2017 IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems

Large-Scale Trip Planning for Bike-Sharing Systems

Zhi Li, Jianhui Zhang*, Jiayu Gan, Pengqian Lu, Fei Lin
College of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018 China
*Corresponding author, Email: jhzhang@ieee.org

***Abstract*—In Bike-Sharing System (BSS), great efforts have been devoted to performing resources prediction, redistribution and trip planning to alleviate the unbalance of resources and inconvenience of bike utilization caused by the explosion of users. However, there is few work in trip planning noticing that the complete trip 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 in BSS by considering system-wide conflicts so as to achieve higher service quality of the system. The problem is formulated as the well-known weighted k-set packing problem. We design two algorithms, a Greedy Trip Planning algorithm (GTP) and a Humble Trip Planning algorithm (HTP), for the problem. 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 our algorithms.**

***Keywords*-Bike-Sharing System; Trip Planning; Complete Bike Trip; Service Quality; Conflict**

I. Introduction

Bike-Sharing System (BSS) is a convenient service deployed in many big cities to alleviate the last-mile problem [1]. More and more users choose to use bike-sharing service brings pressure to both user and BSS. Now, it is quite often that some users are not able to borrow or return bikes from or to stations because of the serious resources unbalance in the BSS, *i.e.* some stations have few bikes or docks. To relieve the pressure, the prediction is designed to estimate the resources [2]–[4] and the redistribution is adopted to alleviate the resources unbalance [5]–[7]. From the user's point of view, the station pair selection is proposed to help user borrow and return bike [8].

To improve the system utilization, it is necessary to alleviate the unbalance by redistributing bikes frequently in the BSS. Several works [5]–[7] 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. Some works leverage techniques such as machine learning and data mining to predict the available resources at each station [2]–[4]. However, the prediction is not very useful for each single user. For each specific user, the trip planning is more valuable since he can complete the trip by knowing exactly which stations to borrow or return bikes. Ji Won Yoon *et al.* mention helping 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 location [8]. Agostino Nuzzolo *et al.* design a trip planning system which considers user's preference [9]. However, they care only about single user rather than the system-wide bike-sharing service quality.

When user rides the bike in the BSS, the complete trip 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 the best of our knowledge, there is few work notices the trip composition and designs corresponding trip planning algorithm. To study the case, this paper addresses a static trip planning problem to maximize the number of served users and minimize their trip time, *i.e.* to achieve higher service quality of the BSS. This paper studies the case that all bike resources can be allocated only once and the arriving time of user is not considered. The trip planning problem is mapped to the *weighted k-set packing problem* [10], which is NP-hard. This paper designs two heuristic algorithms, a Greedy Trip Planning algorithm (GTP) and a Humble Trip Planning algorithm (HTP), to solve the problem. For comparison, this paper designs a Random Trip Planning algorithm (RTP). Extensive simulation is conducted based on the data of Hangzhou Public Bicycle. Simulation results show that GTP and HTP outperform RTP and reveal how the experiment region, the user amount and the user's maximal walking range impact our algorithms.

II. System Model

This paper considers the BSS with the bike station set B and the user set U, where $B = \{b_1, ..., b_N\}$ and $U = \{u_1, ..., u_M\}$. 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.

This paper considers each user u_i's trip is a complete bike utilization process with three segments. 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 the trip composition. 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): 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

2155-6814/17 $31.00 © 2017 IEEE
DOI 10.1109/MASS.2017.36

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.

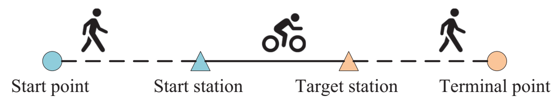


Fig. 1. The trip with three segments

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. The time is related to the length of each segment and its corresponding speed and given as the following definition.

Definition 2 (Trip Time): The 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. Thus, he has several available trips to reach his terminal point. These trips are included in a trip set.

Definition 3 (Trip Set): A trip set of user u_i, denoted by $H_i = \{t_i^1, ..., t_i^K\}$, 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 and we have

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

III. Problem Formulation

To improve the service quality of the BSS, this paper introduces the trip planning, which intends to allocate a trip to each user. This section first describes the trip allocation by considering the conflicts among users' trips, and formulates the trip planning problem and shows its hardness.

