Large-Scale Trip Planning for Bike-Sharing Systems

Zhi Li[a,b], Jianhui Zhang[a,*], Jiayu Gan[a], Pengqian Lu[a], Zhigang Gao[a], Wanzeng Kong[a]

[a]*College of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China*
[b]*Department of Computer Science, Stony Brook University, Stony Brook, NY 11794, USA*

Abstract

Bike-Sharing System (BSS) is a convenient service deployed in many big cities to alleviate the last-mile problem. However, the explosion of users in BSS causes the inconvenience of bike utilization. Much efforts have been devoted to performing resources prediction, redistribution to help with BSS operation. But there are only a few works focus on trip planning, and none of them notice the complete bike trip in BSS composes of three segments: from user's start point to a start station, from the start station to a target station, and from the target station to user's terminal point. To study the case, this paper addresses a static trip planning problem by considering the bike utilization conflict in BSS so as to maximize the number of the served users and minimize their trip time. The problem is formulated as the weighted k-set packing problem. We then design a Greedy Trip Planning algorithm (GTP) and a Humble Trip Planning algorithm (HTP) to solve it. For comparison, we design a Random Trip Planning algorithm (RTP) as a benchmark. Extensive simulation results show that GTP and HTP outperform RTP, and reveal the impact of different factors on the performance of our algorithms.

Keywords: Bike-Sharing System, Trip Planning, Complete Bike Trip, Service Quality, Bike Utilization Conflict

[*]Corresponding author.
Email address: jhzhang@ieee.org (Jianhui Zhang)

1. Introduction

Bike-sharing system (BSS) offers convenient public service by allowing any person to borrow and return bike at the stations nearby [1]. It has been widely deployed in many famous cities all around the world in recent years to solve the last-mile problem, such as Hangzhou, New York, and Chicago [2][3]. More and more users choose the bike-sharing service in their daily life so that the explosion of user amount brings some new pressure to both user and BSS. Now, users are often not able to borrow or return bikes from or to stations because of the serious imbalance of resources in the BSS, *i.e.* some stations lack available bikes or docks. To relieve the pressure, the prediction is designed to estimate the resources [4–10] and the redistribution is adopted to alleviate the imbalance of resources for the BSS [11–18]. From the user's point of view, the station pair selection is proposed to help user borrow and return bike [19–21].

From the BSS's view, it is necessary to alleviate the imbalance by redistributing bikes frequently. Several works [11–14] design the routing mechanisms for trucks to move bikes and the incentive mechanisms for users to help with bike redistribution. However, redistribution is quite expensive and challenging because of the hardness to know available resources at each station in real time or in advance. Before the redistribution, it's basic to predict the available resources at each station. Some works leverage techniques like machine learning and data mining to do so based on historical bike-sharing data [4–10]. The resources prediction can estimate the overall amount of available bikes and docks and is not very useful for each single user.

From the perspective of bike users, the trip planning is more valuable than resources prediction or distribution since they can complete the trip by knowing exactly where to borrow or return bikes. In [19], Ji Won Yoon *et al.* mention to help users select the best pair of stations with the minimal time cost and the maximal probability to complete trip for bike borrowing and returning after giving the origin and destination locations. Agostino Nuzzolo *et al.* design a trip planning system which considers user's preference [22]. However, they care only about single user's preference and do not take system wide resources and bike-sharing service quality into account [19, 22–24].

When user uses the bike in the BSS, the complete trip composes of the following three segments: from user's start point to a start station, from the start station to a target station and from the target station to user's terminal point. However, to the best of our knowledge, only a few works notice the trip composition and design corresponding trip planning algorithm. To study the case, this paper addresses a trip planning problem, which considers bike utilization conflict caused by insufficient bikes/docks at stations in the BSS. The goal of the trip planning is to maximize the number of served users and minimize their trip time, *i.e.* to achieve higher service quality of the BSS. We study the case that all bike resources can be allocated only once and the arriving time of user is not considered. The problem can be mapped to the *weighted k-set packing problem*, which is known to be NP-hard [25]. To solve the problem, intuitively, this paper first designs a Greedy Trip Planning algorithm (GTP), in which the bike utilization conflict is not fully considered during planning trips for users. This paper then designs a Humble Trip Planning algorithm (HTP) which takes the bike utilization conflict into consideration. For comparison, this paper designs a Random Trip Planning algorithm (RTP) as a benchmark in the experiment. Extensive simulation is conducted based on the real dataset of the BSS of Hangzhou city. Simulation results show that GTP and HTP outperform RTP and reveal the impact of the experiment region, the user amount and the user's maximal walking range on our algorithms.

Contribution. To summarize, the contributions of our work are as follows:

- To the best of our knowledge, this is the first work that addresses the static trip planning problem in the BSS which considers the complete bike trip, system-wide resources and quality of bike-sharing service.

- We formulate the trip planning problem as a weighted k-set packing problem and design two heuristic algorithms, namely GTP and HTP, to solve it. For comparison, we design RTP for the problem as a benchmark.

- We implement the three proposed algorithms and conduct extensive simulation based on the real data of the BSS deployed in Hangzhou city in China. The impact of different factors on our algorithms is analyzed.

Road map. In Section 2, we present some models in the BSS. We formulate the trip planning problem and prove its NP-hardness in Section 3. To solve the problem, we design the GTP algorithm and HTP algorithm in Section 4 and 5 respectively. We evaluate our algorithms by extensive simulation in Section 6 and put some discussions in Section 7. Then we review some state-of-the-art works in Section 8 and conclude the whole paper in Section 9.

2. System Model

This paper considers the BSS with the bike station set B and the user set U, where $B = \{b_1, \ldots, b_N\}$ and $U = \{u_1, \ldots, u_M\}$, N and M are the numbers of bike stations and users respectively. In B, each bike station b_i is associated with a location l_i, the number of available bikes A_i^o, and the number of available docks A_i^t. Obviously, $A_i^o \geq 0$ and $A_i^t \geq 0$. In U, each user u_i is associated with a start point l_i^o, a terminal point l_i^t, a start station set B_i^o, and a target station set B_i^t. B_i^o contains all stations that u_i can borrow a bike and B_i^t contains all stations that u_i can return the bike.

In this paper, we consider that each user u_i's trip is a complete bike utilization process with three segments, which is more close to practice. Specifically, u_i first walks from his start point l_i^o to a start station b_i^o. After borrowing a bike, he rides to a target station b_i^t to return the bike and then walks to his terminal point l_i^t. Fig. 1 shows an example of a complete trip. The incomplete bike utilization process, *i.e.* users are not able to borrow or return bikes, is not considered in this paper. This paper defines trip in the BSS as follows.

