Inventory Rebalancing through Pricing in Public Bike Sharing Systems

Zulqarnain Haider[†] Alexander Nikolaev[‡] Jee Eun Kang[‡] Changhyun Kwon[*†]

[†]Department of Industrial and Management Systems Engineering, University of South Florida, Tampa, FL, USA
[‡]Department of Industrial and Systems Engineering, University at Buffalo, Buffalo, NY, USA

February 20, 2018

Abstract

This paper presents a new conceptual approach to improve the operational performance of public bike sharing systems using pricing schemes. Its methodological developments are accompanied by experimental analyses with bike demand data from Capital Bikeshare program of Washington, DC (USA). An optimized price vector determines the incentive levels that can persuade system customers to take bikes from, or park them at, neighboring stations so as to strategically minimize the number of imbalanced stations. This strategy intentionally makes some imbalanced stations even more imbalanced, creating hub stations. This reduces the need for trucks and dedicated staff to carry out inventory repositioning. For smaller networks, a bi-level optimization model with a single level reformulation is introduced to minimize the number of imbalanced stations optimally. The results are compared with a heuristic approach that adjusts route prices by segregating the stations into different categories based on their current inventory profile, projected future demand, and maximum and minimum inventory values calculated to fulfill certain desired service level requirements. We use a routing model for repositioning trucks to show that the proposed optimization model and the latter heuristic approach, called the iterative price adjustment scheme (IPAS), reduce the overall operating cost while partially or fully obviating the need for a manual repositioning operation.

Keywords: transportation; bike-sharing; shared-mobility; rebalancing; pricing; heuristics

1 Introduction

Increasingly, Public Bike Sharing Systems (BSS) are being adopted by many major cities throughout the world. Bikes are being touted as a way to achieve sustainable mobility in an urban setting while also helping to alleviate the last mile problem in urban transportation (Shaheen et al., 2010). As

[*]Corresponding Author; E-mail: chkwon@usf.edu

of February 19, 2018, bike sharing systems are operating in 1560 cities worldwide and another 402 such systems are in planning or under construction with a growing interest in more and more cities (Meddin and DeMaio, 2018). One of the major problems faced by these systems is the operational issue of repositioning of bikes between different stations. Demand variability causes certain stations to become too full or too empty to effectively service new customers. This not only affects the desired service level but also incurs spurious operational costs. According to a report by New York City Department of City Planning (2009) based on different case studies, the total capital cost for a bike sharing system varies from $3000/bike to $4400/bike in different cities. When averaged across programs, the yearly operating cost for a bike share program is around $1,600/bike.

The operating cost consists of system operations, administration, marketing and utility costs associated with hardwired stations. System operation forms the largest share of these costs and includes functions such as: maintenance of all equipment, rebalancing of bikes, customer service operations, and IT support (The Pennsylvania Environmental Council, 2013). Clearly, the repositioning of bikes from stations too full to stations too empty is a huge operational overhead. In fact, for Vélib system in Paris, the average cost of a single repositioning for a single bike is $3 (DeMaio, 2009). A system-wide snapshot of Capital Bikeshare at 9:30 a.m. on May 15, 2014 shows that 88 out of 202 stations are *imbalanced* considering 90% service level (see Section 3.1).

The contribution of this work lies in the development of methods - both exact and heuristic - and algorithms that bike sharing system managers can use to reduce the number of imbalanced stations by rebalancing their inventory through price incentives/disincentives. To do so, we will intentionally make some imbalanced stations more imbalanced, making them function as *hubs*. If only a few highly imbalanced stations exist in the system, bike redistribution can be handled with a few regular short time truck trips. With the reduced number of imbalanced stations, the truck redistribution operation becomes simpler and efficient resulting in operating cost reduction. This observation is key to our idea of designing dynamic pricing policies. We seek to ensure that surplus bikes are gathered predominantly at 'surplus accumulation' stations (hubs for bikes), and similarly, the deficiency of bikes mainly occurs at 'lack accumulation' stations (hubs for docks).

We understand that the practical implementation of the pricing policy can be challenging and must be discussed. Nowadays, many modern bike sharing stations are equipped with a computer terminal with a touch screen. When a bike user tries to checkout a bike, the user will be asked to choose his or her destination. Based on the current state of the system, the user will be provided with alternate journey choices with information about the price to be charged for each choice at the time of return. A mobile webpage or an application can also be used to provide on the go information about the prices even before the bike user approaches a bike station.

We assume that bike users exhibit a homogeneous sensitivity to the price and always seek to maximize their utility. Such indirect control by the system operator in the bi-level programming context is also very common in the Stackelberg game (or leader-follower game) setting for economics

and policy studies (Bard, 1991). We also did not consider the demand elasticity of price. We assumed that travel demand is fixed and users choose the lowest priced alternative to make the journey.

To determine the price incentives, we formulate a bi-level optimization model in Section 3 and provide a single-level reformulation that may be useful for small networks. In Section 4 we propose a heuristic algorithm, called the Iterative Price Adjustment Scheme (IPAS), and compare its performance with the single-level optimization model (P) solved by a commercial solver. We conclude that we can successfully reduce the number of imbalanced stations, by giving travelers multiple journey choices and changing the cost of those journeys through pricing. We also demonstrate that the cost of the same degree of manual rebalancing outweighs the price incentives offered.

The performance of IPAS is demonstrated by computational experiments in Section 5. Using the data from Capital Bikeshare in Washington, D.C., we show how our approaches manage to successfully minimize the number of imbalanced stations. The efficacy of our heuristic approaches vis-à-vis execution time, while bringing satisfactory improvement to the overall objective of minimizing the number of imbalanced stations is also shown. In Section 5.3.2, we use a routing model to show how the smaller number of imbalanced stations achieved as a result of a pricing scheme translates into a simpler and more efficient static repositioning operation using trucks.

2 Literature Review

Bike sharing systems have recently garnered an increased interest from the research community due to their growing importance in sustainable urban transport systems. DeMaio (2009) and Shaheen et al. (2010) separately discuss the history, impacts, models of provision and the future of public BSS. They identify improved redistribution of bikes as a key challenge facing BSS. Schuijbroek et al. (2017) have an excellent and comprehensive description of BSS literature. They divide up the BSS literature into four major streams including strategic design, demand analysis, service level analysis, and rebalancing operations. We, thus, refer readers to Schuijbroek et al. (2017) for general literature review, and limit this section to reviewing relevant literature on bike sharing systems, in particular, rebalancing operations.

Rebalancing operations are a big part of operating costs of a bike sharing system (The Pennsylvania Environmental Council, 2013). Generally, bike sharing systems employ two methods to redistribute the bikes: truck-based manual redistribution and pricing-based rebalancing.

Most bike sharing systems have a fleet of trucks that move around and pick and drop bikes. Vélib has 20 trucks (Benchimol et al., 2011) operating 24 hours to carry out manual rebalancing. Trucks and crew required to operate these have huge associated costs. Paul DeMaio of MetroBike, LLC, mentions a conversation with Luud Schimmelpennick, a pioneer of bike sharing concept, in DeMaio (2009). He reports that according to Schimmelpennick the cost for distribution of a single bike for JCDecaux is $3 and that any scheme that offers incentives to customers would increase the

redistribution efficiency at a fraction of the current cost. Since some kind of manual balancing is always required, most of rebalancing literature is focused on optimal truck routing.

Several papers have recently studied truck-based manual bike redistribution. Benchimol et al. (2011) introduces several approximation algorithms for static rebalancing of bikes at the end of the day. Raviv et al. (2013) introduced several formulations for static rebalancing problem with the objective of minimizing the expected user dissatisfaction. Chemla et al. (2013) present an exact model for the static rebalancing problem and two relaxations that they then solve using a branch-and-cut algorithm in conjunction with tabu search. Dell'Amico et al. (2014) also propose several MILP formulations for bike rebalancing problem and then propose a branch-and-cut algorithm to solve these models. They apply their approach on 65 benchmark instances to compare the performance of their MILP formulations. Contardo et al. (2012) introduce a dynamic public bike sharing balancing problem (DPBSBP) to rebalance a BSS during daytime which constitutes peak hours. They solve the DPBSBP problem using Dantzig-Wolfe decomposition and Benders decomposition to derive lower bounds and fast feasible solutions. Caggiani and Ottomanelli (2012) construct a modular Decision Support System (DSS) for dynamic bike redistribution. Shu et al. (2013) discuss under-utilization of bike sharing systems in Chinese cities and propose a deterministic model to optimally deploy bikes and docking capacity at different stations. They also evaluate the value of redistribution and its impact on the number of trips supported by the system.

There is a recent trend in BBS literature to introduce a scheme of incentives to get users to move bikes away from the crowded stations and into the less occupied stations. Vélib operates a V+ scheme to induce users to avoid certain stations and prefer others. Users get 15 minutes of added travel time if they place the bikes at one of the hundred uphill stations (Fricker and Gast, 2016). The incentives can be in the form of extra added time, as is the case with Vélib, or some cash discounts. The literature on user incentive schemes is not as plentiful as that on rebalancing through trucks (Fricker and Gast, 2016). Fricker and Gast (2016) present a two-choice model in which each user is provided with two station choices at the time of a rental and is given an incentive to choose the station with the lower load as a destination. They show that even if a fraction of the users make the intended choice, the number of imbalanced stations comes down dramatically.

Waserhole et al. (2012) solve an optimization model for setting the trip prices through a Markov Decision Process framework based on Continuous-Time Markov Chain. They present a Fluid Approximation approach and build a mathematical programming model for the fluid approximation of the Stochastic VSS Pricing Problem with continuous prices. A simulation model is also implemented to check the performance of fluid approximation heuristic. Pfrommer et al. (2014) introduced a tailored algorithm for dynamic route planning for multiple trucks for redistribution of bikes and then devised a system of price incentives computed based on Model Predictive Control (MPC) to draw users away from full or empty stations.

The problem of inventory imbalance and subsequent need for redistribution also exists in car

sharing systems. While in bike sharing a user can make a trip on any origin destination pair (DeMaio, 2009), most car sharing systems require users to return their vehicles back to the origin to avoid inventory imbalance (Boyacı et al., 2015). Effective relocation policies are essential for allowing one way trips and making car sharing systems more viable and user friendly.

Barth et al. (2004) use price incentives in a shared-use electric vehicle system called UCR Intellishare to encourage user-based vehicle relocation. Kek et al. (2009) devise a three-phase Optimization-Trend-Simulation (OTS) decision support system for vehicle relocation problem in car sharing systems. The simulation results suggest that using the suggested parameters from OTS can reduce the number of relocations by 37–41 %. Correia and Antunes (2012) recommend relocation operations at the end of each day in one way car sharing systems. They present three MIP models corresponding to three trip selection schemes. They suggest a scheme of price rate discrimination to incentivize or disincentivize certain trips. Correia et al. (2014) determine the added value of information and user flexibility for a one way car sharing system. Their approach is similar to what we propose. They allow users the flexibility to choose neighboring stations other than the intended origin and destination pair to address the stock imbalance problem.

