Transportation Research Part B 103 (2017) 188–207

Contents lists available at ScienceDirect

Transportation Research Part B

journal homepage: www.elsevier.com/locate/trb

A time-space network flow approach to dynamic repositioning in bicycle sharing systems

Dong Zhang[a], Chuhang Yu[b,*], Jitamitra Desai[c], H.Y.K. Lau[d], Sandeep Srivathsan[b]

[a] Department of Industrial and Systems Engineering, National University of Singapore, Singapore
[b] Air Traffic Management Research Institute, Nanyang Technological University, Singapore
[c] Manufacturing and Industrial Engineering Cluster, School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore
[d] Industrial and Manufacturing Systems Engineering, University of Hong Kong, Hong Kong, China

ARTICLE INFO

Article history:
Received 10 May 2016
Revised 9 December 2016
Accepted 14 December 2016
Available online 21 December 2016

Keywords:
Bicycle sharing systems
Time-space network flow model
Dynamic repositioning
Demand forecasting
Convexification and linearization
Heuristic algorithm

ABSTRACT

Faced with increasing population density, rising traffic congestion, and the resulting upsurge in carbon emissions, several urban metropolitan areas have instituted public bicycle sharing system as a viable alternative mode of transportation to complement existing long-distance bus- and metro- transit systems. A pressing issue that needs to be addressed in bike sharing systems is the accrued imbalance of bicycles between commuter demands and inventory levels at stations. To overcome this issue, a commonly employed strategy is to reposition bicycles during off-peak periods (typically at night) when no new user arrivals are expected. However, when such an imbalance occurs during day-time peak hours, such a passive strategy would result in lower resource utilization rates. To overcome this drawback, in this study, we propose a dynamic bicycle repositioning methodology that considers inventory level forecasting, user arrivals forecasting, bicycle repositioning, and vehicle routing in a unified manner. A multi-commodity time-space network flow model is presented, which results in an underlying complex nonlinear optimization problem. This problem is then reformulated into an equivalent mixed-integer problem using a model transformation approach and a novel heuristic algorithm is proposed to efficiently solve this model. Specifically, the first stage involves solving the linear relaxation of the MIP model, and a set covering problem is subsequently solved in the second stage to assign routes to the repositioning vehicles. The proposed methodology is evaluated using standard test-bed instances from the literature, and our numerical results reveal that the heuristic algorithm can achieve a significant reduction in rejected user requests when compared to existing methods, while yet expending only minimal computational effort.

1. Introduction

In recent years, many metropolitan areas have invested in sustainable public transportation systems to stem the rising tide of traffic congestion and the associated increase in harmful carbon emissions. One such green initiative is *bicycle sharing systems*, which have risen to prominence over the last decade. According to a report commissioned by the Land Transport

* Corresponding author.
E-mail addresses: zd.pony@gmail.com (D. Zhang), ych1102@gmail.com, yuch@ntu.edu.sg (C. Yu), jdesai@ntu.edu.sg (J. Desai), hyklau@hku.hk (H.Y.K. Lau), ssandeep@ntu.edu.sg (S. Srivathsan).

http://dx.doi.org/10.1016/j.trb.2016.12.006
0191-2615/

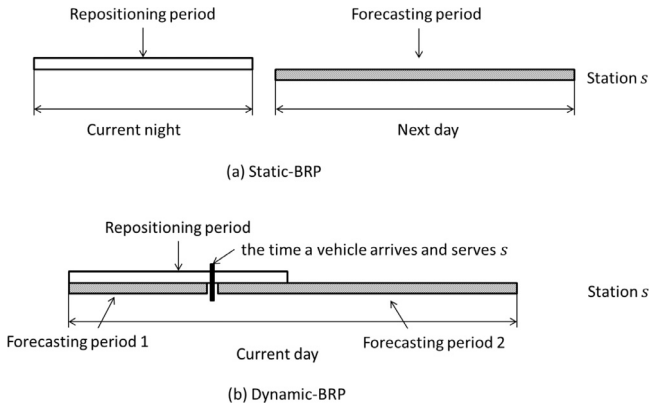

Fig. 1. The repositioning period and forecasting period for the dynamic-BRP.[1]

Authority (LTA) of Singapore, nearly 78 major cities world-wide have deployed bike sharing systems, with Europe leading the way, closely followed by Canada and Asia (Midgley, 2009). A typical bicycle sharing operation comprises the following three resource components: (i) a set of *bicycles*, which can be rented by commuters for a (fixed) price; (ii) a set of *stations* or *depots*, from where a bicycle can be picked-up or dropped-off; (iii) a fleet of *repositioning vehicles* (usually trucks) that are used to transport bicycles from one location to another. From a user perspective, such bicycle sharing options have predominantly been used as an alternative (mostly) one-way transportation mode for short journeys, and serve to complement existing long-distance public transportation modes, such as bus- and metro-transit systems.

In order to guarantee high levels of service quality and increased resource utilization rates, it is necessary to monitor and replenish the inventory level at each station, by periodically repositioning bicycles from one station to another. Such bicycle repositioning activities are usually performed at night, when no pick-ups/drop-offs are anticipated (Raviv et al., 2013; Dell'Amico et al., 2014; Forma et al., 2015), thereby resulting in a *static bicycle repositioning problem* (*static-BRP*). In the literature, there are a bundle of methodologies proposed for the static-BRP (Benchimol et al., 2011; Schuijbroek et al., 2013; Chemla et al., 2013; Raviv et al., 2013; Papazek et al., 2013; Schuijbroek et al., 2013; Di Gaspero et al., 2013; Dell'Amico et al., 2014; Ho and Szeto, 2014; Erdogan et al., 2015; Di Gaspero et al., 2016; Dell'Amico et al., 2016; Li et al., 2016; Szeto et al., 2016; Ho and Szeto, 2017). However, given the growing user demand, scheduling repositioning activities exclusively at night cannot sufficiently adjust the inventory level of stations during the day. This leads to an acute imbalance between pick-ups and drop-offs, and at stations where this phenomenon is most prevalent, a significant proportion of pick-up/drop-off requests will likely be rejected, thereby resulting in severe underutilization rates. Furthermore, it also discourages passengers from adopting bike sharing as an alternative mode of transportation, leading to adverse social and environmental impacts. Therefore, there is a pressing need to mitigate this inevitable accrued imbalance between inventory levels and commuter demand expectations at bicycle stations during regular operational hours.

Repositioning bicycles periodically during day-time operational hours is an option to counteract this phenomenon, as the inventory levels at bicycle stations can be rebalanced by using the forecasts of inventory levels (Raviv and Kolka, 2013; Shu et al., 2013; Schuijbroek et al., 2013) and rental/return requests (Barth and Todd, 1999; Cheu et al., 2006; Kek et al., 2009; Froehlich et al., 2009; Kaltenbrunner et al., 2010; Caggiani and Ottomanelli, 2012; de Chardon and Caruso, 2015; Borgnat et al., 2011; Caggiani and Ottomanelli, 2013), thereby resulting in a *dynamic bicycle repositioning problem* (*dynamic-BRP*). The main objective of the dynamic-BRP is to determine the optimal inventory levels that need to be maintained at each bicycle station such that *user dissatisfaction* is minimized. The user dissatisfaction is defined as the expected number of commuter requests (both rentals and returns) that will be rejected in a future time period due to the lack of available bicycles or empty docks (Raviv and Kolka, 2013). For either the static-BRP or the dynamic-BRP, three key components are user dissatisfaction estimating, bicycle repositioning and vehicle routing. For the static-BRP, the user dissatisfaction is first estimated and then the bicycle repositioning and vehicle routing are solved. However, the situation is much difficult in the dynamic case as the repositioning period and the forecasting period are overlapped (see Fig. 1). It consequently leads to a result that the estimation of the user dissatisfaction should be performed simultaneously when making the bicycle repositioning and vehicle routing decisions. Meanwhile, the forecasting information in the dynamic-BRP is *time dependent* and a repositioning solution needs to be generated relatively quickly; otherwise, the forecasted information might not accurately reflect the actual scenarios at a bicycle station.

The challenges that arise in the dynamic-BRP necessitate dedicated and efficient methodologies. However, considering the complexity of fully integrating three key components (namely, the user dissatisfaction estimating, the bicycle repositioning and the vehicle routing), several partially integrated or sequential solution methodologies of the dynamic-BRP are proposed

[1] The repositioning period is a time-window in which all repositioning operations need to be completed while the forecasting period is a time-window in which the number of rejected requests is forecasted.

in the literature. Caggiani and Ottomanelli (2013) proposed a simulation model and Kloimllner et al. (2014) proposed a dynamic model to minimize the repositioning cost. Regue and Recker (2014) proposed a sequential framework, in which the demand forecasting model and inventory levels model were first solved, then redistribution needs were generated and finally vehicle routing solutions were obtained. Pfrommer et al. (2014) proposed receding horizon-based optimization techniques to combine operators' repositioning decisions with commuter-generated repositionings by means of a reward system. Contardo et al. (2012) and Wang (2014) proposed methodologies based on a time-space network, in which the estimation of the user dissatisfaction is simplified and consequently the estimation accuracy is scarified. Note that Nair and Miller-Hooks (2011) also focused on the fleet repositioning problem, and they incorporated the chance constraints in the model to guarantee a certain service level. For further details, we refer the interested reader to a recent review on the dynamic-BRP by Laporte et al. (2015).

Taking our cue from earlier time-space network flow-based models (see Contardo et al., 2012; Wang, 2014; Zhang et al., 2015; Tong et al., 2015; Mahmoudi and Zhou, 2016; Zhang et al., 2016; Tang et al., 2016; Zhang and Klabjan, 2017), in this paper we propose a *time-space network flow-based formulation* in conjunction with a novel solution methodology to fully integrate three key components of the dynamic-BRP. First, we propose a new approach to estimate the user dissatisfaction by decomposing the forecasting period into two sub-periods based on the vehicle arrival time (see Fig. 1(b)). Within each sub-period, an estimation approach with a high accuracy (e.g., Raviv and Kolka, 2013) is applied to estimate the user dissatisfaction. Then, the bicycle repositioning and the vehicle routing is integrated with the user dissatisfaction estimating, leading to a non-linear time-space network flow model. The objective of the proposed model is to minimize the total user dissatisfaction across all sub-periods plus the vehicle traveling cost. Third, we propose a model transformation methodology by exploring a special characteristic of an affiliation function. It can transform the complex non-linear model into an equivalent MIP model without increasing the model scale and deteriorating the model structure. Finally, a heuristic algorithm is proposed to efficiently solve the model. The experimental results reveal that our approach can achieve significant improvements compared to several benchmark approaches.

The main contributions of this paper to the dynamic-BRP literature can be summarized as follows.

- A novel non-linear time-space network flow model is built to fully integrate the user dissatisfaction estimating, the bicycle repositioning and the vehicle routing.
- An efficient model transformation methodology is proposed to transform the non-linear model into an equivalent MIP model.
- An efficient heuristic algorithm is proposed to solve the MIP model.

The following context is organized as follows. The models and solution methodologies are presented in Section 2 and 3, respectively. In Section 4, extensive computational studies are conducted to study the efficacy of the proposed solution methodology. Finally, we present the conclusions from our study as well as directions for future research in Section 5.

2. Problem formulations

In this section, we begin by briefly describing the forecasting model developed by Raviv and Kolka (2013), which we adopt in our study. Then, we state the assumptions and definitions made in our model, and introduce the key modeling elements. Third, we present a time-space network flow model for the dynamic-BRP and its corresponding nonlinear formulation. Finally, we propose a model transform methodology to transform the complex non-linear model to a much easier formulation.