Definition 1 (Trip). *In the BSS, a trip for user u_i, denoted by $t_i = (l_i^o, b_i^o, b_i^t, l_i^t)$, is a complete bike utilization process with three segments: from user's start point l_i^o to a start station b_i^o, from the start station to a target station b_i^t and from the target station to user's terminal point l_i^t.*

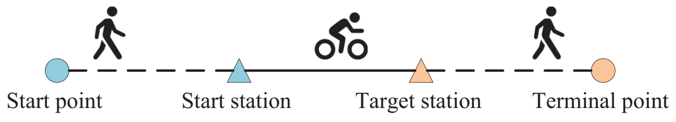


Figure 1: The trip with three segments

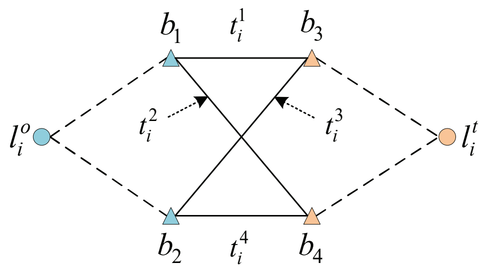


Figure 2: An example of user u_i's trip set H_i

Each trip consumes some time, which is the sum of the time consumption of the three segments, denoted by τ_i^1, τ_i^2, τ_i^3 respectively. The time is related to the length of each segment and the speed at which users travel through each segment. This paper defines the trip time as follows.

Definition 2 (Trip Time). *The trip time $C(t_i)$ of trip t_i is the overall time to complete t_i:*

$$C(t_i) = \tau_i^1 + \tau_i^2 + \tau_i^3 \tag{1}$$

In the BSS, each user may borrow and return a bike at different bike stations. Therefore, he has several available trip options to reach his terminal point. These trips are included in a set, named the trip set.

Definition 3 (Trip Set). *A trip set of user u_i, denoted by $H_i = \{t_i^1, \ldots, t_i^K\}$, is the* *collection of all trips available to u_i, and can be calculated by the following equation.*

$$H_i = \{t_i = (l_i^o, b_i^o, b_i^t, l_i^t) | b_i^o \in B_i^o, b_i^t \in B_i^t\} \tag{2}$$

Let H denote the union of all users' trip sets, we have

$$H = \bigcup_{u_i \in U} H_i \tag{3}$$

For example, in Fig. 2, b_1, b_2 are two start stations and b_3, b_4 are two target stations for user u_i. The four stations and u_i's start and terminal points compose four different trips, denoted by t_i^1, t_i^2, t_i^3 and t_i^4. We have $H_i = \{t_i^1, t_i^2, t_i^3, t_i^4\}$.

Most symbols used in this paper are summarized in Table 1.

Table 1: Symbol and meaning

Symbol	**Description**	**Symbol**	**Description**
u	User	b	Bike station
U	User set	B	Bike station set
t	Trip	A_i^o	b_i's available bikes
$C(t)$	Trip time	A_i^t	b_i's available docks
H	Trip set	h	Allocated trip set
$Q(t)$	Trip quality	d_{max}	Maximal walking range
l	Location	$d(l_i, l_j)$	Distance between l_i and l_j

3. Problem Formulation

To help users access the bike-sharing service more conveniently and improve the service quality of the BSS, this paper introduces the trip planning, which intends to allocate a trip to each user in the system. This section first describes the trip allocation by considering the bike utilization conflict, then formulates the trip planning problem and shows its hardness.

3.1. Trip Allocation

Intuitively, in order to save time, all users would like to look for the nearest stations to borrow or return bikes. However, because of the imbalance of resources at bike stations, some users may not be able to borrow or return bikes at their nearest stations, *i.e.* the *bike utilization conflict* among users' trips happens. Specifically, if a bike station b_i has insufficient bikes or docks to fulfill users' trips, bike utilization conflict among these trips happens at the bike station, and b_i is called conflicting start station or conflicting target station correspondingly. If a trip contains a conflicting station, it is called a conflicting trip. Otherwise it is a non-conflicting trip. For example, there are three trips, as trip t_i of user u_i, trip t_j of user u_j and trip t_k of user u_k shown in Fig. 3. All stations have enough available bikes and docks except b_1 and b_5. Station b_1 has only one bike and station b_5 has only one available dock. Therefore, t_i and t_j are two trips in conflict at b_1 while t_j and t_k are in conflict at b_5. We have that b_1 and b_5 are two conflicting stations. t_i, t_j and t_k are three conflicting trips.

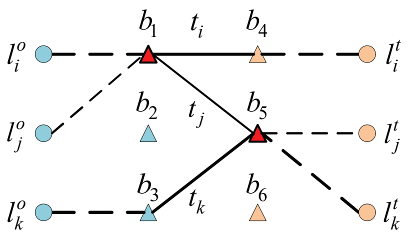


Figure 3: Bike utilization conflict

Considering there are bike utilization conflicts in the BSS, some users can access bike resources and the others may not. Therefore, trips in the BSS should be carefully allocated to users so that more users can access the bike-sharing service and they can reach their terminal points quicker. We include all the allocated trips into an *allocated trip set*, denoted by h. Obviously, $h \subset H$. Note that some users may not have an allocated trip because of the limited bike resources, and the set h is not unique under different trip allocation strategy.

3.2. Trip Planning Problem

In this paper, the process of allocating trips to users in the BSS is called trip planning. The goal of trip planning is to help more users access the bike-sharing service and help them reach their terminal points quicker. That is, when planning trips for users, the number of served users should be maximized and the trip time of these users should be minimized. In order to consider the two optimization problems together, this paper introduces the service quality of the BSS. Before that, we first define the trip quality as follows.

Definition 4 (Trip Quality). *The quality $Q(t_i)$ of a trip t_i in the BSS is inversely proportional to its trip time.*

$$Q(t_i) = \frac{1}{C(t_i)} \tag{4}$$

The total quality of all allocated trips is defined as the service quality of the BSS. If there is no allocated trip, the service quality of the BSS is 0. According to Eq. (4), more users who finish their trips lead to higher service quality of the BSS. And if these users can get their terminal points quicker, the service quality of the BSS will be higher. Thus, by maximizing the service quality of the BSS, we can maximize the number of served users and minimize their trip time at the same time. Therefore, the objective of the trip planning in this paper is to maximize the overall quality of all allocated trips.

Problem formulation. The trip planning problem is described as: Given the bike station set B and the user set U, the trip planning problem is to find the allocated trip set h so that the total quality of all trips in h is maximized. The problem can be formulated as:

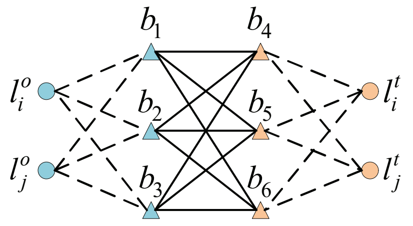
$$\max \sum_{t_i \in h} Q(t_i) \tag{5}$$

$$\textbf{s.t.} \quad A_j^o \geq \sum_{t_i \in h} I(b_i^o, b_j), \forall b_j \in B \tag{6}$$

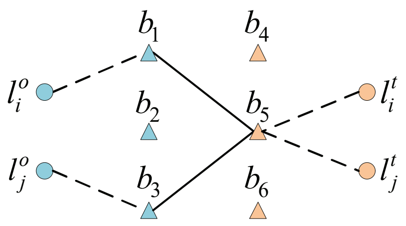
$$A_j^t \geq \sum_{t_i \in h} I(b_i^t, b_j), \forall b_j \in B \tag{7}$$

Where $I(b_i, b_j)$ is a function whose value is 1 when b_i and b_j are the same bike station, and 0 otherwise. Eq. (6) and (7) indicate the resource constraints of each station in the

BSS.