In our paper, we aim to establish a system of cash discounts and penalties on user fee to influence user decisions. The operators have real time status data on all stations and based on this data a price vector can be calculated for all journeys. This information can be provided to users at Point of Sale or through mobile applications. As described by Waserhole et al. (2012), price incentives can be offered in discrete jumps with certain small increments, or they can be continuous within a predefined range. In this paper, pricing policies are dynamically changed in real time depending upon the current state of the system and the expected future demand. They can, however, be static, i.e., independent of the system's current state, and set in advance.

3 Model Formulation

The decision-making process for bike sharing systems is bi-level, as shown in Figure 1. Decisions regarding the location and size of stations as well as pricing are made by the operator running the system, while lower level journey decisions are made by the customers using the bikes. In this paper, we are mainly concerned with the system level pricing decision alone. In our model, the upper level (operator) objective is to minimize the total number of *imbalanced* stations. The lower level customer objective is to make a journey between two points at the minimum possible cost. The underlying assumption is that travelers will always take the minimum cost route. This section works to develop a detailed formulation of the problem that exploits the idea of strategic customer incentives. A few explanations, leading to the definition of *imbalanced* stations, are first in order.

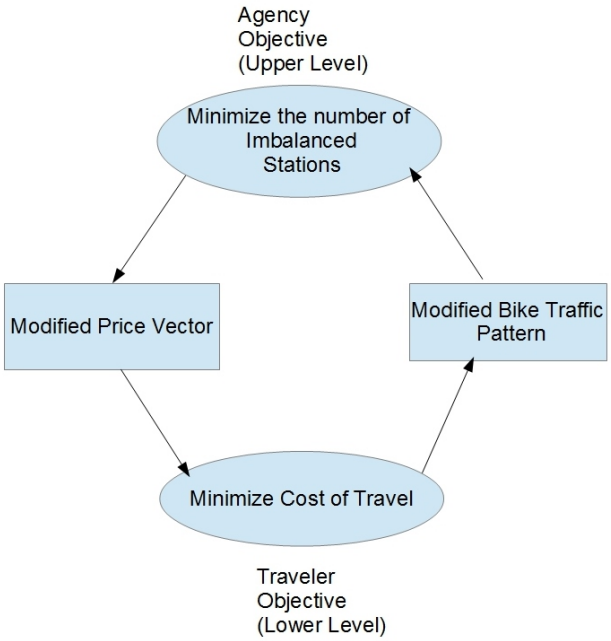


Figure 1: Bilevel Decision Making Process

3.1 Service Level Requirements

As described earlier, bike sharing systems are subject to two demands. On one hand, there exists the demand for bikes while on the other hand, there is the demand for empty docks, i.e., for parking the bikes at the end of the journey. A bike procured from one station is eventually parked back at the same station or any other station in the system. Every time a station is full or empty, a service opportunity is wasted. Most bike sharing systems keep track of the station inventory level. Some systems measure the number of instances (e.g. Capital Bikeshare in Washington, DC, USA) while others measure the fraction of time (e.g. Vélib in Paris) that the stations are empty or full. Operators do it for efficient rebalancing operations and to determine the need for expansion or reduction in the number of docks available at a particular station.

Schuijbroek et al. (2017) define a measurable Type-2 service level: the fraction of demand satisfied directly should be larger than β_i^- for pickups and larger than β_i^+ for returns, assuming no back-orders. We use the same definition:

$$\frac{\mathbb{E}[\text{Satisfied bike pickup demands}]}{\mathbb{E}[\text{Total bike pickup demands}]} \geq \beta_i^-,$$
$$\frac{\mathbb{E}[\text{Satisfied bike return demands}]}{\mathbb{E}[\text{Total bike return demands}]} \geq \beta_i^+.$$

Schuijbroek et al. (2017) then go on to establish a method to evaluate the values of minimum and maximum inventory for each station in the system, respectively designated as $I_i^{\min}$ and $I_i^{\max}$ for given β_i^- and β_i^+. They model the inventory I_i at station i as an $M/M/1/K$ queuing process

with customers in the queue for bikes or docks representing the inventory. We use their system of equations to evaluate the values of $I_i^{\min}$ and $I_i^{\max}$. For given β_i^- and β_i^+, starting inventory I_i^0 should ideally be rebalanced so that:

$$I_i^{\min} \leq I_i \leq I_i^{\max}.$$

If a station does not satisfy the above inequality, the station is termed *imbalanced*. Implicit in this definition of an *imbalanced* station is the idea of service level requirements. Only those stations that cannot fulfill the future demand for bikes and docks with a β_i^-, β_i^+ service level are considered imbalanced. It bears repeating that a station is lack-imbalanced when $I_i \leq I_i^{\min}$ and surplus-imbalanced when $I_i \geq I_i^{\max}$. It must be noted that $I_i^{\min}$ is always greater than or equal to 0 and $I_i^{\max}$ is always less than or equal to the maximum capacity of station i, i.e., the number of docks installed in station i. So a station can be imbalanced even when it is not completely full or empty. The following parameters are required to calculate $I_i^{\min}$ and $I_i^{\max}$ for station i: β_i^-, β_i^+, number of docks installed, arrival rate of users to pick up bikes, and arrival rate of users to return bikes.

3.2 A Bi-Level Formulation

This section presents the mathematical formulation of the bi-level problem. The first level represents the price change vector to minimize the number of *imbalanced* stations while the lower level corresponds to a minimum cost network flow problem which determines the route choices made by the travelers. Let $\mathcal{S}$ be the set of stations in the system. Let us assume that for a single journey, i is the origin station and j is the destination station where $i, j \in \mathcal{S}$. Let us also assume that (i, j) is the OD pair for a single one way trip and $\mathcal{W}$ is the set of all possible OD pairs, i.e., $(i, j) \in \mathcal{W}$. If the number of stations in the network is $|\mathcal{S}|$ then the number of OD pairs is $|\mathcal{S}|^2$. The distance between two stations is the distance along the shortest bike route and not the euclidean distance. In an urban setting, each station has a number of *neighborhood* stations. We assume that two stations less than 600 meters apart are *neighborhood* stations. On average, this accounts for about 6 minutes of walking (Bohannon, 1997). For every OD pair, both origin and destination have a number of neighborhood stations as illustrated in Figure 2. The colored circles around the stations represent the neighborhood radius.

Let us designate a full, directed network $G(\mathcal{S}, \mathcal{A}, \mathcal{P})$ where $\mathcal{S}$ denotes the set of nodes (or stations) in the network, $\mathcal{A}$ denotes the set of arcs and $\mathcal{P}$ denotes the set of paths. In this network a direct arc between any two nodes is also the shortest path between them. For every single OD pair in set $\mathcal{W}$, we construct a directed network $G^{ij}(\mathcal{S}^{ij}, \mathcal{A}^{ij}, \mathcal{P}^{ij})$. In this smaller network $\mathcal{S}^{ij}$ is the set of nodes that includes the origin station, the destination station and their respective neighborhood stations. $\mathcal{A}^{ij}$ is the set of directional arcs for every OD pair (i, j) and $\mathcal{P}^{ij}$ is the set of multiple paths from origin station to destination station. Observe that $\mathcal{S}^{ij} \subset \mathcal{S}$ but $\mathcal{A}^{ij} \not\subset \mathcal{A}$ and $\mathcal{P}^{ij} \not\subset \mathcal{P}$. As

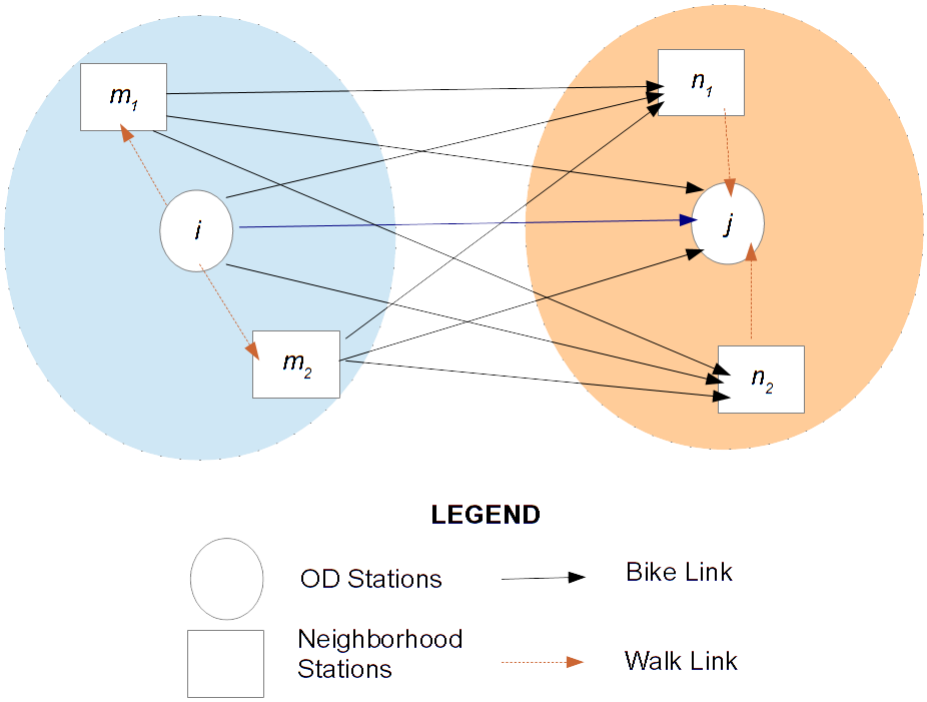


Figure 2: Network structure for every OD station pair

shown in Figure 2, a traveller intending to go from origin i to destination j can now take any one of the many paths available to her. If B_i and B_j are the numbers of neighborhood points of origin and destination, respectively, then the total number of alternate paths available is $(B_i + 1) \times (B_j + 1)$. Each path can consist of one, two or three arcs. For example, path $i \to j$ consists of one arc, while path $i \to m_2 \to n_1 \to j$ consists of three arcs. A traveller taking the latter path walks from origin i to its neighborhood point m_2, rents a bike and bikes to destination's neighborhood point n_1, then parks the bike at an empty dock and walks to the destination j. In Figure 2, *bike links* are represented by black arrows, while red arrows represent *walk links*. Note that every path contains only one bike link.

For a given OD pair (i, j), let c^{ij}_{mn} be the cost matrix for all links (m, n) where $(m, n) \in \mathcal{A}^{ij}$ and $m, n \in \mathcal{S}$. The cost c^{ij}_{mn} of traversing a link consists of various subcosts. These include the cost of walking, cost of biking, and the price of renting a bike to travel on a bike link. The rental price is determined by the operator. Let these subcosts be denoted by w^{ij}_{mn}, v^{ij}_{mn} and p_{mn}, respectively. The costs w^{ij}_{mn}, v^{ij}_{mn} are calculated as

$$w^{ij}_{mn} = \nu_3 t^{ij}_{mn} \qquad \text{and} \qquad v^{ij}_{mn} = \nu_2 t^{ij}_{mn}.$$

In the above expressions t^{ij}_{mn} is the time of travel between two stations m and n using a bike. ν_2 and ν_3 are the coefficients that convert distance between stations to the cost depending upon

biking and walking travel times, respectively. As stated earlier, the price p_{mn} is defined by the operator. The price associated with a single given link (m, n) in multiple OD pair networks is the same. For example, the value of $p_{19,3}$ is the same for all $(19, 3)$ links in all the feasible OD pair networks. Hence, all p_{mn} values in an OD pair network form a price vector associated with the OD pair (m, n) and defined by the operator.