2.1. Forecasting model

As aforementioned in Section 1, Raviv and Kolka (2013) propose an estimation method to compute the expected user dissatisfaction. This forecasting model assumes that bicycle stations are *independent* of each other and that the pick-up and drop-off patterns (at all stations) follow a non-homogeneous Poisson process, and moreover, no two events (rentals or returns) can occur simultaneously or within a small time duration (e.g., every one second). The (Poisson) user arrival rates are allowed to vary across different time-windows and a *continuous-time Markov chain* (CTMC) model is used to study the dynamics of the inventory level at each station. It is demonstrated that this methodology results in a more accurate forecast as compared to the forecasting models adopted in Contardo et al. (2012) and Wang (2014). The expected user dissatisfaction can be estimated from the probability of occurrence of the following two scenarios: (i) a pick-up request arrives at a station when the inventory level at that station is zero; (ii) there is a drop-off request when a station is operating at its full capacity. Although the inventory estimation at a station is not explicitly proposed in Raviv and Kolka (2013), the probability distribution of the inventory level at each station, obtained in the process of the CTMC model, acts as a surrogate to estimate the expected inventory level.

In our study, we adopt the model proposed by Raviv and Kolka (2013) to estimate both the expected user dissatisfaction and the expected inventory level. As there is a requirement for the inventory level to be integer-valued, we round its expected value to the nearest integer. The two estimation functions are described as follows:

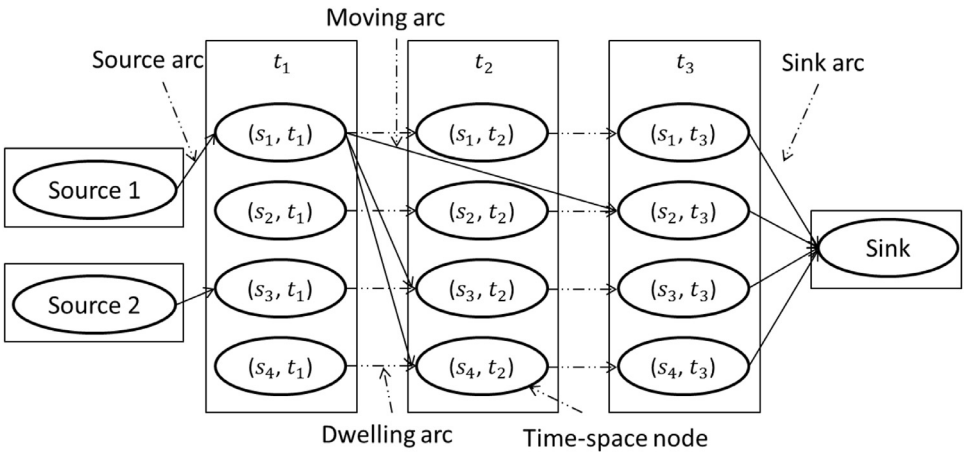

Fig. 2. Time-space network flow model.

- $FD_s[I_{st}, t, t']$ is the function used to estimate the expected total number of rejected (rental and return) user requests at station s during time interval $[t, t']$, given that the inventory level of station s at time t is I_{st} and the inventory level of station s is not adjusted by any repositioning operations between $[t, t']$.
- $FI_s[I_{st}, t, t']$ is the function used to estimate the expected inventory level of station s at time t', given that the inventory level of station s at time t is I_{st} and the inventory level of station s is not adjusted by any repositioning operations between $[t, t']$.

2.2. Assumptions and notations

Before proceeding further, we make the following assumptions in our model:

Assumption 1. Each station is served at most once within a repositioning time-window, and a station is considered to be served only when a repositioning vehicle performs loading/unloading operations.

This assumption is reasonable in practice as the width of the repositioning time-window is typically 2–3 h, during which there is minimal possibility of a station being served more than once. Assumption 1 also implies that a station can be served at most once by one vehicle within a repositioning time-window.

Assumption 2. All repositioning vehicles are identical (i.e., have the same capacity), and can therefore be easily interchanged.

Assumption 3. The minimum traveling time between any two stations is known and fixed during all times of the day.

We are now ready to present the key elements of our proposed time-space network flow model (one example is illustrated in Fig. 2), which are then subsequently used to formally develop the mathematical formulation. The repositioning time-window, denoted as $W^R = [B, E^R]$, begins at time B and ends at time E^R. Similarly, the forecasting time-window is denoted as $W^F = [B, E^F]$. In our problem context, while the forecasting and repositioning time-windows begin at the same time B, the width of the forecasting period is far greater than the width of the repositioning period, i.e., $E^F \gg E^R$ (see Fig. 1(b)). The time-window W^R is discretized into a set of evenly spaced time points $\mathcal{T} = \{t_0, t_1, \ldots, t_{(m-1)}, t_m\}$, where $t_0 \equiv B$ and $t_m \equiv E^R$.

Let $\mathcal{V}$ and $\mathcal{S}$ denote the set of repositioning vehicles and the set of stations, respectively. For each vehicle $v \in \mathcal{V}$, the maximum capacity is K and the initial number of bicycles on vehicle v at time B is I_v^0. For each station $s \in \mathcal{S}$, the maximum (inventory) capacity is C_s and the initial inventory level at time B is I_s^0. The minimum travel time between any two stations s and s' is denoted by $MT(s, s')$, which can be obtained as the sum of the travel time of the repositioning vehicle and the bicycle loading/unloading time. Note that, in our study, we use estimates of the average bicycle loading/unloading time since we do not know a priori the specific number of bicycles loaded/unloaded during repositioning activities.

For each station $s \in \mathcal{S}$ and time $t \in \mathcal{T}$, their combination (s, t) represents a time-space node. Additionally, we define two sets of dummy nodes: (i) set of *source nodes*, denoted by $\mathcal{N}_v$, that are tied to repositioning vehicles, i.e., a source node $(n_v, t_0) \in \mathcal{N}_v$ uniquely corresponds to a vehicle $v \in \mathcal{V}$; and (ii) one *sink node* denoted by (n^{SK}, t_{m+1}). Let $\mathcal{N}$ denote the set of all nodes and $\tilde{\mathcal{N}}$ denotes the set of all nodes excluding source and sink nodes, i.e., $\mathcal{N} = \tilde{\mathcal{N}} \bigcup \mathcal{N}_v \bigcup \{n^{SK}\}$. In addition to time-space nodes, there are four types of arcs in this network: source arcs, sink arcs, moving arcs, and dwelling arcs. Let $\mathcal{A}$ represent the set of all arcs and an arc can be represented as a 4-tuple (s, t, r, t') with (s, t) indicating the tail time-space node and (r, t') indicating the head time-space node. The details of each arc-type are described as follows.

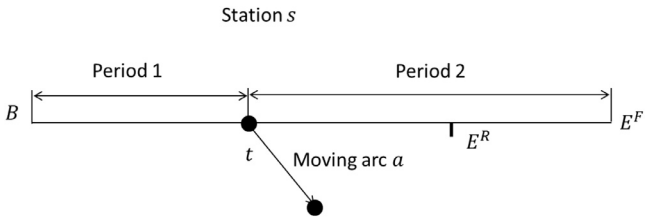

Fig. 3. An example of a repositioning activity scheduled t time t at station s.

- *Source arc* ($\mathcal{A}^{SC}$) *and sink arc* ($\mathcal{A}^{SK}$)*:* Each source arc connects a source node (n_v, t_0) associated with vehicle v to a time-space at the first stage (n_v, t_1), which implies that vehicle v is initially located at station n_v. Similarly, a sink arc connects any time-space node at the last stage (s, t_m) to the sink node.
- *Moving arc* ($\mathcal{A}^{M}$)*:* A moving arc $(s, t, r, t + MT(s, r))$ denotes a moving activity from time-space node (s, t) to $(r, t + MT(s, r))$, where $s \neq r$.
- *Dwelling arc* ($\mathcal{A}^{D}$)*:* A dwelling arc $(s, t, s, t + 1)$ indicates that a vehicle is positioned at station s at time t and continues to dwell at that station during $[t, t + 1]$, without performing any loading/unloading operations.

Based on these key elements of the network flow model, we then further define the repositioning vehicle route and the loading sequence.

Definition 1. A repositioning vehicle route $p \in \mathcal{P}$ is defined as a sequence of time-space nodes $((s_1, t_1), (s_2, t_2), ..., (s_l, t_l))$, where (s_i, t_i) is the ith node in the sequence. It also indicates that station s_i is served by the corresponding vehicle at time t_i.

A feasible route $p \in \mathcal{P}$ must satisfy the following three conditions: (i) it starts from the corresponding vehicle's initial station; (ii) $MT(s_i, s_{i+1}) \leq (t_{i+1} - t_i)$; and (iii) no station is served more than once. Let $\mathcal{P}$ denote the set of all routes.

Definition 2. For each route $p = ((s_1, t_1), (s_2, t_2), \ldots, (s_l, t_l))$, we define a bicycle loading sequence δ as $\delta = [\delta_{s_1 t_1}, \delta_{s_2 t_2}, ..., \delta_{s_l t_l}]$. $\delta_{s_i t_i}$ represents the number of bicycles that need to be loaded/unloaded in station s_i at time t_i. If $\delta_{s_i t_i} \geq 0$, there are $\delta_{s_i t_i}$ bicycles that need to be loaded at station s_i at time t_i; otherwise, there are $-\delta_{s_i t_i}$ bicycles to be unloaded from station s_i. A feasible bicycle loading sequence for route p must ensure that the station's inventory is between zero and its maximum capacity limitation, as shown in the constraint below.

$$0 \leq FI_{s_i}(I^0_{s_i}, B, t_i) + \delta_{s_i t_i} \leq C_{s_i} \tag{1}$$

2.3. Mathematical model

In this section, a novel mathematical model that integrates the expected user dissatisfaction estimation with the subsequent vehicle routing and bicycle repositioning decisions is presented. To facilitate our description, the notations and decision variables used in this paper are summarized in Table 1.

Note that based on our Assumption 1 that each station can be served at most once, for any station s, it indicates that there can be at most one active moving arc flowing out of all nodes corresponding to station s, while there can be multiple active dwelling arcs corresponding to that station.

Our model is a *multi-commodity time-space network flow formlation*, in which vehicles and bicycles represent the two commodities. The model aims to explore feasible flows for vehicles and bicycles such that the vehicle traveling cost and the total expected user dissatisfaction within the forecasting time-window are minimized.

Our objective function can be decomposed into four components, which are described as follows:

- Vehicle travel cost: The first term in the objective function is the total vehicle travel cost (including the fuel cost, driver cost, etc.), and only moving arcs are associated with a positive travel cost. The vehicle travel cost can be expressed as:

$$\sum_{(s,t,r,t') \in \mathcal{A}} c_{strt'} \cdot x_{strt'} \tag{2}$$

- Expected user dissatisfaction before service completion at each station: As illustrated in Fig. 3, a moving arc a flows out of time-space node (s, t) to another time-space node (r, t'), $s \neq r$, which implies that station s is served at time t and there can be no other moving arcs flowing out of s during the time interval $[B, t]$. In other words, within forecasting period 1, there is no vehicle serving station s, and hence, the expected user dissatisfaction of station s within period 1 (between B and t) can be estimated as $FD_s[I^0_s, B, t]$ (see Section 2.1 for details). Therefore, the total expected user dissatisfaction prior to service completion across all stations can then be expressed as:

$$\sum_{s \in S} \sum_{(s,t,r,t') \in \mathcal{A}^M_s} FD_s[I^0_s, B, t] \cdot x_{strt'} = \sum_{(s,t,r,t') \in \mathcal{A}^M} FD_s[I^0_s, B, t] \cdot x_{strt'} \tag{3}$$

Table 1
Model parameters and decision variables.