(a) Trips in H



(b) Trips in h

Figure 4: Trip planning

For example, suppose there are two users u_i and u_j in the BSS, Fig. 4(a) shows all available trips of these users, *i.e.* the trip set H. The goal of the trip planning is to find two trips among all available trips as shown in Fig. 4(b), and the trip quality of each of the two trips should be as high as possible so that the two trips have the maximum total trip quality. The two trips form the allocated trip set h.

3.3. Hardness of Trip Planning Problem

This part shows the trip planning problem can be reduced to the NP-hard problem: the *weighted k-set packing problem* [25], which is described as follow: Given a collection of sets $\mathcal{I} = \{I_1, \dots, I_n\}$, each of which has an associated weight and contains at most k elements drawn from a finite basic set G, the task is to find a collection with disjoint sets of maximum total weight.

For a trip $t_i = (l_i^o, b_i^o, b_i^t, l_i^t)$ of user u_i, he borrows one bike from the start station b_i^o, and consume one dock at a target station b_i^t. Let set $I_i = \{u_i, \text{a bike at } b_i^o, \text{a dock at } b_i^t\}$ and the weight of I_i is the trip quality of t_i. Let $\mathcal{I} = \{I_1, \dots, I_n\}$ be the collection of all possible I_i and G be the set of all users and all available bikes and docks in the BSS. It's easy to find that $\mathcal{I}$ is drawn from G. The trip planning problem can be translated to: find a collection of disjoint sets from $\mathcal{I}$ so that the total weight of these disjoint sets is maximized. By this way, the trip planning problem is reduced to a maximum weighted 3-set packing problem. Since the k-set packing problem is NP-hard for any $k \geq 3$, even in the unweighted case, the maximum weighted 3-set packing problem is NP-hard [25]. Therefore, we have the following theorem.

Theorem 1. *The trip planning problem given in (5), (6) and (7) is NP-hard.*

Due to the NP-hardness of the trip planning problem, we design two heuristic algorithms, GTP and HTP, in the following two sections to solve it.

4. Greedy Trip Planning

In this section, we present GTP to solve the trip planning problem.

4.1. GTP Algorithm

Intuitively, in order to maximize the total quality of all trips in h, GTP greedily and iteratively allocates the trip which has the maximum quality in H. Note that when a trip is allocated to a user u_i, the available bikes at the start station and the available docks at the target station should be reduced by 1, and all trips of the user should be deleted from H because he needs only to complete one trip to reach his terminal point. At the same time, all trips that have no bikes or docks available should also be deleted from H. GTP is summarized in Algorithm 1.

Algorithm 1 Greedy Trip Planning Algorithm

```
Input: Bike station set: $B$; User set: $U$;
Output: Allocated trip set: $h$;
1: $h = \emptyset$;
2: Construct the trip set $H$ and calculate all trips' quality;
3: while $H \neq \emptyset$ do
4:     Find the maximum-quality trip $t_i^* = (l_i^o, b_i^o, b_i^t, l_i^t)$ in $H$;
5:     $h = h \cup \{t_i^*\}$;
6:     Update bike resources at $b_i^o$ and $b_i^t$;
7:     Delete $H_i$ and all trips do not have available bikes/docks from $H$;
8: end while
```

Specifically, GTP first initializes the allocated trip set as an empty set and constructs the trip set H based on Eq. (2) and (3). The quality of all trips in H are calculated based on Eq. (4) at the same time. While the set H is not empty, GTP greedily allocates a trip in each round. In each round, GTP first finds the maximum-quality trip $t_i^* = (l_i^o, b_i^o, b_i^t, l_i^t)$ in H and allocates the trip to its corresponding user u_i by adding it to the allocated trip set h. After that, the resources at the stations b_i^o and b_i^t are updated, *i.e.* the amount of bikes at b_i^o and the amount of available docks at b_i^t are both reduced by 1. Then all trips of u_i are removed from the trip set H by deleting all trips start from l_i^o and end at l_i^t, and all trips that do not have available bikes or docks are also removed from H. When there is no trip in H, GTP outputs the allocated trip set h and stops.

4.2. Trip Pruning

In the BSS, a user can borrow a bike and return it to any station. However, it is too complex to consider all these stations in trip planning because there are hundreds of bike stations in the BSS. To tackle the problem, this part introduces a strategy to prune the trip of users.

In reality, some stations may be too far from the user or his terminal point. He may not be willing to walk that far to borrow a bike or to his terminal point after returning the bike. Therefore, when performing trip planning, trips that contain such stations can be pruned from the trip set of each user by setting a maximal walking range to reduce the complexity of the trip planning. Let d_{max} be the maximal walking range, the start station set B_i^o and the target station set B_i^t of user u_i can be calculated by the following two equations.

$$B_i^o = \{b_j | b_j \in B, A_j^o > 0, d(l_i^o, l_j) < d_{max}\} \tag{8}$$

$$B_i^t = \{b_j | b_j \in B, A_j^t > 0, d(l_j, l_i^t) < d_{max}\} \tag{9}$$

where $d(l_i, l_j)$ is a function which measures the distance between l_i and l_j. Note that the distance measurement of this function could be any existing method, such as haversine distance, Manhattan distance, the shortest distance along streets *etc.* By this way, the trips that require users to walk too far are pruned from the trip set H_i of each user.

4.3. Theoretical Performance

This part presents the theoretical performance of GTP and starts with the time complexity. Given that there are N bike stations and M users in the BSS, GTP first takes 1 step to initialize the set h. Constructing the trip set H takes MN^2 steps because the start station set and target station set of each user are traversed and the start station and target station of each user could be any bike station in the BSS. Then in each round, let m be the users that do not have a trip allocated. It takes mN^2 steps to find the maximum-quality trip in H because each of the m users has at most N^2 available trips and H has at most mN^2 trips. Adding the maximum-quality trip to h and updating the bike resources take two steps. Deleting trips from H takes mN^2 steps because H has to be traversed. Since only one trip is allocated in each round, the "while" loop costs $\sum_{m=0}^{M}(2mN^2+2)$ steps. To sum up, GTP costs $N^2M(M+1)+MN^2+2M+1$ steps. Therefore, the time complexity of GTP is $O((NM)^2)$.

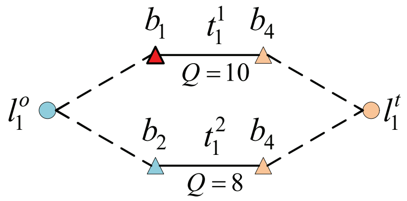
By applying the trip pruning, the trip set size of each user is reduced a lot. Let N_d be the maximum size of all start station sets and target station sets of all users after applying the trip prunning. The time complexity of GTP can thus be given by $O((N_dM)^2)$. Note that if d_{max} is set properly, N_d^2 can be far less than N^2. The time consumption of GTP can thus be reduced a lot.

As the trip planning problem is proved to be a weighted 3-set packing problem and the greedy approach of a weighted k-set packing problem is proved to be a k-approximation algorithm [25]. Therefore, the approximation ratio of GTP is $\frac{1}{3}$, which is the ratio between the result obtained by the algorithm and the optimal result.