Also, for every OD pair (i, j), let $(m, n) \in \mathcal{A}_B^{ij}$ be the links where a bike is used to traverse and hence the cost of travel consists of p_{mn} and v_{mn}^{ij}. Similarly, let $(m, n) \in \mathcal{A}_W^{ij}$ be the links where a traveller walks and hence the cost of travel consists of w_{mn}^{ij} alone. The price vector p_{mn} further consists of a fixed component and a variable component,

$$p_{mn} = p_{mn}^0 + q_{mn}, \tag{1}$$

where p_{mn}^0 is the fixed base price set by the operator and depending on the time of the journey (m, n) while q_{mn} is the variable component capturing a penalty or incentive within a fixed range $[q_{mn}^{\min}, q_{mn}^{\max}]$, determined again by the operator, with

$$q_{mn}^{\min} \leq q_{mn} \leq q_{mn}^{\max}. \tag{2}$$

Note that all the costs mentioned above except q_{mn} are fixed costs depending only on travel time. By modifying the price change vector q_{mn}, the operator can modify the cost matrix and influence the traveller's decision about which path to take. Here we introduce a binary variable x_{mn}^{ij} which is equal to 1 if link (m, n) is used to travel between OD pair (i, j), and 0 otherwise. Since for every OD pair, one or more links (m, n) can be used to travel between origin i and destination j, one or more variables x_{mn}^{ij} can take on the value of 1. The outcome x_{mn}^{ij} of the lower level program which depends on traveller choices is used at the upper level to calculate the bike inventory I_i for each station i at each instant. This information feeds into the upper level objective function to minimize the number of imbalanced stations. To calculate the inventory, we only require links where a bike is used. Let parametric vector α_{mn}^{ij} have value 1 for a walk link and 0 for a bike link. Similarly, let parametric vector δ_{mn}^{ij} have value 1 for a bike link and 0 for a walk link. Then the product $\delta_{mn}^{ij} x_{mn}^{ij}$ is 1 only if a link (m, n) used to travel between an OD pair is a bike link and 0 if it is not a bike link. Observe that for every OD pair combination, $\delta_{mn}^{ij} x_{mn}^{ij}$ is 1 for only one link (m, n). Table 1 details the mathematical notation used in the model.

The Upper Level Pricing Problem of the System Operator

Using the notation defined in Table 1, we formulate an upper level optimization problem to determine the price change vector q_{mn} as follows:

$$\text{(U)} \qquad \min \quad \sum_i y_i + \sum_i z_i \tag{3}$$

Table 1: Mathematical Notation

Sets	
$\mathcal{S}$	Set of stations
$\mathcal{W}$	Set of Origin-Destination (OD) pairs
$\mathcal{A}$	Set of arcs in the directed network for every OD pair
$\mathcal{P}$	Set of Possible Paths for each OD pair
Parameters	
$I_i^{\max}$	Maximum number of permissible bikes at a station $i \in \mathcal{S}$, beyond which the station is considered imbalanced; obtained from preprocessing steps in Section 3.1
$I_i^{\min}$	Minimum number of permissible bikes at a station $i \in \mathcal{S}$, beyond which the station is considered imbalanced; obtained from preprocessing steps in Section 3.1
I_i^0	Starting level of bike inventory at station $i \in \mathcal{S}$ at the beginning of time horizon considered
C_i	Capacity of a station $i \in \mathcal{S}$
M	A very large number or big-M used for modeling if-else type and disjunctive constraints
δ_{mn}^{ij}	Binary parameter, 1 if $(m,n) \in \mathcal{A}^{ij}$ is a bike link and 0 if it is a walking link
α_{mn}^{ij}	Binary parameter, 1 if $(m,n) \in \mathcal{A}^{ij}$ is a walking link and 0 if it is a bike link
d^{ij}	Demand for OD pair $(i,j) \in \mathcal{W}$ during the time horizon considered
B_i	The number of neighborhood stations for a station $i \in \mathcal{S}$
Variables	
I_i	Variable representing the current (final) level of bike inventory at station $i \in \mathcal{S}$ at the end of the time horizon considered
$\tilde{I}_i$	Variable representing truncated level of bike inventory at station $i \in \mathcal{S}$ at the end of the time horizon considered
y_i	Binary variable, 1 if a station $i \in \mathcal{S}$ is a surplus station where surplus station is a station where $I_i > I_i^{\max}$
z_i	Binary variable, 1 if a station $i \in \mathcal{S}$ is a lack station where lack station is a station where $I_i < I_i^{\min}$
q_{mn}	Variable representing price change (Incentive or penalty) for traversing link $(m,n) \in \mathcal{A}$
x_{mn}^{ij}	Binary variable, 1 when link $(m,n) \in \mathcal{A}^{ij}$ is used to travel between OD pair (i,j)
a_i	Auxiliary binary variable for a station $i \in \mathcal{S}$
b_i	Auxiliary binary variable for a station $i \in \mathcal{S}$

subject to

$$I_i - I_i^{\max} \leq My_i \quad \forall i \in \mathcal{S}, \tag{4}$$

$$I_i^{\min} - I_i \leq Mz_i \quad \forall i \in \mathcal{S}, \tag{5}$$

$$I_i = I_i^0 - \sum_{(i,j)\in\mathcal{W}} \sum_{n\in\mathcal{S}^{ij}} x_{in}^{ij}\delta_{in}^{ij}d^{ij} + \sum_{(i,j)\in\mathcal{W}} \sum_{n\in\mathcal{S}^{ij}} x_{ni}^{ij}\delta_{ni}^{ij}d^{ij} \quad \forall i \in \mathcal{S}, \tag{6}$$

$$q_{mn}^{\min} \leq q_{mn} \leq q_{mn}^{\max} \quad \forall (m,n) \in \mathcal{A}, \tag{7}$$

where values of variables x_{mn}^{ij} depend on the minimum cost route choice of bike users and is determined by a lower level problem to be introduced later. It is possible in constraint (6) for the inventory I_i to go beyond capacity C_i or fall below zero as we do not explicitly consider any bounds on the inventory. However, we use a variable $\tilde{I}_i$ to represent the truncated value of inventory if I_i goes beyond capacity or below zero. Following inequalities are used as constraints in our model to represent the relationship between the real final inventory and the truncated inventory. The auxiliary binary variables a_i and b_i take a value of 1 when I_i is below zero or above capacity, respectively. If one of these binary variables a_i or b_i is equal to 1, the corresponding truncated inventory value $\tilde{I}_i$ is equal to 0 or C_i, respectively. If I_i is between 0 and C_i, i.e., a_i and b_i are both 0, the truncated counterpart is simply equal to I_i. The value $I_i - \tilde{I}_i$ summed over all i is the portion of demand that remains unsatisfied.

$$I_i \leq C_i + Mb_i \quad \forall i \in \mathcal{S}, \tag{8}$$

$$I_i \leq M(1 - a_i) \quad \forall i \in \mathcal{S}, \tag{9}$$

$$I_i \geq C_i - M(1 - b_i) \quad \forall i \in \mathcal{S}, \tag{10}$$

$$I_i \geq -Ma_i \quad \forall i \in \mathcal{S}, \tag{11}$$

$$a_i + b_i \leq 1 \quad \forall i \in \mathcal{S}, \tag{12}$$

$$\tilde{I}_i \leq I_i + Ma_i + Mb_i \quad \forall i \in \mathcal{S}, \tag{13}$$

$$\tilde{I}_i \geq I_i - M(a_i + b_i) \quad \forall i \in \mathcal{S}, \tag{14}$$

$$\tilde{I}_i \geq M(1 - a_i) \quad \forall i \in \mathcal{S}, \tag{15}$$

$$\tilde{I}_i \geq C_i - M(1 - b_i) \quad \forall i \in \mathcal{S}, \tag{16}$$

$$0 \leq \tilde{I}_i \leq C_i \quad \forall i \in \mathcal{S}, \tag{17}$$

Using this approach, we avoid the possible infeasibility in our model by not considering any explicit bounds on the final inventory I_i. At the same time, however, we make sure that any deviations from the realistic range of $[0, C_i]$ are accounted for. Later in our computational experiments, we report the unsatisfied demand for comparison purposes between our pricing approaches and the situation without pricing.

The Lower Level Routing Problem of the Bike Users

In the lower level problem bike users who want to travel from origin i to destination j use minimum cost paths so the objective is,

$$\text{(L)} \qquad \min \quad \sum_{(i,j)\in\mathcal{W}} \sum_{(m,n)\in\mathcal{A}^{ij}} c_{mn}^{ij} x_{mn}^{ij} \tag{18}$$

subject to

$$\sum_{(m,n)\in\mathcal{A}^{ij}} x_{mn}^{ij} - \sum_{(n,m)\in\mathcal{A}^{ij}} x_{nm}^{ij} = e_m^{ij} \quad \forall m \in \mathcal{S}^{ij}, (i,j) \in \mathcal{W}, \tag{19}$$

$$x_{mn}^{ij} \in \{0,1\} \quad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}, \tag{20}$$

where

$$c_{mn}^{ij} = \begin{cases} p_{mn}^0 + q_{mn} + v_{mn}^{ij} & \forall (m,n) \in \mathcal{A}_B^{ij} \\ w_{mn}^{ij} & \forall (m,n) \in \mathcal{A}_W^{ij}. \end{cases} \tag{21}$$

In (19), e_m^{ij} takes the value 1 (respectively, -1), if node m is the origin (respectively, destination) of the trip, and 0 otherwise.

