeys, and sometimes by a considerable distance. Weak Preemptive Control attempts to address this shortcoming.

E. Weak Preemptive Control via Mobile App (WPC-MA)

We “weaken” the control we have over customers by reducing the number of times we force cooperative customers to change their journeys. When cooperative customers enter their desired start and destination stations in the mobile app and the fill levels of these stations are above/below 50%, WPC-MA asks the customers to use these stations.
Otherwise, we send them to the nearest station that has a fill level above/below 50% for start/destination respectively. This strategy should be contrasted with PC-MA, where customers are sent to the most full/empty station. By using WPC-MA, customers not only need to change their journeys less often compared to PC-MA, but they also need to make smaller (geographical) changes once they are asked to do so.

### IV. Simulation Results

#### A. Parameters

The performance of the strategies discussed in Section III largely depends on two parameters: (i) the neighbourhood radius $r$, and (ii) a parameter defining the “obedience” of customers. This second parameter is called customer cooperation $c \in [0, 1]$ and it describes the percentage of customers willing to cooperate. In PC, $c$ describes the probability that a customer decides to use the interface at a station even when she does not experience a no-service event, whereas in PC-MA and WPC-MA, $c$ simply denotes the percentage of customers using the mobile app. Note that in theory, an additional compliance parameter can be introduced which models the probability that the user uses the app but does not follow the suggestions. However, the compliance parameter can be implicitly incorporated in the customer cooperation factor $c$, as a compliance parameter $< 1$ effectively reduces $c$ by that amount. We note that customer cooperation $c$ has no influence on MIC, as here we only intervene when no-service events happen and then the customer must cooperate in order to rent/return a bike.

#### B. Performance Analysis

To analyze the performance of our control strategies, we examine their behavior for varying levels of the neighbourhood radius and the customer cooperation. The results are depicted in Fig. 2. Recall that MIC is independent of $c$; its performance only varies for different values of $r$. However, the other three control strategies profit considerably from a large $c$. Note that for $c = 0$, all control strategies reduce to MIC, since no customer would cooperate unless a no-service event happens.

The graphs display the behavior of the controllers for $r \geq 400$ meters. When $r < 400$ meters, the service levels are very low since most stations do not have neighbouring stations within that radius. In such cases, the control strategies are reduced to the No Control (NC) strategy described in Section III-A. Simulation results clearly show that PC-MA is the best performing control strategy, followed by WPC-MA, PC, and MIC. The reason why PC-MA and WPC-MA outperform MIC and PC is that they include knowledge about ongoing journeys in their calculations, since such information is revealed by customers using the app. This prevents the oversteering effect we see in Fig. 2d for PC. There we observe that PC starts performing worse once the neighbourhood radius exceeds 700 meters. This happens because many customers are on their way to the same allegedly “good” station in a short time span. Since PC only looks at the current station fill level, it does not detect that this nearly empty station will soon be full because many customers are on their way there, and will keep sending customers to this station.

Another interesting result is that WPC-MA performs nearly as well as PC-MA, indicating that PC-MA might be too “aggressive”, making more customers change their journeys than necessary. Furthermore, it is striking how the performance of all strategies increases rapidly until $r$ reaches at least 600 meters and then saturates for higher values of $r$. Naturally, this behavior will be different for other bike sharing systems, but in our case it indicates that for Barclays Cycle Hire in London, a neighbourhood radius of around 600 meters would be desirable.

According to Fig. 2d, the “performance bound” is achieved by PC-MA, where a service level of 99.97% is reached for $r = 1$ km and $c = 1$. WPC-MA with the same parameter settings achieves 99.83%. PC performs best for $r = 700$ meters and $c = 1$, where it reaches a service level of 97.3%. From these numbers we can conclude that preemptive strategies clearly outperform reactive strategies such as MIC, which reaches a maximum service level of 81.6%. Compared to the No Control (NC) case (service level of 6%), all four strategies achieve service levels that are one order of magnitude better than NC.

#### C. Station Fill Levels

To better illustrate how our control strategies manage to balance the system, let us take a look at the average fill levels of the outer region and the city center under MIC and PC-MA (compare with Fig. 1b). For this we use the parameters $r = 700$ meters and $c = 0.5$. In Fig. 3a, we see that the stations still reach their capacities when we use MIC, since MIC is purely reactive and only intervenes once stations are full or empty. In Fig. 3b, we see the preemptive nature of PC-MA. It prevents stations from becoming completely empty or full in the first place, which naturally prevents no-service events from happening. However, the travel pattern towards the city center in the morning and back out in the evening is still clearly visible.

