Static Repositioning in a Bike-Sharing System: Models and Solution Approaches

Tal Raviv, Michal Tzur, Iris A. Forma

Industrial Engineering Department

Tel Aviv University, Tel Aviv, 69978 ISRAEL

talraviv@eng.tau.ac.il, tzur@eng.tau.ac.il, irisforma@eng.tau.ac.il

August 2011

Abstract

Bike-sharing systems allow people to rent a bicycle at one of many automatic rental stations scattered in the city, use them for a short journey and return them at any other station in the city. Recently many cities around the world deployed such systems in order to encourage their citizens to use bicycles as an environmentally sustainable, socially equitable mode of transportation, and as a good complementary to other modes of the mass transit systems. A crucial factor for the success of a bike sharing system is its ability to meet the fluctuating demand for bicycles and for vacant lockers at each station. This is performed through a *repositioning* operation which consists of removing bicycles from some stations and transferring them to other stations, using a dedicated fleet of trucks. Operating such a fleet in a large bike sharing system is an intricate problem consisting of decisions on the routes that the vehicles should follow, and the number of bicycles that should be removed or placed in each station at each visit of the vehicles. In this paper we present our modeling approach to the problem, which is unique in its objective function, and incorporates additional characteristics that are new to the literature. Several Mixed Integer Linear Program formulations are then presented. We discuss the assumptions and implications associated with each, and compare their performances through an extensive numerical study. The results indicate that one of the formulations is very effective in obtaining high quality solutions to real life instances of the problem consisting of up to 104 stations and two vehicles. The optimality gap in these instances was quite small (2.97% on average and no more than 5.41%). It is also indicated that under certain conditions, other formulations may become preferable.

1. Introduction

Bike-sharing systems allow people to rent a bicycle at one of many automatic rental stations scattered in the city, use them for a short journey and return them at any other station in the city. Recently many cities around the world deployed such systems in order to encourage their citizens to use bicycles as an environmentally sustainable and socially equitable mode of transportation, and as a good complementary to other modes of mass transit systems (mode sharing). In addition, a municipal bike-sharing system may yield revenue for the city in the compliance carbon offset market, see for example Capoor and Ambrosi (2009). One of the largest bike-sharing system as of today is Vélib launched on

July 2007 in Paris (www.velib.paris.fr). It now consists of some 2000 renting stations and offers more than 20,000 bikes for rent. The company operates a fleet of 25 trucks to perform the repositioning operation. As of December 2010 some 238 cities around the world deployed such systems and 53 are in planning stages, see MetroBike LLC (2011).

A crucial factor for the success of a bike sharing system is its ability to meet the fluctuating demand for bicycles at each station. In addition, the system should be able to provide enough vacant lockers to allow the renters to return the bicycles at their destinations. Indeed, the main complaint heard from users of bike-sharing systems in forums and blogs regards to unavailability of bicycles and (even worse) unavailability of lockers at their destination. In Brussels, for example, a voluntary group of users created a web service that pulls inventory data from the city's bike-sharing (Villo) website in order to monitor the service level and create a public pressure on the operator to improve it. According to the group's web site (http://www.wheresmyvillo.be/), their main cause is to make "JCDecaux [the operator] drastically improve the availability of bikes and parking spaces, through better reallocation of bikes". Their website displays statistics about the percentage of time in which at least one bicycle (resp., locker) was available in each station during the last seven days. As of October 14, 2010 the ten worst stations, out of the system's 180 stations, could not provide a single bicycle more than 33% of the time (resp., provide a single locker more than 23% of time).

Meeting the demand for bicycles and vacant lockers is a particularly challenging problem due to inherent imbalances in the renting and return rates at the various stations. In some cases, the imbalance is temporary, e.g., high return rate in a suburban train station in the morning and high renting rate in the afternoon. In other cases the imbalance is persistent, e.g., relatively low return rate in stations located on top of hills. Satisfying the users' demand subject to such imbalances requires regularly removing bicycles from some stations and transferring them to other stations. This task is performed using a dedicated fleet of light trucks. We refer to this activity as *repositioning* bicycles.

Thus, the bicycle repositioning problem involves routing decisions concerning the vehicles, and inventory decisions concerning bicycles in the rental stations. The latter problem involves determining the number of bicycles removed or placed in each station at each visit of a vehicle, with the objective of achieving a high level of users' satisfaction from the system, defined more precisely in the sequel. The repositioning operation can be carried out in two different modes: one is during the night when the usage rate of the system is negligible; the other is during the day when the status of the system is rapidly changing. We refer to the former as the *static repositioning* problem and to the latter as the *dynamic repositioning* problem. Some operators use the former mode, some use the latter and some use a combination thereof, Callé (2009).

Since automated bike sharing systems are quite new, the operations research literature on problems related to it is sparse. Such systems raise modeling questions, as well as challenges concerning devising appropriate solution methods to solve the resulting problems. While the domain of bike sharing systems' operation is yet to be studied, some aspects of it have already appeared in the

literature on vehicle and inventory routing, as described in Section 2. In this paper we focus on the static mode of operation which benefits from a practical advantage because it allows the repositioning fleet to travel swiftly in the city without contributing to traffic congestion and parking problems. The static problem needs to be solved once at the beginning of every night, based on the status of the system at that time and the demand forecast for the next day.

The contributions of this paper are the following:

First – Introduction of a new problem:

We introduce a new inventory-routing problem, arising from a new application area, namely bike sharing systems, which is timely, relevant and challenging.

Second – Presentation of mathematical models for the new problem:

We present our modeling approach to the problem of repositioning in a bike-sharing system, which is unique in its objective function, and incorporates additional characteristics that are new to the literature. The chosen objective function is based on practical considerations and is quite unusual compared to typical routing problems as it focuses on minimizing the user dissatisfaction in face of stochastic demand. Based on our modeling approach we offer several Mixed Integer Linear Program (MILP) formulations, each is associated with a certain vision on the problem's physical characteristics. As a result, they are distinct from each other with respect to permissible actions of the repositioning vehicles. In particular, they differ in their limitations on the allowed set of routes, and in the restrictions on transshipments of bicycles via intermediate stations.

Third – Development of solution methods:

We develop solution methods to the problem, that are based on the MILP formulations described above, combined with some algorithmic enhancements. The algorithmic enhancements are steps taken beyond a straightforward application of a MILP solver, and are required in order to obtain good solutions in a reasonable time. Technicalities that needed to be worked out in formulating and solving the problem efficiently may prove to be useful for other routing problems as well, so that the theoretical contribution of this work may be applicable more generally.

Fourth – Verification through computational experiments:

We solve a variety of instances using the suggested formulations and solution methods, compare their solutions and running times, and discuss the advantages and disadvantages of each. The results indicate that one of the formulations and its associated solution method is very effective in solving real life instances of the problem with up to 104 nodes and two vehicles.

Thus, our contributions are both from a scientific and a practical point of view. As this is the first work to consider all aspects of the repositioning problem, we see it also as a building block in developing further models and methods to this complex optimization problem. Towards that, we provide the community with our set of instances, to be used as benchmark problems.

The rest of this paper is organized as follows: in Section 2 we review the literature, describing related work from several application areas. In Section 3 we present our modeling approach by

specifying the underlying assumptions and the chosen objective function, present our formulations, and sets of valid inequalities to each, that are used to tighten them. In Section 4 we discuss additional algorithmic enhancements that are needed in order to solve the mathematical models effectively. In Section 5 we describe our numerical experiments, the results, and their analysis. Finally, in Section 6 we discuss some of our assumptions, their implications, possible extensions and directions for further research.

2. Literature Review

The bicycle repositioning problem was first introduced in Forma et al. (2010) where a single formulation to a more restricted problem was presented. The bicycle repositioning problem can be classified as a variation of the Pickup and Delivery Problem (PDP). Berbeglia et al. (2007) surveyed the literature on static PDP and classified these problems according to various parameters. By this classification, the repositioning problem presented in this study is a many-to-many single commodity PDP with arbitrary vehicle capacities and non-linear objective function, for which no studies are available. In pickup and delivery problems, as opposed to our bicycle repositioning problem, the quantities picked-up or delivered to a node are given.

The bicycle repositioning problem also bears great similarities to the Inventory Routing Problem (IRP), since it needs to determine the routes of the vehicles, as well as the quantities of bicycles to load or unload at each visited node. However, in IRP, the objective is to determine distribution policies from one or more depots that minimize the total cost, i.e., the sum of inventory holding and transportation costs, while avoiding stock-outs and respecting storage capacity limitations, see a recent survey by Bertazzi et al. (2008).

Another closely related routing problem is the Swapping Problem (SP), first introduced by Anily and Hassin (1992). In its generic form, the SP is defined as follows: "Each vertex of a graph may contain an object of a known type. A final state, specifying the type of object desired at each vertex, is also given. A single vehicle of a unit capacity is available for shipping objects among the vertices. The swapping problem is to compute a shortest route such that a vehicle can accomplish the rearrangement of the objects while following this route". Anily and Hassin (1992) showed that the problem is NP-Hard and presented a 2.5 approximation algorithm. The SP is, on one hand, more general than our problem since the objects may belong to more than one type; however, in the SP the demand at the nodes is limited to one unit, there is one vehicle only, and its capacity is limited to one. Further studies on the SP focus mainly on special cases of the problem, for which a polynomial time optimization or approximation algorithms are possible, for example Anily et al. (1999), where a quadratic time optimization algorithm for SP on a line is presented. A slight variation of SP is the mixed SP where the vehicle is allowed to temporarily unload some of the objects at nodes other than their destinations. A recent study by Bordenave et al. (2010) presents a constructive approach and

several improvement heuristics that provide near optimal solutions (optimality gap of less than 1% on average) of instances with up to 10,000 nodes.

Hernández-Pérez and Salazar-González (2004a) introduced the one-commodity pickup and delivery traveling salesman problem (1-PDTSP), a generalization of the well-known TSP where each customer has supply or demand of a given amount of a single product. One vehicle of a given capacity must visit each customer and the depot exactly once, picking up units of the product from customers with supply and delivering it to customers with demand, while minimizing the total travel distance. They present an ILP model for this problem and describe a branch and cut procedure for solving it. Hernández-Pérez and Salazar-González (2004b) presented heuristic methods for the problem and demonstrated their applicability for instances with up to 500 nodes. Louveaux and Salazar-González (2009) considered the 1-PDTSP with stochastic demand or supply. They study the problem of finding the smallest vehicle capacity that assures feasibility, i.e., being able to satisfy all demands; for a given vehicle's capacity they search for a tour which minimizes the objective function which includes a penalty that is proportional to the unsatisfied demand.

Raviv and Kolka (2011) develop a method which calculates the expected user dissatisfaction over a given period in a single bike-sharing station. More specifically, it calculates the expected weighted number of users who arrive to the station and cannot fulfill their request immediately due to lack of bicycles or lack of vacant lockers. Their method requires as an input the rates of two independent non-homogenous Poisson demand streams for users seeking to rent bicycles and users seeking to return bicycles at the station, and the relative weights of dissatisfaction for unfulfilled requests. They also proved that the expected weighted user dissatisfaction function is convex in the inventory level at the station at the beginning of the period (e.g., day), and developed a method to calculate the function efficiently.

The closest studies to ours is that of Benchimol et al. (2011) and Chemla et al. (2011) who study a one commodity pickup and delivery problem under the assumptions of a single vehicle and no time constraint, motivated too by the application of repositioning bicycles. The goal is to minimize the total travel distance of the vehicle while completing a prescribed repositioning task. Chemla et al. (2011) describes a branch-and-cut algorithm for solving a relaxation of the problem, from which a solution is obtained through a Tabu search.

Vogel and Mattfeld (2010) presented a stylized model to assess the effect of dynamic repositioning efforts on service level. Their model is useful for strategic planning but is not detailed enough to support the repositioning operation. Nair and Miller-Hooks (2011) used a stochastic programming approach to handle dynamic repositioning planning in shared mobility systems. Their model assumes that the cost of moving a single vehicle between two given stations is known and fixed. In our view this assumption is realistic for one way car sharing systems that motivated their work. However, Nair et. al (2011) apply a similar model on data obtained from Vélib.

Some authors considered strategic decisions regarding bike rental stations capacity and locations. Shu et al. (2010) proposed a stochastic network flow model to support these decisions. They use their model to design a bike sharing program in Singapore based on demand forecast derived from current usage of the mass transit system. Lin and Ta-Hui (2011) considered a similar problem but formulate it as a deterministic mathematical model. Their model is aware of the bike path network and mode sharing with other means of public transportation.

Finally, there are some similarities between bike-sharing systems and car-sharing systems. The latter may belong to the round-trip type of system, or to the one-way type of system, based on whether a user has to return a car to the same parking space or can return it to any station, respectively, see, for example, Mukai and Watanabe (2005) and Uesugi, et al. (2007). However, there are clearly major differences between bike and car sharing systems, since typically a car is moved from one location to another by assigning a driver to it, and cars are not moved in batches.

The static bicycle repositioning problem combines some aspects of the studies described above, but addresses new and broader ones that have not been studied before. One such important aspect is the objective function, which aims to minimize some measure of expected user's dissatisfaction, where all the studies mentioned above consider total travel distance as the sole objective function.