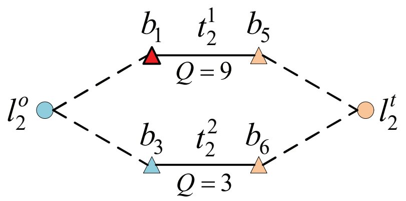
Theorem 2. *The approximation ratio of GTP is* $\frac{1}{3}$.

4.4. Local Optimality of GTP

When there are bike utilization conflicts in the BSS, GTP may incur the problem of only achieving local optimality. This part presents a simple example as shown in Fig. 5.



(a) Trip set H_1 of u_1



(b) Trip set H_2 of u_2

Figure 5: An example of local optimality of GTP

For simplicity, we set the example as follows. There are two users u_1 and u_2. Each user has two available trips, as t_1^1, t_1^2 of user u_1 and t_2^1, t_2^2 of user u_2 shown in Fig. 5(a) and 5(b) respectively. The trip quality of these trips are: $Q(t_1^1)$=10, $Q(t_1^2)$=8, $Q(t_2^1)$=9, $Q(t_2^2)$=3. All bike stations have enough bikes and docks except b_1, which has only one bike. As both u_1 and u_2 tend to choose their maximum-quality trips, *i.e.* u_1 tends to choose t_1^1 and u_2 tends to choose t_2^1, there is bike utilization conflict between u_1 and u_2 because of insufficient resources at bike station b_1.

By applying GTP, we have that t_1^1 is allocated to u_1, and t_2^2 is allocated to u_2. The total quality of the two trips is 13. Let's consider another way of trip allocation: t_1^2 is allocated to u_1 and t_2^1 is allocated to u_2. The total quality of the two allocated trips is 17. The new way of trip allocation achieves more total quality than that of GTP, which demonstrates the local optimality of GTP.

5. Humble Trip Planning

Since the bike utilization conflict is not fully considered in GTP, in this section, we present HTP to take the conflict into consideration.

5.1. Basic Idea

In the example presented in Section 4.4, when changing u_1's trip from t_1^1 to t_1^2, his trip quality is reduced by 2, and when changing u_2's trip from t_2^1 to t_2^2, his trip quality is reduced by 6. In this paper, the trip t_1^2 and t_2^2 are called the alternative trips and the quality reduction 2 and 6 are called the trip-changing costs of t_1^1 and t_2^1 respectively. Intuitively, it is harder to find an alternative trip for the trip which has higher trip-changing cost, so the trip should be allocated to its corresponding user first to achieve higher service quality of the BSS. By doing so, the trip t_2^1 is allocated to u_2 first, and the total quality of the BSS is 17, which is higher than that achieved by GTP.

Based on the above analysis, we present the basic idea of HTP. When there is no bike utilization conflict in the BSS, the maximum-quality trips of all users can be allocated directly. But when there are bike utilization conflicts, the maximum-quality trip which has higher trip-changing cost should be allocated first to its corresponding user so as to achieve higher service quality of the BSS.

A basic problem left is how to calculate the trip-changing cost. To do so, the alternative trip should be found in the first. In this paper, the alternative trip of a trip $t_i \in H_i$ is defined as the maximum-quality trip among all trips in H_i that do not contain the conflicting stations of t_i. For instance, in Fig. 6, all trips of user u_i is sorted descendingly by their trip quality from up to down, *i.e.* $Q(t_i^1) \geq Q(t_i^2) \geq Q(t_i^3)$. Bike station b_1 is a conflicting station in both trip t_i^1 and t_i^2. According to the definition, we have that the alternative trip of t_i^1 is t_i^3. After finding the alternative trip of t_i, its trip-changing cost can be calculated, which is defined as the difference between the trip quality of t_i and its alternative trip. If the alternative trip of t_i cannot be found, the trip-changing cost of t_i is set as $Q(t_i)$.

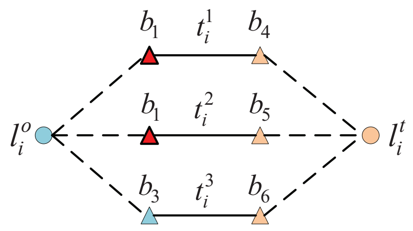


Figure 6: An example of alternative trip

5.2. HTP Algorithm

This part presents the design of HTP in detail, which eliminates bike utilization conflicts among users' maximum-quality trips in the first. That is, HTP first greedily and iteratively allocates a maximum-quality trip which has the maximum trip-changing cost among all conflicting trips to its corresponding user. After all conflicts been eliminated, the maximum-quality trips of the remaining users are allocated directly. HTP is summarized in Algorithm 2.

Algorithm 2 Humble Trip Planning Algorithm

```
Input: Bike station set: $B$; User set: $U$;
Output: Allocated trip set: $h'$;
 1: $U_p = U$, $h' = \emptyset$;
 2: Construct $H_i$ for each $u_i \in U_p$ and calculate all trips' quality;
 3: while $U_p \neq \emptyset$ do
 4:     Select the maximum-quality trip for each $u_i \in U_p$ from their $H_i$ to form a set $H'$;
 5:     Select all conflicting trips from $H'$ to form a set $H''$;
 6:     if $H'' \neq \emptyset$ then
 7:         Calculate the alternative trip and the trip-changing cost for $t_i \in H''$;
 8:         Find the trip $t_i^* = (l_i^o, b_i^o, b_i^t, l_i^t)$ which has the maximum trip-changing cost among
            all trips in $H''$;
 9:         $h' = h' \cup \{t_i^*\}$;
10:         Update bike resources at $b_i^o$ and $b_i^t$;
11:         Delete $t_i^*$'s corresponding user from $U_p$;
12:         Delete trips that do not have available bikes/docks from $H_i$ for each $u_i \in U_p$;
13:     else if $H'' = \emptyset$ then
14:         $h' = h' \cup H'$;
15:         Update bike resources at the start and terminal stations of each trip $t_i \in H'$;
16:         break;
17:     end if
18: end while
```

Specifically, HTP first initializes $U_p = U$ and $h' = \emptyset$. The set U_p consists of all users that no trip is allocated to them. Then HTP constructs the trip set H_i for each user u_i in U_p based on Eq. (2) and calculates all trips' quality based on Eq. (4). Next, while the user set U_p is not empty, HTP allocates trips round by round. In each round, HTP first select the maximum-quality trip for each user $u_i \in U_p$ from their corresponding trip sets H_i to form a set H'. Then HTP collects the bike resources demand at each bike station to compute the conflicting stations among all trips in H' and selects all conflicting trips from H' into a set H''. Then there are two cases: 1) If H'' is not empty, which means there are trips in conflict. To eliminate conflicts, HTP finds the alternative trip for each trip in H'' and calculate their trip-changing cost. It then finds a trip $t_i^* = (l_i^o, b_i^o, b_i^t, l_i^t)$ which has the maximum trip-changing cost among all trips in H'' and allocates it to its corresponding user by adding t_i^* to the allocated trip set h. Then the resources at the bike stations b_i^o and b_i^t are updated. Next, the trip t_i^*'s corresponding user is deleted from the set U_p, and all trips that do not have a bike at the start station or do not have an available dock at the target station are deleted from the trip set H_i of each user u_i in the set U_p. After that, if there are still some users in U_p, HTP goes into the next round. Otherwise, it outputs the allocated trip set h' and stops. 2) If H'' is empty, which means there is no conflict remaining in the BSS, all trips in H' can be added to h' directly. Then the resources at the start and terminal stations of each trip $t_i \in H'$ are updated. After that, HTP outputs the allocated trip set h' and stops.