Model parameters	
B:	beginning time of the repositioning/forecasting time-window
E^R, E^F:	end time of the repositioning and forecasting time-window, respectively
$\mathcal{S}$:	set of all stations, indexed as s, r
$\mathcal{V}$:	set of all vehicles, indexed as v
$\mathcal{T}$:	set of all discrete time points, indexed as t
$\mathcal{N}$:	set of all time space nodes, indexed as a pair of station and time (s, t)
$\tilde{\mathcal{N}}$:	set of all time-space nodes, excluding source and sink nodes
$\mathcal{N}_v$:	set of source nodes
n^{SK}:	sink node
n_v:	source node corresponding to vehicle v
$\mathcal{A}$:	set of all arcs
$\mathcal{A}_{st}^{in}$, $\mathcal{A}_{st}^{out}$:	sets of all arcs flowing into and out of time-space node (s, t), respectively
$\mathcal{A}^M$:	set of all moving arcs
$\mathcal{A}_s^M$:	set of all moving arcs flowing out of station s, i.e., $\mathcal{A}_s^M = (s, t, r, t') \in \mathcal{A}^M$
$\mathcal{A}_{st}^M$:	set of all moving arcs flowing out of a time-space node (s, t), i.e., $\mathcal{A}_{st}^M = (s, t, r, t') \in \mathcal{A}^M$
$c_{strt'}$:	travel cost of an arc $(s, t, r, t') \in \mathcal{A}$. For a dwelling arc, $c_{strt'} = 0$
I_{st}:	inventory level of station s at time t
C_s:	capacity of station s
I_v^0:	number of bicycles initially loaded on the vehicle
K:	capacity of vehicle
$FD_s[I_{st}, t, t']$:	expected user dissatisfaction between $[t, t']$ for station s
$FI_s[I_{st}, t, t']$:	expected inventory level of station s at a future time t'
$MT(s, r)$:	sum of the travel time from station s to r
$\mathcal{P}$:	set of all vehicle routes, indexed by p
$\mathcal{P}_v$:	set of candidate routes belonged to vehicle v
$\mathcal{P}_s$:	set of candidate routes in which station s is served
FD_p^{MIN}:	minimum value of expected user dissatisfaction of route p
$FD_s(\delta)$:	expected user dissatisfaction of station s under bicycle loading sequence δ
$FD_p(\delta)$:	expected user dissatisfaction of route p under bicycle loading sequence δ
Decision variables	
$x_{strt'} \in \{0, 1\}$:	flow of vehicles on arc (s, t, r, t'). It takes 1 if arc (s, t, r, t') is used and 0 otherwise
$y_{strt'} \in [0, K]$:	flow of bicycles on arc (s, t, r, t')
$\delta_{st} \in [-\Theta, \Theta]$:	it indicates the inventory adjustment for station s at time t. If $\delta_{st} > 0$, δ_{st} bicycles are loaded to station s at time t; otherwise, bicycles are unloaded from station s. Note that $\Theta = min\{K, C_s\}$
$z_s \in \{0, 1\}$:	it indicates whether station s is served during the repositioning time-window. It takes 1 if station s is served, i.e., there is an active moving arc flowing out of any time-space nodes corresponding to station s; 0 otherwise
$\theta_p \in \{0, 1\}$:	if $\theta_p = 1$, route p is selected; otherwise $\theta_p = 0$

- Expected user dissatisfaction after service completion for each stations: As in the previos case, consider station s given in Fig. 3. The expected inventory level of s just before time t can be first estimated as $FI_s[I_s^0, B, t]$ as no vehicle has served s before t. At time t, the inventory level is adjusted by δ_{st}. Then I_{st}, defined as the inventory level of station s at time t, can be computed as $I_{st} = FI_s[I_s^0, B, t] + \delta_{st}$. (Note that δ_{st} is a decision variable in our model.) Since there will be no other repositioning vehicle serving s after t, the expected user dissatisfaction at station s in period 2 (between t and E^F) can then be estimated as $FD_s[I_{st}, t, E^F]$, and the total expected user dissatisfaction after service completion across all stations can then be expressed as:

$$\sum_{s\in\mathcal{S}} \sum_{(s,t,r,t')\in\mathcal{A}_s^M} FD_s[I_{st}, t, E^F] \cdot x_{strt'} = \sum_{(s,t,r,t')\in\mathcal{A}^M} FD_s[I_{st}, t, E^F] \cdot x_{strt'} \tag{4}$$

- Expected user dissatisfaction of stations not served by any vehicle during the repositioning time-window: If the inventory level of station s is not adjusted within the repositioning time window $[B, E^R]$, then the expected user dissatisfaction for s across the entire forecasting period is simply $FD_s[I_s^0, B, E^F]$, and summing across all stations yields:

$$\sum_{s\in\mathcal{S}} FD_s[I_s^0, B, E^F] \cdot (1 - z_s) \tag{5}$$

Expressions (3)–(5) represent the total expected user dissatisfaction between B and E^F of all stations. We are now ready to formally present the mathematical model as follows. *Model 1*

$$\text{Minimize} \sum_{(s,t,r,t')\in\mathcal{A}} c_{strt'} \cdot x_{strt'} + \sum_{(s,t,r,t')\in\mathcal{A}^M} FD_s[I_s^0, B, t] \cdot x_{strt'}$$

$$+ \sum_{(s,t,r,t')\in\mathcal{A}^M} FD_s[I_{(s,t)}, t, E^F] \cdot x_{strt'} + \sum_{s\in\mathcal{S}} FD_s[I_s^0, B, E^F] \cdot (1 - z_s) \tag{6a}$$

$$\text{s.t.} \sum_{(s,t,r,t') \in \mathcal{A}_{st}^{in}} x_{strt'} = \sum_{(s,t,r,t') \in \mathcal{A}_{st}^{out}} x_{strt'}, \quad (s,t) \in \check{N} \tag{6b}$$

$$\sum_{(s,t,r,t') \in \mathcal{A}_{st}^{in}} y_{strt'} = \sum_{(s,t,r,t') \in \mathcal{A}_{st}^{out}} y_{strt'} + \delta_{st}, \quad (s,t) \in \check{\mathcal{N}} \tag{6c}$$

$$\sum_{(s,t,r,t') \in \mathcal{A}_{s}^{M}} x_{strt'} - z_s = 0, \quad s \in \mathcal{S} \tag{6d}$$

$$0 \le \delta_{st} + FI_s[I_s^0, B^F, t] \le C_s, \quad (s,t) \in \mathcal{N} \tag{6e}$$

$$\delta_{st} \le min\{K, C_s\} \cdot \sum_{(s,t,r,t') \in \mathcal{A}_{st}^{M}} x_{strt'}, \quad (s,t) \in \mathcal{N} \tag{6f}$$

$$\delta_{st} \ge -min\{K, C_s\} \cdot \sum_{(s,t,r,t') \in \mathcal{A}_{st}^{M}} x_{strt'}, \quad (s,t) \in \mathcal{N} \tag{6g}$$

$$\sum_{(n_v,t_0,r,t') \in \mathcal{A}_{n_v t_0}^{out}} x_{n_v t_0 rt'} = 1, \quad v \in \mathcal{V} \tag{6h}$$

$$\sum_{(n_v,t_0,r,t') \in \mathcal{A}_{n_v t_0}^{out}} y_{n_v t_0 rt'} = I_v^0, \quad v \in \mathcal{V} \tag{6i}$$

$$y_{strt'} \le K \cdot x_{strt'}, \quad (s,t,r,t') \in \mathcal{A} \tag{6j}$$

The constraints in our model are similar to those in traditional multi-commodity time-space network flow models. The objective function (6a) minimizes the total vehicle travel costs plus the expected user dissatisfaction in the system; constraints (6b) and (6c) ensure the vehicle and bicycle flow balance restrictions, respectively, at each time-space node $(s,t) \in \check{\mathcal{N}}$; constraint (6d) ensures that for station s at most one moving arc in $\mathcal{A}_s^M$ can be selected. Since an active moving arc indicates that station s is served, constraint (6d) guarantees that each station is served at most once during the repositioning time-window; constraint (6e) ensures that the number of bicycles at a station at any time should always be non-negative and less than the capacity limit; constraints (6h) and (6i) ensure the flow balance at the source nodes. Note that the flow balance constraints at all other nodes implicitly ensures the flow balance at the sink node; constraint (6j) ensures that there is a positive bicycle flow value on an arc if and only if there is a positive vehicle flow value on that arc; constraints (6f) and (6g) ensure that bicycle loading/unloading operations can be conducted at time-space node (s, t), only if a moving arc $(s,t,r,t') \in \mathcal{A}_{st}^M$ is selected.

Remark 1. The expected user dissatisfaction within the forecasting time-window can be estimated as the sum of expressions (3)–(5). Expressions (3) and (4) are both linear functions, as their parameters $FD_s[I_s^0, B, t]$ are constants. However, expression (5) is a product of two decision variables as $FD_s[I_{st}, t, E^F]$ is also dependent on decision variable δ_{st}, which renders Model 1 as a difficult nonlinear program.

Remark 2. The assumption that a station can be served at most once during the repositioning time-window guarantees that the expected user dissatisfaction is exactly estimated in our method. If a station is served at time t, it implies that no repositioning operations are performed before and after time t. Therefore, we can estimate the user dissatisfaction before and after time t based on the forecasting method proposed in Raviv and Kolka (2013). Moreover, Assumption 1 can be easily relaxed by implementing our method in a rolling horizon manner (see discussion in Section 3.3).

Remark 3. The assumption that each vehicle has the same capacity can also be relaxed by building a multi-commodity network flow model, in which each vehicle is regarded as a single commodity.

Remark 4. Although our model adopts the forecasting approach proposed by Raviv and Kolka (2013), any other forecasting approaches, which can estimate the expected inventory level and the expected user dissatisfaction value, are appropriate candidates for our model.

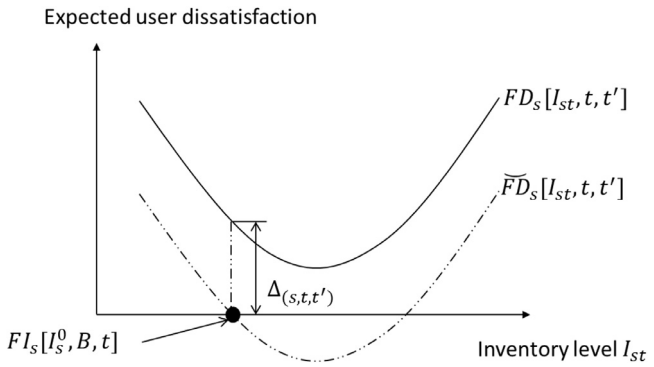

Fig. 4. Illustration of an affiliate function.

2.4. Model transformation

In this section, we first transform the complex nonlinear model (Model 1) to an equivalent convex model, which leads to a much easier formulation (a major contribution of our study), and then transform the convex model to a mixed integer programming (MIP) model using the approach proposed by Raviv and Kolka (2013).

Our proposed approach to transform the nonlinear model to an MIP model consists of the following steps:

Step 1: Define an affiliate function

Let $\Delta_{(s,t,t')}$ in Fig. 4 be the expected user dissatisfaction of station s during time interval $[t, t']$ given that there is no vehicle serving station s during time interval $[B, t']$. Since the initial inventory level of station s at time t is estimated as $FI_s[I_s^0, B, t]$, $\Delta_{(s,t,t')}$ is then estimated as:

$$\Delta_{(s,t,t')} = FD_s[FI_s[I_s^0, B, t], t, t'] \tag{7}$$

If s, t and t' are all fixed, $\Delta_{(s,t,t')}$ is a constant. Meanwhile, if t and t' are both fixed, then the estimated user dissatisfaction of station s during time interval $[t, t']$ (see definition of $FD_s[I_{st}, t, t']$ in Section 2.1) is a function of the inventory level I_{st}, as illustrated in Fig. 4.

Then, we define an affiliate function $\breve{FD}_s[I_{st}, t, t']$ based on $FD_s[I_{st}, t, t']$ and $\Delta_{(s,t,t')}$, which is represented as:

$$\breve{FD}_s[I_{st}, t, t'] = FD_s[I_{st}, t, t'] - \Delta_{(s,t,t')} \tag{8}$$

This affiliate function can be intuitively described in Fig. 4. Note that an important property of the affiliate function is that if $I_{st} = FI_s[I_s^0, B, t]$, then $\breve{FD}_s[I_{st}, t, t']=0$.