3. Model Formulation

The satisfaction of users from a bike sharing system is a crucial factor in its popularity and success. Hence, a major challenge in modeling and formulating the repositioning problem is to express the objective function accordingly. We do so by defining the objective function precisely as the dissatisfaction of users, to be minimized. Raviv and Kolka (2011) show how to calculate the user dissatisfaction function based on time-dependent demand distributions for the next day (or any other chosen planning horizon). This enables us to treat the values of the dissatisfaction functions as known for every possible inventory level. They further prove that this function is convex, a property which is crucial for our solution methods. For completeness of our paper, we summarize these results in Appendix A.

Therefore, the repositioning problem is to determine the route of each vehicle and the number of bicycles to load or unload at each station that the vehicle visits, such that the sum of the dissatisfaction costs at all stations is minimized. The above decisions are subject to capacity constraints of the vehicles, the stations and the depot, as well as time constraints concerning traveling times, and loading and unloading times. The latter are assumed to be linear with the number of bicycles loaded/unloaded.

Additional modeling choices are associated with permissible actions of the fleet of vehicles performing the repositioning operation, in particular, limitations on the allowed set of routes and/or limitations on the quantity of bicycles loaded / unloaded at each station. Such limitations may result

from operational considerations or from computational limitations. They are described and discussed in details later in this section, when presenting the various mathematical formulations. Table 1 at the end of this section summarizes the capabilities and assumptions of the formulations presented in this paper.

In the static version of the problem studied here, repositioning is performed while the demand for bicycles and vacant lockers is assumed to be null. Indeed, in reality the demand during the night is negligible. A given length of time (say, five hours, from 1 a.m. to 6 a.m. every night) is allotted to the repositioning operation, and its purpose is to improve the starting conditions (bicycle inventory levels at the stations) of the next day.

While one could argue that a bi-objective function should be used, where a second objective represents the operating costs of the repositioning fleet, we believe that adding this term is not needed because the cost of hiring the drivers and maintaining the vehicles is sunk at the point of time when the actual repositioning operation is performed. Nevertheless, in all formulations presented in this paper it is straightforward to add the operating costs to the objective function, if desired.

Next we present notation that is common to all our formulations.

3.1 Notation

The static repositioning problem is described by the following sets and parameters:

N	set of stations, $i = 1, \dots, \lvert N \rvert$
N_0	set of nodes, including the stations and the depot (denoted by $i = 0$), $i = 0, \dots, \lvert N \rvert$
V	set of vehicles, $v = 1, \dots, \lvert V \rvert$
s_i^0	quantity of bicycles at node *i* before the repositioning operation starts
c_i	capacity of bicycles (number of lockers installed) at station $i \in N$
c_0	capacity of bicycles at the depot (available space); note that this parameter may be represented by a large number, if no space constraint exists.
k_v	capacity (number of bicycles) of vehicle $v \in V$
$f_i(u)$	cost (dissatisfaction, in units of, e.g., expected weighted unfulfilled requests of users) of reaching a level of u bicycles at station i at the end of the repositioning operation, $u = 0, \dots, c_i$
t_{ij}	Traveling time from station *i* to station *j*
T	time allotted to the repositioning operation, also referred to as a planning horizon
L	time required to remove a bicycle from a station and load it onto the vehicle
U	time required to unload a bicycle from the vehicle and hook it to a locker in a station

3.2 Arc-Indexed Formulation

Our first mathematical formulation of the problem is referred to as an *arc-indexed* (AI) formulation. Initially its objective function is represented by a sum of convex functions, while later these functions are linearized in an exact manner through a set of constraints. This formulation is similar in the definition of its decision variables to classical three index formulations of other routing problems. Still, some of its characteristics are unique to this particular application. An assumption of this formulation is that each station may be visited at most once by each vehicle, although a certain station may be visited by several vehicles. We define the following decision variables:

x_{ijv} a binary variable which equals one if vehicle v travels directly from node i to node j, and zero otherwise;

y_{ijv} quantity of bicycles carried on vehicle v when it travels directly from node i to node j, and zero if the vehicle does not travel directly from i to j;

y_{iv}^{L} quantity of bicycles loaded onto vehicle v at node i;

y_{iv}^{U} quantity of bicycles unloaded out of vehicle v at node i;

q_{iv} auxiliary variables, used for sub-tour elimination constraints;

s_i inventory level at node i at the end of the repositioning operation;

The following parameter needs to be defined and used specifically in this formulation:

M An upper bound on the number of arcs in a tour whose length is at most T time units, which visits each station at most once; ($M = |N_0|$ is a trivial such upper bound; it may be strengthened when T is small, by solving a simple integer program).

(P1) – Arc indexed (AI) formulation

$$Min \sum_{i \in N} f_i(s_i) \quad (1)$$

s.t.

$$s_i = s_i^0 - \sum_{v \in V}(y_{iv}^L - y_{iv}^U) \qquad \forall\, i \in N_0 \quad (2)$$

$$y_{iv}^L - y_{iv}^U = \sum_{j \in N_0, j \neq i} y_{ijv} - \sum_{j \in N_0, j \neq i} y_{jiv} \qquad \forall\, i \in N_0,\ \forall\, v \in V \quad (3)$$

$$y_{ijv} \leq k_v x_{ijv} \qquad \forall\, i,j \in N_0, i \neq j,\ \forall\, v \in V \quad (4)$$

$$\sum_{j \in N_0, j \neq i} x_{ijv} = \sum_{j \in N_0, j \neq i} x_{jiv} \qquad \forall\, i \in N_0, \forall\, v \in V \quad (5)$$

$$\sum_{j \in N_0, j \neq i} x_{ijv} \leq 1 \qquad \forall\, i \in N,\ \forall\, v \in V \quad (6)$$

$$\sum_{v \in V} y_{iv}^L \leq s_i^0 \qquad \forall\, i \in N_0 \quad (7)$$

$$\sum_{v \in V} y_{iv}^U \leq c_i - s_i^0 \qquad \forall\, i \in N_0 \quad (8)$$

$$\sum_{i \in N_0} \left(y_{iv}^L - y_{iv}^U\right) = 0 \qquad \forall\, v \in V \quad (9)$$

$$\sum_{i\in N}\left(Ly_{iv}^{L}+Uy_{iv}^{U}\right)+\sum_{i\in N}(Ly_{0iv}+Uy_{i0v})+\sum_{i,j\in N_0:i\neq j}t_{ij}x_{ijv}\leq T \qquad \forall\, v\in V \qquad (10)$$

$$q_{jv}\geq q_{iv}+1-M\left(1-x_{ijv}\right) \qquad \forall\, i\in N_0, j\in N, i\neq j, \forall\, v\in V \qquad (11)$$

$$x_{ijv}\in\{0,1\} \qquad \forall\, i,j\in N_0: i\neq j,\ \forall\, v\in V \qquad (12)$$

$$y_{iv}^{L}\geq 0,\ y_{iv}^{U}\geq 0, \text{ integer} \qquad \forall i\in N_0\ , \forall v\in V \qquad (13)$$

$$y_{ijv}\geq 0 \qquad \forall\, i,j\in N_0: i\neq j,\ \forall\, v\in V \qquad (14)$$

$$s_i\geq 0 \qquad \forall\, i\in N_0 \qquad (15)$$

$$q_{iv}\geq 0 \qquad \forall i\in N_0\ , \forall v\in V \qquad (16)$$

The objective function (1) minimizes the total cost (dissatisfaction) incurred at all stations. Constraints (2) are inventory-balance constraints at the nodes (the stations and the depot). Constraints (3) represent the conservation of inventory on the vehicles, and constraints (4) limit the quantity carried on each vehicle to its capacity. These constraints also set to zero the quantity carried on a vehicle when it travels directly from i to j if the vehicle does not use that arc. Constraints (5) are vehicle flow conservation equations. Constraints (6) assert that each station is visited at most once by each vehicle and Constraints (7) (resp., (8)) limit the quantity picked-up by all vehicles from a station (resp., delivered to a station) to the quantity available there initially (resp., the residual capacity of the station), see discussion below. Note that Constraints (7) also imply non-negativity of the inventory variables, while constraints (8) also assure that the inventory at each station and at the depot is bounded by its capacity, therefore these two restrictions are not written explicitly. Constraints (9) stipulate that all the bicycles that are loaded onto the vehicles are also unloaded. Constraints (10) limit the total loading and unloading times plus the travel times to the length of time available for the repositioning operation. Constraints (11) are sub-tour elimination constraints that are similar to those of Miller et al. (1960). Finally, (12) and (13) are binary and general integrality constraints, respectively, and (14)-(16) are non-negativity constraints. Note that the integrality of y_{ijv} and s_i is implied by the integrality of y_{iv}^{L}, y_{iv}^{U} and s_i^0.

Note that the objective function is defined over integer inventory values and supported by a piecewise linear and convex function. We exploit these properties in order to linearize the objective function, so that the above formulation becomes a MILP.

Towards that, we define and calculate for all $i=1,\dots,N$ and $u=0,\dots,c_i$-1:

$$b_{iu}\equiv f_i(u+1)-f_i(u)$$

$$a_{iu}\equiv f_i(u)-b_{iu}\cdot u$$

Thus, b_{iu} is the marginal cost (which may be negative) of the $(u+1)^{st}$ bicycle at station i. Alternatively, a_{iu} and b_{iu} represent the intercept and slope, respectively, of the linear function that supports the convex cost function $f_i(.)$ at the level of u, see Figure1.

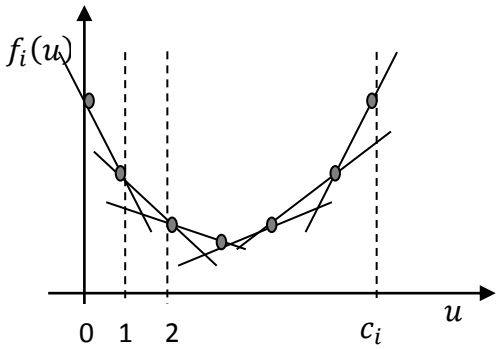

Figure 1. linear functions supporting $f_i(u)$

Then, linearity of the formulation is achieved by further defining the following decision variables:

g_i cost incurred at station i;

We can now replace the previous convex objective function by:

$$Min \sum_{i \in N} g_i \tag{17}$$

and add to the formulation the following constraints:

$$g_i \geq a_{iu} + b_{iu} s_i \qquad \forall\, i \in N, u = 0, \ldots, c_i - 1 \tag{18}$$

The resulting MILP formulation consists of the objective function (17) subject to Constraints (2)-(16) and (18). Note that since the convex function is defined over integer values only (quantity of bicycles), the linearization scheme, described by (17) and (18), is exact and not approximated. The following assumptions are embedded in the above formulation:

First, as noted above, each station may be visited at most once by each vehicle, see constraints (6), however a certain station may be visited by several vehicles. Thus, the total quantity picked-up from a station or delivered to it by all vehicles is limited as stated in constraints (7)-(8). Together these constraints verify that bicycles are not transshipped from a station before they are brought to it (or delivered to a station before space is available) during another visit of the same or a different vehicle. In other words, these constraints exclude situations in which the inventory level at a station is negative or exceeds its capacity, in the interim of the repositioning period. In fact, these restrictions are more severe than required in practice, because they exclude the possibility of a vehicle picking-up bicycles that were previously brought by another vehicle, or delivering bicycles to a station that has available capacity due to another vehicle which removed bicycles from it in an earlier visit. The Time Indexed formulation presented in Section 3.3 relaxes this restriction, but is harder to solve.

Second, it is assumed that all bicycles brought to the depot by the vehicles are unloaded there, even if the vehicle is about to continue its trip and need to load new bicycles in order to deliver them to other stations. This is due to the inability of the arc indexed model to keep track of the total number

of bicycles loaded and unloaded at the depot, and hence the time that these operations consume. While this assumption is a somewhat conservative, its effect on the solution is marginal.

Finally, vehicles may redistribute bicycles in the system in any way that reduces costs. This means, for example, that a vehicle may deliver bicycles to a station even if the station already has more than its "ideal" quantity, that is, the quantity that minimizes its cost function. Such an action may be justified if the bicycles are brought from a station where their presence is more costly. Since the objective is minimizing the overall cost in the system, this capability is desirable.

Solving the problem using the above formulation may become quite time consuming for problems of realistic size. To enhance the running time, we propose two directions: one is adding valid inequalities that are specific to this formulation. Another is solving the problem through a two-stage heuristic, a method which is used also to enhance the running times of the other formulations, and is described in Section 4.1.

The following valid inequalities may be added to the formulation:

$$\sum_{j\in N} x_{0jv} \geq 1 \qquad \forall\, v \in V \qquad (19)$$

Constraints (19) verify that each vehicle departs from the depot at least once. We observed numerically that this valid inequality proved to be particularly useful, since without it, the number of vehicles that "leave" the depot, in the LP relaxation, is fractional, according to the required capacity. Forcing a whole vehicle to leave the depot propagates by the vehicle flow conservation equation (5) to all other visited nodes.

$$y^L_{iv} \leq \min(s^0_i, k_v) \sum_{j\in N_0} x_{ijv} \qquad \forall\, i \in N, v \in V \qquad (20)$$

$$y^U_{iv} \leq min(c_i - s^0_i, k_v) \sum_{j\in N_0} x_{ijv} \qquad \forall\, i \in N, v \in V \qquad (21)$$

Constraints (20)-(21) are somewhat similar to constraints (7)-(8), but refer to loading and unloading quantity limitations by a single vehicle (rather than all vehicles). This enables tightening the right hand side of the constraint by including the vehicle's capacity, and conditioning it only to cases in which the vehicle visits the station.

$$y^L_{iv} + y^U_{iv} \geq \sum_{j\in N_0} x_{ijv} \qquad \forall\, i \in N, v \in V \qquad (22)$$

Constraints (22) eliminate some solutions where a vehicle enters a station without loading or unloading any bicycle there. It is valid when the distance matrix satisfies the triangle inequality because then it is always possible to skip a station with no loading and unloading activities.