3.3 A Single Level Reformulation

In the lower level problem the integrality requirement for variables x_{mn}^{ij} can be replaced by the constraints $x_{mn}^{ij} \geq 0$. This is so because the lower level program is a minimum cost network flow problem: its right hand side can only be integer and the coefficient matrix in the left hand side forms a totally unimodular matrix. Now we can represent the lower problem by its optimality conditions or Karush-Kuhn-Tucker (KKT) conditions of its LP relaxation. Since the lower-level problem is a linear optimization problem, we can replace it by following KKT optimality conditions:

$$\delta_{mn}^{ij}(p_{mn}^0 + q_{mn} + v_{mn}^{ij}) + \alpha_{mn}^{ij} w_{mn}^{ij} - \lambda_m^{ij} + \lambda_n^{ij} - \mu_{mn}^{ij} = 0 \quad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}, \tag{22}$$

$$\sum_{(m,n)\in\mathcal{A}^{ij}} x_{mn}^{ij} - \sum_{(n,m)\in\mathcal{A}^{ij}} x_{nm}^{ij} = e_m^{ij} \quad \forall m \in \mathcal{S}^{ij}, (i,j) \in \mathcal{W}, \tag{23}$$

$$x_{mn}^{ij} \geq 0 \quad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}, \tag{24}$$

$$\mu_{mn}^{ij} \geq 0 \quad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}, \tag{25}$$

$$\lambda_m^{ij} \text{ free} \quad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}, \tag{26}$$

$$-\mu_{mn}^{ij} x_{mn}^{ij} = 0 \quad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}, \tag{27}$$

where the variables λ_n^{ij} and μ_{mn}^{ij} are the dual variables corresponding to constraints (19) and (20) in the primal problem, respectively. The complementary slackness conditions (27) are non-convex, and we can linearize them taking advantage of binary nature of x_{mn}^{ij}. Suppose M is a very large

number then the linearized constraint would be:

$$\mu_{mn}^{ij} \leq M(1 - x_{mn}^{ij}) \quad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}. \tag{28}$$

We state the final formulation of our single level optimization model here:

$$\text{(P)} \quad \min \quad \sum_i y_i + \sum_i z_i \tag{29}$$

subject to

$$I_i - I_i^{\max} \leq M y_i \qquad \forall i \in \mathcal{S}, \tag{30}$$

$$I_i^{\min} - I_i \leq M z_i \qquad \forall i \in \mathcal{S}, \tag{31}$$

$$I_i = I_i^0 - \sum_{(i,j)\in\mathcal{W}} \sum_{n\in\mathcal{S}^{ij}} x_{in}^{ij} \delta_{in}^{ij} d^{ij} + \sum_{(i,j)\in\mathcal{W}} \sum_{n\in\mathcal{S}^{ij}} x_{ni}^{ij} \delta_{ni}^{ij} d^{ij} \quad \forall i \in \mathcal{S}, \tag{32}$$

$$q_{mn}^{\min} \leq q_{mn} \leq q_{mn}^{\max} \qquad \forall (m,n) \in \mathcal{A}, \tag{33}$$

$$\delta_{mn}^{ij}(p_{mn}^0 + q_{mn} + v_{mn}^{ij}) + \alpha_{mn}^{ij} w_{mn}^{ij} - \lambda_m^{ij} + \lambda_n^{ij} - \mu_{mn}^{ij} = 0 \qquad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}, \tag{34}$$

$$\sum_{(m,n)\in\mathcal{A}^{ij}} x_{mn}^{ij} - \sum_{(n,m)\in\mathcal{A}^{ij}} x_{nm}^{ij} = e_m^{ij}, \qquad \forall m \in \mathcal{S}^{ij}, (i,j) \in \mathcal{W}, \tag{35}$$

$$x_{mn}^{ij} \geq 0 \qquad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}, \tag{36}$$

$$\mu_{mn}^{ij} \geq 0 \qquad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}, \tag{37}$$

$$\lambda_m^{ij} \text{ free} \qquad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}, \tag{38}$$

$$\mu_{mn}^{ij} \leq M(1 - x_{mn}^{ij}) \qquad \forall (m,n) \in \mathcal{A}^{ij}, (i,j) \in \mathcal{W}. \tag{39}$$

In the final model represented by (29)–(39) alongside (8)–(17), the objective function represents the total number of imbalanced stations. Constraints (30) and (31) make sure the corresponding binary variables y_i and z_i take up a value of 1 when inventory bounds are violated. Constraint (32) is the inventory update constraint while (33) bounds the price incentives or penalties that can be given to riders. Constraints (34) and (35) are respectively the dual and primal feasibility constraints and (39) is complementary slackness condition. Finally, (8)–(17) work to limit the truncated inventory variable I_i^T between 0 and C_i. It is important to point out that our choice of the objective function is an obvious departure from the functions that try to minimize the system-wide number of unsatisfied demands. Rather than trying to minimize the system-wide service level, we look at the service level for every single station and try to achieve a certain desirable value for it for as many stations as possible. Also, the alternative objective functions may provide us a price vector but they will naturally redistribute the unsatisfied demand between different stations in equal measure. This outcome is unfavorable to our basic idea of trying to create *hub* stations and concentrating the unsatisfied demand on those stations, thus making the subsequent manual

truck repositioning operation much easier and simpler.

In the single level model (P) that we use for numerical experiments, for every OD pair (i, j), the number of decision variables is calculated as $2B_iB_j + 3(B_i + B_j)$. For a network of 200 stations, this amounts to approximately 1 million variables and 1 million constraints. Therefore, while this formulation can be useful for a comparative evaluation of other methods (with small-scale problems), more scalable solutions are desirable for practical purposes. Another issue that we face is that for every OD pair (i, j), the optimal price vector prescribed by the single level model (P) is prone to give us path costs that can be approximately equal to each other. We give an example of this phenomenon in Appendix A of Supplementary Materials.

3.4 The Issue of Approximately Equal Path Costs

The issue of approximately equal path costs is instructive for anyone trying to optimally determine a price vector. Since lower level decision makers are not explicitly aware of the upper level objective of minimizing the number of imbalanced stations, given ambiguous choices, they are likely to make decisions unfavorable to the upper level agency.

To take the effect of this fuzziness into account, we carry out some post processing steps after getting the pricing vector from our model (P). We construct a new model called ($\bar{\text{P}}$), a slightly modified version of the original model (P), and incorporate the price vector recommended by P, designated as q_{mn}^{opt}, into the new model as a constraint. We state the formulation of the model ($\bar{\text{P}}$) here:

$$(\bar{\text{P}}) \qquad \max \quad \sum_i y_i + \sum_i z_i \tag{40}$$

s.t.

(32), (34)–(39), (8)–(17),

$$I_i^{\max} - I_i \leq M(1 - y_i) \quad \forall i \in \mathcal{S}, \tag{41}$$

$$I_i - I_i^{\min} \leq M(1 - z_i) \quad \forall i \in \mathcal{S}, \tag{42}$$

$$q_{mn} = q_{mn}^{\text{opt}} \quad \forall (m, n) \in \mathcal{A}, \tag{43}$$

In the new model, the objective function is maximized, and constraints (30), (31) and (33) are replaced by constraints (41), (42) and (43). As evident, (41) and (42) are modified to correspond to the changed objective function. Constraint (43) assigns the optimal price vector values from (P) to the price variables in the new model ($\bar{\text{P}}$). The rest of the constraints in ($\bar{\text{P}}$) are exactly the same as the model (P). The new model ($\bar{\text{P}}$) does not try to determine a pricing vector. It uses the optimal vector suggested by (P) to come up with an alternate set of lower level decisions (x). These decisions correspond to people making choices, in response to the predetermined pricing vector, that are unfavorable to the upper level objective (represented by maximization).

Using ($\bar{\text{P}}$), we are interested in the worst case scenario and answer the following question: What happens if we apply the optimal pricing vector in reality, but rather than minimizing the number of imbalanced stations (people making decisions favorable to the agency), we maximize the number of imbalanced stations (people making decisions unfavorable to the agency)? What would be the worst case result of applying the suggested price vector? The objective function value of ($\bar{\text{P}}$) thus serves as a strict upper bound on the number of imbalanced stations if the optimal price vector recommended by model (P) is to be implemented.

4 Iterative Price Adjustment Scheme

This section presents a novel heuristic method to solve the problem in model (P). Keeping the upper level problem intact, we work with the lower level problem: starting with an initial price vector, we use that vector to calculate, rather than optimally determine, the initial set of route choices (x). Given these route choices, the upper level model is then used to calculate the value of the objective function. Based on this value, the decisions about the subsequent modification in the initial price vector are made. That is, instead of performing the exact minimum cost optimization to determine the optimum price change vector q_{mn}, we determine q_{mn} heuristically. The overall objective of minimizing the number of imbalanced stations remains the same. This section presents an iterative heuristic method called the *Iterative Price Adjustment Scheme (IPAS)* that produces a price change vector using discrete increments and decrements on the price between different station categories.

The proposed heuristic scheme IPAS relies on a simple decision making process that classifies the bike stations into different categories based on their starting inventory level I_i^0, the maximum and minimum inventory values, and the demand in the next time period. Based on these factors we divide the stations into six different types defined in Section 4.1. Although, we formally define the categories in Section 4.1, the basic purpose of categorizing the stations is to identify station pairs where price should increase in discrete jumps, decrease in discrete jumps or stay the same. If a pair of stations m and n are both balanced stations (as defined in Section 4.1), there is no need to change the price between them and hence the corresponding $q_{mn} = 0$. Similarly, if a pair of stations m and n are both "slightly" imbalanced stations (as defined in Section 4.1), it may be advisable to change the price vector between them so as to reduce their respective imbalance. Finally, if a station m is "highly" imbalanced (or hub station) (as defined in Section 4.1) such that it is not possible to make it balanced through pricing changes, it may be desirable to make such a station even more imbalanced for the sake of preserving the balance at other neighboring stations.

As the primary objective of the pricing problem lies in identifying the *hub* stations, while simultaneously reducing the number of such stations, we will first develop a categorizing heuristic to identify such stations where accumulation happens naturally. If a station is expected to have a significant surplus and there are no stations in its neighborhood where lack accumulation is

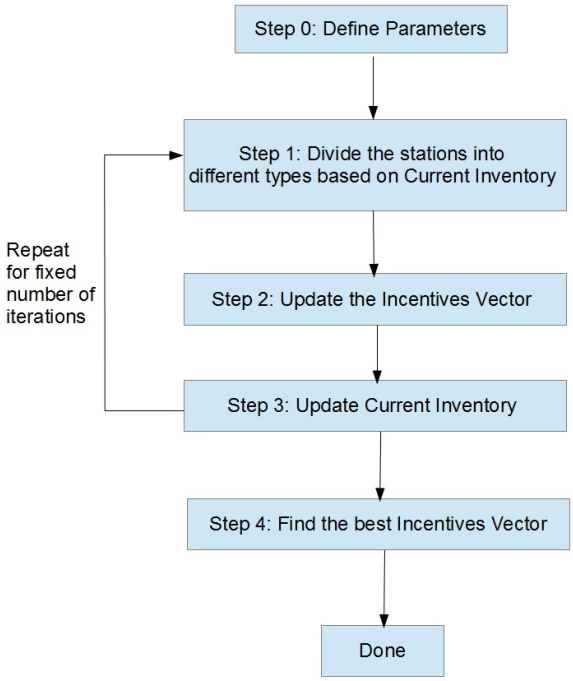


Figure 3: Flow chart of the Iterative Price Adjustment Scheme (IPAS)

expected, then such a station is a suitable candidate for a surplus hub. For this kind of station, to help further accumulation, we can decrease prices for bike returns to this station and simultaneously increase regular prices on bike checkouts from the station. For a likely lack accumulation station (Hub station for docks), we can do the opposite: increase regular prices on bike returns and offer discounts for checkouts. Based on such selection of hub stations and price changes, we may evaluate the objective function in (P) and iteratively adjust prices. This process is repeated until the designated number of iterations. After the iterations run out, this scheme gives the price change vector with minimum number of imbalanced stations as its output. Figure 3 shows a flow chart for the proposed iterative price adjustment scheme.

Step 0 (Initialization): Define the Algorithm Parameters

First of all, we define some parameters that will be used in the proposed scheme. We borrow much of our notation from Table 1 while we describe the new notation used for our heuristic method in Table 2. The basic parameters are as follows: The parameters $I_i^{\max}$, $I_i^{\min}$, I_i^0, and I_i as defined in Table 1; k is an integer number representing the running iteration of the algorithm; and q_{mn}^k is the Price change (Incentive or Penalty) vector for traversing link (m, n) during iteration number k of the algorithm where q_{mn} for a given link (m, n) is the same for all OD pairs. The starting price change vector called q_{mn}^k for $k = 0$ can generally have all values equal to 0.