Step 2: Transform the nonlinear model into an equivalent convex model

Since the third term in the objective function is a multiplication of two unknowns (it is even difficult to know the explicit expression of $FD_s[I_{(s,\ t)}, t, E^F]$), Model 1 is very complex to solve. As a consequence, we transform the objective function into an equivalent convex form by utilizing the property of the affiliate function, which leads to a much easier formulation.

In the beginning, the objective function (6a) is transformed to an equivalent expression (9) based on the affiliate function (derived in Step 1).

$$\begin{aligned}\text{Minimize} \quad & \sum_{(s,t,r,t')\in\mathcal{A}} c_{strt'} \cdot x_{strt'} + \sum_{(s,t,r,t')\in\mathcal{A}^M} FD_s[I_s^0, B, t] \cdot x_{strt'} \\ & + \sum_{(s,t,r,t')\in\mathcal{A}^M} \breve{FD}_s[I_{st}, t, E^F] \cdot x_{strt'} + \sum_{s\in S} FD_s[I_s^0, B, E^F] \cdot (1 - z_s) + \sum_{(s,t,r,t')\in\mathcal{A}^M} \Delta_{(s,t,E^F)} \cdot x_{strt'}\end{aligned} \tag{9}$$

Theorem 1. *Objective functions (6a) and (9) are equivalent.*

Proof. The proof can be found in Appendix A.1. □

Note that the term given by (9), is still the multiplication of two unknowns (i.e. $\breve{FD}_s[I_{st}, t, E^F]$ and $x_{strt'}$). In Theorem 2, we further prove that (9) can be equivalently transformed to a convex form (10) based on the property of the affiliate function.

$$\begin{aligned}\text{Minimize} \quad & \sum_{(s,t,r,t')\in\mathcal{A}} c_{strt'} \cdot x_{strt'} + \sum_{(s,t,r,t')\in\mathcal{A}^M} FD_s[I_s^0, B, t] \cdot x_{strt'} \\ & + \sum_{(s,t)\in\breve{N}} \breve{FD}_s[I_{st}, t, E^F] + \sum_{s\in S} FD_s[I_s^0, B, E^F] \cdot (1 - z_s) + \sum_{(s,t,r,t')\in\mathcal{A}^M} \Delta_{(s,t,E^F)} \cdot x_{strt'}\end{aligned} \tag{10}$$

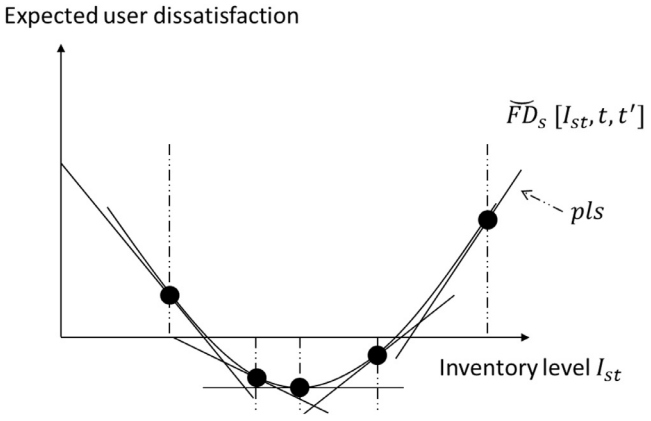

Fig. 5. Supporting functions of $\tilde{FD}_s[I_{st}, t, t']$ (Raviv et al., 2013).

Theorem 2. *Objective functions (9) and (10) are equivalent.*

Proof. The proof can be found in Appendix A.2. □

Lemma 1. *Objective function (10) is a convex function.*

Proof. Raviv and Kolka (2013) proved that $FD_s[I_{st}, t, t']$ is a convex function of I_{st}. $\tilde{FD}_s[I_{st}, t, t']$ is an affine function of $FD_s[I_{st}, t, t']$, therefore $\tilde{FD}_s[I_{st}, t, t']$ is also a convex function of I_{st}. Considering that all other components in (10) are linear elements, objective function (10) is a convex function. □

Step 3: Transfer the convex model into a mixed integer programming model

We then adopt the approach proposed in Raviv et al. (2013) to transform the convex objective function (10) into a MIP formulation by using a series of piecewise linear functions.

The basic idea behind the transformation is illustrated in Fig. 5. $\tilde{FD}_s[I_{st}, t, t']$ can be equivalently represented by a series of piecewise linear supporting ($\mathcal{PLS}$) functions. For each time-space node (s, t), let $\mathcal{PLS}_{st}$ be the set of supporting functions. For a supporting function $pls \in \mathcal{PLS}_{st}$, slp_{pls} and itp_{pls} are the slope and intercept of pls, respectively.

Using these supporting functions, the objective function (10) can be transformed to a pure MIP form (Model 2). Additional decision variable w_{st} for each time-space node $(s, t) \in \check{\mathcal{N}}$ is introduced to transfer the convex function into an equivalent MIP model. Constraint (11b) ensures that w_{st} can take the minimum value of the piecewise linear functions.

Model 2

$$\text{Minimize} \sum_{(s,t,r,t')\in\mathcal{A}} c_{strt'} \cdot x_{strt'} + \sum_{(s,t,r,t')\in\mathcal{A}^M} FD_s[I_s^0, B, t] \cdot x_{strt'}$$

$$+ \sum_{(s,t)\in\check{\mathcal{N}}} w_{st} + \sum_{s\in\mathcal{S}} FD_s[I_s^0, B, E^F] \cdot (1 - z_s) + \sum_{(s,t,r,t')\in\mathcal{A}^M} \Delta_{(s,t,E^F)} \cdot x_{strt'} \tag{11a}$$

$$\text{s.t. } w_{st} \geq slp_{pls} \cdot I_{st} + itp_{pls}, \quad (s,t) \in \check{\mathcal{N}} \text{ and } pls \in \mathcal{PLS}_{st} \tag{11b}$$

Constraints (6b – 64j)

3. Solution methodology

Having transformed the nonlinear model to a MIP model, we are now ready to present the solution methodology used to solve the model. Note that the dynamic-BRP is NP-hard as the vehicle routing with pickup and delivery problem is a special case. It is not computationally tractable to solve Model 2 directly via a commercial MIP solver, thereby necessitating a heuristic algorithm.

3.1. A heuristic algorithm

One possible way to solve Model 2 is to (i) enumerate all feasible routes $\mathcal{P}_v$ (see definition in Section 2.2) for each vehicle v; (ii) calculate the best loading sequence for each route p to minimize the total expected user dissatisfaction of all stations served in p (see Section 3.2); and (iii) find a feasible combination of routes to minimize the whole expected user dissatisfaction by solving Model 3. In Model 3, let $\mathcal{P}_v$ represent the set of routes belonging to vehicle v and $\mathcal{P}_s$ represent

the set of routes in which station s is served. An additional binary decision variable θ_p is introduced to indicate whether route p is selected. If $\theta_p = 1$, it indicates that the route p is selected. z_s is inherited from the previous model to indicate whether station s in served by any vehicle. The objective (12a) of Model 3 minimizes the expected user dissatisfaction of all stations during the forecasting time-window. Constraint (12b) ensures that at most one route is selected for each vehicle and constraint (12c) ensures that a station is served at most once.

Model 3

$$\text{Minimize} \sum_{v\in\mathcal{V}} \sum_{p\in\mathcal{P}_v} FD_p^{MIN} \cdot \theta_p + \sum_{s\in\mathcal{S}} FD_s[I_s^0, B, E^F] \cdot (1 - z_s) \tag{12a}$$

$$\text{s.t.} \sum_{v\in\mathcal{V}} \sum_{p\in\mathcal{P}_v} \theta_p \le 1, v \in \mathcal{V} \tag{12b}$$

$$\sum_{s\in\mathcal{S}} \sum_{p\in\mathcal{P}_s} \theta_p - z_s = 0, s \in \mathcal{S} \tag{12c}$$

$$\theta_p, z_s \in \{0, 1\} \tag{12d}$$

However, in the proposed network, there exists a huge number of feasible routes, and it is not computationally tractable to enumerate all of them and then solve Model 3. Although it is possible to design a column generation algorithm to only introduce parts of routes in each iteration, the computational efficiency cannot be guaranteed as many iterations might be required in a column generation algorithm. In our study, we heuristically generate a potential route set (much smaller than the whole route set) based on the linear relaxation of Model 2, denoted as LP(Model 2). It is observed that the arcs with positive value ((0,1]) in the solution of LP(Model 2) probably represent the potential arcs that can result in a better solution for Model 2. Meanwhile, the number of the potential arcs is typically not very huge. These observations motivate us to first detect the potential arcs by solving LP(Model 2), then enumerate all potential routes only consisting of these potential arcs and finally solve Model 3 based on the potential route set. Details of our proposed algorithm are described in Algorithm 1.

Algorithm 1 Our heuristic algorithm to assign a route to each vehicle.

1: **while** The maximum iterations are not reached **do**
2: Solve LP(Model 2) and all arcs with positive flow are regarded as potential arcs;
3: Generate potential routes by enumerating all combinations of the potential arcs;
4: For each potential route, calculate the optimal repositioning sequence $\delta = (\delta_{s_1 t_1}, \delta_{s_2 t_2}, \ldots, \delta_{s_l t_l})$ based on the algorithm described in Section 3.2;
5: **if** Each repositioning vehicle is assigned a route **then**
6: Terminate the algorithm;
7: **else**
8: For vehicles with routes assigned, fix their route selections in the following iterations;
9: Go back to step 2. Solve the restricted LP(Model 2) without considering vehicles with route selections fixed;
10: **end if**
11: **end while**

Remark 5. At step 5 of Algorithm 1, we check whether each vehicle is assigned a route. The reasons for not assigning a route to a repositioning vehicle are as follows: (i) there are excess repositioning vehicles for fulfilling the repositioning tasks; and (ii) the potential route set is relatively small. In the latter case, it is still possible to assign a route for this vehicle by introducing additional routes into the potential route set so that more repositioning tasks can be scheduled. Therefore, our strategy is to run the algorithm for multiple iterations to iteratively generate more potential routes. If a vehicle has a route assigned during a particular iteration, the route selection for this vehicle is fixed (i.e., the route assigned to this vehicle is not altered) in all later iterations.

3.2. Exploring the best loading sequence of vehicle route p

In this section, we propose an approach to explore the best loading sequence for route p, which minimize the total expected user dissatisfaction of stations served in p. We first describe how to estimate the total expected user dissatisfaction of all stations in route p, given that the specific loading sequence δ is executed. Let us consider the expected user dissatisfaction of station s_i in route p. As explained in Fig. 3, we consider two separate periods: (i) the period before that station s_i is served, $[B, t_i]$; and (ii) the period after that station s_i is served, $[t_i, E^F]$. As we assume that a station is served at most once within a repositioning time-window, the expected user dissatisfaction during $[B, t_i]$ is estimated as $FD_{s_i}[I_{s_i}^0, B, t_i]$. At time t_i, the inventory level of station s_i is adjusted to $FI_{s_i}[I_{s_i}^0, B, t_i] + \delta_{s_i t_i}$ and therefore the expected user dissatisfaction of station s_i

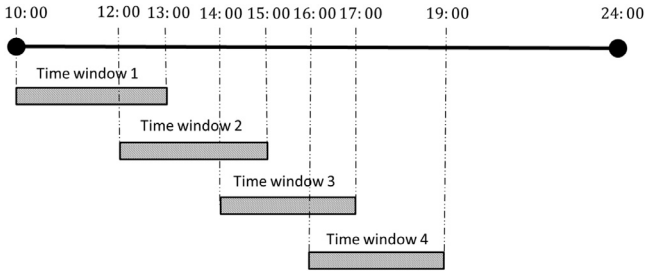

Fig. 6. A rolling-horizon scheme.

Table 2
Description of data sets.