Similar to GTP, the trip pruning presented in Section 4.2 can also be employed in HTP to reduce its complexity.

5.3. Theoretical Performance

Given that there are N bike stations and M users in the BSS, this part describes the time complexity of HTP. In HTP, it first takes 1 step to finish the initialization. Constructing the trip set H_i for each user in U_p and calculating all trips' quality take MN^2 steps. Then in each round, let m be the users that remain in U_p. Selecting the maximum-quality trip for each user in U_p takes mN^2 steps. Selecting all conflicting trips from H' takes $2m$ steps. The worst case of HTP is that all trips in H' are in conflict and only one trip can be allocated in each round, so we only consider the case $H'' \neq \emptyset$. Calculating the alternative trip and the trip-changing cost for each trip in H'' takes mN^2 steps because there are at most N^2 trips in H_i for each user $u_i \in U_p$. Then finding the trip t_i^* takes m steps. Adding the trip t_i^* to h' and updating the bike resources take two steps. Deleting t_i^*'s corresponding user from U_p takes m steps. And deleting the trips that do not have available bikes/docks from H_i for each $u_i \in U_p$ takes mN^2 steps. So the "while" loop costs $\sum_{m=0}^{M}(3mN^2 + 4m + 2)$ steps in total. To sum up, HTP costs $MN^2 + \frac{3}{2}M(M+1)N^2 + 2M(M+1) + 2M + 1$ steps. Therefore, we have that the time complexity of HTP is $O((NM)^2)$.

By applying the trip pruning presented in Section 4.2, the time consumption of HTP can be reduced a lot. Let N_d be the maximum size of all start station sets and target station sets of all users after applying the trip planning, the time complexity of HTP can be given by $O((N_dM)^2)$.

6. Experiment Evaluation

This section conducts extensive simulation to evaluate our algorithms based on real bike station dataset of Hangzhou Public Bicycle.

6.1. Methodology

Since there is few existing approach aiming at promoting BSS's service quality by trip planning, this section proposes a Random Trip Planning algorithm (RTP) as a benchmark. It describes the character of the bike utilization in daily life by randomly and iteratively selecting a user u_i in U, then allocating the maximum-quality trip in the trip set H_i to him.

In the experiment, several impact factors of the trip planning algorithms are considered, including the experiment region, the user amount and the maximal walking range.

Three regions in Hangzhou city of China are utilized as experiment regions, as the region A, B, C shown in Fig. 7. These regions locate in the central area of Hangzhou city, containing commercial centers, scenic zones and resident zones. The bike station dataset of Hangzhou Public Bicycle, one of the world's largest BSSs, is utilized in our experiment. The dataset contains ID, location, capacity, *etc.* of each bike station deployed in Hangzhou Public Bicycle. The total number and the total capacity of these bike stations in these

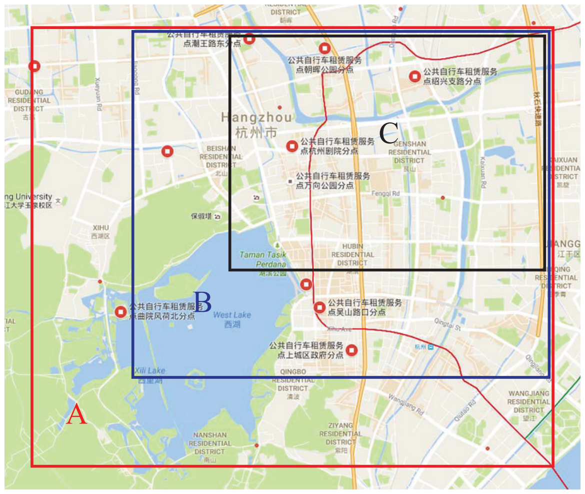


Figure 7: Experiment regions

experiment regions are presented in Table 2. In the experiment, the total number of bikes and available docks at each bike station are initialized in two ways: equally and randomly. Since bike resources can be used only once in this paper, the trips in the allocated trip set is bounded by the minor one of the total bikes and total docks in the experiment region.

Table 2: Bike stations in 3 experiment regions

Region	**Total stations**	**Total capacity**
A	645	14207
B	513	11260
C	307	6820

This paper considers different user amounts in the experiment, *i.e.* $|U| = 0.5$k, 1k, 2k, 3k, 5k, 7k, 10k, 15k, 20k, 25k, 30k, 35k, 40k, 45k, 50k. For all users, the start points and terminal points are randomly initialized in the experiment region. The walking speed of each user is set as $5km/h$ [26] and the riding speed of user is set as $20km/h$ [27].

To explore the impact of the maximal walking range of users, this paper considers 6 different maximal walking ranges in the experiment, *i.e.* $d_{max} = 100m$, $200m$, $300m$, $500m$, $1000m$, $1500m$. In the experiment, the haversine distance is adopted to measure the distance between any two locations.

For each specific setting of different region, user amount and maximal walking range, this paper repeatedly runs the simulation for 20 times and records the average results. The simulation is performed on a laptop with Intel(R) Core(TM) i7-3610QM 2.3GHz CPU and 8GB RAM.

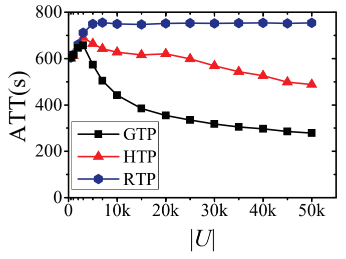
Metrics. Considering that the goal of the trip planning is to maximize the number of served users and minimize their trip time, this paper applies the following three metrics to evaluate the performance of the proposed algorithms.

- **Average trip time (ATT)**: It is the average trip time of all allocated trips.
- **Allocated trips (AT)**: It is the total number of allocated trips, which indicates the number of served users.
- **Trip allocation ratio (TAR)**: It is the ratio of users who have an allocated trip among all users and calculated by AT divided by $|U|$.

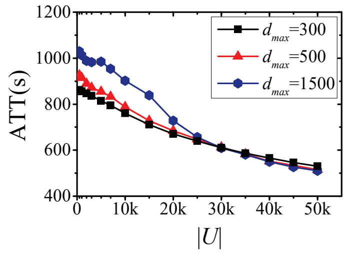
6.2. Simulation Results

The simulation results of equally and randomly initialized bike resources at each bike station are almost the same. We thus only present the results of the case that bike resources are initialized randomly.