In addition, if vehicles are identical it is possible to break symmetry by adding the following constraints:

$$\sum_{j\in N} j \cdot x_{0jv} \leq \sum_{j\in N} j \cdot x_{0,j,v+1} \qquad \forall\, v \in V \qquad (23)$$

In Section 5 we demonstrate the capabilities of this formulation in solving problems of moderate size.

3.3 Time-Indexed Formulation

Our second formulation is based on discretizing the time available for repositioning into slots of short periods, say five minutes each. The length of each such period is denoted by τ. We define decision variables with one of the indices representing the time period. Hence, we refer to this formulation as a *time indexed* (TI) formulation. The motivation behind it is to be able to "follow" the state of the system at any point of time, which is helpful in formulating complex situations properly. Indeed, the time indexed formulation extends the feasible region compared to the arc based formulation as far as the visits and quantity loaded/unloaded at each station are concerned. Namely, it no longer needs to limit the quantities of bicycles transshipped via other stations as described in the arc based formulation (see Constraints (7) and (8)), and it no longer needs to limit each vehicle to visit each station at most once.

In the formulation below, the discretized times are referred to as periods. We define a discretized travel time matrix, denoted by t'_{ij}. These travel times are calculated by dividing the actual travel time by the period length, τ, and rounding it up to the next integer, to assure feasibility. In addition we set $t'_{ii} = 1$, where traveling from node i to itself represents dwelling at the node for one period. Let $T' = T/\tau$ represent the number of periods in the planning horizon. T' is assumed to be integer.

The above discretization procedure is applied to the traveling times, and the smaller τ is chosen to be, the closer the resulting model is to the continuous times case. However, loading and unloading times are typically much smaller than traveling times, and discretizing them in the same manner would result in an unreasonable deviation from the continuous case. Therefore, in the sequel we show how loading/unloading times can be kept continuous and still incorporated in the discrete periodic model. To ease on the presentation, we first introduce the TI formulation assuming that loading/unloading times are zero and afterwards explain how to generalize the formulation to include them back. In this first, more simplistic model, it is assumed that the loading and unloading operations are carried out at the beginning of a discretized period before the vehicle leaves the node.

In this and subsequent formulations, we directly present the linearized objective function and supporting constraints, both of which are identical to those of the AI formulation.

The decision variables used in the time indexed formulation are as follows:

x_{ijtv} a binary variable that equals one if vehicle *v* starts to travel from node *i* to node *j* in period *t*, and zero otherwise;

y^L_{itv} quantity of bicycles loaded onto vehicle v at node i during period t;

y^U_{itv} quantity of bicycles unloaded from vehicle v at node i during period t;

y_{ijtv} number of bicycles carried from node *i* to node *j* by vehicle *v* during period *t*;

s_{it} inventory level at node i at the end of period t;

g_i cost incurred at station i; (as in the arc based formulation)

(P2) – Time indexed (TI) formulation

$$Min \sum_{i \in N} g_i \quad (24)$$

s.t.

$$g_i \geq a_{iu} + b_{iu} s_{iT'} \qquad \forall\, i \in N, u = 0, \dots, c_i - 1 \quad (25)$$

$$s_{i0} = s_i^0 \qquad \forall\, i \in N_0 \quad (26)$$

$$s_{it} = s_{i,t-1} + \sum_{v \in V} \left(y_{itv}^U - y_{itv}^L \right) \qquad \forall\, i \in N_0, t = 1, \dots, T' \quad (27)$$

$$s_{it} \leq c_i \qquad \forall\, i \in N_0, t = 1, \dots, T' \quad (28)$$

$$\sum_{j \in N_0} x_{0j0v} = 1 \qquad \forall\, v \in V \quad (29)$$

$$\sum_{j \in N_0} x_{j,0,T'-t'_{j0},v} = 1 \qquad \forall\, v \in V \quad (30)$$

$$\sum_{j \in N_0} x_{j,i,t-t'_{ji},v} = \sum_{k \in N_0} x_{iktv} \qquad \forall\, i \in N_0, t = 1, \dots, T' - 1, v \in V \quad (31)$$

$$\sum_{j \in N_0} y_{j,i,t-t'_{ji},v} = \sum_{k \in N_0} y_{iktv} + y_{itv}^U - y_{itv}^L \qquad \forall\, i \in N_0, t = 1, \dots, T' - 1, v \in V \quad (32)$$

$$y_{ijtv} \leq k_v x_{ijtv} \qquad \forall\, i \in N_0, t = 1, \dots, T', \forall\, v \in V \quad (33)$$

$$y_{itv}^L \leq min(c_i, k_v) \sum_{j \in N_0} x_{ijtv} \qquad \forall\, \mathrm{i} \in \mathrm{N_0}, \mathrm{t} = 1, \dots, \mathrm{T'},\ \forall\, \mathrm{v} \in \mathrm{V} \quad (34)$$

$$\mathrm{y_{itv}^U} \leq \min(\mathrm{c_i}, \mathrm{k_v}) \sum_{\mathrm{j} \in \mathrm{N_0}} \mathrm{x_{ijtv}} \qquad \forall\, \mathrm{i} \in \mathrm{N_0}, \mathrm{t} = 1, \dots, \mathrm{T'},\ \forall\, \mathrm{v} \in \mathrm{V} \quad (35)$$

$$y_{itv}^L \geq 0,\ y_{itv}^U \geq 0, \text{ integers} \qquad \forall\, i \in N_0, t = 1, \dots, T', \forall\, v \in V \quad (36)$$

$$x_{ijtv} \in \{0,1\} \qquad \forall i, j \in N_0, t = 1, \dots, T', v \in V \quad (37)$$

$$y_{ijtv} \geq 0 \qquad \forall i, j \in N_0, t = 1, \dots, T', v \in V \quad (38)$$

$$s_{it} \geq 0 \qquad \forall i \in N, t = 0, \dots, T' \quad (39)$$

The objective function (24) is identical to the linearized objective function in the arc based formulation. Similarly, constraints (25) are the linear constraints that support the convex cost function, defined here with respect to the inventory at each station at the end of the last period, T'. Constraints (26) and (27) define the initial inventory and the inventory balance at the nodes while constraints (28) specify that the inventory at each node in each period is bounded by its capacity. Constraints (29) and (30) specify that each vehicle departs from the depot at the beginning and returns to it at the end of the repositioning operation, respectively. Constraints (31) are vehicle flow conservation equations, they stipulate that when a vehicle enters a node at some period (after traveling to it a certain number of periods according to its origin), it will leave the node at that period, possibly going to the same node itself. Thus, these constraints schedule the movement of vehicles consistently with the (discretized) distance matrix. Constraints (32) represent the conservation of inventory (bicycles) on the vehicles in each period, and constraints (33) limit the quantity carried at each vehicle in each period to the vehicle's capacity and stipulate that bicycle are carried only on the chosen links. Constraints (34) (resp., (35)) assure that no bicycles are loaded (resp., unloaded) at a node in each

certain period if the node is not visited at that period. Finally, constraints (36)-(39) are integrality and non-negativity constraints. In this formulation any reference to an index of a vector or matrix that is out of bound should be replaced by zero.

The important advantages of the TI formulation mentioned at the beginning of this section are made possible due to monitoring the inventory at each station in every period. Note that it also means that sub-tours of the vehicles are allowed in the solution, although each vehicle still departs from the depot and returns to it. This additional flexibility in forming solutions is likely to improve the repositioning operation and consequently the optimal objective function's value. The ability to redistribute bicycles in the system in any way that reduces costs exists in the time indexed formulation, in the same way as it does in the arc based formulation, see the discussion there.

We define $t'_{ii} = 1$ to allow dwelling at the stations. Note that dwelling may be desirable exactly for the reasons indicated above, namely, to synchronize the visits of different vehicles at the station. As the arc based formulation does not keep track of time, this capability cannot be observed and utilized by its solution.

On the other hand, the TI formulation suffers from a limitation which results from discretizing the travel times. To assure feasibility, the integer traveling times are rounded up, which causes unnecessary slacks in the schedule. To partially overcome this problem, as well as to add the positive loading/unloading times back into the formulation, we extend the above formulation by modifying and adding some constraints. The modifications are non-trivial, since the revised model combines a discrete and continuous representation of time. This enables us to gain the advantages of discreteness, discussed above, together with increased accuracy of the real time constraints of the system.

First, we add the following two sets of constraints:

$$\sum_{i,j\in N_0}\sum_{t\le w} t_{ij}x_{ijt-t'_{ij},v} + \sum_{i\in N_0}\sum_{t\le w}(Ly^L_{itv} + Uy^U_{itv}) \le w\cdot\tau$$
$$\forall\, w = 1,\dots,T',\ \forall v \in V \qquad (40)$$

$$\sum_{i,j\in N_0}\sum_{t\le w} t_{ij}x_{ijt-t'_{ij},v} + \sum_{i\in N_0}\sum_{t\le w}(Ly^L_{itv} + Uy^U_{itv}) \ge (w-2)\cdot\tau$$
$$\forall\, w = 1,\dots,T',\ \forall v \in V \qquad (41)$$

In constraints (40), the restriction on time is enforced every period (i.e., every τ continuous time units). It allows monitoring closely the total time spent on all activities (traveling and loading/unloading). While the first term on the left hand side of the constraint represents the total completed travel time of vehicle v on all arcs up to a certain discretized period, w, the second term on the left hand side of the constraint represents the total loading/unloading time spent by that vehicle up to the same period. Note that both terms on the left hand side of this constraint represent continuous times, and so does the right hand side. Since the actual travel times may be lower than the time in a complete number of periods, a slack may be created by the first term on the LHS of the constraint, which may be used by the second term of the LHS of the constraint, by loading/unloading a larger quantity of bicycles than is actually possible in a certain number of periods, see also constraints (42)

and (43) below. In particular if a vehicle is dwelling at some station i during a period, which is represented by traveling from node i to node i, the whole τ units of time can be spent by loading and unloading bicycles since $t_{ii} = 0$ while $t'_{ii} = 1$.

In Constraints (41), a lower bound is enforced on the time restriction, with the purpose of keeping the time of the planned schedule close to its execution. This is important for the accuracy of the inventory levels at the nodes, which is necessary for the synchronization among vehicles, i.e., making sure that no vehicle plans to pick-up bicycles which are not brought there yet (by another vehicle). While lack of synchronization may still apply in the interval of two periods defined between the upper and lower bounds of Constraints (40) and (41), we believe that it is quite negligible.

Next, consider Constraints (34) and (35) and replace them by the following constraints:

$$y^L_{itv} \leq \min(2\bar{L}, c_i, k_v) \sum_{j \in N_0} x_{ijvt} \qquad \forall\, i \in N_0, t = 1, \dots, T',\ \forall\, v \in V \qquad (42)$$

$$y^U_{itv} \leq \min(2\bar{U}, c_i, k_v) \sum_{j \in N_0} x_{ijvt} \qquad \forall\, i \in N_0, t = 1, \dots, T',\ \forall\, v \in V \qquad (43)$$

where the following two parameters are defined and assumed to be integers:

$\bar{L}$ maximum number of bicycles that can be loaded during one period ($= \tau/L$)

$\bar{U}$ maximum number of bicycles that can be unloaded during one period ($= \tau/U$)

These constraints add a limitation on the loading / unloading quantities during one period (beyond the original limitations), which are related to the time it takes to perform these operations. Yet, the limitation is expressed as twice the actual number, to allow utilizing the slack that may have been created by the actual (rather than rounded up) travel times in Constraints (40).

Given the flexibility of the TI model in terms of its allowed options (re-visiting a node, etc.), we believe this formulation is a better representation of the real system than the AI formulation. Clearly, as the length of a period in the discretized model (τ) becomes shorter, the model becomes closer to the continuous one, but also includes a larger number of periods which make it harder to solve.