	No. of stations	Location	Data source
Data 1	104	DC capital bike sharing system	Raviv et al. (2013)
Data 2	100	Paris Velib bike sharing system	Raviv et al. (2013)
Data 3	200	Paris Velib bike sharing system	Forma et al. (2015)

during $[t_i, E^F]$ is estimated as $FD_{s_i}[\{FI_{s_i}[I^0_{s_i}, B, t_i] + \delta_{s_i t_i}\}, t_i, E^F]$. In total, the expected user dissatisfaction of station s_i if loading sequence δ is executed can be estimated as:

$$FD_{s_i}(\delta) = FD_{s_i}[I^0_{s_i}, B, t_i] + FD_{s_i}[\{FI_{s_i}[I^0_{s_i}, B, t_i] + \delta_{s_i t_i}\}, t_i, E^F] \tag{13}$$

$FD_p(\delta)$, the summation of the expected user dissatisfaction of all stations in route p, given that loading sequence δ is executed, can be estimated as:

$$FD_p(\delta) = \sum_{(s_i, t_i) \in p} FD_{s_i}(\delta). \tag{14}$$

For route p, the set of all possible loading sequences is a limited set. Therefore, there must exist a loading sequencing δ so that $FD_p(\delta)$ is minimized and let FD_p^{MIN} denote the minimum expected user dissatisfaction of route p, which can be represented as:

$$FD_p^{MIN} = \min_{\delta} FD_p(\delta). \tag{15}$$

However, considering that the number of all possible loading sequences is still very huge, it is not computationally tractable to find the best loading sequence by enumerating all possible sequences. In Appendix B, we proposed an efficient dynamic programming algorithm to explore the best loading sequence for route p within a polynomial time.

3.3. An extension to dynamic-BRP with multiple time-windows

Algorithm 1 only allows for one repositioning service per day. However, in reality, there may be a need to perform multiple services during a day. To address this issue, we extend the model by employing a rolling horizon scheme, that considers multiple time-windows during the day as follows (see Fig. 6). In Fig. 6, there are four overlapped repositioning time-windows: [10:00, 13:00], [12:00, 15:00], [14:00, 17:00] and [16:00, 19:00]. In each of these time-windows, we solve the dynamic-BRP using the heuristic algorithm proposed in Section 3. In this rolling horizon scheme, a station can be served in multiple time-windows during the day. For example, if a station is served at 11:00 during the first repositioning time-window, it can also be served at a time in the following repositioning time-windows, e.g., 12:30 in time-window 2.

Another advantage of this overlapped rolling horizon scheme is to rectify the deviations caused by forecasting errors. For example, at 12:00 of time-window 1, we already have some updated forecasting information about the user demand and bicycle inventory level at each station. However, current repositioning plan is based on the old forecasting information (at 10:00), which might not accurately represent a future scenario. Under our strategy, we terminate the solution made in time-window 1 in advance (at the beginning of the next time-window) and a snapshot of status of vehicles and stations at 12:00 is taken and then serves as inputs of the problem during time-window 2. A new repositioning plan for time-window 2 based on the updated forecasting information is then made.

4. Numerical studies

In this section, we test our heuristic algorithm by solving several data sets from real-world BSS (see Table 2). The first data set (Data 1) is from the DC capital bike sharing system in Washington, D.C., and includes 104 stations. The second and third data sets (Data 2 and Data 3) are from the Paris Velib bike sharing system, and include 100 and 200 stations,

respectively. Each data set contains information regarding the capacity of each station, the capacity of each vehicle, and the distance between any two stations. We use the MIP solver, ILOG CPLEX 12.5, to solve the MIP model. The algorithm was tested on an Intel Xeon E5-1650 3.8G CPU with 8 gigabytes memory. The length of the repositioning time-window (throughout our experiments) was set as three hours. The time-window is discretized into time epochs and the granularity of the discretization is five minutes. The unit travel cost is set as 0.1/vehicle/per minute.

We present our numerical study in four segments. The first three segments consider the dynamic-BRP with a single repositioning time-window and the aim is to analyze our MIP model and the heuristic algorithm. In the first segment, we compare our results with the CPLEX solution. In the second segment, we conduct a sensitivity analysis to explore the impact of different granularities of the time discretization on the solution quality and computational efficiency. In the third segment, we compare our results with two benchmark models. In the fourth segment, a dynamic-BRP with multiple repositioning time-windows (see Section 3.3) is considered. Simulation experiments are conducted to compare our rolling horizon method proposed in Section 3.3 with the 'do-nothing' strategy. The aim of this step is to reveal the benefit of applying our method from a practical perspective.

The arrivals of renters and returners are assumed to follow the non-homogeneous Poisson process (Raviv and Kolka, 2013) and the user arrival rate is allowed to vary after every (five minute) time epoch. In our experimental studies, three types of arrival distributions (summarized in Dell'Amico et al. (2014)) are considered. They are listed as follows:

- Arrival distribution 1: Distribution 1 has a peak of incoming bicycles between 7:00 and 9:00 and a smaller one between 13:00 and 15:00. The peaks of outgoing bicycles occur between 12:00 and 14:00 and between 16:00 and 18:00. This distribution simulates the arrival distribution for stations located at the city center.
- Arrival distribution 2: Distribution 2 is complementary to distribution 1. It has a peak of outgoing bicycles occurring between 7:00 and 9:00 and several peaks of incoming bicycles between 12:00 and 18:00. This distribution simulates the arrival distribution for stations close to car parking areas or residential areas, where users may leave by car to a bicycle station or go directly from home to bicycle station.
- Arrival distribution 3: Distribution 3 has arrivals and departures following a similar pattern during the day. It simulates the arrival distribution for stations close to places that are used all day long, such as subway stations.

In the setup phase for the simulation experiments, each station is first assigned an arrival distribution type. Then, arrival rates of each station at different time epoches are determined with the average number of renters/returners at each station during each day to be 60. It is assumed that if two stations have the same arrival distribution type, their arrival rates are also same. Finally, 200 user arrival instances are randomly generated based on the arrival rates at the stations and simulation studies were performed based on these instances.

To ensure that our simulation experiments mimic realistic scenarios, we assign an arrival distribution type to each station using the following procedures:

- Step 1: For each station, we use the distance matrix provided in the data sets to obtain the cumulative distance between that station and all other stations (e.g., for station s, we calculate this as the sum of the elements on the row corresponding to station s).
- Step 2: We then sort the stations in ascending order of the cumulative distances.
- Step 3: Finally, starting from the station with the lowest cumulative distance, we assign arrival distribution 1 to the first 30% of the stations (as the cumulative distance of bicycle stations around the city center might be small), arrival distribution 2 to the next 40% of the stations, and arrival distribution 3 for the remaining (30%) stations (as the cumulative distance of bicycle stations around the residential areas may be high).

The repositioning time-window is set as [12:00, 15:00] and the forecasting time-window is [12:00, 22:00]. The following criteria are used to set the initial inventory level at 12:00: (i) the inventory level at 5:00 in the morning (based on the assumption that no arrivals occur prior to 5:00) is set as the optimal inventory level to minimize the expected user dissatisfaction of the whole day; (ii) there is no repositioning operations between 5:00 and 12:00, therefore, the inventory level of each station between 5:00 and 12:00 is updated based on the user arrival sequence of the current instance; and (iii) finally, the inventory level at 12:00 is set as the initial inventory level of the repositioning problem. Note that, in our experiment dealing with multiple time-windows, we have additional time-windows (see Section 4.4 for details).

4.1. Comparisons with CPLEX

We first compare our heuristic algorithm with the branch-and-cut algorithm embedded in CPLEX to solve Model 2. The performance indicators of interest in this segment are the optimality gap (between our solution and CPLEX solution) and CPU seconds. A small optimality gap could reveal the advantage of our algorithm from the computational perspective. Additionally, we also compare our solutions with the 'do-nothing' strategy, which is a strategy without any repositioning operations during the repositioning time-window. For each instance, the objective value of the 'do-nothing' strategy is the simulated user dissatisfaction value during the corresponding forecasting time-window with no repositioning operations. Similarly, the optimality gap of the 'do-nothing' strategy is calculated by comparing its solution with the CPLEX solution. The comparison of the optimality gap between our heuristic algorithm and the 'do-nothing' strategy reveals the expected benefit that can be attained by applying our repositioning method. The computation time limit for CPLEX is set as one hour.

Table 3
Results of comparisons with CPLEX.

	No. of stations	Optimality gap%				CPU (s)			
		Our		Do-nothing		Our		CPLEX	
		Avg.	Max.	Avg.	Max.	Avg.	Max.	Avg.	Max.
Data 1	20	4.22	4.92	25.25	37.52	0.22	0.27	18.27	31.73
	30	4.28	6.73	14.99	23.33	0.46	0.47	45.13	50.35
	40	2.80	3.97	17.70	22.93	1.09	1.28	100.57	136.71
	50	2.15	2.93	10.68	17.12	2.51	3.92	178.14	218.55
	60	2.14	3.57	15.47	23.75	3.31	3.75	434.59	680.07
	70	1.89	3.41	12.64	17.01	5.48	6.53	876.34	904.35
	80	1.70	2.91	12.64	15.51	6.67	8.58	864.45	1140.75
	90	1.38	2.20	8.92	10.70	9.53	10.8	1227.71	1321.31
	100	1.29	2.05	9.26	10.58	11.18	13.14	1652.92	2152.92
Data 2	20	5.05	9.02	22.21	35.01	0.16	0.22	4.05	5.42
	30	3.71	5.77	23.42	32.88	0.68	0.92	18.64	37.9
	40	3.00	4.40	13.06	19.50	1.56	2.12	45.23	86.34
	50	2.20	2.89	14.23	17.11	2.70	3.48	141.02	307.34
	60	1.78	2.37	13.28	17.07	4.12	5.04	194.53	428.82
	70	1.57	2.01	13.65	15.82	5.80	7.38	311.07	649.9
	80	1.45	1.82	14.04	15.74	6.70	8.58	257.46	536.78
	90	1.17	1.77	10.01	12.12	10.26	12.96	922.38	2551.15
	100	1.07	1.62	11.01	15.75	12.12	16.60	1260.52	3353.11
Data 3	20	5.27	7.93	19.14	32.07	0.08	0.17	2.81	4.14
	30	2.71	4.23	17.73	29.20	0.36	0.44	19.74	25.14
	40	2.44	3.54	14.56	19.57	0.93	1.23	88.43	196.71
	50	1.91	2.67	13.88	17.39	1.23	1.56	135.78	239.83
	60	2.08	2.43	16.55	22.83	1.74	2.31	154.01	293.13
	70	2.17	4.05	15.58	23.20	2.07	2.90	302.18	616.96
	80	2.37	3.89	16.58	25.03	2.63	3.32	202.24	311.96
	90	1.83	2.70	12.45	20.22	3.95	5.34	1497.79	3600.00
	100	1.59	1.95	11.68	17.56	5.15	7.10	1510.22	3600.00
	110	1.46	1.81	12.00	20.81	4.85	6.81	1255.46	3308.16
	120	1.34	1.70	10.67	19.97	5.81	8.72	1266.89	3600.00
	130	1.56	2.50	10.39	19.70	6.18	8.64	2027.52	3600.00
	140	1.34	2.14	12.37	17.21	7.64	10.07	1888.14	3600.00
	150	1.25	2.14	9.62	16.23	8.73	12.95	1498.24	3600.00
	160	1.27	2.04	8.71	14.60	10.38	15.60	2061.83	3600.00
	170	0.92	1.36	8.15	13.29	11.51	16.34	3078.29	3600.00
	180	1.08	1.86	8.39	13.91	10.68	13.25	2756.24	3600.00
	190	0.88	1.27	7.41	10.26	12.29	16.85	1416.21	2999.83
	200	0.82	1.24	6.75	9.84	15.20	20.52	2338.78	3600.00

For each data set, we conducted experiments with different numbers of stations, i.e., when we set $n = 20$, even if there are 100 stations in the dataset, we only consider the first 20. The number of vehicles is set as 2. Of the 200 user arrival instances that were generated for our study, we only tested the first 50 instances as we focus on the computational aspect in this segment. The computational results are summarized in Table 3, where the first column presents the data set used, and the second column furnishes the number of stations considered for that experiment. The third and fourth columns present the average and maximum optimality gap of solutions given by our algorithm, while the corresponding values for the 'do-nothing' strategy are presented in the fifth and sixth columns. The seventh and eighth columns present the average and maximum CPU seconds of our algorithm, and the corresponding values for the CPLEX code used to solve the MIP model is presented in the ninth and tenth columns. Some key observations and inferences based on the numerical results are illustrated as follows:

- The general trend shows that the average optimality gap for both methods (the heuristic algorithm and the 'do-nothing' strategy) decreases as the problem scale increases. However, it is also observed that the optimality gap for our algorithm is much smaller in all cases. When the number of stations involved is 100 and 200, the average optimality gap of our algorithm is 1.32% and 0.82%, respectively, and the corresponding values for the do-nothing strategy are 10.65% and 6.75%, respectively. The decrease in optimality gap with increasing problem size for our algorithm can be attributed to the fact that there may be more arcs with positive values in the LP relaxation solution (as more stations are available), leading to more potential routes that need to be explored. From the aspect of computation, more potential routes imply a larger solution space and therefore, a higher probability of exploring a better solution. However in the case of the 'do-nothing' strategy, this trend can be explained as follows. For small-scale problems, two vehicles can very efficiently readjust the inventory level of stations and in this range, the CPLEX code has the capability of yielding a high quality solution with small user dissatisfaction, thereby leading to a large optimality gap of the 'do-nothing' strategy.