#### D. Price of Cooperation via Mobile Application

Using a mobile app (as in PC-MA or WPC-MA, but not MIC or PC) makes journeys of cooperative customers longer. The following metric captures this extra effort made by the customers:

$$\text{Extra Effort} = \frac{\sum_{\text{customers}} [d(s_s, \hat{s}_s) + d(s_d, \hat{s}_d)]}{\# \text{ App Users}},$$

where $d(\cdot, \cdot)$ denotes the euclidean distance between two stations. $s_s$ and $s_d$ are the original start and destination stations chosen by the customer, with $\hat{s}_s$ and $\hat{s}_d$ being the start and destination station chosen by the control strategy. Note that $s_s$ and $\hat{s}_s$ (or $s_d$ and $\hat{s}_d$) are the same station if (i) the control strategy coincidentally selects the same station, or if (ii) the customer does not use the app. Therefore only cooperative customers add to the sum, and dividing by the number of app users gives the average combined distance.
Fig. 2. Service levels of different control strategies for different parameter settings.

Fig. 3. Average station fill levels for MIC (left) and PC-MA (right), generated with parameters $r = 700$ meters and $c = 0.5$.

Fig. 4. Extra Effort (left) and Benefit (right) of App Usage.
at the start and end caused by the changed journey. This number gives a rough estimate of the extra effort performed by app users.

Simulated values of (2) are depicted in Fig. 4a for PC-MA and WPC-MA. They are calculated for a constant customer cooperation $c = 0.7$ and a varying neighbourhood radius $r$. We see a significant difference in extra effort demanded of cooperative customers in PC-MA and WPC-MA. This is explained by the fact that customers in WPC-MA need to change their journey less often, and even in the case they do, they more likely will go to a closer station.

Fig. 4b separately depicts the service level of app users and non-app users for PC-MA and WPC-MA. Unsurprisingly, app users enjoy a higher service level than non-app users. It is interesting to notice that non-app users profit from app users, since the service level of non-app users increases as the fraction of app users increases. This “herd immunization phenomenon” happens because cooperative customers (i.e., app users) are balancing the system, which also benefits non-cooperative customers.

V. Discussion

Simulation results shown in the previous section clearly indicate that customers are able to balance a public bike sharing system on their own, without the need for relocating trucks. However, a service level approaching 100% can only be achieved if most customers cooperate, as seen in Fig. 2a–2d. Nevertheless, even if only 25% of customers are cooperating, our control strategies still show a large performance increase. In this case the service level may still surpass 90%, so that a combined solution which incorporates a small amount of relocating trucks may be a sensible option.

A. Customer Cooperation Level

It is hard to predict realistic values for the customer cooperation $c$. The best way to determine $c$ are real-life experiments. In this regard, [12] found that, through a customer survey in Mainz, Germany, $c$ approximately decays linearly as $r$ increases. Moreover, the authors found that almost 60% of surveyed customers declared their willingness to cooperate given a maximum walking distance of at most 500 meters. Clearly, the actual values the two parameters take on in reality are largely based on how customers are incentivized. Nevertheless, as seen in Fig. 2b, if the system operator is able to motivate a customer participation rate of 50% with a neighbourhood radius of 700 meters, then the system could be balanced rather well.

B. Implementation of Control Strategies

The control strategies presented in this paper can be readily implemented in real world. The algorithms are simple, and scalable to systems of realistic sizes. Communication with cooperative customers can comfortably happen via a mobile application. Due to the savings from reducing or avoiding the use of trucks, and the minimal investment required by the system operator, the financial circumstances seem favorable. All in all, we believe that our proposed control strategies not only effectively balance bike sharing systems, but are also practically attractive and implementable.

VI. Conclusion

In this paper, we discussed the potential of involving customers in balancing station fill levels of public bike sharing systems with no staff-operated relocation trucks. We proposed four different control strategies and evaluated their efficacy through extensive simulations on a realistic system model based on Barclays Cycle Hire in London. Simulation results indicate that customers can effectively help balance truck-free bike sharing systems. In fact, if all customers cooperate in the balancing process, service levels of almost 100% can be achieved. Surprisingly, even if fewer cooperative customers can be mobilized (say 25%), good service levels are achieved. Our results clearly indicate that customer cooperation is a viable way of balancing bike sharing systems, which can partially (or even entirely) replace expensive truck-based relocation strategies.

References