3.4 Sequence-Indexed Formulation

The next formulation is based on the idea of defining the journey of each vehicle according to the sequence of nodes which the vehicle visits. In this way, only one node index is required in the routing decision variables (the index of the node where the vehicle is located), instead of two in the previous two formulations (denoting the arc on which the vehicle traverses). The other index required, in addition to the vehicle index, is the position of the node in the sequence. The sequence indexed formulation handles time in an exact manner as in the AI formulation, but it allows several stops at a station as in the TI formulation. However, if the number of vehicles is greater than one, it constraints the number of bicycles transported to and from a certain station, as in constraints (7) and (8) of the AI formulation. The following parameters need to be defined and used in this formulation:

A an upper bound on the number of stops in the trip of each vehicle. In Section 4 we discuss how this parameter is derived;

$\bar{T}_j$ $\equiv \max_{i \in N_0} t_{ij}$ is the longest traveling time into node j from all nodes

The decision variables are defined as follows:

x_{iav} a binary variable which equals one if the a^{th} stop of vehicle *v* is at node *i*, and zero otherwise;

y^L_{iav} quantity of bicycles loaded at node i onto vehicle v in its a^{th} stop;

y^U_{iav} quantity of bicycles unloaded at node i out of vehicle v in its a^{th} stop;

r_{av} time of completing loading/unloading at the a^{th} stop of vehicle v;

y_{av} quantity of bicycles carried on vehicle v after loading/unloading at its a^{th} stop;

s_i inventory level at station i at the end of the repositioning operation;

(P3) – Sequence Indexed (SI) Formulation

$$Min \sum_{i\in N} g_i \tag{44}$$

s.t.

$$g_i \geq a_{iu} + b_{iu}s_i \qquad \forall i \in N, u = 0, \dots, c_i - 1 \tag{45}$$

$$s_i = s_i^0 - \sum_{v\in V}\sum_a (y^L_{iav} - y^U_{iav}) \qquad \forall i \in N \tag{46}$$

$$\sum_{i\in N} x_{iav} = 1 \qquad \forall a = 1, \dots, A,\ \forall v \in V \tag{47}$$

$$x_{01v} = 1 \qquad \forall v \in V \tag{48}$$

$$x_{0Av} = 1 \qquad \forall v \in V \tag{49}$$

$$r_{av} \geq r_{a-1v} - \bar{T}_j\left(1 - x_{jav}\right) + \sum_{i\in N_0} t_{ij}x_{i,a-1,v} + \sum_{i\in N_0}\left(Ly^L_{iav} + Uy^U_{iav}\right) \qquad \forall j \in N_0, \forall a \geq 2, \forall v \in V \tag{50}$$

$$r_{1v} = Ly^L_{01v} \qquad \forall v \in V \tag{51}$$

$$r_{Av} \leq T \qquad \forall v \in V \tag{52}$$

$$\sum_{a,v} y^L_{iav} \leq s_i^0 \qquad \forall i \in N_0 \tag{53}$$

$$\sum_{a,v} y^U_{iav} \leq c_i - s_i^0 \qquad \forall i \in N_0 \tag{54}$$

$$y^L_{iav} \leq \min(k_v, s_i^0)x_{iav} \qquad \forall i \in N_0, \forall a = 1, \dots, A,\ \forall v \in V \tag{55}$$

$$y^U_{iav} \leq min(k_v, c_i - s_i^0)x_{iav} \qquad \forall i \in N_0, \forall a = 1, \dots, A,\ \forall v \in V \tag{56}$$

$$y_{av} = y_{a-1v} + \sum_{i\in N_0}(y^L_{iav} - y^U_{iav}) \qquad \forall a = 1, \dots, A,\ \forall v \in V \tag{57}$$

$$y_{av} \leq k_v \qquad \forall a = 1, \dots, A,\ \forall v \in V \tag{58}$$

$$y_{0v} = 0 \qquad \forall v \in V \tag{59}$$

$$y_{Av} = 0 \qquad \forall v \in V \tag{60}$$

$$y^L_{iav} \geq 0, y^U_{iav} \geq 0, \text{ integers} \qquad \forall\, i \in N, \forall a = 1, \dots, A,\ \forall v \in V \tag{61}$$

$$x_{iav} \in \{0,1\} \qquad \forall\, i \in N, \forall a = 1, \dots, A,\ \forall v \in V \qquad (62)$$

$$y_{av} \geq 0 \qquad \forall a = 1, \dots, A,\ \forall v \in V \qquad (63)$$

$$s_i \geq 0 \qquad \forall\, i \in N \qquad (64)$$

The objective function (44) is identical to the one in the previous formulations and constraints (45) are the linear constraints that support the convex cost function, defined with respect to the ending inventory at each station, which is the inventory after loading/unloading during all visits. Constraints (46) define the inventory at the stations at the end of the repositioning operation. Constraints (47) make sure that each vehicle stops at one station at a time and constraints (48) (resp., (49)) verify that all vehicles depart from (resp., return to) the depot at the beginning (resp., end) of the repositioning operation. Constraints (50) define the minimal time required to reach node j which is the a^{th} stop, when the $(a-1)^{st}$ stop is at node i. The minimal time includes, beyond the time to reach node i, the loading/unloading times at node i and the travel time between i and j. Constraints (51) define the time in which vehicle v leaves its first stop (the depot) to be the completion time of loading there (note that the vehicle cannot unload in the first stop at the depot). Constraints (52) restrict the last visit of each vehicle to occur no later than time T. Constraints (53)-(54) limit the quantity picked-up from a node (resp., delivered to a node) in all stops of all vehicles to the quantity available there initially (resp., the remaining capacity of the station), as limited in the arc based formulation. These constraints make sure that the capacity of a node is not exceeded and the inventory at the node is non-negative. Constraints (55)-(56) allow loading and unloading of bicycles at a certain node only when a vehicle stops at that node. Constraints (57) are inventory balance constraints on each vehicle after each stop while constraints (58) are vehicle capacity constraints. Constraints (59) (resp., (60)) make sure that each vehicle is empty before it reaches the first stop (resp., after visiting the last stop, which is the depot). This stipulates that the total loading and unloading quantities equal to each other. Finally, (61)-(62) are integrality constraints and (63)-(64) are non-negativity constraints.

We note that if there is only one vehicle, there is no need for synchronization among vehicles and hence constraints (53)-(54) can be replaced by: $0 \leq s_i^0 - \sum_{a \leq A'}(y_{ia1}^L - y_{ia1}^U) \leq c_i \quad \forall A' = 1, \dots, A$ and the right hand side of constraints (55)-(56) should be replaced by $min(k_v, c_i)x_{iav}$ in order to allow unlimited transshipments.

Several valid inequalities, described next, were developed and found to be helpful in speeding-up the running time of the above formulation. The following constraints were added with respect to the minimal time required to reach the a^{th} stop, where $\widetilde{\mathrm{T}}_{\mathrm{j}} = \min_{\mathrm{i} \in \mathrm{N}_0 : \mathrm{i} \neq \mathrm{j}} \mathrm{t}_{\mathrm{ij}}$ and $\widehat{\mathrm{T}}_{\mathrm{i}} = \min_{\mathrm{j} \in \mathrm{N}_0 : \mathrm{i} \neq \mathrm{j}} \mathrm{t}_{\mathrm{ij}}$:

$$r_{av} \geq r_{a-1,v} + \sum_{j \in N}(Ly_{jav}^L + Uy_{jav}^U) + \sum_{j \in N} \tilde{T}_j x_{jav} \qquad \forall a \geq 2,\ \forall v \in V \qquad (65)$$

$$r_{av} \geq r_{a-1,v} + \sum_{j \in N}(Ly_{iav}^L + Uy_{iav}^U) + \sum_{i \in N} \hat{T}_i x_{i,a-1,v} \qquad \forall a \geq 2,\ \forall v \in V \qquad (66)$$

Constraints (65) state that the time to complete loading/unloading at the a^{th} stop is at least that of completing loading/unloading at the $(a-1)^{st}$, plus the time spent on loading/unloading at the a^{th} stop, plus the minimal time it takes to reach the a^{th} stop from any other node. Constraints (66) are similar, except that the last term is replaced by the minimal time it takes to reach another node when leaving the node visited in the $(a-1)^{st}$ stop. Note that there are even stronger special cases of constraints (65) for a=A and of constraints (66) for $a = 1$, since the last and first stops are forced to be the depot.

Next we add additional valid inequalities to break some symmetry in the problem, which is useful in reducing the search space:

$$x_{iav} + x_{i,a+1,v} \leq 1 \qquad \forall i \in N, a = 1, \dots, A-2, \ \forall v \in V \tag{67}$$

$$x_{iav} \leq y^{L}_{iav} + y^{U}_{iav} \qquad \forall i \in N, \ a = 2, \dots, A-1, \ \forall v \in V \tag{68}$$

$$x_{0av} + x_{0,a+1,v} - 1 \leq x_{0,a+2,v} \qquad \forall\, a = 1, \dots, A-2, \ \forall v \in V \tag{69}$$

Constraints (67) eliminate solutions with two consecutive stops at the same station and Constraints (68) eliminate solutions where a vehicle stops at some station but does not load or unload anything there. While these solutions are feasible they are weakly dominated by others. Constraints (69) make sure that no vehicle has two consecutive stops at the depot, except possibly last stops, after all loading/unloading activities are completed. This is achieved by requiring that once a vehicle stays two consecutive stops at the depot, it must remain there.

3.5 Swapping Based (SB) formulation

Our last formulation is motivated by the similarity of our problem to the swapping problem. As mentioned in the literature review, the swapping problem is similar to ours in that objects need to be moved from nodes where they are initially available, to nodes where they are demanded. Thus, our last formulation is based on the concept of supply and demand nodes. We denote by s_i^* the quantity of bicycles which minimizes the cost function $f_i(\cdot)$ at station i, and refer to each station i as a supply node if its initial inventory is larger than s_i^* and as a demand node if it is smaller. Moreover, to adhere to the swapping application and formulation, we aim to represent each node as a supply or demand node of one unit (bicycle) only. Thus, when the supply or demand quantity of a certain station is larger than one, we duplicate the station, and create as many stations as the supply or demand quantity. Visiting the first duplicated station associated with a certain original station is equivalent to picking or removing one unit from the original station, and we set the cost function of the duplicated stations accordingly, see our notation. All stations i with initial inventory of exactly s_i^* need not be included in the problem. Finally, the depot remains as a single node with the same role as in previous formulations.

The resulting problem has many more nodes than the number of stations. In fact, the number of nodes is exactly the number of units by which all stations (together) are away from their ideal (cost

minimizing) quantity. One may refer to it as the "amount of work" in the system. Thus, we expect that using this formulation may be useful only for systems in which the amount of work is not very high. On the other hand, for a given number of nodes, the problem is somewhat easier since there are no quantity decisions to make. Rather, when a node is visited, the activity performed in it is well defined by the node type (supply or demand) and always one unit is involved.

Compared with previous formulations, in the SB formulation vehicles may *not* redistribute bicycles in the system in any way that reduces costs. This is because every duplicated station is associated with either a surplus or a shortage (compared to the minimizing quantity) of one bicycle, so that only a pre-defined operation of removing (resp., delivering) is allowed from any particular node. On the other hand, visiting a station multiple times is possible; in fact it is obtained when visiting duplicated stations of the same original station not consecutively.

The swapping based (SB) formulation is similar to the arc indexed formulation, where each station represents a copy of the original one. More details on the SB formulation and how its unique characteristics are formulated compared to the AI formulation, are presented in Appendix B. As mentioned above, this formulation by itself is not expected to be an efficient one for most realistic systems (an expectation that was confirmed in a limited numerical experiment). However, the way in which the swapping and the repositioning problems are similar on one hand and different on the other hand is established and made clear by presenting this formulation. This may become useful for solving the repositioning problem if in the future an efficient algorithm for the swapping problem may be developed, in a way that can be easily amended to the repositioning problem.

We conclude this section, in which various formulations were presented for the repositioning problem, with a summary of the capabilities and assumption embedded in these formulations. This summary is organized in a tabular format, see Table 1.

Table 1: Comparing the assumptions and capabilities of the various formulations

Assumption	**Arc Indexed**	**Time Indexed**	**Sequence Indexed**	**Swapping Based**
Vehicles can visit each station an arbitrary number of times.	No. Each vehicle can visit each station at most once. Each station can be visited by several vehicles.	Yes	Yes	Yes
Transshipments are allowed, that is, vehicles can unload bicycles at nodes, to be loaded later on.	Yes, but the maximum number of bicycles that can be unloaded (resp. loaded) cannot exceed the initial residual capacity (resp., initial inventory level) at the node.	Yes, in an unlimited manner.	Same as in the Arc Indexed model except in the single vehicle case where transshipment are unlimited.	No.
Vehicles can dwell at stations in order to synchronize transshipments.	No. synchronization is guaranteed via restriction on loading and unloading quantities. See previous assumption.	Yes	Same as in the Arc Indexed model.	N/A. No Transshipments are allowed.
Bicycles can be redistributed in the system in any way that reduces total costs, even if it is not locally optimal to do so.	Yes	Yes	Yes	No
All bicycles are assumed to be unloaded at each visit of the vehicles to the depot.	Yes	No	No	Yes
Other considerations and comments	This model delivered the best results for most of the instances in our numerical experiment, in spite of its restrictive assumptions.	- Accuracy is lost due to the time discretization. - The model can be adapted to the dynamic problem.	Presumption on the maximum number of stops is required.	The size of the model is affected by the total amount of work. Indeed, the model could not solve most of our instances.

4. Algorithmic Enhancements

Solving the formulations presented in the previous section may be quite time consuming, even for instances of moderate size. In this section we discuss ways to speed-up the computation times of the various formulations. It includes solving the problems in two stages (Section 4.1), reducing the

number of binary variables based on geographical considerations (Section 4.2) and a careful selection of some of the parameters used in the formulations (Section 4.3). Some of these techniques are heuristic rather than optimal, but in our numerical experiments in Section 5 it is demonstrated that this fact has a marginal impact on the solution, and when a budget of time is given to solve the problem, it contributes to improving the overall solution in most cases.