Table 4
Sensitivity analysis on discretization gap.

	No. of vehicles	5 min		10 min		15 min	
		RLR%	CPU (s)	RLR%	CPU (s)	RLR%	CPU (s)
Data 1	2	9.43	11.16	8.82	5.74	8.22	5.28
	3	12.47	13.63	12.07	8.37	11.86	5.05
	4	16.00	16.62	15.72	9.72	15.63	5.80
	5	19.41	17.95	19.36	10.05	19.05	8.63
Data 2	2	9.36	11.39	8.41	6.94	7.29	5.64
	3	13.89	12.06	12.81	7.72	12.10	5.06
	4	16.62	12.59	16.13	7.94	14.74	5.60
	5	18.79	14.12	17.96	9.83	16.81	5.34
Data 3	2	7.27	16.13	6.90	9.66	6.52	4.85
	3	10.69	15.09	9.95	10.02	8.47	5.31
	4	13.55	17.49	12.94	11.41	11.32	5.37
	5	16.61	19.12	16.05	10.73	15.33	6.27

- The numerical results also reveal that our algorithm can efficiently provide high quality solutions for practical problems. For instance, the average CPU seconds for solving instances with 100 and 200 stations using our algorithm are 9.48 s and 15.20 s, while the corresponding run times for solving the MIP model using CPLEX are much higher (1474.55 s and 2338.78 s, respectively).
- Additionally, we observe that the average computation time of our method roughly has a linear relationship with the number of stations considered in the experiment. The rationale behind this observation is as follows. Our algorithm consists of three major steps, namely, (i) solving the linear relaxation model; (ii) estimating the expected user dissatisfaction for each route; and (iii) solving a set covering problem. The computational time of the first two steps is roughly proportional to the problem scale since they can be solved using algorithms within polynomial time. With regards to the set covering problem, it has been widely observed that its gap between the linear relaxation solution and the integer optimal solution is usually very small. The number of decision variables in our set covering model is not very huge (since we only consider the potential arcs), and the problem can be efficiently solved. Therefore, the summation of these three parts roughly has a linear relationship with the problem scale.

4.2. Sensitivity analysis

In previous segment, our time-space network flow model was built with the granularity of discretization as five minutes. In the following experiments, we conduct a sensitivity analysis to explore the impact of changing the granularity of discretization on the solution quality. The different values chosen for the granularity of discretization are 5 min, 10 min and 15 min. The different values chosen for the number of vehicles are 2, 3 4 and 5. All 200 instances are run in this segment. For each instance, let L_{dn} represent the simulated number of lost requests during the forecasting time-window with the 'do-nothing' strategy and L_m represent the simulated number of lost requests when employing the heuristic algorithm proposed in Section 3. Two key performance indicators considered in this segment are:

- Percentage reduction in lost request (*RLR%*)

$$RLR\% = \left(\frac{L_{dn} - L_m}{L_{dn}}\right) \cdot 100 \tag{16}$$

- CPU seconds

The detailed results are summarized in Table 4, where the first column presents the data set used while the second column furnishes the number of vehicles. The third and fourth columns present the average estimates of *RLR%* and CPU seconds for the case where the width of a time epoch is fixed as five minutes. The corresponding results for a width of 10 min and 15 min are presented in columns five through eight. The observations and inferences from the numerical experiments are as follows:

- On average, the *RLR%* decreases by 4.72% and 6.83% when the granularity of discretization increases from five minutes to ten minutes, and from ten minutes to fifteen minutes, respectively. As the granularity of discretization increases, the expected user dissatisfaction value of our solution may also increase because of the following two reasons: (i) less stations can be served; and (ii) stations cannot be served at the best serving time. However, we can infer from the results that these two reasons have a negligible effect on the solution quality. This can be attributed to the following reasons: (i) vehicles are actually not fully utilized and there exists waiting time in vehicles' routes. It is possible to utilize the waiting time so that the number of served stations is not significantly decreased; Second, small earliness or lateness cannot have a substantial impact on the expected user dissatisfaction value considering the characteristic of the forecasting method.
- It is observed that the CPU seconds decrease significantly as the width of the time epoch increases. On average, the CPU seconds decrease by 38.95% and 35.14% when the granularity of discretization increases from five minutes to ten minutes,

and from ten minutes to fifteen minutes, respectively. The reason is that a larger granularity of discretization results in a comparatively small-scale problem. The computational efficiency of our algorithm largely depends on the model scale, and a small-scale problem results in a short computation time.

- The reduction of CPU seconds is much larger than the deterioration of the solution quality when the granularity of discretization increases. Therefore, the heuristic algorithm has a potential to solve large-scale BRPs (e.g., for a BSS with more than 500 stations) by building a network with a large granularity of discretization (e.g., 15 min) in order to accelerate the computations without sacrificing the solution quality too much.

4.3. Comparisons with benchmark methods

In this segment, we compare our MIP model with two benchmark models. Note that the focus of this experimental segment is on comparing the impact of employing different models on the service quality, therefore we ignore the vehicle travel cost in this segment. The details of these two benchmark models are described as follows.

- A dynamic benchmark model
  The first benchmark model is based on the network flow model proposed by Contardo et al. (2012) for the dynamic-BRP. The benchmark model discretized the repositioning time window into a set of time epoches and all arrivals (rental/return) are assumed to occur at the end of one time epoch. By keeping the flow balance at each epoch, the number of rejected requests can be estimated. Their model requires the expected number of user arrivals (rental/return) during each time epoch (e.g., [12:00, 12:05]). These values can be easily estimated from our arrival distribution function. Additional constraints are incorporated to guarantee that each station is served at most once. Considering the similarity between our model and the dynamic benchmark model, in our experiments, we solve the dynamic benchmark model using our heuristic algorithm proposed in Section 3.1 except that the way of estimating the expected user dissatisfaction of vehicle routes is modified. Our preliminary results (based on 100 instances) reveal that the average and maximum optimality gap of solving the dynamic benchmark model using our algorithm are 1.85% and 2.72%, respectively. It showcases that our algorithm is an appropriate candidate for solving the dynamic benchmark model. For the sake of simplification, the dynamic benchmark method is denoted as *CDBR*.
- A static benchmark model
  The second benchmark model is based on the model proposed for the static-BRP. In the literature, there are already several methodologies proposed for solving the static-BRP (Raviv et al., 2013; Chemla et al., 2013; Erdogan et al., 2015), therefore, a straightforward option to solve the dynamic-BRP is to use an approach similar to these static bicycle repositioning methodologies (in which the stochastic user arrivals during the repositioning time-window are neglected). In this segment, we implement a static benchmark model by slightly modifying our dynamic model (Model 1) to compare with our model. The aim of comparisons is to highlight the importance of considering stochastic user arrivals during the repositioning time-window. In the following context, the static benchmark model is referred to as *SBR*. Note that *SBR* is similar to the model proposed by Raviv et al. (2013) except that the vehicle flows in *SBR* are built in a time-space network while the vehicle flows in the model proposed by Raviv et al. (2013) are built in a connection network. Compared to Model 1, two differences of *SBR* are as follows:
  - The objective (6a) of Model 1 is modified to (17)

$$\text{Minimize} \sum_{(s,t,r,t')\in\mathcal{A}} c_{strt'} \cdot x_{strt'} + \sum_{s\in\mathcal{S}} FD_s\left[\left(I_s^0 + \sum_{t\in\mathcal{T}} \delta_{st}\right), E^R, E^F\right] \tag{17}$$

  - Constraint (6e) of Model 1 is modified to (18):

$$0 \le \sum_{t\in\mathcal{T}} \delta_{st} + I_s^0 \le C_s, \quad s \in \mathcal{S} \tag{18}$$

  In objective (17), the first term minimizes the vehicle traveling cost and the second term minimizes the expected user dissatisfaction between $[E^R, E^F]$. Note that $(I_s^0 + \sum_{t\in\mathcal{T}} \delta_{st})$ indicates the inventory level of station s at the end of repositioning time-window (E^R) and $FD_s[(I_s^0 + \sum_{t\in\mathcal{T}} \delta_{st}), E^R, E^F]$ indicates the expected user dissatisfaction of s between $[E^R, E^F]$. The objective (17) is convex, therefore, it is easily to be linearized. Constraint (18) guarantees that the inventory level of each station s is between $[0, C_s]$. Similarly, we adopt the heuristic algorithm proposed in Section 3.1 to solve *SBR*. Our preliminary results (based on 100 instances) reveal that the average and maximum optimality gap of using heuristic to solve *SBR* are 1.53% and 2.17%, respectively. It justifies that our heuristic algorithm is an appropriate candidate for solving *SBR*.

4.3.1. Computational results

Let L_{CDBR} and L_{SBR} represent the total number of lost requests by employing *CDBR* and *SBR* during the forecasting period, respectively. The percentage reduction in lost request *RLR*1% and *RLR*2% are defined as:

$$RLR1\% = \left(\frac{L_{CDBR} - L_m}{L_{CDBR}}\right) \cdot 100 \tag{19}$$

Table 5
Comparison results with two benchmark models.