Parameter θ_i^{in} describes the maximum number of bikes that can possibly come into a certain

Table 2: Mathematical Notation for IPAS

q_{mn}^k	Price change (Incentive or penalty) value for traversing link $(m,n) \in \mathcal{A}$ in iteration k
Δq_{mn}	The change (decrease or increase) in incentives for link $(m,n) \in \mathcal{A}$ between two consecutive iterations of the algorithm.
k	An integer number representing the running iteration of the algorithm.
θ_i^{in}	Parameter representing the maximum number of bikes that can possibly arrive into a station $i \in \mathcal{S}$
θ_i^{out}	Parameter representing the maximum number of bikes that can possibly go out of a station $i \in \mathcal{S}$
ρ_i	Parameter representing the rank ratio of a station $i \in \mathcal{S}$ which is used to rank the stations based on the quantum of their imbalance.
κ	User determined parameter used to control the level of accumulation in the network intended. The value of κ varies between 0 and 1 with larger value giving smaller number of hub (accumulation) stations.
σ_s	The parameter representing the small discrete jumps in the price vector.
σ_l	The parameter representing the large discrete jumps in the price vector.

station i, which is the sum of the number of bikes coming in from all the other stations of the network into the station i and the number of bikes coming in from all the other stations of the network into the neighborhood stations of i. We assume that if big enough incentives were offered, all the traffic coming into neighborhood points of i will be redirected to i and users will take rest of the trip walking.

Parameter θ_i^{out} describes the maximum number of bikes that can possibly go out of a certain station i, which is the sum of the number of bikes going to all the other stations of the network from the station i and the number of bikes going to all the other stations of the network from the neighborhood stations of i. We assume that if big enough incentives were offered, all the traffic heading from neighborhood points of i will be redirected through i and users will walk to i and take a bike forward to their destinations.

The *rank ratio*, denoted by ρ_i for each $i \in \mathcal{S}$, is the parameter used to rank different stations based on their current inventory status,

$$\rho_i = \frac{(I_i - I_i^{\max})^2 + (I_i - I_i^{\min})^2}{(I_i^{\max} - I_i^{\min})^2}. \tag{44}$$

The value of ρ_i varies from 0.5 to ∞ with lesser values implying a more balanced station and larger values implying a more imbalanced station. A station with value of $\rho_i \geq 1$ implies that either of the two inequalities $I_i \leq I_i^{\max}$ and $I_i \geq I_i^{\min}$ are unsatisfied. The metric ρ_i looks at the current inventory and min-max range of a station to define the "distance" of the current inventory of the

station from its min-max extremes. If for a station i, $I_i^{\min} = 0$, $I_i^{\max} = 10$, and $I_i = 5$, then its ρ_i value will be 0.5, which is the minimum possible. Any deviation from this middle value will increase ρ_i and the station will be considered less balanced. We could very well have constructed some other measures but the underlying logic would stay the same. The purpose of the rank ratio is to sort stations that fall into one of the six categories listed below.

4.1 Step 1: Define Station Types

Each station falls in only one of the six subsets of stations based on the following definitions:

HUB Stations with Bikes Needed (HSBN) have very low bike inventory, so that we would not have enough bikes to make this station balanced again, even if the incoming bike inventory from all the neighboring stations were to be redirected; hence they become good candidates for lack accumulation stations. Instead of making such a station balanced by gaining bikes, we make it even more imbalanced by losing bikes further. Such a station must satisfy the following inequality:

$$\kappa I_i^{\min} - I_i^0 > \theta_i^{\text{in}}.$$

where κ is a constant with value ranging from 0 to 1. A higher value of κ means less HUB stations and vice versa. For κ equal to 1, the above inequality implies that it will be impossible to raise I_i^0 to a value equal to or greater than $I_i^{\min}$ even if all the bikes from the neighborhood stations of i were redirected to i.

HUB Stations with Docks Needed (HSDN) have very high bike inventory so that it is very difficult to lose enough bikes to make the station balanced again, even if all the outgoing bikes from its neighborhood stations were to be redirected through it. Hence these stations become good candidates for surplus accumulation station. So instead of making it balanced by losing bikes, we make it even more imbalanced by accumulating more bikes. Such station must satisfy the following inequality

$$I_i^0 - \kappa I_i^{\max} > \theta_i^{\text{out}}.$$

Imbalanced stations with Bikes Needed (ISBN) are stations which cannot satisfy user demand at β_i^- service level with their current bike inventory, although they might not be completely vacant. Such station must satisfy the following two inequalities:

$$I_i < I_i^{\min},$$
$$\kappa I_i^{\min} - I_i^0 \leq \theta_i^{\text{in}}.$$

The first inequality defines an imbalanced Station with Bikes Needed while the second makes sure that the station is not an HSBN.

Imbalanced stations with Docks Needed (ISDN) are stations which cannot satisfy user demand at β_i^+ service level with their current inventory, although they might not be filled to capacity. Such stations must satisfy the following two inequalities:

$$\begin{aligned} I_i &> I_i^{\max}, \\ I_i^0 - \kappa I_i^{\max} &\leq \theta_i^{\text{out}}. \end{aligned}$$

The first inequality defines an imbalanced Station with Docks Needed while the second makes sure that the station is not an HSDN.

Balanced Stations with Bikes Needed (BSBN) are stations which satisfy the inequality $I_i \geq I_i^{\min}$, and hence, are balanced by definition but their current inventory is relatively closer to $I_i^{\min}$ than $I_i^{\max}$. Such stations must satisfy the following two inequalities:

$$\begin{aligned} I_i &\geq I_i^{\min}, \\ \kappa I_i^{\min} - I_i^0 &\leq \theta_i^{\text{in}}. \end{aligned}$$

Balanced Stations with Docks Needed (BSDN) are stations which satisfy the inequality $I_i \leq I_i^{\max}$, and hence, are balanced by definition but their current inventory is relatively closer to $I_i^{\max}$ than $I_i^{\min}$. Such stations must satisfy the following two inequalities:

$$\begin{aligned} I_i &\leq I_i^{\max}, \\ I_i^0 - \kappa I_i^{\max} &\leq \theta_i^{\text{out}}. \end{aligned}$$

We denote each subset of stations by $\mathcal{S}_{\text{HSDN}}$, $\mathcal{S}_{\text{HSDN}}$, $\mathcal{S}_{\text{ISBN}}$, $\mathcal{S}_{\text{ISDN}}$, $\mathcal{S}_{\text{BSBN}}$, and $\mathcal{S}_{\text{BSDN}}$ respectively. Within each subset, we order stations from the most balanced to the least balanced.

4.2 Step 2: Update the Incentive Vector

After the network has been divided into six exclusive sets of stations, it is time to update the price change vector based on the following equation:

$$q_{mn}^k = q_{mn}^{k-1} + \Delta q_{mn},$$

where Δq_{mn} is the price update that we will calculate as follows.

For each link (m, n), the price change is updated in discrete jumps. We define σ_s and σ_l as small and large discrete jumps. Their values are constant numbers. We use \$0.017 and \$0.1 for

Table 3: Calculating values of Δq_{mn}, where h_m and h_n are the order indices starting from zero within the set $\mathcal{S}_{\text{ISBN}}$ of stations m and n, respectively—for example, if station m is the most balanced station in the set $\mathcal{S}_{\text{ISBN}}$, then h_m=0—and similarly, g_m and g_n are the order indices starting from zero within the set $\mathcal{S}_{\text{ISDN}}$. We let $\Delta q_{mn} = 0$ for all $m \in \mathcal{S}_{\text{BSBN}}$ and $n \in \mathcal{S}_{\text{BSDN}}$.

Type of Station m	Type of Station n: HSBN	HSDN	ISBN	ISDN
HSBN	0	$-\sigma_s$	$-\sigma_l \sin \frac{\pi \cdot h_n}{\lvert\mathcal{S}_{\text{ISBN}}\rvert}$	$\sigma_s \sin \frac{\pi \cdot g_n}{\lvert\mathcal{S}_{\text{ISDN}}\rvert}$
HSDN	σ_s	0	$-\sigma_s \sin \frac{\pi \cdot h_n}{\lvert\mathcal{S}_{\text{ISBN}}\rvert}$	$\sigma_l \sin \frac{\pi \cdot g_n}{\lvert\mathcal{S}_{\text{ISDN}}\rvert}$
ISBN	$\sigma_l \sin \frac{\pi \cdot h_m}{\lvert\mathcal{S}_{\text{ISBN}}\rvert}$	$\sigma_s \sin \frac{\pi \cdot h_m}{\lvert\mathcal{S}_{\text{ISBN}}\rvert}$	$\frac{\sigma_s}{2}\left(-\cos \frac{\pi \cdot h_n}{\lvert\mathcal{S}_{\text{ISBN}}\rvert} + \cos \frac{\pi \cdot h_m}{\lvert\mathcal{S}_{\text{ISBN}}\rvert} \right)$	$\sigma_l \sin \frac{\pi \cdot (h_n + g_m)}{\lvert\mathcal{S}_{\text{ISBN}}\rvert + \lvert\mathcal{S}_{\text{ISDN}}\rvert}$
ISDN	$-\sigma_s \sin \frac{\pi \cdot g_m}{\lvert\mathcal{S}_{\text{ISDN}}\rvert}$	$-\sigma_l \sin \frac{\pi \cdot g_m}{\lvert\mathcal{S}_{\text{ISDN}}\rvert}$	$-\sigma_l \sin \frac{\pi \cdot (h_n + g_m)}{\lvert\mathcal{S}_{\text{ISBN}}\rvert + \lvert\mathcal{S}_{\text{ISDN}}\rvert}$	$\frac{\sigma_s}{2}\left(-\cos \frac{\pi \cdot g_n}{\lvert\mathcal{S}_{\text{ISDN}}\rvert} + \cos \frac{\pi \cdot g_m}{\lvert\mathcal{S}_{\text{ISDN}}\rvert} \right)$

experimental purposes. As shown in Table 3 the discrete jumps can thus vary from −\$0.017 to \$0.017 when σ_s is used or from −\$0.1 to \$0.1 when σ_l is used. The trigonometric distribution functions used to calculate Δq_{mn} are plotted in Figure 4. We have not included stations of type BSBN and BSDN in the table because the price change for travel between and to the *balanced* stations is zero. The stations belonging to each of the described categories are sorted according to their rank ratio values and stored as a sorted array. The stations in the beginning of the array are more balanced (or less imbalanced) and those at the end are less balanced (or more imbalanced). When deciding the prices between stations of each category, we take into account the position of each station in their respective arrays.