4.1 A Two-Phase Solution Method

The two-phase solution method is motivated by the variables representing the quantity of bicycles loaded or unloaded at the various nodes. While these variables are required to be non-negative integers, which add a tremendous difficulty to solving the problem, their precise value tend to inflict only a minor effect on the optimal routing decisions.

Thus, in the two-phase solution method the problem is first solved while ignoring the integrality constraints of the loading/unloading variables, so that a solution to the routing decisions is obtained. In the second phase the obtained routing variables are fixed to their solution of the first phase, and the rest of the problem is then solved. We demonstrate in more details the motivation and implementation of this approach in the arc-indexed formulation, where the adaptation to other formulations is done in a similar manner.

Thus, consider again the AI formulation given in Section 3.2, and note that the integrality constraint in (13) may be omitted if there are no loading and unloading times, i.e., $L = 0$ and $U = 0$. In such a case there always exists an integer optimal solution to (P1), because once the values of the *x*'s are determined, the problem can be casted as a minimum cost flow problem with integer capacity and demand parameters. Positive values of loading or unloading times add a *knapsack* component to the problem through constraints (10) and hence further complicate it. However, we observed empirically, that even if the integrality constraints are removed, the number of non-integer values obtained in the optimal solution is small (in most cases, not more than two non-integers along the route of each vehicle).

In the two-phase solution approach, the first phase includes solving problem (P1') which is identical to problem (P1), except that the integrality constraints in (13) are removed. It is still a mixed integer program, but one which is simpler to solve. Then, the second phase involves solving another mixed integer program, denoted by (P1''), obtained by fixing the values of the *x*-variable to the optimal values they obtained in (P1') and where the integrality of y_{iv}^{L} and y_{iv}^{U} is added back. Obtaining an optimal solution to problem (P1'') is achieved very quickly.

Specifically, consider the AI formulation before it is strengthened by the valid inequalities, given by the objective function (17) subject to constraints (2)-(16) and (18). In (P1''), the objective function (17) remains unchanged, and so do constraints (2), (3), (7)-(9) and (13)-(16), which do not include (directly, see below) the x-variables. Then, constraints (5), (6), (11) and (12) which include x-variables only (and the related q-variables) are removed. Finally, the remaining constraints ((4) and

(10)), which include both x and y variables, are modified as follows, using x^*_{ijv} to denote the solution of the x-variables obtained in the first phase:

$$y_{ijv} \leq k_v x^*_{ijv} \qquad \forall\, i,j,v \qquad (70)$$

$$\sum_{i\in N_0}(Ly^L_{iv} + Uy^U_{iv}) \leq T - \sum_{i,j\in N_0: i\neq j} t_{ij}x^*_{ijv} \qquad \forall\, v \in V \qquad (71)$$

In these modified constraints, note that (70) in fact sets to zero all y_{ijv} variables whose equivalent x^*_{ijv} variables are equal zero. This is equivalent to removing all those variables, as indeed performed by the pre-solver. This modifies, indirectly, constraints (3) and (14), so that they are defined only for y_{ijv} variables whose equivalent x^*_{ijv} variables are equal to one. The modified constraints (71) is a tighter version of the time constraint (10), where the total travel time of a vehicle is subtracted from both sides of the inequality. The modifications with respect to the valid inequalities (19)-(23) are similar to those described above.

The solution to (P1") delivers a feasible solution to (P1), with integer values of y^L_{iv} and y^U_{iv} and with an objective value that is numerically shown (in Section 5) to be very close to the lower bound obtained by solving (P1) without integrality constraints on these variables.

4.2 Arc Deletion

While the previous section described a way to (heuristically) overcome the integrality requirement concerning the loading and unloading variables, in this section we focus on the routing variables (the x-variables) in the TI model with the goal of reducing their number significantly. In this case the reduction is exact rather than heuristic, and results from the following geographical considerations. In an urban environment, the movement of vehicles is limited to the road network and is subject to various regulations such as "one way streets", "no left turns", etc. Consequently, the shortest valid journey between a pair of stations is likely to pass via other stations. Thus, we can represent the stations network as a sparse graph where stations (i,j) are connected by an arc if only if there is no other station, say k, such that $t_{ik} + t_{kj} = t_{ij}$. Otherwise, the journey from i to j can be represented by a path of two arcs, (i,k) and (k,j), which takes the same amount of time as the direct path from i to j. Using this observation, a significant number of variables that correspond to arcs may be reduced. This idea is implemented by defining for all $i \in N_0$ the set $\delta_i = \{j \in N_0\colon\ t_{ij} < t_{ik} + t_{kj}\ \ \forall k \in N_0\}$. Now in order to accommodate this *arc deletion* concept, we revise the definition of the indices of the routing variables, and possibly other related variables. For example, the *x* and *y* variables in (37) and (38) are defined as follows:

$$x_{ijtv} \in \{0,1\},\ y_{ijtv} \geq 0 \qquad \forall i \in N_0, j \in \delta_i\ , \forall v \in V, t = 1,\dots,T', v \in V$$

In addition, reference to undefined variables should be removed by revising all constraints in which these variables appear.

Note that the above arc deletion procedure is different from the *heuristic concentration* procedure, see e.g., Rosing and ReVelle (1997). The latter is a known approach to reducing the number of arcs by using information from previous runs, but it cannot assure that the removed arcs are not part of the optimal solution. Our procedure, on the other hand, removes only arcs that, for certainty, can be replaced by alternative paths with the same cost. To the best of our knowledge, this idea is new. By using the arc deletion procedure, we drastically reduce the size of our mixed integer program and increase the size of instances that can be solved in a reasonable amount of time. In the largest instances that we tested, about 80% of arcs could be removed.

Note that the above method may not be applied to the AI (resp., SB) formulation because in this formulation each vehicle is allowed to visit each station (resp., station's node) only once. If visits are "wasted" on constructing routes to other nodes when the inventory or the residual capacity on the vehicle is not sufficient to serve the station, then this limitation becomes too restrictive. However the idea of arc deletion may be beneficial for arc index formulations of other vehicle and inventory routing problems, when the order of the visits at the nodes is not important. The arc deletion idea is also not directly applicable to the SI formulation since its decision variables correspond to stops rather than arcs.

4.3 Parameter Setting

In this section we highlight several issues that need to be addressed when selecting parameter values for our models. Our goal is to find values that enable solving large problem instances, possibly at a moderate cost of optimality or the validity of the obtained lower bound. We note that since the static repositioning problem is solved by the operator on a daily (or nightly) basis, the value of the parameters that suite best the system at hand, can be learnt over time.

In the TI formulation, a major parameter that affects the difficulty of solving the problem on one hand, and the model's accuracy (or flexibility) on the other hand, is the value of the period length, τ. As in most discretized models, when the length of a period gets larger, it becomes easier to solve the problem due to the reduced number of variables, but the solution is less adaptive to desired changes, since decisions are taken less frequently. The choice of τ must clearly be related also to the prevailing times in the bike sharing system for which a solution is sought.

In the SI formulation, A represents an upper bound on the number of stops in the trip of each vehicle. A can be safely chosen to be $T/\min_{i\neq j\in N_0} t_{ij}$, but typically this number is significantly higher than the actual number of stops in the optimal solution. Reducing A reduces the number of decision variables in a linear rate, and this helps solving the problem. If A is chosen to be smaller than the (unknown in advance) optimal number of stops, the optimal solution of the resulting problem is worse than the optimal solution with the unbinding number of stops, however, since the ability to

solve the former is likely to be better than the latter, it still may produce better results, especially under a time limit constraint. This issue is examined in our numerical experiment in Section 5.

5. Numerical Experiments

In this section we demonstrate the performance of the mixed integer programming formulations introduced in Section 3, together with the algorithmic enhancements presented in Section 4, when solving instances of practical size. We first present results of three of our formulations that were applied on 24 instances that are based on data of the Vélib system. Then we applied the formulation that performed best on an entire real system consisting of 104 stations of Capital Bikeshare in Washington DC.

In our first numerical experiment we created a set of 24 benchmark problems based on actual locations of some 60 Vélib stations located in Paris' First quarter. We conducted an experiment with all combinations of the following parameter values:

- Number of nodes – 30, 45, and 60 nodes, where the smaller instances are subsets of the larger ones.
- Number of vehicles – one and two vehicles.
- Length of the planning horizon – 2.5 hours and 5 hours.
- Penalty functions, based on demand patterns – two penalty functions were considered where the first is based on a fictitious (but likely a representative) demand pattern while the second is based on an estimated demand pattern from data that we collected from the Vélib website during some ten working days. The initial inventory levels at the stations were selected randomly. The distance matrix was calculated based on the L_1 (Manhattan) metric. The distances are given in meters and the average speed of the vehicles is assumed to be one meter/second. The loading and unloading times were set to be 1 minute/bicycle. The vehicle's capacity was set to 20 bicycles which is the capacity of the light trucks used by Vélib. The location of the depot was selected arbitrarily in the First quarter. The capacity and the initial inventory of the depot were set to be large enough so that they were not binding. The dataset for our benchmark problems is available through the first author's website at http://www.eng.tau.ac.il/~talraviv/ under "Publications".

We implemented the Arc Indexed formulation (AI), the Arc Indexed formulation with two phases (AI2), the Time Indexed formulation with two phases (TI2) and the Sequence Indexed formulation with two phases (SI2) using IBM-Ilog OPL and solved the above instances using IBM-Ilog CPLEX on an Intel Xenon X3440 @ 2.53GHz with 8GB of RAM. In all our experiments we used CPLEX default settings and set the solver time limit to two hours. For the two phase procedure the time limit was imposed only on the first phase, since the time needed to perform the second phase was negligible in all instances and formulations (less than one second). Under such time limit, the main

memory was never exploited. The optimality tolerance was left with CPLEX's default of 0.0001. That is, the solver stopped whenever a provable optimality gap of 0.01% was obtained.

We also experimented with the Swapping Based model and the single phase Time Indexed and Sequence Indexed formulations, but we found that many of our instances could not be solved to optimality and even did not result in solutions that are within a reasonable optimality gap. Hence, we do not report on these results.

We first conducted an experiment to evaluate the effect of the two-phase procedure, by comparing AI and AI2, as reported in Table 2.

The four left columns of the table describe each instance in terms of number of vehicles, penalty function, number of nodes and the length of the planning horizon in hours. The next four columns summarize the performance obtained from the AI formulation. The first two report on the lower bound and the best integer obtained for each instance. We then present the provable relative optimality gap, calculated according to the CPLEX convention, that is,

$$Relative\ Gap\ (AI) = \frac{Best\ Integer\ Solution - Lower\ Bound}{Best\ Integer\ Solution}.$$

The next column presents the CPU time required to solve the instance within the optimality tolerance, where 7200 seconds (two hours) means that CPLEX stopped before achieving a solution within the optimality tolerance and thus only values lower than 7200 represent the time needed to solve the problem to optimality.

The next six columns present similar information on our experiments with the AI2 formulation. The lower bound column is the lower bound obtained in the first phase, which is the only valid lower bound we obtain using this formulation. In the column headed "Phase 1", the value of the best integer solution (with respect to the routing variables) obtained in phase 1 is presented. Recall that this solution is not necessarily an integer feasible solution of the problem at hand because it may (and typically does) include fractional loading and unloading quantities. Under "Phase 2" we present the solution obtained by this formulation and under "Relative Gap" we compare the second phase solution with the first phase lower bound, by computing:

$$Relative\ Gap(AI2) = \frac{Solution\ phase\ 2 - Lower\ Bound\ phase\ 1}{Solution\ phase\ 2}.$$

The column "Phase 2 Gap" represents a post-priori upper bound on the loss of optimality incurred by the decomposition of the problem into two phases and is calculated as follows:

$$Phase\ 2\ Gap = \frac{Best\ Integer\ solution\ phase\ 2 - Best\ Integer\ solution\ phase\ 1}{Best\ Integer\ solution\ phase\ 2}.$$

For each instance we denote in bold face the lower bound, best integer and CPU time that were best among the two formulations.

It is apparent from Table 2 that the phase 2 gap incurred by this decomposition is close to negligible (0.17% on average and 0.34% in the worst case). We observe that while the AI formulation is capable of providing slightly better solutions for instances with one vehicle and small number of

nodes, AI2 outperforms AI for the larger (and harder) instances, both in terms of best integer solution and valid lower bound obtained. The average optimality gap for AI is 3.69% while for AI2 it is 2.41%. Moreover, for the two vehicle instances the figures are 7.24% vs. 4.54%, respectively. The instance with the largest optimality gap when solving by AI had a gap of 11,1% while the instance with the largest optimality gap when solving by AI2 had a gap of 6.28% only. For all the smaller instances that can be solved to optimality within two hours, AI2 converge faster with only one exception.

As expected, the performances of both formulations are adversely affected by both the number of nodes and the number of vehicles. Interestingly, the performances seem not to be significantly affected by the length of the planning horizon. In AI2, the average optimality gap is not significantly affected by the length of the planning horizon or by the penalty function type. The former may imply that while geographical decomposition of the problem may be beneficial, there is no benefit in temporal decomposition when using this formulation.