	No. of vehicles	*RLR1%*			*RLR2%*		
		Avg.	Max.	Min.	Avg.	Max.	Min.
Data 1	2	2.58	3.46	1.66	3.45	4.31	1.97
	3	2.87	7.99	0.18	2.74	2.99	1.09
	4	4.43	6.13	2.34	2.81	7.17	0.12
	5	2.42	4.83	1.67	2.61	3.02	1.69
Data 2	2	5.54	7.15	3.46	2.93	5.83	1.31
	3	4.68	9.62	0.12	3.98	6.92	0.12
	4	6.76	13.06	3.04	2.98	6.63	−0.33
	5	4.34	8.78	−0.60	2.82	5.07	2.05
Data 3	2	3.78	4.74	3.12	3.14	4.18	−0.62
	3	4.08	10.18	−1.07	3.70	5.00	−0.59
	4	4.47	9.09	0.55	3.61	6.99	1.70
	5	4.94	7.90	1.72	2.86	6.93	0.46

$$RLR2\% = \left(\frac{L_{SBR} - L_m}{L_{SBR}}\right) \cdot 100 \tag{20}$$

A positive *RLR*1% (*RLR*2%) indicates that our method outperforms *CDBR* (*SBR*), and vice versa. The computational results are summarized in Table 5, where the first column presents the data set used while the second column presents the number of vehicles. The third, fourth and fifth columns furnish the average, maximum and minimum *RLR*1% and the sixth, seventh and eighth columns furnish the average, maximum and minimum of *RLR*2%. Some key observations and inferences based on the numerical results are illustrated as follows:

- Our model outperforms *CDBR* on average in all cases, although there are a few instances in which *CDBR* outperforms our model. The reasons for the overall superiority of our model are presented as follows: (i) *CDBR* assumed that all user arrivals can only occur at the end of each time epoch and this can lead to forecasting errors, meaning that a repositioning solution may not reflect the current situation, thereby rendering the repositioning operations weak. On the other hand, our model integrates the forecasting method, which considers the dynamics of bicycle inventory at each station as a *CTMC* model, into our repositioning model, thereby resulting in a higher forecasting accuracy. (ii) *CDBR* only considered the user dissatisfaction within the repositioning time-window. However, in some situations, the inventory level at the end of repositioning time-window can lead to a bad performance in a future period. In contrast, we consider a longer forecasting time-window to consider the impact of the inventory level at the end of the repositioning period.
- Similarly, our model outperforms *SBR* on average in all cases. On average, our model can decrease the expected user dissatisfaction by 3.13% compared to *SBR*. It is also observed that in some scenarios, our model can achieve up to 7.17% decrease in the expected user dissatisfaction. The main reason for the overall superiority is that our model sufficiently considers the stochastic arrivals during the repositioning time-window. In our model, the bicycle repositioning plan for station s is made based on the forecasted inventory level when the vehicle is arriving at s. Meanwhile, the estimation of expected user dissatisfaction is conducted for two sub-periods separately to sufficiently consider the dynamic repositioning in this process. On the contrary, *SBR* makes decisions based on the information obtained at the beginning of the repositioning time-window. However, when a vehicle arrives at s, the inventory level has already changed and deviated from the initial inventory level because of the user arrivals during this period (this phenomenon is more prevalent when s has high user arrival rates but it is scheduled to be served at the very end of the repositioning time-window). As a result, the bicycle repositioning plan made by *SBR* takes much less effect for the inventory adjustment. Therefore, the system performance by applying our dynamic bicycle repositioning plan is better than applying the solution obtained by solving *SBR*.

4.4. Experiments for the dynamic-BRP with multiple time-windows

In this segment, we consider the dynamic-BRP with multiple time-windows, which is solved by the rolling horizon algorithm proposed in Section 3.3. Four time-windows [10:00,13:00], [12:00,15:00], [14:00,17:00] and [16:00,19:00] are considered. The key performance indicators considered in this segment of our experimentation are as follows:

- Reduction in lost requests (*RLR*)

$$RLR = L_{dn} - L_m \tag{21}$$

- Percentage reduction in lost request (*RLR%*):

$$RLR\% = \left(\frac{L_{dn} - L_m}{L_{dn}}\right) \cdot 100 \tag{22}$$

Table 6
Simulation results.

	No. of vehicles	*RLR*		*RLR%*	*TTT (min)*		CPU (s)	
		Avg.	Max.	Avg.	Avg.	Max.	Avg.	Max.
Data 1	2	118.77	147	11.94	249.10	300	21.30	25.49
	3	169.57	220	17.03	358.50	408	24.82	28.43
	4	209.77	269	21.11	458.17	509	30.14	34.24
	5	256.60	310	25.81	542.80	599	34.23	38.09
Data 2	2	94.43	143	19.40	317.50	346	26.28	30.71
	3	130.47	175	26.98	469.13	533	29.41	34.56
	4	160.03	201	33.02	618.87	656	31.20	36.33
	5	184.37	229	38.06	761.70	815	34.74	39.83
Data 3	2	103.90	148	9.73	333.93	375	38.46	44.52
	3	149.03	188	13.97	490.50	554	43.00	49.64
	4	195.77	232	18.38	639.67	696	35.05	45.86
	5	238.47	307	22.33	802.40	903	39.51	47.12

- Total time traveled (*TTT*): It represents the total time spent by all repositioning vehicles. For the 'do-nothing' strategy, *TTT* is always zero as there is no repositioning.
- CPU seconds

We tested all 200 instances in this segment. For each instance, the number of vehicles is set as 2, 3, 4 and 5. The numerical results are summarized in Table 6, where the first column presents the data set used, while the second column presents the number of vehicles. The third column and fourth column furnish the average and maximum number of reduction in lost request (*RLR*), while the average percentage reduction in lost request (*RLR%*) is furnished in the fifth column. The sixth and seventh columns present the average and maximum total traveling time (*TTT*), while the eighth and ninth columns indicate the average and maximum CPU seconds of our solution.

The observations and inferences from the numerical results are presented below.

- On average, our approach results in a significant reduction in lost requests, and a resultant improvement in the service quality of a bicycle sharing system. This observation is consistent with our expectations.
- As the number of vehicles increases, there is a corresponding increase in both *RLR* and *TTT*. This can be attributed to the fact that more vehicles can be used to reposition more bicycles, thereby leading to more effective (bicycle) inventory management. However, more repositioning will result in a longer traveling time for vehicles. Therefore, we note that there needs to be a trade-off between the vehicle operating cost and service performance of BSS. For instance, the number of repositioning vehicles can be increased to improve the service quality when the vehicle operating cost is relatively low.
- The heuristic algorithm can solve problems with up to 200 stations very efficiently. The longest time taken to solve instances with 200 stations (as in Data 3) is around 50 s. It could sufficiently satisfy the computational efficiency requirement in practice.
- As the number of vehicles increases, CPU seconds also increase. The rationale behind this is presented as follows. In the repositioning problem for each horizon, our algorithm tries to explore a route for each vehicle until the maximum limit of iterations is reached. If every vehicle is assigned to a route, the algorithm terminates immediately. However, the algorithm continues until the limit on the number of iterations is reached when there is even one repositioning vehicle that has not been assigned a route, i.e., there are excess repositioning vehicles in the BSS. The limit on the number of iterations in our algorithm is set as the number of vehicles. If the number of vehicles increases, then the total number of iterations in the computation process may also increase, thereby resulting in an increase in the total computational time.

5. Conclusions

In this study, we propose a methodology for solving the dynamic-BRP. First, a time-space network flow model is built in which the user dissatisfaction forecasting, the vehicle routing and the bicycle repositioning are simultaneously considered. However, the proposed model results in a nonlinear objective function. A novel approach is then proposed to transform the model into an equivalent MIP model. Finally, an efficient heuristic algorithm is proposed to solve the model.

Extensive simulation experiments are conducted to study the performance of our method. Three data sets from the literature are used in our study, and our solutions are compared with solutions obtained by directly solving Model 2 using CPLEX and the strategy with no repositioning. The numerical experiments reveal that our solution methodology can achieve a high-quality solution very efficiently in all test instances. A sensitivity analysis is then conducted to study the impact of granularity of discretization on the solution quality and numerical results indicate that as the granularity of discretization is increased, there is a considerable reduction in computational time and the solution quality is not affected much. Third, we compare our MIP model with two benchmark models and the numerical results reveal that on average, our integrated

model can yield significant improvements in service quality. Finally, extensive experiments are conducted to compare the performance of employing our rolling horizon method with the corresponding performance of 'do-nothing' strategy during the whole day.

In our study, we ignore the relationship between stations. Modeling the relationship between stations would significantly change the problem characteristics, and would warrant new models and algorithms. Our model also assumes that the minimum travel cost between stations remain the same throughout the day, and therefore models that consider the time-based minimum travel time would be very useful to model more realistic scenarios.

Acknowledgments

This research has been partially supported under Air Traffic Management Research Institute (NTU-CAAS) Grant no. M4061216. The authors would like to thank five anonymous reviewers, guest editor and editor-in-chief for their valuable comments on the earlier drafts.

Appendix A

A1. Proof of theorem 1

Proof. If we can prove relationship (23) holds, then it is obvious that objective functions (6a) and (9) are equivalent since all other components of (6a) and (9) are the same.

$$\sum_{(s,t,r,t')\in\mathcal{A}^M} \check{FD}_s[I_{st},t,E^F]\cdot x_{strt'} + \sum_{(s,t,r,t')\in\mathcal{A}^M} \Delta_{(s,t,E^F)}\cdot x_{strt'} = \sum_{(s,t,r,t')\in\mathcal{A}^M} FD_s[I_{st},t,E^F] \tag{23}$$

Based on our definition, $FD_s[I_{st},t,E^F] = \check{FD}_s[I_{st},t,E^F] + \Delta_{(s,t,E^F)}$ holds for any time-space node $(s,\ t)$ $\Rightarrow$ Relationship (23) holds $\Rightarrow$ Theorem 1 holds. □

A2. Proof of theorem 2

Proof. The solution space is not changed by replacing the objective function (9) to the objective function (10), therefore, it is only required to prove that for each feasible solution, objective value of (9) equals to objective value of (10). Furthermore, if we can prove that relationship in (24) holds for any feasible solution, then objective value of (9) equals to objective value of (10) for every feasible solution as all other components of objectives (9) and (10) are the same.

$$\sum_{(s,t,r,t')\in\mathcal{A}^M} \check{FD}_s[I_{st},t,E^F]\cdot x_{strt'} = \sum_{(s,t)\in\check{\mathcal{N}}} \check{FD}_s[I_{st},t,E^F] \tag{24}$$

Before proving relationship (24), we first transfer the relationship (24) into an equivalent form (25).

$$\sum_{(s,t)\in\check{\mathcal{N}}} \{\check{FD}_s[I_{st},t,E^F]\cdot \sum_{(s,t,r,t')\in\mathcal{A}^M_{st}} x_{strt'}\} = \sum_{(s,t)\in\check{\mathcal{N}}} \check{FD}_s[I_{st},t,E^F] \tag{25}$$

Let us consider a feasible solution. If we can prove that for any time-space node $(s,t)\in\check{\mathcal{N}}$ relationship (26) holds, then relationship (25) also holds.

$$\check{FD}_s[I_{st},t,E^F]\cdot \sum_{(s,t,r,t')\in\mathcal{A}^M_{st}} x_{strt'} = \check{FD}_s[I_{st},t,E^F] \tag{26}$$

For a feasible solution, $\sum_{(s,t,r,t')\in\mathcal{A}^M_{st}} x_{strt'}$ can only be 0 or 1 because there is at most one moving arcs flowing out of a time-space node $(s,\ t)$ (,which is enforced by constraint (6d)). Let us consider its two possible values:

- If $\sum_{(s,t,r,t')\in\mathcal{A}^M_{st}} x_{strt'} = 1$, then automatically relationship (26) holds;
- Else $\sum_{(s,t,r,t')\in\mathcal{A}^M_{st}} x_{strt'} = 0$, it indicates that no vehicle is serving station s at time t. Therefore, no bicycle loading or unloading operations are conducted for station s at time t. In other words, $\delta_{st} = 0$ according to constraints (6f and 6g). Thereby, we have $I_{st} = FI_s[I^0_s,B,t] + \delta_{st} = FI_s[I^0_s,B,t]$. Based on the definition of affiliation funcion in Section 3.2, we have $\check{FD}_s[FI_s[I^0_s,B,t],t,E^F] = 0$. Then, left and right parts of relationship (26) both equal to 0 and therefore relationship (26) holds.

In conclusion, for any feasible solution and any $(s,t)\in\check{\mathcal{N}}$, relationship (26) holds $\Rightarrow$ Relationship (25) holds for any feasible solution $\Rightarrow$ Theorem 2 holds. □

Appendix B

In this section, we present the algorithm used to decide the loading sequence along stations in a route p using a dynamic programming algorithm. Given a route p, the traveling cost is already fixed. Therefore, we only need to guarantee that the summation of the expected user dissatisfaction of all served stations is minimized.