A. Trip Allocation

When there are limited bikes and docks at each bike station in the BSS, some users may not borrow or return bikes at their nearest stations, which is called *conflict* in this paper. In the BSS, there is conflict at a bike station b_i if it has insufficient bikes or docks to fulfill users' demand. If a trip contains a conflicting station, it is called a conflicting trip. Otherwise, it is a non-conflicting trip. For example, three trips, t_i of u_i, t_j of u_j and t_k of u_k, are shown in Fig. 2. All stations have enough bikes and docks except b_1 and b_5. b_1 has only one bike and b_5 has only one dock. Therefore, we have that b_1 and b_5 are two conflicting stations. t_i, t_j and t_k are three conflicting trips.

Considering there are conflicts in the BSS, trips in the BSS should be carefully allocated to users so as to achieve higher service quality of the system. We include all the allocated trips into an *allocated trip set*, denoted by h. Obviously, $h \subset H$ and h is not unique under different trip allocation strategy.

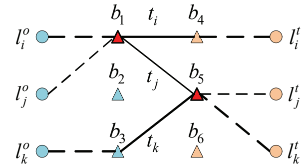


Fig. 2. Conflict

B. Trip Planning Problem

In this paper, allocating trips to users is called trip planning. Intuitively, the service quality of the BSS is related to the amount of users who complete their trips and their trip time. That is, to improve the service quality of the BSS, there should be more users who finish a trip and with less time consumption. To combine the two factors together, we define the trip quality as follows.

Definition 4 (Trip Quality): The quality $Q(t_i)$ of a trip t_i is inversely proportional to its trip time as the following equation.

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

According to the equation, it's easy to find that the trip quality can be maximized by minimizing the trip time. More short-time trips result in higher overall trip quality.

This paper considers the service quality of the BSS is the total quality of all allocated trips. If there is no allocated trip, the service quality is 0. The trip planning problem is then described as: Given the bike station set B and the user set U, 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.

C. Hardness of Trip Planning Problem

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

Proof: We prove the theorem by reducing the problem to the *weighted k-set packing problem* [10], which is described as follows: Given a collection of sets $\mathcal{I} = \{I_1, ..., I_n\}$, each of which has an associated weight and contains at most k elements drawn from a finite basic set G, find a collection with disjoint sets of maximum total weight.

For user u_i's trip $t_i = (l_i^o, b_i^o, b_i^t, l_i^t)$, he borrows one bike from station b_i^o, and consume one dock at 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 its weight be $Q(t_i)$. Let $\mathcal{I} = \{I_1, ..., I_n\}$ be the collection of all possible I_i and

G be the set of all users, bikes and docks. We have that $\mathcal{I}$ is drawn from G. By this way, the trip planning problem can be reduced to a maximum weighted 3-set packing problem, which is known to be NP-hard [10]. ∎

IV. Greedy Trip Planning

A. GTP Algorithm

To maximize the total quality of 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 in H_i should be deleted from H because he needs only to complete one trip. 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;

```
h = ∅;
Construct the trip set H and calculate all trips' quality;
while H ≠ ∅ do
    Find the maximum-quality trip t_i^* = (l_i^o, b_i^o, b_i^t, l_i^t) in H;
    h = h ∪ {t_i^*};
    Update bike resources at b_i^o and b_i^t;
    Delete H_i of t_i^*'s corresponding user u_i from H;
end while
```

B. Trip Pruning

In reality, users may not be willing to walk too far to borrow bikes or to their terminal points after returning bikes. Therefore, trips that contain such stations can be pruned from the trip set of each user by setting a maximal walking range, denoted by d_{max}, to reduce the complexity of the trip planning. The station set B_i^o and B_i^t of user u_i can thus be calculated by the following two equations.

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

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

C. Theoretical Performance

Given the N bike stations and the M users in the BSS, we describe the time complexity of GTP. GTP first takes 1 step to initialize the set h. Constructing the trip set H takes MN^2 steps. Then in each round, let m be the users that do not have a trip allocated. So it takes mN^2 steps to find the maximum-quality trip in H. Adding the maximum-quality trip to h and updating the bike resources take two steps. Deleting trips from H takes mN^2 steps. Because only one trip is allocated in each round, the "while" loop costs $\sum_{m=0}^{M}(2mN^2+2)$ steps. To sum up, the total cost of GTP is given by $O((NM)^2)$. So we have the following theorem.