We note that in order to obtain a tight upper bound on the number of arcs in the route of each vehicle, M, we formulated an integer program. We solved this program for each of the six combinations of distance matrices and planning horizon lengths in very few minutes. This preprocessing operation can be carried out once for each bike-sharing system and used repeatedly every day as long as the distance matrix and time allotted to the repositioning operation remains unchanged. Thus, we decided not to include these times in the processing times reported in Table 2.

It is worth mentioning that although most of the larger instances could not be solved to optimality within two hours, most of the progress towards the obtained solution was carried out by the solver during the very first minutes. That is, the AI and AI2 formulations are applicable even under much tighter time constraints.

Next, we present the results of our experiment of the Time Indexed formulation with the two phase procedure (TI2) and while applying the arc deletion enhancement. We applied this formulation to all of our 24 instances with two different time discretization levels, namely 300 seconds and 450 seconds per period, resulting in 48 runs in total. None of the problems converged to optimality within the two hour time-limit, but high quality solutions could be obtained for many instances. In five of the instances a feasible solution could not be obtained within the two hour time-limit.

Instance				AI				AI2					
vehicles	Penalty function	Nodes	T (Hours)	Lower Bound	Best Integer	Relative Gap	CPU Time (Sec)	Lower Bound	Phase 1	Phase 2	Relative Gap	CPU Time (Sec)	Phase 2 Gap
1	1	30	2.5	**214.22**	**214.24**	0.01%	73	213.76	213.76	214.5	0.35%	**24**	0.34%
			5.0	**167.05**	**167.07**	0.01%	2519	166.74	166.74	167.1	0.22%	**1036**	0.21%
		45	2.5	**335.15**	335.18	0.01%	1217	**334.40**	334.40	335.18	0.24%	**939**	0.23%
			5.0	**273.10**	**273.13**	0.01%	3425	272.97	272.97	273.54	0.22%	**1614**	0.21%
		60	2.5	457.94	**460.86**	0.63%	7200	460.54	460.54	461.32	0.28%	7200	0.17%
			5.0	**382.34**	382.38	0.01%	1002	**382.30**	382.30	382.38	0.03%	**966**	0.02%
	2	30	2.5	**139.52**	139.52	0.00%	**5**	**139.49**	139.49	139.52	0.02%	14	0.02%
			5.0	**109.32**	**109.33**	0.01%	5811	109.16	109.16	109.52	0.34%	**2796**	0.33%
		45	2.5	**203.21**	**203.23**	0.01%	151	203.11	203.11	203.73	0.31%	**125**	0.30%
			5.0	**166.27**	**166.29**	0.01%	2958	166.00	166.00	166.37	0.23%	**2343**	0.22%
		60	2.5	286.89	**289.55**	0.92%	7200	**289.32**	289.32	289.76	0.98%	7200	0.15%
			5.0	**242.57**	242.59	0.01%	313	**242.32**	242.32	242.59	0.12%	**129**	0.11%
2	1	30	2.5	157.24	**171.49**	8.31%	7200	**171.72**	171.72	171.84	5.84%	7200	0.07%
			5.0	113.37	116.62	2.79%	7200	**116.44**	116.43	116.62	2.50%	7200	0.16%
		45	2.5	257.85	280.73	8.15%	7200	**279.17**	279.17	**279.68**	4.51%	7200	0.18%
			5.0	185.50	198.86	6.72%	7200	**197.71**	197.71	**197.95**	4.00%	7201	0.12%
		60	2.5	364.40	400.34	8.98%	7200	**399.52**	399.52	**399.53**	5.69%	7200	0.00%
			5.0	269.04	302.63	11.10%	7200	**291.78**	291.78	**292.27**	4.55%	7200	0.17%
	2	30	2.5	103.98	109.18	4.76%	7200	**108.75**	108.75	**108.9**	2.65%	7200	0.13%
			5.0	**66.45**	**69.04**	3.76%	7200	68.94	68.94	69.13	4.00%	7200	0.27%
		45	2.5	154.37	168.06	8.15%	7200	**167.56**	167.56	**168.01**	4.69%	7200	0.27%
			5.0	107.49	115.9	7.26%	7200	**115.04**	115.04	**115.07**	4.49%	7200	0.02%
		60	2.5	230.51	253.28	8.99%	7200	**251.00**	251.00	**251.52**	6.28%	7200	0.21%
			5.0	173.14	188.02	7.92%	7200	**187.44**	187.44	**187.64**	5.32%	7200	0.11%

Table 2: Results for the Arc Indexed and Arc Indexed with two phases Models

A summary of the results is presented in Table 3. The columns headings of this table are similar to those of Table 2 and thus we omit their description here. The phase 2 objective function values were all within 1% of those of phase 1 and thus the latter are not reported in the table. Cells related to instances that could not be solved within the time limit were left blank.

It is apparent from Table 3 that the performances of the time indexed formulation, similarly to the arc indexed formulation, are adversely affected by the number of vehicle and the number of nodes in the network. In contrast with the former formulations, TI2 is also adversely affected by the length of the planning horizon because it directly affects the number of variables in the model. This suggests that when using the TI2 formulation, temporal decomposition of the problem may be beneficial.

Instance				Crude Time Disc. (τ =450 sec.)			Fine Time Disc. (τ =300 sec.)		
Trucks	Penalty function	Nodes	T (hours)	Lower Bound	Second Phase	Relative Gap	Lower Bound	Second Phase	Relative Gap
1	1	30	2.5	204.84	215.19	4.81%	202.78	**214.50**	5.47%
			5.0	151.98	**172.74**	12.02%	151.88	176.23	13.82%
		45	2.5	318.05	**334.74**	4.98%	315.78	338.64	6.75%
			5.0	246.08	279.03	11.81%	246.03	**277.45**	11.33%
		60	2.5	437.84	**460.82**	4.99%	437.73	460.88	5.02%
			5.0	354.15	**386.35**	8.34%	354.32	397.75	10.92%
	2	30	2.5	134.94	140.37	3.87%	133.50	**139.52**	4.32%
			5.0	99.04	110.20	10.13%	99.02	**108.74**	8.94%
		45	2.5	192.32	203.73	5.60%	191.80	**203.41**	5.71%
			5.0	151.04	169.54	10.91%	151.08	**169.41**	10.82%
		60	2.5	273.54	**290.49**	5.83%	273.45	290.79	5.96%
			5.0	226.97	251.07	9.60%	227.13	**248.38**	5.96%
2	1	30	2.5	155.47	**174.44**	10.88%	155.71	179.50	13.25%
			5.0	112.05	**152.51**	26.53%	112.30	246.64	54.47%
		45	2.5	254.45	**285.03**	10.73%	254.84	294.81	13.56%
			5.0	182.49	**369.80**	50.65%	-	-	-
		60	2.5	365.02	**416.36**	12.33%	365.97	447.09	18.14%
			5.0	-	-	-	-	-	-
	2	30	2.5	101.43	**112.05**	9.48%	101.68	112.79	9.85%
			5.0	63.09	**79.58**	18.29%	65.39	106.13	38.39%
		45	2.5	154.15	**172.83**	10.81%	154.54	179.61	13.96%
			5.0	106.77	**204.78**	47.86%	-	-	-
		60	2.5	231.50	**255.80**	9.50%	231.89	268.21	13.54%
			5.0	265.6133	**538.14**	50.64%	-	-	-

Table 3: Results for the Time Indexed formulation with the two phase procedure and arc deletion (TI2)

We observed that the crude time discretization ($\tau = 450\ sec.$) delivers better solutions for the harder problem instances, while the finer time discretization ($\tau = 300\ sec.$) which results in a much larger mathematical model, yields better solutions only for few single vehicle instances, mostly with smaller number of nodes. Indeed, the solver fails to solve, within the two hour time-limit, even the root node of four out of the 24 problem instances when using the fine time discretization and only one instance when using the crude time discretization.

Overall, the optimality gaps are much larger with the TI2 formulation with both fine and crude time discretization, compared to the AI2 formulation. Additionally, the actual best solution obtained by TI2 was inferior to that of AI2 in 20 out of the 24 instances.

We note that the lower bounds and optimality gaps obtained by both time discretization levels are not exactly comparable with those of the arc indexed formulation. This is since the TI2 model allows

more flexibility in the routing decisions and in loading and unloading operations, so that the optimal solution value of this model is generally lower. Moreover, the lower bounds of TI2 with different time discretization levels are not comparable, since the finer discretization level provides more flexibility.

Next, we present the results of our experiment with the SI2 formulation over the same set of 24 problem instances. As Discussed in Section 4.3, there is a need to set an upper bound, A, on the number of stops per vehicle. We used two values for each time horizon: $A = 20$ and $A = 30$ for the 2.5 hours instances and $A = 40$ and $A = 60$ for the 5 hours instances, which provide the opportunity to stop every 450 or every 300 seconds on average, and thus to keep in line with the time discretization in our TI2 experiment, described above. None of the 48 runs converged to optimality within two hours, but quite small optimality gaps were obtained for many of them. As with the former formulations, the deterioration of the objective function value from the first to second phase was negligible, 0.13% on average and 0.35% in the worst instance.

A summary of our experiment with SI2 is presented in Table 4. The structure of the table is similar to that of Table 3. For each instance we report on the solutions obtained with a large number of stops (30/60) and with a small one (20/40). We also report on the number of branch and bound nodes explored by CPLEX and the actual number of stops (A.S) before reaching the depot for the last time, in the obtained solution. For the instances with two vehicles, this figure represents the total number of stops of both vehicles.

It is apparent from Table 4 that the SI formulation is relatively robust to the selection of the number of stops parameter within the range that we tested. Indeed, in most cases better solution were obtained with smaller number of stops (20/40), but the differences were minor, and in five cases, better solutions were obtained with 30/60 stops. This fact is somewhat intriguing because we note that in all of our runs the total number of stops was substantially smaller than the maximum stops allowed by the A parameter, so one could expect that a lower value of A which results in a smaller number of variables in the formulation would perform a lot better. It is also surprising that the number of branch and bound nodes explored by CPLEX was substantially larger for the runs with smaller number of stops. The fact that the number of stops is by far not binding in all of our runs suggests that the lower bound obtained by this formulation is likely to be valid although this is not guaranteed.

Lastly, we compared the results obtained from the different formulations and parameter settings. In Table 5 we present the values of the best solutions obtained for each instance using formulations AI, AI2, TI2 with period lengths of 300 and 450 seconds, and SI2 with 30/60 stops and 20/40 stops. The best known solution of each instance is set in boldface. It is apparent from the table that the arc indexed formulation "wins" in most instances and in particular AI is the best in most of the single vehicle instances and AI2 is the best in most of the harder, two vehicle instances. The two TI2

formulations combined deliver the best solution, by a small margin, to three out of the 24 instances, but largely lag behind in most of the others.

Instance				30/60 stops					20/40 stops				
Trucks	Penalty function	Nodes	T	Lower Bound	Second Phase	Relative Gap	B&B Nodes	A.S	Lower Bound	Second Phase	Relative Gap	B&B Nodes	A.S
1	1	30	2.5	194.12	**214.52**	9.51%	111315	10	198.97	215.07	7.49%	624229	10
			5.0	140.28	**169.15**	17.07%	19950	18	140.29	170.66	17.80%	35942	21
		45	2.5	305.79	336.88	9.23%	60756	10	307.91	**336.85**	8.59%	112293	11
			5.0	236.18	283.49	16.69%	6549	17	236.03	**275.59**	14.36%	22616	18
		60	2.5	427.53	464.07	7.87%	142283	10	430.38	**464.02**	7.25%	304196	9
			5.0	337.75	400.68	15.70%	5217	17	342.30	**385.85**	11.29%	30807	19
	2	30	2.5	129.12	139.52	7.46%	297993	12	130.83	139.52	6.23%	436290	12
			5.0	82.93	**108.90**	23.85%	18052	18	83.68	109.17	23.35%	83329	19
		45	2.5	186.40	203.41	8.36%	116618	11	188.26	203.41	7.45%	330590	11
			5.0	134.45	168.90	20.40%	9468	20	135.20	**167.96**	19.50%	26741	18
		60	2.5	268.43	**289.76**	7.36%	99155	11	269.23	289.98	7.16%	164690	11
			5.0	207.95	**251.01**	17.15%	11019	19	209.40	251.12	16.61%	26035	19
2	1	30	2.5	140.71	172.16	18.27%	33917	21	140.87	**172.10**	18.15%	51560	21
			5.0	107.58	123.77	13.08%	5334	33	107.58	**117.41**	8.37%	27557	39
		45	2.5	236.71	285.77	17.17%	11265	19	236.72	**284.03**	16.66%	25313	23
			5.0	172.39	215.63	20.05%	609	35	172.49	**221.12**	21.99%	1244	40
		60	2.5	333.94	420.06	20.50%	1832	18	339.95	**405.37**	16.14%	38445	22
			5.0	245.02	**324.26**	24.44%	823	32	245.40	328.62	25.33%	1460	31
	2	30	2.5	83.50	109.51	23.75%	42923	21	83.85	**108.95**	23.04%	81216	22
			5.0	52.87	73.36	27.93%	6460	36	52.87	69.46	23.88%	24932	40
		45	2.5	135.42	168.14	19.46%	23421	21	135.79	168.40	19.37%	48069	22
			5.0	84.35	123.83	31.88%	1145	46	84.39	131.96	36.05%	2376	33
		60	2.5	206.41	258.41	20.12%	11471	22	209.09	256.09	18.35%	39999	22
			5.0	132.91	215.00	38.18%	809	28	132.80	202.18	34.32%	2123	35

Table 4: Results for the Sequence Indexed formulation with the two phase procedure (SI2)

In the last row of Table 5 we present the average relative deviation of the result obtained by the corresponding method from the best known solution, obtained by one of the six methods. Instances that could not be solved were not included in this average and hence the results for TI2 (the only method that failed to solve some of the instances) is biased down. However, even with this bias it is clear that on average TI2 is inferior to the two other formulations in terms of average performances. The AI2 formulation was substantially better on average than all other methods while SI2 was in between.