We use trigonometric functions because their shape allows us to have smoothly increasing or decreasing price changes based on the position of a station in an array. For instance, the value of Δq_{mn} between stations of HSBN and ISBN types follows the distribution in Figure 4a with stations at both ends of the $\mathcal{S}_{\text{ISBN}}$ vector getting smaller decrements while those in the middle getting the maximum decrements. This makes sure that our pricing favors the movement of bikes from stations in the subset $\mathcal{S}_{\text{HSDN}}$ to the middle stations of the subset $\mathcal{S}_{\text{ISBN}}$. The reason for favoring middle stations is that the stations at the start are substantially close to being balanced and hence do not need large changes in price while the stations at the end are critically imbalanced and it is difficult to make them balanced again using price changes. The middle stations in the array are

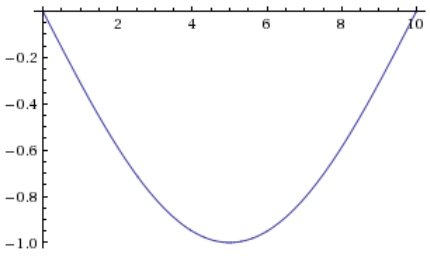
moderately imbalanced and hence are more prone to respond to price changes. Similarly, the value of Δq_{mn} between stations of ISBN and ISDN types follows the distribution in Figure 4d. The price increment is maximum for stations in the middle making it difficult for bikes to move from stations in subset $\mathcal{S}_{\text{ISBN}}$ to stations in subset $\mathcal{S}_{\text{ISDN}}$, so on and so forth. The logic behind using trigonometric functions is their ability to give us two sided minima. Admittedly, we could have used a simple triangle function for simpler situations in 4a and 4b but trigonometric functions are better suited for constructing the more complex functions in 4c, 4d and 4e.

4.3 Step 3: Calculate the Number of Imbalanced stations

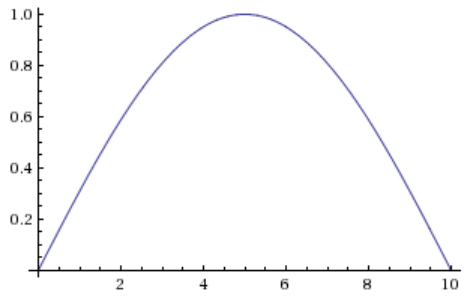
We use (U) to calculate the number of imbalanced stations given the price change vector q_{mn}. We store this value for every iteration. The upper level problem (U) is also used to calculate the updated values of I_i. After this, Step 1 is repeated with updated I_i and stations are again categorized into different types based on Step 1. This process is repeated until the designated number of iterations.

4.4 Step 4: Find the Best Price Change Vector

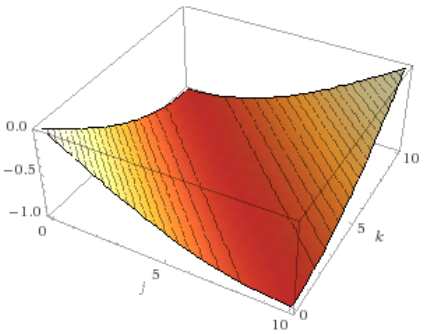
The number of imbalanced stations calculated in Step 3 is compared to the minimum value. If the current price change vector improves on the objective function, the vector and the objective value are stored. After this, Step 0 is repeated with updated I_i and the values of some of the parameters are calculated based on new I_i. Stations are again categorized into different types based on Step 1. This process is repeated until the designated number of iterations. After the iterations run out, this scheme gives the price change vector with minimum number of imbalanced stations as its output. We chose the number of iterations as our stopping criteria because of various reasons. Generally, in the literature, the algorithms are stopped when the improvement between consecutive iterations is very small. We do not use this stopping criterion in our paper. The objective that we are comparing across iterations is the number of imbalanced stations which is an integer number with a very small range. It is not possible to look at two consecutive iterations and decide to stop the algorithm because no improvement was achieved. For most consecutive iterations of IPAS, there is indeed no improvement in the objective function. Also the objective function value goes up and down through the iterations. It does not linearly decrease. Hence, we run a large number of iterations and record the iteration, and the corresponding pricing vector, that gave us the best solution first. If the optimal iteration is very close to the allowable number of iterations, we can simply choose to increase the number of iterations.



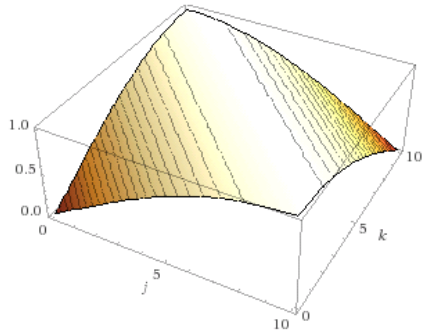
(a) $-\sin\frac{\pi\cdot h}{|\mathcal{S}_{\text{ISBN}}|}$ for $h \in [0, |\mathcal{S}_{\text{ISBN}}| - 1]$



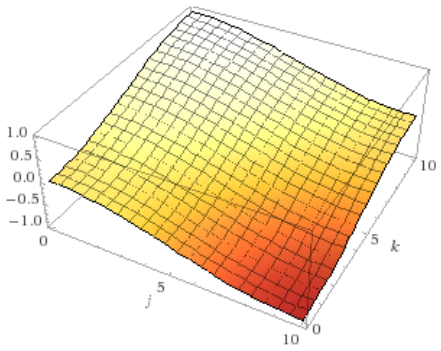
(b) $\sin\frac{\pi\cdot h}{\mathcal{S}_{\text{ISBN}}}$ for $h \in [0, |\mathcal{S}_{\text{ISBN}}| - 1]$



(c) $-\sin\frac{\pi\cdot(h+g)}{|\mathcal{S}_{\text{ISBN}}|+|\mathcal{S}_{\text{ISDN}}|}$ for $h \in [0, |\mathcal{S}_{\text{ISBN}}| - 1]$ and $g \in [0, |\mathcal{S}_{\text{ISDN}}| - 1]$



(d) $\sin\frac{\pi\cdot(h+g)}{|\mathcal{S}_{\text{ISBN}}|+|\mathcal{S}_{\text{ISDN}}|}$ for $h \in [0, |\mathcal{S}_{\text{ISBN}}| - 1]$ and $g \in [0, |\mathcal{S}_{\text{ISDN}}| - 1]$



(e) $(-\cos\frac{\pi\cdot h}{|\mathcal{S}_{\text{ISBN}}|} + \cos\frac{\pi\cdot g}{|\mathcal{S}_{\text{ISBN}}|-1})$ for $h, g \in [0, |\mathcal{S}_{\text{ISBN}}| - 1]$

Figure 4: Graphs of Different Δq_{mn} functions for $|\mathcal{S}_{\text{ISBN}}| = 10$ and $|\mathcal{S}_{\text{ISDN}}| = 10$

Table 4: Journey Data Sample

Start Time	Station ID	End Time	Station ID	bike ID
7/1/2013 12:02:00 a.m.	31250	7/1/2013 12:21:00 a.m.	31506	W20231
9/4/2013 4:27:00 p.m.	31247	9/4/2013 5:11:00 p.m.	31248	W01484
8/9/2013 3:54:00 p.m.	31253	8/9/2013 4:01:00 p.m.	31229	W20198
9/23/2013 5:34:00 p.m.	31004	9/23/2013 5:52:00 p.m.	31007	W20523

Table 5: Station Status Data Sample

Station ID	Longitude	Latitude	No of bikes	No of Empty Docks
31000	-77.0512	38.8561	4	7
31605	-77.0023	38.8851	1	10
31609	-77.0213	38.8767	7	12
31403	-77.0198	38.9566	8	7

5 Numerical Experiments

5.1 Data Description

The following publicly available data sets were retrieved from the Capital Bikeshare website:

1. Journey Data: The data of 0.85 million bike rides during the three months period starting from July 1st to September 30th, 2013 (See Table 4).

2. Station Data: The data of longitude and latitude of all stations, their ID numbers and their current inventory status retrieved from publicly available xml file (See Table 5).

Journey data is used to calculate the demand vector d^{ij}. Capital Bikeshare is a relatively newer and smaller system. The number of journeys between each OD pair in a short enough time period is rather small. At the Capital Bikeshare website, the station data with sample in Table 5 is available in the form of an XML file and the current status of the stations in terms of inventory, check-ins and checkouts is continuously updated on to the server. This data is used to calculate the position of each station in the network. The longitude and latitude are used to calculate distances and time of travel between all OD pairs. The distance calculated is not euclidean and represents actual biking distance. We also get the number of bikes, number of empty docks and capacity of each station using this data. The starting inventory I_i^0 used in our calculations is the starting inventory of the network. The distance matrix is also used to determine the neighborhood stations of each station. We use 600 meters as a conservative measure with stations less than 600 meters apart becoming each other's neighbors.

Table 6: Inventory Status for Different Network Sizes.

Network Name	No of Stations	No of Total Docks	No of Available Bikes	No of Empty Docks	System-wide Demand for bikes	System-wide Demand for Docks
MEDIUM	60	1606	1240	366	907	907
LARGE	202	4498	1012	3486	2531	2531

We build the data sets for two sets of problems of different network size. A MEDIUM sized network with 60 stations and a LARGE sized network with 202 stations in it. It should be taken into account that the number of imbalanced stations cannot always be 0. In fact, our observations show that during high demand periods, like the one we have used for experimentation, the total inventory available in the system is much smaller than demand for the next time period. In Table 6 we present the inventory situation for each of our data sets. As evident, there is always a measure of imbalance present in the system.

5.2 Experiments with Optimization Model and Heuristic Approaches

For our experimental setup, we divide each day into four equal intervals of six hours. The first three intervals from 6:00 a.m. to 11:59 p.m. are used for price based repositioning while in the fourth interval, from 12:00 a.m. to 6:00 a.m. the next day, the trucks carry out static repositioning. We call these intervals $Q1$, $Q2$, $Q3$ and $Q4$, respectively. We use the demand data from 20th September 2013. We assume that the demand on this day in all three intervals is known at the beginning of the intervals. At the beginning of each of the first three intervals, we use the single level model (P) or the IPAS scheme to calcualte a price vector. The journey data is also used to determine service level requirements and values of $I_i^{\max}$ and $I_i^{\min}$ at the beginning of the three intervals. We use β_i^- and β_i^+ values of 0.95 for our calculations. The starting inventory profile MEDIUM represents the system status at 6 a.m. on 20th September, 2013 for the 60 station network. In the subsequent intervals, the starting inventory is recursively updated. For second interval, the starting inventory is equal to the final inventory at the end of the first interval. For the fourth interval beginning at 12 a.m., the starting inventory (for truck based repositioning) is the same as the ending inventory for the third interval.