Theorem 2: The time complexity of GTP is $O((NM)^2)$.

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

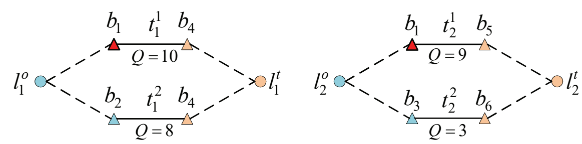
Then we present the approximation ratio of GTP.

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

Proof: 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 [10]. Thus, the approximation ratio of GTP is $\frac{1}{3}$. ∎

D. Local Optimality of Greedy Algorithm

This part presents an example of the local optimality of GTP. As shown in Fig. 3, there are two users u_1 and u_2 and each has two available trips, *i.e.* t_1^1, t_1^2 of user u_1 and t_2^1, t_2^2 of user u_2. 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. By applying GTP, t_1^1 and t_2^2 are allocated to u_1 and u_2 respectively. The total trip quality is 13. However, when t_1^2 and t_2^1 are allocated to u_1 and u_2 respectively, the total trip quality is 17.



(a) Trip set H_1 of u_1 (b) Trip set H_2 of u_2

Fig. 3. An example of local optimality of GTP

V. Humble Trip Planning

In this section, we present HTP to solve the trip planning problem and overcome the local optimality of GTP.

A. Basic Idea

In the example presented in Section IV-D, when changing u_1's trip from t_1^1 to t_1^2, his trip quality is reduced by 2. In this paper, t_1^2 is called the alternative trip and the quality reduction 2 is called the trip-changing costs for t_1^1. Thus, the trip-changing cost for t_1^1 and t_2^1 are 2 and 6 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 first to achieve higher service quality of the BSS. By doing so, t_2^1 is allocated to u_2, and the total quality of the BSS is 17, higher than that achieved by GTP.

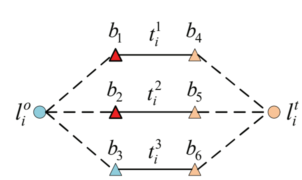


Fig. 4. Alternative trip

This paper defines the alternative trip of trip $t_i \in H_i$ 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. 4, there are three trips of u_i and $Q(t_i^1) > Q(t_i^2) > Q(t_i^3)$. b_1 is a conflicting station of t_i^1 and in t_i^2. The alternative trip of t_i^1 is t_i^3. In this paper, if the alternative trip of t_i exists, t_i's trip-changing cost is defined as the difference between the trip quality of t_i and its alternative trip. Otherwise, t_i's trip-changing cost is set as $Q(t_i)$.

B. HTP Algorithm

HTP first eliminates conflicts among users' maximum-quality trips by greedily and iteratively allocating a maximum-quality trip which has the maximum trip-changing cost 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';

$U_p = U$, $h' = \emptyset$;
Construct H_i for each $u_i \in U_p$ and calculate all trips' quality;
while $U_p \neq \emptyset$ **do**
  Select the maximum-quality trip for each $u_i \in U_p$ from their H_i to form a set H';
  Select all conflicting trips from H' to form a set H'';
  if $H'' \neq \emptyset$ **then**
    Calculate the alternative trip and the trip-changing cost for $t_i \in H''$;
    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'';
    $h' = h' \cup \{t_i^*\}$;
    Update bike resources at b_i^o and b_i^t;
    Delete t_i^*'s corresponding user from U_p;
  else if $H'' = \emptyset$ **then**
    $h' = h' \cup H'$;
    break;
  end if
end while

Similar to GTP, the trip pruning presented in Section IV-B can also be adopted in HTP.

C. Theoretical Performance

Given the N bike stations and the M users in the BSS, we describe the time complexity of HTP. In HTP, the initialization takes 1 step. 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 $|U_p| = m$. 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 $H'' \neq \emptyset$. Calculating the alternative trip and the trip-changing cost for each trip in H'' takes mN^2 steps. 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. So the "while" loop costs $\sum_{m=0}^{M}(2mN^2+4m+2)$ steps in total. To sum up, the total cost of HTP is given by $O((NM)^2)$. So we have the following theorem.