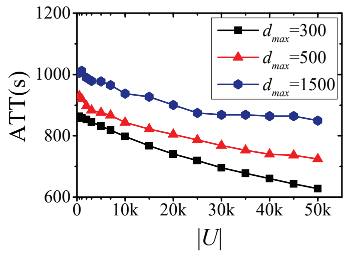
Average trip time. Fig. 8 shows the ATT of the three algorithms. The ATT of the three algorithms changing with $|U|$ in region C is shown in Fig. 8(a). When $|U|$ is small, as $|U| < 5$k shown in the figure, the ATT of each algorithm is almost the same and increases along with $|U|$. It is because some users have to walk and ride a little further to complete their trips when there are more users. When $|U|$ gets bigger, as $|U| \geq 5$k shown in the figure, the ATT of RTP almost keeps stables and is bigger than that of HTP and GTP. It is because the allocated trips of RTP are not intentionally selected and only reflect the baseline performance. Therefore, in the scope of ATT, GTP and HTP outperform RTP when $|U|$ is big enough. The ATT of HTP is bigger than that of GTP, which is because GTP always tries to allocates the maximum-quality trip among all trips of users first.



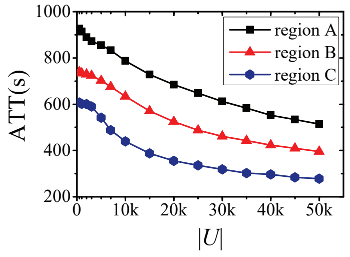
(a) Region C, $d_{max} = 1000$



(b) GTP, region A



(c) HTP, region A



(d) GTP, $d_{max} = 500m$

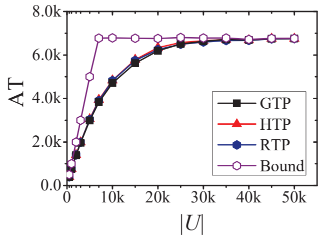
Figure 8: The ATT of GTP, HTP and RTP

Fig. 8(b) and Fig. 8(c) show the ATT of GTP and HTP changes along with $|U|$ under different d_{max} in region A. For the two algorithms, bigger d_{max} results in bigger ATT, which is because some users can access bike resources at further stations when d_{max} is bigger so that their trip time increases.

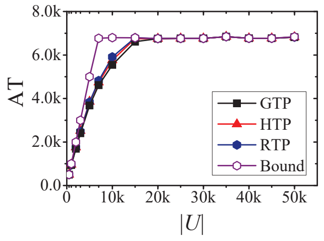
Fig. 8(d) shows the ATT of GTP changes along with $|U|$ in different regions when d_{max} is $500m$, where smaller region results in lower ATT, as the average distance between user's start point and terminal point is much smaller in the smaller region than that in the bigger region. HTP also has the same results.

Allocated trips. Fig. 9 plots the total AT of the three algorithms and the upper bound of AT. The total AT of HTP and RTP are almost the same and more than that of GTP. As RTP does not contribute much on reducing ATT and increasing the total number of AT, we can conclude that GTP and HTP outperform RTP. In the following content, we evaluate the performance of GTP and HTP.

Fig. 10 shows the difference of AT between HTP and GTP, *i.e.* the total AT of

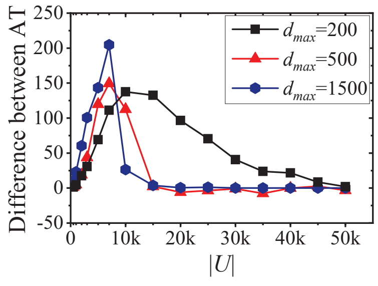


(a) $d_{max} = 500m$

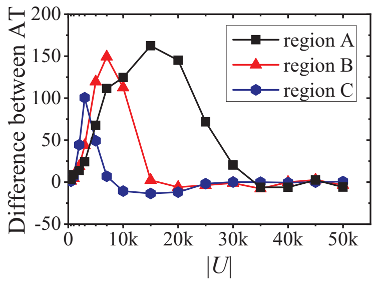


(b) $d_{max} = 1500m$

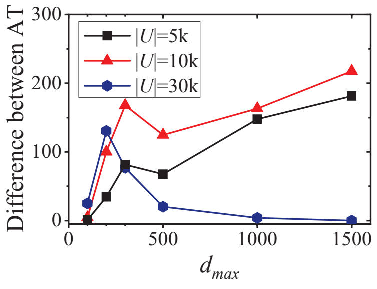
Figure 9: The total number of AT of 3 algorithms in region A



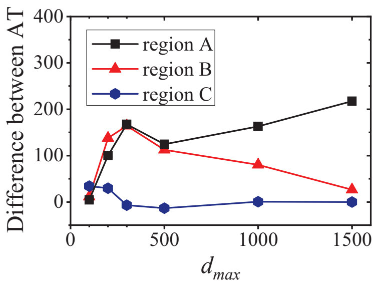
(a) Region B



(b) $d_{max} = 500m$

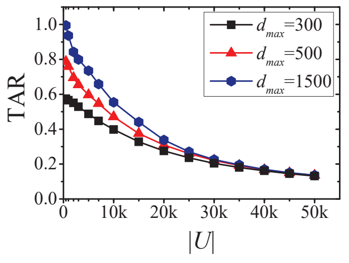


(c) Region A

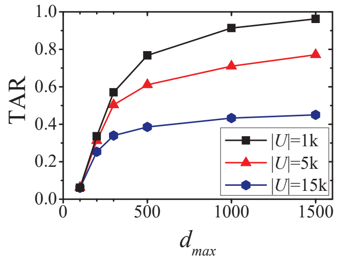


(d) $|U| = 15\text{k}$

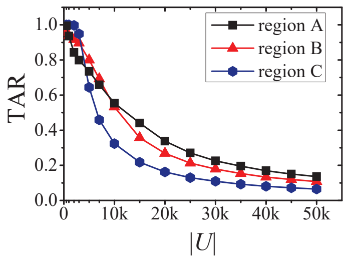
Figure 10: The difference of AT between HTP and GTP



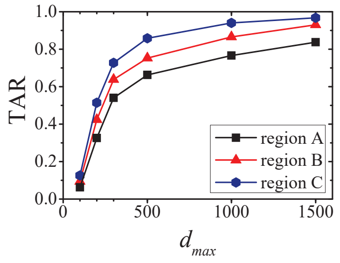
(a) GTP, region A



(b) HTP, region B



(c) GTP, $d_{max} = 1500m$



(d) HTP, $|U|$=3k

Figure 11: The TAR of GTP and HTP

HTP minus that of GTP. It shows that HTP can always allocate more trips than GTP. Fig. 10(a) and Fig. 10(b) plot the difference of AT between HTP and GTP changing with $|U|$ under different d_{max} and in different experiment regions. For a specific region and d_{max}, the difference between AT of HTP and GTP increases when $|U|$ is small. However, it decreases when there are more users. The difference between AT of HTP and GTP becomes 0 at last for there is an upper bound of AT. Fig. 10(c) and Fig. 10(d) show the impact of d_{max} on the difference of AT between HTP and GTP. For different $|U|$ and different regions, a very small d_{max} and a very big d_{max}, as $d_{max} < 100m$ and $d_{max} > 500m$ shown in the two figures, lead to bigger difference of AT between HTP and GTP. While a medium d_{max}, as $100m < d_{max} < 500m$ shown in the two figures, results in a smaller difference.