Vehicles	Penalty function	Stations	T	AI	AI2	TI2 τ=300	TI2 τ=450	SI2 30/60	SI2 20/40
1	1	30	2.5	**214.24**	214.50	214.50	215.19	214.52	215.07
			5.0	**167.07**	167.10	176.23	172.74	169.15	170.66
		45	2.5	335.18	335.18	338.64	**334.74**	336.88	336.85
			5.0	273.13	273.54	277.45	279.03	283.49	275.59
		60	2.5	460.86	461.32	460.88	**460.82**	464.07	464.02
			5.0	**382.38**	382.38	397.75	386.35	400.68	385.85
	2	30	2.5	**139.52**	139.52	139.52	140.37	139.52	139.52
			5.0	109.33	109.52	**108.74**	110.20	108.90	109.17
		45	2.5	**203.23**	203.73	203.41	203.73	203.41	203.41
			5.0	**166.29**	166.37	169.41	169.54	168.90	167.96
		60	2.5	**289.55**	289.76	290.79	290.49	289.76	289.98
			5.0	**242.59**	242.59	248.38	251.07	251.12	251.12
2	1	30	2.5	**171.49**	171.84	179.50	174.44	172.16	172.10
			5.0	**116.62**	**116.62**	246.64	152.51	123.77	117.41
		45	2.5	280.73	**279.68**	294.81	285.03	285.77	284.03
			5.0	198.86	**197.95**	-	369.80	215.63	221.12
		60	2.5	400.34	**399.53**	447.09	416.36	420.06	405.37
			5.0	302.63	**292.27**	-	-	324.26	328.62
	2	30	2.5	109.18	**108.90**	112.79	112.05	109.51	108.95
			5.0	**69.04**	69.13	106.13	79.58	73.36	69.46
		45	2.5	168.06	**168.01**	179.61	172.83	168.14	168.40
			5.0	115.90	**115.07**	-	204.78	123.83	131.96
		60	2.5	253.28	**251.52**	268.21	255.80	258.41	256.09
			5.0	188.02	**187.64**	-	538.14	215.00	202.18
Average above best known				0.29%	0.08%	7.03%	9.64%	3.18%	2.45%

Table 5: Comparing the results obtained from the different models and parameters

Overall, the AI2 formulation yielded good solutions to instances with up to 60 nodes and two vehicles and thus may be a good tool to provide to operators of bike sharing systems. For systems of moderate size the entire system may be solved together. For larger systems we believe that a practical approach is to decompose the entire area into several geographical areas (clusters), such that a small number of vehicles (say, up to three) perform repositioning in each area, with possibly a common local warehouse (or a particularly large station), allowing the transfer of bicycles between different areas, if needed. Such an approach is practical and easier to manage since it allows appointing an area manager, responsible for a small number of drivers who become familiar with their area. It is, in fact,

a common practice in many distribution systems which consist of a large number of nodes. When such a decomposition approach is used, our methods are likely to solve (to optimality or near optimality) the resulting separate problems. A related question is clearly how to decompose the system into separate areas, but this is beyond the scope of this paper.

We note that the flexibility of the AI2 model is limited compared to the SI2 and TI2 models and thus the *optimal* solutions of these models are generally better than those of AI and AI2. However, as the objective functions of all our models are exactly comparable, we learn from Table 5 that in spite of its modeling limitations, AI2 achieves better solutions in most cases. We may infer from this observation that for this problem, as in some other problems from the literature, good solutions may be achieved by slightly simplifying the problem in a way that makes it computationally simpler.

Recall that we set the loading and unloading times of bicycles to one minute per unit. We believe that this is a reasonable value when taking into account some overhead tasks of the repositioning staff at the stations (e.g., checking the mechanical status of the bicycles and lockers). However, if the actual loading and unloading times are smaller, possibly thanks to automated equipment that aids to perform these tasks, the transshipping of bicycles via third stations may became more attractive. Under such conditions, since the TI2 formulation allows more flexibility with transshipments, this method may perform better. In addition, both SI2 and TI2 may perform better in situations where the capacity of the vehicles is smaller compared to the capacity of the stations and thus multiple visits at each station is required.

Next, we applied our model and solution method on real problem instances obtained from the operator of Capital Bikeshare in Washington DC. The distance matrix represents real driving times between each pair of stations. The arrival rates of renters and returners at each hour of the day on regular weekdays were estimated based on a complete transaction log collected during a three months period starting November 1, 2010. Moreover, since the demand in Capital Bikeshare during this period was low and the system was in its early stages of implementation, we created another set of instances with the same network but with demand rates multiplied by two and stations' size multiplied by 1.5. This set is referred to as the "inflated" set. A user dissatisfaction function for each of the systems' 104 stations was created based on the above demand data and station capacity.

We solved the static repositioning problem of these systems using one and two vehicles with time horizon of 2.5 hours and 5 hours. The capacity of the vehicles in the instances below is set to 25, the same as the light trucks used by the operator of Capital Bikeshare. In Table 6 we present the results obtained from solving the AI2 formulation with a time limit of two CPU hours. In the table, we present the following three measures for each problem instance (columns 4-6):

- **Initial value** - the value of the objective function when no repositioning is performed, i.e., each station remains with its initial inventory of bicycles. This is a naive upper bound for the problem.

- **Ideal value** - the value of the objective function given the optimal inventory of bicycles at each station, ignoring capacity and time constraint on the repositioning vehicles. This is an obvious lower bound.
- **To add / remove -** the total number of bicycles that needed to be added and removed from all the stations in order to get from the initial state of the system to the ideal one.

For each instance we report on the Integer Programming lower bound, upper bound (best integer solution), and calculate the relative optimality gap as in the previous tables. In addition, we report on the actual number of bicycles that were added/removed in the best integer solution found. Finally, in the right most column, we report on the ratio between the total number of bicycles that were added and removed in the actual solution and the number that should be added and removed in order to get to the ideal solution. This ratio is referred as "Job done".

Instance						AI2 Solution				
Demand / station size	Vehicles	T (Hours)	Initial Value	Ideal Value	to add / remove	Lower Bound	Best Integer	Relative Gap	Added / Removed	Job Done
Original	1	2.5	284.29	189.33	150/ 197	222.60	227.53	2.17%	43/43	24.78%
		5.0				197.33	200.21	1.44%	83/83	47.84%
	2	2.5				194.87	206.01	5.41%	77/76	44.09%
		5.0				189.33	197.28	4.03%	119/126	70.61%
Inflated	1	2.5	592.48	413.57	242/ 314	509.50	510.87	0.27%	48/48	17.27%
		5.0				451.23	456.65	1.19%	96/96	34.53%
	2	2.5				447.26	471.88	5.22%	84/84	30.22%
		5.0				414.11	431.53	4.04%	150/150	53.96%

Table 6: Comparing the results obtained from the different models and parameters

It is apparent from the table that high quality feasible solutions with optimality gaps of few percents can be obtained using our solution method. There is no significant difference between the optimality gaps of the original instances and the inflated ones. This implies that the method is robust to the traffic volume of the system. The optimality gaps of the two vehicles instances are larger than those of the single vehicle ones. Surprisingly, the optimality gaps of the 2.5 hours instances are larger than those of the 5 hours instances in three out of four cases. As expected, the value of the objective function decreases as the number of vehicles and time allotted to the repositioning increases. In correspondence, the share of the job done is also increased.

We observe that even when the objective function value of the best solution found is close to the ideal value and far apart from the initial one, the share of the job done is significantly smaller than 100%. See for example, the fourth and last instances in the table. This is due the fact that the dissatisfaction function is convex and typically almost flat around its minimum. Indeed, it seems that in some stations the operator is almost indifferent between the exact optimal inventory and many

other possibilities close to it. We believe that this very phenomenon justifies the use of dissatisfaction functions rather than setting target values for the inventory at each station and solving a "many to many" single commodity pickup and delivery problem with a standard objective function. Indeed, the operator should allocate its limited repositioning capacity to the stations that are expected to face the largest number of shortage events during the next day.

6. Conclusions and discussion

This paper defines and formulates a new rich inventory routing model that is motivated by the need to regulate the new emerging bike sharing systems. Proper regulation of these systems by repositioning bicycles among the stations is an important factor in the success of these systems. This paper is the first attempt to consider all aspects of the static repositioning problem, to formulate it as a mixed integer linear program and address various technical obstacles that arise in solving large instances. Indeed, we offer four, essentially different formulations, based on different modeling assumptions. In all our models the objective is to minimize the expected cost of events in which the service cannot be provided. Due to the stochastic nature of the demand for bicycles, this objective function turns to be non-linear. We build on previous results that establish the convex nature of this function in order to linearize it.

In addition to the immediate contribution to the domain of bike-sharing system management, we believe that some of the concepts developed in this paper are relevant to other inventory routing and pickup and delivery problems. In particular, our methods may be applicable to other inventory routing problems with stochastic demand. Furthermore, the concept of arc deletion, described in Section 4.2, may be useful to many other complex routing problems.

Since we address a new problem, it can be extended in many different directions. One is examining the effect of substitutability and complementarity of the demand function among stations. This relaxes our current assumption that demand at different stations is independent. Although we believe that our notion of user dissatisfaction is a good approximation of reality, verifying this point is currently under investigation.

One may argue that a fixed amount of time needs to be added every time the vehicle stops at a station, in addition to the linear loading and unloading times that are already included in our model. This case can be handled by adding the fixed time to the travel times between stations. Note however, that then it will not be possible to use the arc deletion method as is. To use our TI (and TI2) models, a new set of binary variables denoting actual stops at stations (rather than passing through them) should be introduced. One binary variable is required for each period and for each vehicle.

Proceeding to a higher level of the system design, an important extension of our models may be the inclusion of several depots in the system. We believe that the operator may benefit from using several stations, located where space is not costly, as secondary depots. These stations may be used to

reduce the total distance traveled by the reposting vehicles and serve as a local buffer to meet fluctuating demand during peak times. In fact, the inclusion of such an extension in the SI and TI models only requires a change of the parameters (station capacities and penalty functions). The AI model can accommodate multi-depots with minor changes. A similar related extension is allowing each vehicle to set its base node at a different location.

We demonstrate that our various MILP formulations are capable of solving problem instances of a moderate size of up to 60 stations with acceptable optimality gaps. The AI2 formulation can be used to solve realistic instances with two vehicles and 104 stations. However, larger systems such as Vélib, Bixi or Bicing consist of many hundreds of stations. One method can be used to handle such systems in conjunction with a geographical decomposition approach, mentioned in Section 5. Another possible approach is to use heuristic methods such as Tabu search or Genetic Algorithm, that exhibited good performances in other large scale rich routing problems, see, for example, Bräysy and Gendreau (2005).

Finally, we focused in this paper on the static repositioning problem, where bicycles are moved during slack hours when the system is nearly inactive. The dynamic version of the problem requires a different solution approach. In that case, the demand (resp., returns) in each station depletes (resp., increases) its inventory level due to the user's activity at the same time the repositioning is carried out. Since the TI model already uses decision variables which keep track of the inventory level at each station in each time period (s_{it}), the corresponding inventory balance constraints (constraints (27)) may be adjusted to reflect the dynamic situation. This can be achieved by reducing from the right hand side of (27) the forecasted net demand, and adding variables that represents unsatisfied demand for bicycles and lockers or waiting time of users at the stations. The objective function should be modified to represent the cost incurred by this "lost sales" and waiting times of users. While other solution approaches may be more appropriate for the dynamic version of the problem, the solution of a linear integer programming formulation may be useful as a benchmark.

Acknowledgment: The authors wish to thank Mr. Brodie Hylton and Mr. Danny Quarrel from Alta Bicycle Share Ltd. for providing data on the Capital Bikeshare and Mr. Avraham Edison for help in processing the demand data.

References

Anily, S. and Hassin, R. (1992) The swapping problem. *Networks*, **22**, 419-433.

Anily, S., Gendreau, M. and Laporte, G. (1999) The swapping problem on a line. *SIAM J. on Computing*, **29(1)**, 327-335.

Benchimol, M., Benchimol, P., Chappert, B., Taille, A.D.L, Laroche, F., Meunier, F. and Robinet, L. (2010) Balancing the stations of a self service “bike hire” system. *RAIRO Operations Research*, forthcoming.

Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I. and Laporte, G. (2007) Static pickup and delivery problems: a classification scheme and survey. *TOP*, **15**, 1-31.