Theorem 4: The time complexity of HTP is $O((NM)^2)$.

Let N_d be the maximum size of all start station sets and target station sets after applying the trip pruning. The total cost of HTP can be given by $O((N_d M)^2)$.

VI. Experiment Evaluation

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

A. Methodology

This paper proposes a Random Trip Planning algorithm (RTP), which describes the character of the bike utilization in daily life, to compare with GTP and HTP algorithms. It randomly and iteratively selects a user u_i in U, then allocates the maximum-quality trip in the trip set H_i to him.

In the experiment, we consider the impact of the experiment region, the user amount and the maximal walking range. The detailed settings of the experiment are summarized in Table I.

TABLE I
Experiment setting

Factor	Setting
Experiment region	Region A, B, C
$\lvert U\rvert$	0.5k, 1k, 2k, 3k, 5k, 7k, 10k, 15k, 20k, 25k, 30k, 35k, 40k, 45k, 50k
d_{max}	$100m$, $200m$, $300m$, $500m$, $1000m$, $1500m$
Walking speed	$5km/h$
Riding speed	$20km/h$

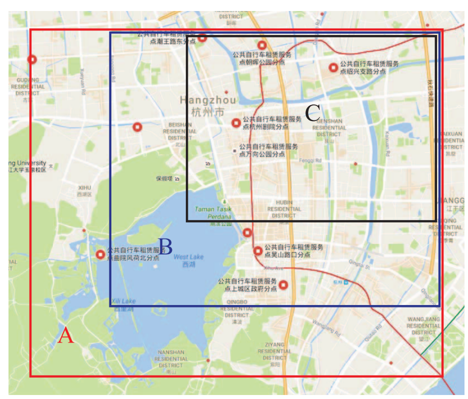


Fig. 5. Experiment regions

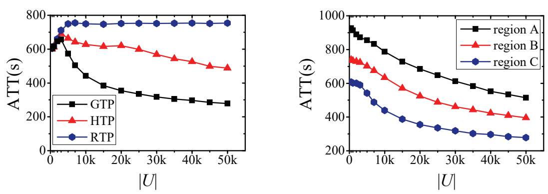
The three regions A, B and C are selected from the central area of Hangzhou city, as shown in Fig. 5. The bike station dataset of Hangzhou Public Bicycle is utilized in the experiment and the amount of bikes and docks at each bike station are initialized randomly. Since bike resources can be used only once in this paper, the total allocated trips is bounded by the minor one of the total bikes and docks in each region.

For each specific setting of different factors, this paper runs the simulation for 20 times and records the average results.

B. Simulation Results

We evaluate our algorithms by the Average Trip Time (ATT) of all allocated trips and the number of Allocated Trips (AT).

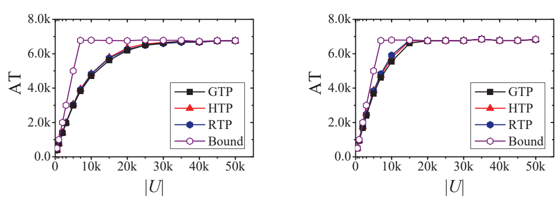
Average trip time. Fig. 6 shows the ATT of the three algorithms. Fig. 6(a) shows that when $|U|$ is small, the ATT of each algorithm is almost the same and increases along with $|U|$. While the ATT of RTP almost keeps stable and is higher than that of HTP and GTP when there are more users. Therefore, in the scope of ATT, GTP and HTP outperform RTP. It is because the allocated trips of RTP are not intentionally selected and only reflect the baseline performance. The ATT of HTP is higher than that of GTP, which is because GTP always tries to allocate the maximum-quality trip first. Fig. 6(b) shows the ATT of GTP changes along with $|U|$ in different regions when d_{max} is $500m$, where smaller region results in lower ATT, which is mainly because the average distance between user's start and terminal points is much smaller in the smaller region. The same goes for HTP.