To conclude, we have that GTP and HTP perform better than RTP. When $|U| <$ 5k, HTP performs better than GTP. Because when $|U| <$ 5k, the ATT of GTP and HTP are almost the same, as shown in Fig. 8(a), and the AT of HTP is always higher than that of

GTP, as shown in Fig. 10(a), 10(b). However, when $|U| \geq 5$k, there is a tradeoff between GTP and HTP in maximizing the number of served users and minimizing their trip time, because the ATT of GTP is lower than that of HTP, while the AT of HTP is higher than that of GTP.

Trip allocation ratio. Fig. 11 shows how these factors affect the TAR of GTP and HTP. In this part, both GTP and HTP have similar results. As shown in Fig. 11(a), due to there is an upper bound of AT, the TAR decreases along with $|U|$. Fig. 11(b) shows that the TAR increases along with d_{max} for a specific $|U|$, which is because more available trips can be allocated to each user when d_{max} is bigger. Fig. 11(c) shows the TAR of GTP changes along with $|U|$ in different regions when d_{max} is $1500m$. At the beginning, smaller region results in higher TAR. With the increment of user amount, the TAR in smaller region becomes lower, which is because the AT in a small region reaches the upper bound first. Fig. 11(d) shows the TAR changes with d_{max} in different regions of HTP when $|U|$=3k, which confirms bigger d_{max} results in higher TAR.

7. Discussion

Trip planning in real scenario. In real scenario, there are many challenges to perform trip planning: bike resources can be used repeatedly, the bike utilization demand is not known and changes dynamically, the arriving time of users are unknown and the trip allocation is invariable. Our algorithms can be applied in real scenario to plan bike trips for users in every short time period, such as the system can plan trips for all users who request trip planning service in every minute. However, considering all the challenges in bike trip planning in real scenario, the decision model of users, waiting time of users and incentive mechanism could be considered in the future work.

Conflict measurement. Simulation results in this paper reveal the performance of GTP and HTP are related to user amount, user's maximal walking range and region size. Intuitively, these factors affect the bike utilization conflict in BSS. To our best knowledge, almost all existing works do not consider these impact factors together to provide efficient ways to measure bike utilization conflict in a BSS. We also leave it to our future work.

8. Related Work

Public transportation and bike-sharing system has attracted tremendous attention in recent years [28–33]. In this part, we review the state-of-the-art works related to BSS.

System prediction. Many works leverage machine learning techniques to estimate bike resources in the BSS. Andreas Kaltenbrunner *et al.* [4] predict the available bikes at each station some minutes/hours ahead by using the Auto-Regressive Moving Average (ARMA) model. Min Yang *et al.* [6] predict the number of available bikes at bike stations by an improved back propagation neural network. Bei Chen *et al.* [7] design a system to predict bike availability at bike stations, which takes into account exogenous variable, such as weather and time. The system also predicts how long a user has to wait for a bike when there's no bike available. Romain Giot *et al.* [8] predict the global bike utilization of the BSS by using various regressors from the state-of-the-art. Yexin Li *et al.* [9] propose a hierarchical prediction model to predict the check-out/check-in of each cluster in the BSS. Zidong Yang *et al.* [5] propose a spatio-temporal bicycle mobility model based on historical bike-sharing data, and design a traffic prediction mechanism on a per-station basis with sub-hour granularity. Some works predict the trips in the BSS. Jiawei Zhang *et al.* [34] design a model to predict trip destination and trip duration by utilizing a Multiple Additive Regression Tree and a Lasso regression model. Divya Singhvi *et al.* [35] predict pairwise bike demand for the Citi Bike system of New York City.

System operation. Another important topic is BSS operation. Benefited from the system prediction, many works are designed to help the BSS operate efficiently. Since bike utilizations cause imbalanced bike resources in the BSS, operator of the system has to redistribute these bikes by trucks. Tal Raviv *et al.* [11] plan routs for trucks to solve a static redistribution problem in the BSS. Adish Singla *et al.* [12] propose a schema to incentivize users to contribute to redistributing bikes. Julius Pfrommer *et al.* [13] design a hybrid solution for the dynamic bike redistribution problem, in which the routing of trucks and a dynamic incentive scheme are both considered. Some works provide real time trip planning for users, but they only consider the problem in single user scope. Ji Won Yoon *et al.* [19] mentioned to help users select the best pair of stations with the minimal time cost and the maximal probability to finish trip after giving the origin and destination. In [22], Agostino Nuzzolo *et al.* design a trip planning system which considers user's preference. In the BSS, mobile crowdsourcing techniques could be utilized to obtain user trip information and provide bike station information for users [36, 37].

System design. Some works make great contributions to BSS design. In [38], Longbiao Chen *et al.* leverage open data to predict bike utilization and recommend the optimal placement of bike stations. Jenn-Rong Lin *et al.* [39] develop a mathematical model to consider the interests of customers and investors when building the BSS. Juan Carlos García-Palomares *et al.* [40] design a GIS-based approach to locate stations based on a location-allocation model and determine the capacities of these stations by calculating the potential bike utilization demand.

k-set packing problem. The weighted k-set packing problem is usually solved through approximation algorithm. Barun Chandra uses greedy local improvement and achieves the approximation ratio of $3/2(k+1)$ [25]. Piotr Berman presents an algorithm to achieve $2/k$ approximation ratio [41]. However, these approaches are too complex to use. In this paper, we present algorithms that has the same complexity as the simple greedy algorithm to solve the bike trip planning problem.

Different from the existing works presented above, this paper addresses a trip planning problem which considers the system-wide resources and service quality of the BSS.

9. Conclusion

This paper addresses a trip planning problem, which considers the bike utilization conflict in BSS so as to maximize the number of served users and minimize their trip time. We formulate the trip planning problem as the well-known weighted k-set packing problem, which is NP-hard. In order to solve the problem, this paper designs two algorithms, namely

GTP and HTP. For comparison, this paper designs RTP as benchmark. We conduct extensive simulation based on real bike station dataset of Hangzhou Public Bicycle to evaluate our algorithms. Simulation results show that GTP and HTP outperform RTP. Meanwhile, the results reveal the impact of different experiment region, user amount and user's maximal walking range on the performance of GTP and HTP. Some future works may include conflict modeling and online trip planning.

10. Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grants No. 61473109, 61572164 and 61671193, Zhejiang Major Science and Technology Programme under Grant No. 2018C04012 , and the Graduate Scientific Research Foundation and the Excellent Dissertation Fostering Foundation of Hangzhou Dianzi University.

References

[1] Bicycle-sharing system, `https://en.wikipedia.org/wiki/Bicycle-sharing_system`, accessed November 21 2016.

[2] Last mile (transportation), `https://en.wikipedia.org/wiki/Last_mile_(transportation)`, accessed November 21 2016.

[3] P. Midgley, The role of smart bike-sharing systems in urban mobility, Journeys 2 (2009) 23–31.

[4] A. Kaltenbrunner, R. Meza, J. Grivolla, J. Codina, R. Banchs, Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system, Pervasive and Mobile Computing 6 (4) (2010) 455–466.