The objective function values for single level model (P), its maximization counterpart ($\bar{\text{P}}$), IPAS scheme and the situation without pricing for different network sizes, and different time intervals, are tabulated in Table 7. The second row in Table 7 reports the objective function value for the maximization model ($\bar{\text{P}}$). The values in first and second rows provide us with a range in which the objective function may lie for the given optimal price vector. For instance, if we use q_{mn}^{opt} in

Table 7: Value of Objective Function for different starting inventories with (P), ($\bar{\text{P}}$), and Iterative Price Adjustment Scheme (IPAS) for the MEDIUM and LARGE networks

			Number of Imbalanced Stations		
Network	Method	Unsatisfied Demand	$Q1$	$Q2$	$Q3$
MEDIUM	(P)	355/83	26	21	15
	($\bar{\text{P}}$)	350/91	44	38	37
	IPAS	294/19	37	30	25
	N	252/48	47	46	38
LARGE	(P)	-	-	-	-
	($\bar{\text{P}}$)	-	-	-	-
	IPAS	390/972	110	89	84
	N	203/813	156	132	116

reality, the number of imbalanced stations for $Q1$ could be in the range (26-44), 26 in the best case scenario and 44 in the worst case scenario. On the other hand, and very crucially, the IPAS heuristic generates the price vector based on iterative increments and decrements and hence the price vector thus generated is well differentiated. All the numerical experiments were done on a machine with 2.30 GHz CPU clock speed, 8 GB RAM and 64-bit Windows 8 operating system. The single level model (P) was solved using the Java API of CPLEX V12.4 while the coding for the IPAS heuristic was also done in Java. The optimality gap for the single level model (P) was set at 4.0%. For IPAS, as mentioned earlier, the heuristic was run for a designated number of iterations. We use 5000 iterations for our experiment. Mostly, our best result is found within the first 1000 iterations.

We find out that for the medium sized network, a single run of the single level model (P) is solved by CPLEX in approximately two hours. However, CPLEX fails to solve the full network model consisting of 202 stations in a reasonable amount of time. We see a huge improvement in the objective function when price change vectors are generated through single level model (P) and IPAS heuristic as compared with situation without pricing. Although the heuristic developed in this paper may potentially underperform model (P) in terms of objective function, the price vector generated by IPAS is well differentiated, and we may reasonably claim that IPAS can potentially outperform the single level model (P) in terms of the objective function in scenarios close to the worst case for (P). On the other hand, time is a critical factor in calculating and updating a real time price vector. A single run of IPAS scheme only takes 40 seconds and 120 seconds for the 60 station and 202 station networks, respectively.

We also limit the maximum absolute value of price changes available to the operator. Table 8 shows the objective function values for different ranges of incentive and penalty values and different ν_3 values. Smaller values of ν_3 underestimate the cost of walking and hence alternative paths become

Table 8: Value of Objective Function $f(q_{mn})$ generated by IPAS for different q_{mn} ranges and ν_3 values for a LARGE network when neighborhood radius is 600m

	Range of q_{mn}					
	$[-5, 5]$	$[-4, 4]$	$[-3, 3]$	$[-2.5, 2.5]$	$[-2, 2]$	$[-1.5, 1.5]$
ν_3=1/25	91	91	91	95	95	94
ν_3=1/50	83	85	84	88	90	95
ν_3=1/75	80	82	78	85	82	85

more suitable for travellers. We have used ν_3=1/50 in our earlier calculations.

As evident, larger the price changes an operator is willing to introduce, the more balanced a network becomes. We see that as the price change range increases the total price change offered increases dramatically while offering much less in terms of objective function improvement. It is always better to offer smaller price changes first and gradually increase them if doing so provides considerable advantage.

5.3 Models for Cost Comparison

To prove the efficacy of our model (P) and the IPAS scheme in delivering cost advantages, we discuss two approaches in this paper. The first approach is a naive method based on a simple costing heuristic for repositioning of bikes. The second approach, which we discuss in relative detail, is a routing model that we borrow from Raviv et al. (2013).

5.3.1 Naive Method

In Table 9, we use the naive method for cost comparison between manual and price-induced repositioning schemes. The values in column A represent the total *positive* price change which is the amount of incentives doled out to customers. The values in column B are the total *negative* price change which is the total penalties levied on the travellers to stop them from making undesirable journey choices. Naturally, due to the price change vector, the travellers automatically move certain bikes between stations to make them balanced. The sum of A and B values is the total cost to the system for moving a certain number of bikes automatically. Column C indicates the cost of moving the same number of bikes using manual trucks and crew. Simply, column C is the cost that is saved as a result of pricing scheme. To find the cost values in column C, we use the average \$3.5, inflation adjusted value of the estimate from DeMaio (2009), as a conservative measure for cost of a single manual repositioning and multiply it with the total number of bikes moved as a direct result of our pricing scheme. As long as A+B $\leq$ C, price induced repositioning is more viable than manual repositioning.

Table 9: Using the Naive Method for Comparison of Cost(in Dollars) to the System for Price Induced Repositioning (P) and Manual Repositioning (No pricing), A=Total Incentives given, B= Total Penalties levied, A+B= Cost of Price Based Repositioning, C= Cost of Manual Repositioning

	ν_3=1/25				ν_3=1/50				ν_3=1/75			
Range of q_{mn}	A	B	A+B	C	A	B	A+B	C	A	B	A+B	C
$[-5, 5]$	1311	−1955	−644	744	913	−1426	−513	1183	690	−1102	−412	1148
$[-4, 4]$	1251	−1742	−491	605	894	−1361	−467	1134	742	−1095	−337	1235
$[-3, 3]$	1039	−1337	−298	399	726	−1078	−352	952	604	−919	−315	1123
$[-2.5, 2.5]$	983	−1162	−181	224	668	−973	−305	896	537	−804	−267	1039
$[-2, 2]$	820	−954	−134	168	520	−829	−309	822	480	−697	−217	945
$[-1.5, 1.5]$	590	−724	−134	63	476	−689	−213	592	376	−577	−201	801

Note that if the range of incentives increases, the number of imbalanced stations comes down. Also, the number of bikes 'automatically' moved as a direct result of the pricing scheme increases and hence we see greater values in column A, B and C. Naturally, if the range of price change is larger, greater cost is saved.

5.3.2 Arc-Indexed Formulation for Repositioning Vehicle Routing

To evaluate the impact of pricing on repositioning costs we use, with minor modifications, the arc-indexed formulation presented in Raviv et al. (2013). The modified formulation has the same constraints, with minor modifications, for inventory conservation, flow conservation, station capacity, repositioning vehicle capacity and upper bounds on loading, unloading and total vehicle travel times among others. Let $\mathcal{S}$ be the set of stations, $\mathcal{S}_0$ the set of stations including the depot, $\mathcal{V}$ the set of repositioning vehicles, and t_{ij} the truck travel time from station i to station j. The objective function, consisting of a penalty term and a travel cost term, is given by the following expression:

$$\min \quad \sum_{i \in \mathcal{S}} \zeta_i + \alpha \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}} \sum_{v \in \mathcal{V}} t_{ij} x_{ijv} \tag{45}$$

In our modified model, however, we modify the definition of the first term (penalty term) of the objective function. Since we have already calculated $I_i^{\min}$ and $I_i^{\max}$ in Section 3.1, we already have the inventory range that will ensure a 95% service level for a station taking into account the future demand. Given this range, we do not need to calculate the convex penalty function values for all inventory levels considering shortage costs and service costs as Raviv et al. (2013) do. Instead, we assume the penalty to be 0 if the inventory I_i lies between $I_i^{\min}$ and $I_i^{\max}$ and we use a simple linear function to calculate the penalties for all possible deviations from this range as shown in Figure 5.

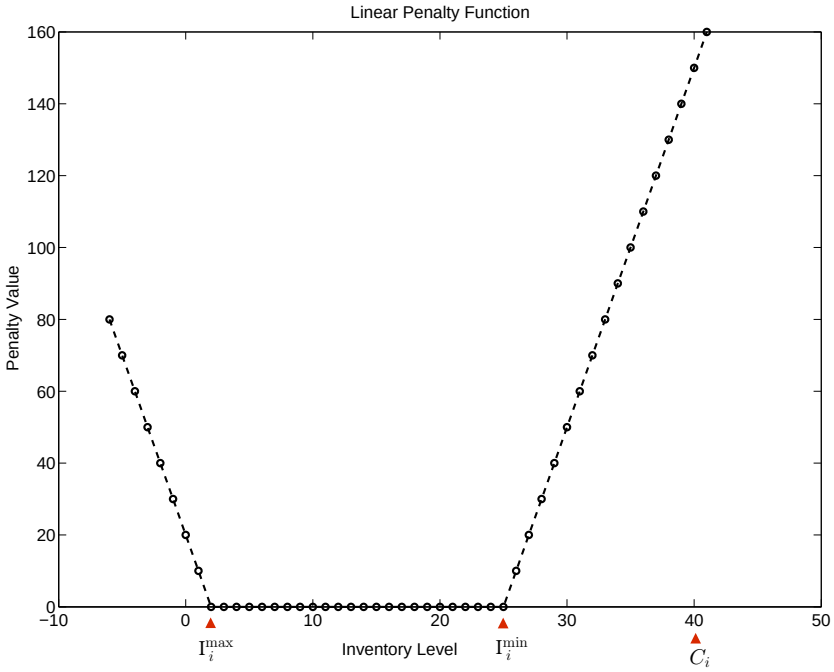


Figure 5: Linear Objective Function as a function of Inventory

The complete modified formulation for the problem along with the notation is provided in Appendix B. The static or truck-based repositioning operation discussed in the model starts at 12 a.m. and continues for a specific period of time (We use 9000sec and 18000sec). The purpose of repositioning operation is to improve the starting conditions of the next day. This is done through adjusting the initial inventory levels before the start of the next day (at 6am) using the truck-based static repositioning. The dynamic pricing scheme, which precedes static repositioning, in turn adjusts the inventory level at 12 a.m. before the start of static repositioning. To compare the two situations i.e. with and without pricing, we compare the performance of the static respositioning model vis-à-vis its objective function given the different starting inventory situations at 12 a.m.