(a) Region C, $d_{max} = 1000$ (b) GTP, $d_{max} = 500m$

Fig. 6. The ATT of GTP, HTP and RTP

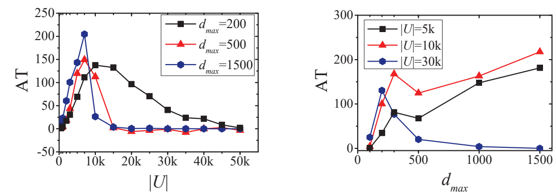
Allocated trips. Fig. 7 shows the total AT of HTP and RTP are almost the same and more than that of GTP. As RTP does not contribute much on reducing the ATT and increasing the AT, we can conclude that GTP and HTP outperform RTP.



(a) $d_{max} = 500m$ (b) $d_{max} = 1500m$

Fig. 7. The total number of AT of 3 algorithms in region A

Fig. 8 shows the difference of AT between HTP and GTP, *i.e.* the total AT of HTP minus that of GTP, and we have that HTP can always allocate more trips than GTP. Fig. 8(a) shows that for a specific region and d_{max}, the difference between AT of HTP and GTP increases when $|U|$ is small, while it decreases when $|U|$ increases. The difference becomes 0 at last for there is an upper bound of AT. Fig. 8(b) shows the impact of d_{max} on the difference of AT between HTP and GTP, which suggests that when $100m \leq d_{max} \leq 300m$, HTP



(a) Region B (b) Region A

Fig. 8. The difference of AT between HTP and GTP

achieves the most difference of AT towards GTP. To conclude, in the scope of AT, HTP performs better than GTP.

VII. Conclusion

This paper addresses a static trip planning problem, which considers system-wide conflicts in BSS so as to maximize the number of users served by the system and minimize their trip time. We formulate the problem as the well-known weighted k-set packing problem and design three algorithms, namely GTP, HTP and RTP. Extensive simulation is conducted based on dataset of Hangzhou Public Bicycle. Simulation results show that GTP and HTP outperform RTP and reveal the impact of different factors on GTP and HTP. Some future works may include conflict modeling and online trip planning.

VIII. Acknowledgements

This work is under the support of General Program of the National Natural Science Foundation of China under Grants No. 61473109, 61572164 and 61602141 and the Public Welfare Technology Application Research Program of Zhejiang Province under Grant No. 2015C33067.

References

[1] P. Midgley, "The role of smart bike-sharing systems in urban mobility," *Journeys*, vol. 2, pp. 23–31, 2009.

[2] Z. Yang, J. Hu, Y. Shu, P. Cheng, J. Chen, and T. Moscibroda, "Mobility modeling and prediction in bike-sharing systems," in *Proc. of ACM MobiSys*, Singapore, Jun 26-30 2016.

[3] A. Kaltenbrunner, R. Meza, J. Grivolla, J. Codina, and R. Banchs, "Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system," *Pervasive and Mobile Computing*, vol. 6, no. 4, pp. 455–466, 2010.

[4] H. Z. L. C. Yexin Li, Yu Zheng, "Traffic prediction in a bike-sharing system," in *Proc. of ACM SIGSPATIAL*, Seattle, WA, USA, Nov 3-6 2015.

[5] A. Singla, M. Santoni, G. Bartók, P. Mukerji, M. Meenen, and A. Krause, "Incentivizing users for balancing bike sharing systems," in *AAAI*, Austin, TX, USA, Jan 25-29 2015, pp. 723–729.

[6] J. Pfrommer, J. Warrington, G. Schildbach, and M. Morari, "Dynamic vehicle redistribution and online price incentives in shared mobility systems," *IEEE Transactions on ITS*, vol. 15, no. 4, pp. 1567–1578, 2014.

[7] J. Schuijbroek, R. Hampshire, and W.-J. van Hoeve, "Inventory rebalancing and vehicle routing in bike sharing systems," 2013.

[8] J. W. Yoon, F. Pinelli, and F. Calabrese, "Cityride: a predictive bike sharing journey advisor," in *IEEE MDM*, Bengaluru, India, Jul 23-26 2012, pp. 306–311.

[9] A. Nuzzolo and A. Comi, "Individual utility-based path suggestions in transit trip planners," *IET Intelligent Transport Systems*, vol. 10, no. 4, pp. 219–226, 2016.

[10] B. Chandra and M. M. Halldorsson, "Greedy local improvement and weighted set packing approximation," *Journal of Algorithms*, vol. 39, no. 2, pp. 223–240, 2001.