Bertazzi, L., Savelsbergh, M. and Speranza, M.G. (2008) Inventory Routing. In: "The Vehicle Routing Problem: Latest Advances and New Challenges" Edited by Golden, B., Raghavan, S. and Wasil, E. Springer.

Bordenave, C., Gendreau, M. and Laporte, G. (2010) Heuristics for the mixed swapping problem. *Computers & Operations Research*, **38**, 108-114.

Bräysy, O. and Gendreau, M. (2005) Vehicle routing problem with time windows, Part II: metaheuristics. *Transportation Science*, **39**, 119-139.

Callé, E. (Director of Operation in Vélib), April 2009, Personal communication.

Capoor, K. and Ambrosi, P. (2009) State and trends of the carbon market 2009, The World Bank, Washington, DC.

Chemla, D., Meunier, F. and Wolfler Calvo, R. (2010) Bike hiring system: solving the rebalancing problem in the static case. Working paper.

Forma, I., Raviv, T. and Tzur, M. (2010) The Static Repositioning Problem in a Bike-Sharing System. In the proceeding of the *7th Triennial Symposium on Transportation Analysis (TRISTAN)*, Tromsø, Norway, 279-282.

Hernández-Pérez, H. and Salazar-González, J.-J. (2004a) A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discrete Applied Mathematics*, **145**, 126-139.

Hernández-Pérez, H. and Salazar-González, J.-J. (2004b) Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. *Transportation Science*, **38**, 245–255.

Lin, J-R. and Ta-Hui, Y. (2011) Strategic design of public bicycle sharing systems with service level constraints. *Transportation Research Part E*, **47**, 284–294.

Louveaux, F. and Salazar-González, J.-J. (2009) On the one-commodity pickup-and-delivery traveling salesman problem with stochastic demands. *Mathematical Programming Ser. A*, 119, 169–194.

Miller, C.E., Tucker, A.W. and Zemlin, R.A. (1960) Integer programming formulations and traveling salesman problems. *J. ACM*, **7**, 326–329.

MetroBike LLC. (2011) http://bike-sharing.blogspot.com/2010_12_01_archive.html, observed in July 21, 2011.

Mukai N. and Watanabe, T. 2005. Dynamic location management for on-demand car sharing system, Knowledge-Based Intelligent Information and Engineering Systems. *9th International Conference, KES* 2005, Melbourne, Australia, 768-774.

Nair, R. and Miller-Hooks, E. (2011). Fleet management for vehicle sharing operations. *Transportation Science*. Forthcoming.

Nair, R., Miller-Hooks, E., Hampshire, R.C. and Busic, A. (2011) Large-scale bicycle sharing systems: analysis of Velib. Working paper.

Raviv, T. and Kolka, O. (2011). Dynamic management of stock levels in a bike sharing system. Working paper.

Rosing, K.E. and Revelle, C.S. (1997) Heuristic concentration: two stage solution construction. *European Journal of Operational Research*, **97**(1), 75-86.

Shu, J., Chou, M., Liu, Q., Teo, C-P. and Wang, I-L. (2010) Bicycle-sharing system: deployment, utilization and the value of re-distribution. Working paper.

Uesugi, K., Mukai, N. and Watanabe, T. (2007) Optimization of vehicle assignment for car sharing system. In: Lecture notes in artificial intelligence, *Intelligent knowledge: knowledge-based intelligent information and engineering systems*. 1105-1111.

Vogel, P. and Mattfeld, D.C. (2010) Modeling of repositioning activities in bike-sharing systems. *Proceeding of the 12th World Conference on Transport Research*, July 11-15, 2010, Lisbon, Portugal.

Appendix A – User Dissatisfaction Function

For completeness of the presentation we summarize here some of the results of Raviv and Kolka (2011). In particular, we formally define the user dissatisfaction function and show how to approximate it.

We consider a single bike-sharing station over a finite horizon $[0,T]$ with the following setting: the inventory level (number of bicycles) in the station at time 0 is given. At time $t \in [0,T]$, users who wish to rent (resp., return) a bicycle arrive at the station according to some non-homogenous Poisson process with rate μ_t [resp., λ_t]. If the requested service can be provided right away, the bicycle is rented or returned and the inventory level is updated accordingly. If the service cannot be provided (i.e., due to an empty station for a renter or a full one for a returner) then the user abandon the station. In reality, the abandoning user may decide to seek the service in other stations of the system but this is out of the scope of this model. Thus, there are two undesirable types of events that may occur in a station. We assume that the system is penalized for each one of them.

p Penalty charged for each renter that abandon due to shortage of bicycles.

h Penalty charged for each returner that abandon due to shortage of vacant lockers

The state of the station is a bounded birth and death process with birth rate μ_t and death rate λ_t. The process is depicted as a Markov Chain in Figure 2.

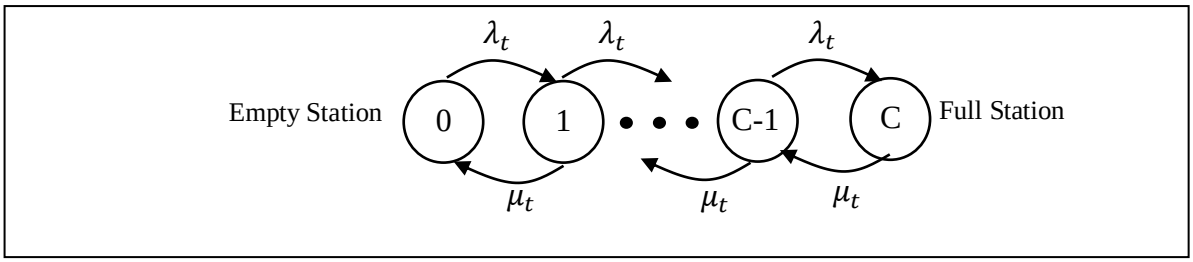

Figure 2: Continues time Markov chain that represents the dynamics of the bicycles inventory level

Let $\pi_{ij}(t)$ denote the probability of the station being at state j at time t provided that its initial state at time 0 was i. It is possible to express the user dissatisfaction function, F(x), where x denotes the initial inventory level (of bicycles) as follow:

$$F(x) \equiv \int_0^T (\pi_{x0}(t)p + \pi_{xC}(t)h)\, dt.$$

The first summation term in the integral represents the expected user dissatisfaction accumulated due to abandonments of renters and the second due to abandonments of users. In the numerical section of this paper we used $p = h = 1$. Thus the value of the user dissatisfaction function represents the expected number of unserved users during the period $[0, T]$.

Next, we note that while the arrival rates of renters and returners are non-homogenous over time, it is reasonable to assume that these rates vary in a finite number of steps over the planning horizon, say every 15 or 30 minutes. Now, our strategy to estimate F(x) is as follows: the planning horizon is discretized into short periods, each of length δ, say $\delta = 1$ minute. These periods are indexed by θ. For each period θ we evaluate the transition probability matrix from the beginning of the period to its end. We denote this transition probability by P_θ. Now the transition probability matrix from time 0 to time $t = \tau \cdot \delta$ is given by

$$\boldsymbol{\pi}(t) = \prod_{\theta=1}^{\tau} P_\theta.$$

P_θ can be calculated numerically, for more details see Ross (2010) Section 6.8. Based on the value of $\pi(t)$ we can obtain a reliable approximation of the user dissatisfaction function using the following discretization procedure:

$$F(x) \approx \delta \sum_{\theta=0}^{\frac{T}{\delta}-1} (\pi_{x0}(t + 0.5\delta)p\mu_{\delta\theta} + \pi_{xC}(t + 0.5\delta)h\lambda_{\delta\theta}). \tag{72}$$

Raviv and Kolka (2011) also established bounds for the estimation error and showed that for a discretization level of one minute the user dissatisfaction function can be estimated very accurately in a fraction of a second for each of the stations in Washington's BSS. They also proved that the user dissatisfaction function is convex.

The objective function in this paper is the sum of dissatisfaction functions, one for each station, The function for each station i is evaluated through (72) for each possible inventory level $0 \leq x \leq C$.

Appendix B – Swapping Based Formulation

In this appendix we provide details on the SB formulation and how its unique characteristics are formulated compared to the AI formulation.

Recall the additional parameter introduced in Section 3.5:

s_i^* the quantity of bicycles which minimizes the cost function $f_i(.)$ at station i;

We further define:

$N'(i)$ set of duplicated stations, representing original station i, $d_i \equiv |N'(i)| = \left|s_i^0 - s_i^*\right|$, $\forall i \in N$; The duplicated nodes are denoted by $N'(i) = \{1(i), 2(i), \dots, d_i(i)\}$; We use i' and j' as indices for stations in $N'(i)$ and $N'(j)$, respectively, $N'(0) = \{0\}$;

$p^+_{i'(i)} = f_i(s_i^* + i') - f_i(s_i^* + i' - 1)$ penalty (cost) for not visiting the $(i')^{th}$ duplicate station of station i, when station i is a supply node, i.e., $s_i^0 > s_i^*$, $i' \in N'(i),\ i \in N$;

$p^-_{i'(i)} = f_i(s_i^* - i') - f_i(s_i^* - i' + 1)$ penalty (cost) for not visiting the $(i')^{th}$ duplicate station of station i, when station i is a demand node, i.e., $s_i^0 < s_i^*$, $i' \in N'(i),\ i \in N$;

The decision variables are defined as follows:

$x_{i'(i)j'(j)v}$ a binary variable which equals one if vehicle v travels directly from station $i'(i)$ to station $j'(j)$, and zero otherwise;

$y_{i'(i)j'(j)v}$ quantity of bicycles carried on vehicle v when it travels directly from station $i'(i)$ to station $j'(j)$, and zero if the vehicle does not travel directly between these nodes;

$q_{i'(i)v}$ auxiliary variables, used for sub-tour elimination constraints;

The swapping based (SB) formulation is similar to the arc indexed formulation, where each station represents a copy of the original one, and therefore denoted by $i'(i)$ instead of i. Similarly, the variables $x_{i'(i)j'(j)v}$ and $y_{i'(i)j'(j)v}$ are used in the SB formulation in a similar way to x_{ijv} and y_{ijv} in the AI formulation, and $q_{i'(i)v}$ in a similar way to q_{iv}. Additional differences are associated with constraints related to station capacities and to the vehicle conservation constraints, as explained next. Note first that by the network representation described above, the capacity of all duplicated stations is in fact one. This is not expressed explicitly in the formulation, however it is implied by removing the variables y^L_{iv}, y^U_{iv} and s_i, which all include quantity of bicycles. In the swapping based formulation, all these quantities are either one or zero. In particular, the vehicle conservation constraints (3) are replaced by the following two sets of constraints, which distinguish between supply and demand nodes:

$$\sum_{i\in N}\sum_{i'(i)\in N'(i)}(y_{i'(i)j'(j)v} + x_{i'(i)j'(j)v}) = \sum_{k\in N}\sum_{k'(k)\in N'(k)} y_{j'(j)k'(k)v} \quad (73)$$

$$\forall\, j \in N, \forall\, j' \in N'(j):\ s_j^0 > s_j^*, \forall v \in V$$

$$\sum_{i\in N}\sum_{i'(i)\in N'(i)}(y_{i'(i)j'(j)v} - x_{i'(i)j'(j)v}) = \sum_{k\in N}\sum_{k'(k)\in N'(k)} y_{j'(j)k'(k)v} \qquad (74)$$

$$\forall\, j \in N, \forall\, j' \in N'(j):\ s_j^0 < s_j^*, \forall v \in V$$

The loading/unloading times can be easily incorporated within the traveling matrix, since every visit to a node is associated with a known operation of either loading or unloading one unit, depending whether it is a supply or a demand node, respectively. Therefore, given an instance in which the supply and demand node sets are known, all travel times of arcs entering a supply (resp., demand) node are modified by adding to them L (resp., U) time units.

Finally, the objective function of the SB formulation is to minimize the total penalty incurred for not visiting supply and demand stations. The penalty for not visiting a certain station is calculated from the cost function as described above. Note that since the original cost function is convex, the "correct" visit order will be obtained automatically for all duplicated stations of the same original station, that is, the $(i')^{th}$ duplicated station (representing, e.g., the $(s^* + i')^{th}$ unit) will be visited before the $(i'-1)^{st}$ station is visited. To represent the objective function, we define the following two sets of original nodes:

$$N^+ = \{i: s_j^0 > s_j^*\} \text{ and } N^- = \{i: s_j^0 > s_j^*\}$$

Then, the objective function is:

$$\sum_{i\in N} f_i(s_i^*) + \sum_{i\in N^+}\sum_{i'(i)\in N'(i)} p^+_{i'(i)}\left(1 - \sum_{v\in V}\sum_{j\in N}\sum_{j'(j)\in N'(j)} x_{i'(i)j'(j)v}\right) +$$
$$\sum_{i\in N^-}\sum_{i'(i)\in N'(i)} p^-_{i'(i)}\sum_{j\in N}\left(1 - \sum_{v\in V}\sum_{j'(j)\in N'(j)} x_{i'(i)j'(j)v}\right) \qquad (75)$$

In conclusion, the SB formulation consists of extensions (to duplicated stations) of constraints (4)-(6) and (10)-(12), where in (12) the first term on the left hand side of the constraint is omitted and the travel times are modified as explained above, plus the modified constraints (73)-(74), with the objective function (75).