Suppose a route p is $((s_1, t_1), (s_2, t_2), .., (s_l, t_l))$. For each station $s_i \in p$, we then create $K+1$ labels, namely $label_{(s_i,0)}, label_{(s_i,1)}, .., label_{(s_i,K)}$. Note that K is the maximum capacity of vehicle. Label $label_{(s_i,q)}$ indicates the minimum expected user dissatisfaction of all stations before s_i (s_i is also included) in route p, given that there are q bicycles loaded in the vehicle before this vehicle arrives at station s_i. Let $\delta_{s_it_i}$ denote the amount of bicycles loaded from the vehicle to station s_i. If $\delta_{s_it_i} > 0$, some bicycles are loaded to station s_i, otherwise, some bicycles are unloaded from the station. For station s_i, the forecasted inventory level at time t_i is denoted as $FI_{s_i}[I^0_{s_i}, B, t_i]$ and the expected user dissatisfaction between $[B, t_i]$ is $FD_{s_i}[I^0_{s_i}, B, t_i]$. At t_i, $\delta_{s_it_i}$ bicycles are loaded to s_i and the inventory level becomes $FI_{s_i}[I^0_{s_i}, B, t_i] + \delta_{s_it_i}$. The expected user dissatisfaction between $[t_i, E^F]$ is $FD_{s_i}[\{FI_{s_i}[I^0_{s_i}, B, t_i] + \delta_{s_it_i}\}, t_i, E^F]$. In total, the expected user dissatisfaction for s_i between $[B, E^F]$ is $FD_{s_i}[I^0_{s_i}, B, t_1] + FD_{s_i}[\{FI_{s_i}[I^0_{s_i}, B, t_i] + \delta_{s_it_i}\}, t_i, E^F]$. We will use this result in the following context. For the purpose of simplicity, we denote $\Pi_{(s_i,t_i,\delta_{s_it_i})}$ to be $FD_{s_i}[I^0_{s_i}, B, t_1] + FD_{s_i}[\{FI_{s_i}[I^0_{s_i}, B, t_i] + \delta_{s_it_i}\}, t_i, E^F]$. The details of the algorithm are described in Algorithm 2.

Algorithm 2 Calculating the minimum value of the expected user dissatisfaction.

initial load of the corresponding vehicle is K^0_v
for any $s_i \in [s_1, .., s_l]$ contained in route p and any $q \in [0, K]$, set $label_{(s_i,q)}$ as $+\infty$
considering the first station s_1:
for $\delta_{s_1t_1} \in [K, -K]$ **do**
  if $0 \le FI_{s_1}[I^0_{s_1}, B, t_{s_1}] + \delta_{s_1t_1} \le C_s$ and $0 \le K^0_v - \delta_{s_1t_1} \le K$ **then**
    set $label_{s_1, K^0_v - \delta_{s_1t_1}}$ as $\Pi_{s_1,t_1,\delta_{s_1t_1}}$
    set proceeding label of $label_{s_1, K^0_v - \delta_{s_1t_1}}$ as a dummy source label
  end if
end for
for $(s_i, t_i) \in p$ **do**
  for $label_{(s_i,q):q\in[0,K]}$ **do**
    if $label_{(s_i,q)} \le +\infty$ **then**
      for $\delta_{s_{i+1}t_{i+1}} \in [-K, K]$ **do**
        if $0 \le FI_{s_{i+1}}[I^0_{s_{i+1}}, B, t_{i+1}] + \delta_{s_{i+1}t_{i+1}} \le C_s$ and $0 \le K^0_v - \delta_{s_{i+1}t_{i+1}} \le K$ **then**
          if $label_{(s_i,q)} + \Pi_{s_{i+1},t_{i+1},\delta_{s_{i+1}t_{i+1}}} \le label_{s_{i+1}, q-\delta_{s_{i+1}t_{i+1}}}$ **then**
            set $label_{s_{i+1}, q-\delta_{s_{i+1}t_{i+1}}}$ as $label_{(s_i,q)} + \Pi_{s_{i+1},t_{i+1},\delta_{s_{i+1}t_{i+1}}}$
            set proceeding of label $label_{s_{i+1}, q-\delta_{s_{i+1}t_{i+1}}}$ as $label_{s_i,q}$
          end if
        end if
      end for
    end if
    for last station s_l, let $labelmin_{s_l,q}$ to be the label with the smallest value
    backtracking $labelmin_{s_l,q}$, we can then know the specific loading/unloading demands in each station of p
  end for
end for

Lemma 2. *For route p with l stations, computation complexity of Algorithm 2 is* $O((K+1)^{2\cdot l})$.

Proof. For each station, there are at most $K+1$ labels. For each label, there are at most $O(K+1)$ unit computation operations required. Therefore, for each station, there are at most $O((K+1)^2)$ computation operations required. Considering all stations in route p, computational complexity is $O((K+1)^{2\cdot l})$. □

References

Barth, M., Todd, M., 1999. Simulation model performance analysis of a multiple station shared vehicle system. Transp. Res. Part C 7 (4), 237–259.
Benchimol, M., Benchimol, P., Chappert, B., De La Taille, A., Laroche, F., Meunier, F., Robinet, L., 2011. Balancing the stations of a self service bike hire system. RAIRO – Oper. Res. 45 (01), 37–61.
Borgnat, P., Abry, P., FLANDRIN, P., Robardet, C., Rouquier, J.B., Fleury, E., 2011. Shared bicycles in a city: a signal processing and data analysis perspective. Adv. Complex Syst. 14 (03), 415–438.
Caggiani, L., Ottomanelli, M., 2012. A modular soft computing based method for vehicles repositioning in bike-sharing systems. Procedia – Soc. Behav. Sci. 54 (1), 675–684.

Caggiani, L., Ottomanelli, M., 2013. A dynamic simulation based model for optimal fleet repositioning in bike-sharing systems. Procedia – Soc. Behav. Sci. 87 (1), 203–210.
de Chardon, C.M., Caruso, G., 2015. Estimating bike-share trips using station level data. Transp. Res. Part B 78 (1), 260–279.
Chemla, D., Meunier, F., Wolfler Calvo, R., 2013. Bike sharing systems: solving the static rebalancing problem. Discret. Optim. 10 (2), 120–146.
Cheu, R., Xu, J.X., Kek, A., Lim, W., Chen, W., 2006. Forecasting shared-use vehicle trips with neural networks and support vector machines. Transp. Res. Rec.: J. Transp. Res. Board 1 (1968), 40–46.
Contardo, C., Morency, C., Rousseau, L. M., 2012. Balancing a dynamic public bike-sharing system. https://www.cirrelt.ca/DocumentsTravail/CIRRELT-2012-09.pdf.
Dell'Amico, M., Hadjiconstantinou, E., Iori, M., Novellani, S., 2014. The bike sharing rebalancing problem: mathematical formulations and benchmark instances. Omega 45 (1), 7–19.
Dell'Amico, M., Iori, M., Novellani, S., Stutzle, T., 2016. A destroy and repair algorithm for the bike sharing rebalancing problem. Comput. Oper. Res. 71 (1), 149–162.
Di Gaspero, L., Rendl, A., Urli, T., 2013. A hybrid ACO+CP for balancing bicycle sharing systems. In: Proceedings of International Workshop on Hybrid Metaheuristics, pp. 198–212.
Di Gaspero, L., Rendl, A., Urli, T., 2016. Balancing bike sharing systems with constraint programming. Constraints 1 (2), 318–348.
Erdogan, G., Battarra, M., Wolfler Calvo, R., 2015. An exact algorithm for the static rebalancing problem arising in bicycle sharing systems. Eur. J. Oper. Res. 245 (3), 667–679.
Forma, I.A., Raviv, T., Tzur, M., 2015. A 3-step math heuristic for the static repositioning problem in bike-sharing systems. Transp. Res. Part B 71 (1), 230–247.
Froehlich, J., Neumann, J., Oliver, N., 2009. Sensing and predicting the pulse of the city through shared bicycling. In: Proceedings of International Joint Conference on Artificial Intelligence, IJCAI.
Ho, S.C., Szeto, W.Y., 2014. Solving a static repositioning problem in bike-sharing systems using iterated tabu search. Transp. Res. Part E 69, 180–198.
Ho, S.C., Szeto, W.Y., 2017. A hybrid large neighborhood search for the static multi-vehicle bike-repositioning problem. Transp. Res. Part B 95, 340–363.
Kaltenbrunner, A., Meza, R., Grivolla, J., Codina, J., Banchs, R., 2010. Urban cycles and mobility patterns: exploring and predicting trends in a bicycle-based public transport system. Pervasive Mob. Comput. 6 (4), 455–466.
Kek, A.G., Cheu, R.L., Meng, Q., Fung, C.H., 2009. A decision support system for vehicle relocation operations in carsharing systems. Transp. Res. Part E 45 (1), 149–158.
Kloimllner, C., Papazek, P., Hu, B., Raidl, G.R., 2014. Balancing bicycle sharing systems: an approach for the dynamic case. In: Proceedings of European Conference on Evolutionary Computation in Combinatorial Optimization, pp. 73–84.
Laporte, G., Meunier, F., Wolfler Calvo, R., 2015. Shared mobility systems. 4OR 13 (4), 341–360.
Li, Y.F., Szeto, W.Y., Long, J.C., Shui, C.S., 2016. A multiple type bike repositioning problem. Transp. Res. Part B 90 (1), 263–278.
Mahmoudi, M., Zhou, X., 2016. Finding optimal solutions for vehicle routing problem with pickup and delivery services with time windows: a dynamic programming approach based on state space time network representations. Transp. Res. Part B 89, 19–42.
Midgley, P., 2009. The role of smart bike-sharing systems in urban mobility. Journeys 2 (1), 23–31.
Nair, R., Miller-Hooks, E., 2011. Fleet management for vehicle sharing operations. Transp. Sci. 45 (4), 524–540.
Papazek, P., Raidl, G.R., Rainer-Harbach, M., Hu, B., 2013. A PILOT/VND/GRASP hybrid for the static balancing of public bicycle sharing systems. In: Proceedings of International Conference on Computer Aided Systems Theory, pp. 372–379.
Pfrommer, J., Warrington, J., Schildbach, G., Morari, M., 2014. Dynamic vehicle redistribution and online price incentives in shared mobility systems. IEEE Trans. Intell. Transp. Syst. 15 (4), 1567–1578.
Raviv, T., Kolka, O., 2013. Optimal inventory management of a bike-sharing station. IIE Trans. 45 (10), 1077–1093.
Raviv, T., Tzur, M., Forma, I.A., 2013. Static repositioning in a bike-sharing system: models and solution approaches. EURO J. Transp. Logist. 2 (3), 187–229.
Regue, R., Recker, W., 2014. Proactive vehicle routing with inferred demand to solve the bikesharing rebalancing problem. Transp. Res. Part E 72 (1), 192–209.
Schuijbroek, J., Hampshire, R., van Hoeve, W.J., 2013. Inventory Rebalancing and Vehicle Routing in Bike Sharing Systems. Technique report.
Shu, J., Chou, M.C., Liu, Q.Z., Teo, C.P., Wang, I.L., 2013. Models for effective deployment and redistribution of bicycles within public bicycle-sharing systems. Oper. Res. 61 (6), 1346–1359.
Szeto, W.Y., Liu, Y., Ho, S.C., 2016. Chemical reaction optimization for solving a static bike repositioning problem. Transp. Res. Part D 47, 104–135.
Tang, J., Song, Y., Miller, H.J., Zhou, X., 2016. Estimating the most likely space time paths, dwell times and path uncertainties from vehicle trajectory data: a time geographic method. Transp. Res. Part C 66, 176–194.
Tong, L., Zhou, X., Miller, H.J., 2015. Transportation network design for maximizing space time accessibility. Transp. Res. Part B 81, 555–576.
Wang, T., 2014. Solving Dynamic Repositioning Problem for Bicycle Sharing Systems: Model, Heuristics, and Decomposition. Master thesis. The University of Texas at Austin.
Zhang, D., Klabjan, D., 2017. Optimization for gate re-assignment. Transp. Res. Part B 95, 260–284.
Zhang, D., Yu, C., Desai, J., Lau, H., 2016. A math-heuristic algorithm for the integrated air service recovery. Transp. Res. Part B 84, 211–236.
Zhang, D., Yu, C., Lau, H., 2015. A two stage heuristic algorithm for the integrated aircraft and crew schedule recovery problems. Comput. Ind. Eng. 87, 436–453.