Using the arc-indexed formulation presented in Appendix B, we compare the objective function of the routing problem with four different 12 a.m. inventory situations: one reached with optimal pricing considering the best case scenario (P), second reached with optimal pricing considering the worst case scenario ($\bar{\text{P}}$), third reached without pricing, and the fourth reached as a result of IPAS pricing. We use Java API of CPLEX to solve the routing problem. We use setting of 4.0% for optimality gap and set the solver time to be 1 day. A sample of results is tabulated in table 10. We actually solve a large number of instances of the Routing Model. We solve the 60 station network with different settings for number of vehicles, the total repositioning time available, the total travelling time available, the capacity of the repositioning vehicle(s), the time it takes to load/unload a single bike, the value of constant α in the objective function, and the type of pricing

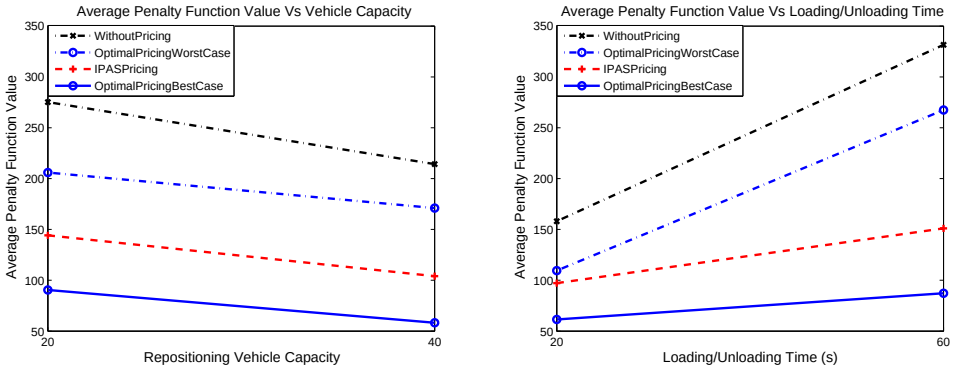
Table 10: No of Vehicles = 1, No of Stations = 60 Total Time = 9000 sec, Travel Time = 9000 sec, Pricing Scheme: (P) = Optimal Pricing Best case scenario, ($\bar{P}$) = Optimal Pricing Worst case scenario (with number of imbalanced stations maximized), N = No Pricing, IPAS = IPAS Pricing, L/U Time = Loading/Unloading time per vehicle, T1 = Travel Time Used, T2 = Total Time Used

Vehicle Capacity	L/U Time	Pricing Scheme	No. of Stations Visited	Bikes Added/ Removed	T1/T2	Objective Function	Average no. of Bikes moved per Station	Maximum Bikes carried	Optimality gap (%)
20	20	(P)	16	30/50	4594.0/6194.0	76	5.000	20.0	3.3%
		($\bar{P}$)	32	80/99	6276.0/9856.0	93	5.594	20.0	4.0%
		N	39	81/101	6966.0/10606.0	120	4.667	20.0	4.0%
		IPAS	25	62/77	5927.0/8807.0	89	5.560	20.0	4.0%
20	60	(P)	18	32/49	4695.0/9555.0	77	4.500	17.0	4.0%
		($\bar{P}$)	34	72/92	6225.0/16065.0	92	4.824	20.0	3.6%
		N	39	76/96	6373.0/16693.0	124	4.410	20.0	3.8%
		IPAS	28	51/70	5792.0/13232.0	88	4.321	20.0	3.7%
40	20	(P)	16	33/62	4278.0/6178.0	43	5.938	29.0	3.7%
		($\bar{P}$)	31	60/87	5608.0/8548.0	56	4.742	31.0	3.7%
		N	37	79/120	6215.0/10255.0	62	5.378	39.9	3.4%
		IPAS	26	58/90	5321.0/8320.0	53	5.692	31.9	1.8%
40	60	(P)	16	28/52	4298.0/9098.0	43	5.000	29.0	3.6%
		($\bar{P}$)	31	61/88	5770.0/14710.0	58	4.806	31.0	3.9%
		N	37	60/100	5968.0/15568.0	70	4.324	40.0	2.0%
		IPAS	28	48/81	5420.0/13220.0	54	4.607	32.0	3.8%

scheme employed.

We find considerable improvement in the overall objective function value when we use the final (12 a.m.) inventory situation which was achieved as a result of a pricing scheme. We also find improvement in other critical to performance factors, like the number of stations visited by the repositioning vehicle, the travel time used for repositioning and the total time used in the repositioning operation. As is evident in table 10, IPAS outperforms the situation without pricing by a considerable margin. This is especially true when full repositioning is allowed to happen, allowing for sufficient time for all the stations to become balanced, as is the case for data in table (10). Not only were smaller numbers of stations visited (up to 55% less), the vehicle had to move smaller number of bikes. The travel time and the overall time used for respositioning were also smaller by a margin of up to 16% and 30%, respectively. We were not able to solve the problem with acceptable optimality gap within the set time limit for the large network of 202 stations. This is consistent with the findings by Raviv et al. (2013).

Furthermore, we also compare the improvements provided by the pricing schemes for different values of the parameters as explained earlier. We postulate that the value of the objective function and, in turn, the improvement due to the pricing scheme should depend on the following factors:



(a) Penalty Function vs repositioning vehicle capacity

(b) Penalty Function vs loading/unloading time

Figure 6: This figure shows how the Penalty Function varies with different parameters. Penalty Function is smaller if vehicles of larger capacity are used. If the overall carrying capacity is the same, larger number of small vehicles are better than smaller number of large vehicles. Similarly, repositioning is more effective when the loading/unloading operation is faster.

- The size of the repositioning vehicle: We postulate that larger the capacity of the repositioning trucks used, smaller (better) will be the objective function.
- The value of loading/unloading time for a single bike: If loading/unloading of individual bikes is faster, there is more time for vehicles to travel to different stations and objective function value is smaller.

We test our results for different values of the mentioned parameters for all four pricing arrangements, i.e. Optimal Pricing in best case scenario (P), Optimal pricing in worst case scenario ($\bar{\text{P}}$), IPAS Pricing, and Without Pricing situations. The results are plotted in Figure 6.

As evident from Figure 6, as the capacity of the repositioning vehicle increases, the objective function value decreases. This is expected because of two reasons. Firstly, a larger vehicle can carry more bikes at a time so it can afford to choose a route that minimizes the penalty function. Secondly, a larger capacity vehicle can travel between different *hub* stations and easily carry a larger inventory between surplus and lack accumulation hubs. Similarly, smaller loading/unloading time gives us better value of the objective function. Since total time is limited and is divided between travel time and loading/unloading time, smaller value of loading/unloading time leaves more time for the vehicle to travel and choose a longer route thus minimizing the objective function. In all these instances however, the objective function and other critical to performance factors are always better for instances with (P) or IPAS pricing as compared to those with no pricing.

6 Conclusion

At the start of this paper, we set out to prove the efficacy of a pricing scheme for partially or fully rebalancing a BSS. This paper explores pricing and incentive schemes as a way to rebalance the network of a public bike sharing system. As already stated, the objective is to minimize the number of imbalanced stations to fully or partially obviate the need for a manual repositioning operation using trucks and crew. We develop a bi-level optimization model and its subsequent single level reformulation (P) that works well for relatively smaller networks to deliver the optimal pricing scheme. We develop heuristics to solve the issue of time while still improving the objective function. We set out to develop a more intuitive heuristic approach based on the classification of different stations using the data available. This heuristic called iterative price adjustment scheme (IPAS) delivers much better computing time. For the full network, the IPAS takes only 120 seconds. The value of objective function is also markedly better than the situation without pricing. We conclude that this time is small enough to make it feasible for BSS operators to update their price vector in real time. The cost of offering incentives is also much smaller than the cost reduction from smaller number of imbalanced stations, smaller crew and trucking fleet. We use a routing model to show that the overall cost of repositioning, consisting of a linear penalty function penalizing the deviation from the min-max range for starting inventory at the beginning of the next day and an operating cost term, is markedly lesser when we employ a pricing scheme. The repositioning operation itself is easier to carry out and faster.

The demand vector we use is based on actual rides data after the fact. One obvious improvement is to model the demand more accurately using demand forecasting taking into account various factors that affect the demand for bikes. The values of coefficients for deriving cost of travel have also been roughly determined. We assume the value of time to be the same for all customers which is obviously not the case. The values of these coefficients can be better estimated using insights into customer behavior. We also assumed that bike users are homogeneous and have the same level of sensitivity to price changes. In further research, we can consider heterogeneous probabilistic behavior or bounded rationality of bike users. As an extension of this work, we can also create a model in which the price based and dynamic manual repositioning happen side by side through out the day, complementing each other. We conclude that a real time dynamic pricing scheme cannot only solve the problem of system wide imbalance in BSS but also cut down on operating costs of controlling that imbalance. As a practical solution to the repositioning problem, based on our findings, we recommend to have a dynamic pricing scheme during the day time complementing a static repositioning at the end of the day when the bike system is closed to the riders.

Acknowledgement

The fourth author's research was partially supported by the University Transportation Research Center (UTRC) under Grant Number 49997-34-24.

References

Bard, J. F. 1991. Some properties of the bilevel programming problem. *Journal of optimization theory and applications* **68**(2) 371–378.

Barth, M., M. Todd, L. Xue. 2004. User-based vehicle relocation techniques for multiple-station shared-use vehicle systems .

Benchimol, M., P. Benchimol, B. Chappert, A. De La Taille, F. Laroche, F. Meunier, L. Robinet, et al. 2011. Balancing the stations of a self-service bike hire system. *RAIRO-Operations Research* **45**(1) 37–61.

Bohannon, R. W. 1997. *Age and ageing* **26**(1) 15–19.

Boyacı, B., K. G. Zografos, N. Geroliminis. 2015. An optimization framework for the development of efficient one-way car-sharing systems. *European Journal of Operational Research* **240**(3) 718–733.

Caggiani, L., M. Ottomanelli. 2012. A modular soft computing based method for vehicles repositioning in bike-sharing systems. *Procedia-Social and Behavioral Sciences* **54** 675–684.

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

Contardo, C., C. Morency, L.-M. Rousseau. 2012. Balancing a dynamic public bike-sharing system. URL `https://www.cirrelt.ca/DocumentsTravail/CIRRELT-2012-09.pdf`.

Correia, G. H. d. A., A. P. Antunes. 2012. Optimization approach to depot location and trip selection in one-way carsharing systems. *Transportation Research Part E: Logistics and Transportation Review* **48**(1) 233–247.

Correia, G. H. D. A., D. R. Jorge, D. M. Antunes. 2014. The added value of accounting for users' flexibility and information on the potential of a station-based one-way car-sharing system: An application in lisbon, portugal. *Journal of Intelligent Transportation Systems* **18**(3) 299–308. doi: 10.1080/15472450.2013.836928. URL `http://dx.doi.org/10.1080/15472450.2013.836928`.

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

DeMaio, P. 2009. Bike-sharing: History, impacts, models of provision, and future. *Journal of Public Transportation* **12**(4) 3.

Fricker, C., N. Gast. 2016. Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity. *EURO Journal on Transportation and Logistics* **5**(3) 261–291.

Kek, A. G., R. L. Cheu, Q. Meng, C. H. Fung. 2009. A decision support system for vehicle relocation operations in carsharing systems. *Transportation Research Part E: Logistics and Transportation Review* **45**(1) 149–158.

Meddin, R., P. DeMaio. 2018. The Bike-sharing World. URL `http://www.bikesharingmap.com/`. Accessed on February 19, 2018.

New York City Department of City Planning. 2009. Bike-share opportunities in New York City. `http://www.nyc.gov/html/dcp/pdf/transportation/bike_share_complete.pdf`.

Pfrommer, J., J. Warrington, G. Schildbach, M. Morari. 2014. Dynamic vehicle redistribution and online price incentives in shared mobility systems. *Intelligent Transportation Systems, IEEE Transactions on* **15**(4) 1567–1578.

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

Schuijbroek, J., R. C. Hampshire, W.-J. Van Hoeve. 2017. Inventory rebalancing and vehicle routing in bike sharing systems. *European Journal of Operational Research* **257**(3) 992–1004.

Shaheen, S. A., S. Guzman, H. Zhang. 2010. Bikesharing in europe, the americas, and asia. *Transportation Research Record: Journal of the Transportation Research Board* **2143**(1) 159–167.

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

The Pennsylvania Environmental Council. 2013. Philadelphia bike share strategic business plan. URL `http://www.bikesharephiladelphia.org/PhilaStudy/CompleteBusinessPlan.pdf`.

Waserhole, A., V. Jost, et al. 2012. Vehicle sharing system pricing regulation: A fluid approximation. URL `http://hal.archives-ouvertes.fr/hal-00727041/`.