[5] Z. Yang, J. Hu, Y. Shu, P. Cheng, J. Chen, T. Moscibroda, Mobility modeling and prediction in bike-sharing systems, in: Proc. of ACM MobiSys, Singapore, 2016.

[6] M. Yang, X. Zhang, A novel travel adviser based on improved back-propagation neural network, in: Proc. of IEEE ISMS, Bangkok, Thailand, 2016, pp. 283–288.

[7] B. Chen, F. Pinelli, M. Sinn, A. Botea, F. Calabrese, Uncertainty in urban mobility: Predicting waiting times for shared bicycles and parking lots, in: Proc. of IEEE ITSC, Hague, Netherlands, 2013, pp. 53–58.

[8] R. Giot, R. Cherrier, Predicting bikeshare system usage up to one day ahead, in: Proc. of IEEE CIVTS, San Diego, CA, USA, 2014, pp. 22–29.

[9] Y. Li, Y. Zheng, H. Zhang, L. Chen, Traffic prediction in a bike-sharing system, in: Proc. of ACM SIGSPATIAL, Seattle, Washington, USA, 2015.

[10] P. Vogel, T. Greiser, D. C. Mattfeld, Understanding bike-sharing systems using data mining: Exploring activity patterns, Procedia-Social and Behavioral Sciences 20 (2011) 514–523.

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

[12] A. Singla, M. Santoni, G. Bartók, P. Mukerji, M. Meenen, A. Krause, Incentivizing users for balancing bike sharing systems, in: Proc. of AAAI, Austin, Texas, USA, 2015, pp. 723–729.

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

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

[15] S. Ghosh, P. Varakantham, Y. Adulyasak, P. Jaillet, Dynamic repositioning to reduce lost demand in bike sharing systems, Journal of Artificial Intelligence Research 58 (2017) 387–430.

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

[17] S. Ghosh, P. Varakantham, Incentivising the use of bike trailers for dynamic repositioning in bike sharing systems, in: Proc. of ICAPS, Pittsburg, PA, USA, 2017.

[18] E. O'Mahony, D. B. Shmoys, Data analysis and optimization for (citi) bike sharing., in: Proc. of AAAI, Austin, TX, USA, 2015, pp. 687–694.

[19] J. W. Yoon, F. Pinelli, F. Calabrese, Cityride: a predictive bike sharing journey advisor, in: Proc. of IEEE MDM, Bengaluru, India, 2012, pp. 306–311.

[20] M. Conti, A. Passarella, S. K. Das, The internet of people (iop): A new wave in pervasive mobile computing, Pervasive and Mobile Computing 41 (2017) 1–27.

[21] F. Delmastro, V. Arnaboldi, M. Conti, People-centric computing and communications in smart cities, IEEE Communications Magazine 54 (7) (2016) 122–128.

[22] A. Nuzzolo, A. Comi, Individual utility-based path suggestions in transit trip planners, IET Intelligent Transport Systems 10 (4) (2016) 219–226.

[23] S. Kaplan, F. Manca, T. A. S. Nielsen, C. G. Prato, Intentions to use bike-sharing for holiday cycling: An application of the theory of planned behavior, Tourism Management 47 (2015) 34–46.

[24] A. Faghih-Imani, N. Eluru, Analysing bicycle-sharing system user destination choice preferences: Chicago's divvy system, Journal of transport geography 44 (2015) 53–64.

[25] B. Chandra, M. M. Halldorsson, Greedy local improvement and weighted set packing approximation, Journal of Algorithms 39 (2) (2001) 223–240.

[26] R. C. Browning, E. A. Baker, J. A. Herron, R. Kram, Effects of obesity and sex on the energetic cost and preferred speed of walking, Journal of Applied Physiology 100 (2) (2006) 390–398.

[27] Bicycle performance, `https://en.wikipedia.org/wiki/Bicycle_performance`, accessed November 21 2016.

[28] P. DeMaio, Bike-sharing: History, impacts, models of provision, and future, Journal of Public Transportation 12 (4) (2009) 41–56.

[29] S. Shaheen, S. Guzman, H. Zhang, Bikesharing in europe, the americas, and asia: past, present, and future, Transportation Research Record: Journal of the Transportation Research Board (2143) (2010) 159–167.

[30] J. Zhang, P. Lu, Z. Li, J. Gan, Distributed trip selection game for public bike system with crowdsourcing, in: Proc. of IEEE INFOCOM, Honolulu, HI, USA, 2018, pp. 1–9.

[31] X. Kong, F. Xia, Z. Ning, A. Rahim, Y. Cai, Z. Gao, J. Ma, Mobility dataset generation for vehicular social networks based on floating car data, IEEE Transactions on Vehicular Technology 67 (5) (2018) 3874–3886.

[32] X. Kong, X. Song, F. Xia, H. Guo, J. Wang, A. Tolba, Lotad: long-term traffic anomaly detection based on crowdsourced bus trajectory data, World Wide Web 21 (3) (2018) 825–847.

[33] A. Rahim, X. Kong, F. Xia, Z. Ning, N. Ullah, J. Wang, S. K. Das, Vehicular social networks: A survey, Pervasive and Mobile Computing`doi:10.1016/j.pmcj.2017.12.004`.

[34] J. Zhang, X. Pan, M. Li, P. S. Yu, Bicycle-sharing system analysis and trip prediction, in: Proc. of IEEE MDM, Porto, Portugal, 2016.

[35] D. Singhvi, S. Singhvi, P. I. Frazier, S. G. Henderson, E. O' Mahony, D. B. Shmoys, D. B. Woodard, Predicting bike usage for new york citys bike sharing system, in: Proc. of AAAI Workshop on Computational Sustainability, Association for the Advancement of Artificial Intelligence, Austin, Texas, USA, 2015.

[36] Z. Wang, X. Pang, Y. Chen, H. Shao, Q. Wang, L. Wu, H. Chen, H. Qi, Privacy-preserving crowd-sourced statistical data publishing with an untrusted server, IEEE Transactions on Mobile Computingdoi:10.1109/TMC.2018.2861765.

[37] Z. Wang, J. Hu, R. Lv, J. Wei, Q. Wang, D. Yang, H. Qi, Personalized privacy-preserving task allocation for mobile crowdsensing, IEEE Transactions on Mobile Computingdoi:10.1109/TMC.2018.2861393.

[38] L. Chen, D. Zhang, G. Pan, X. Ma, D. Yang, K. Kushlev, W. Zhang, S. Li, Bike sharing station placement leveraging heterogeneous urban open data, in: Proc. of UBICOMP, Osaka, Japan, 2015, pp. 571–575.

[39] J.-R. Lin, T.-H. Yang, Strategic design of public bicycle sharing systems with service level constraints, Transportation research part E: logistics and transportation review 47 (2) (2011) 284–294.

[40] J. C. García-Palomares, J. Gutiérrez, M. Latorre, Optimizing the location of stations in bike-sharing programs: a gis approach, Applied Geography 35 (1) (2012) 235–246.

[41] P. Berman, A d/2 approximation for maximum weight independent set in d-claw free graphs, in: Scandinavian Workshop on Algorithm Theory, Springer, 2000, pp. 214–219.