Rechargeable Battery Cabinet Deployment for Public Bike System

Jianhui Zhang, *Member, IEEE*, Wanqing Zhang, Jiacheng Wang, Jianwen Feng, Zhigang Gao, and Siwen Zheng

***Abstract*— Public Bike Systems (PBSs) offer the popular service for the short distance in daily life. The battery powered bike is an interesting and feasible method to extend the bike trip length, which can promote the PBS service but faces the challenges caused by the limited budget for the battery cabinet deployment and user demand. Thus, the realistic problem is how to deploy the cabinets near a part of public bike stations by considering the challenges. This paper is the first to study the novel problem, Cabinet Deployment Problem (CDP) in PBS, based on the features extracted from the real dataset of PBS in Hangzhou China, and proposes our strategies in the case of the Euclidean space and Manhattan model. In the Euclidean space, CDP can be specified as the *e*lectric-bike Set Cover problem (*e*-SC), and this paper proposes a Greedy Station Coverage algorithm (GSC). Its distributed version, called the Localized Greedy Selection algorithm (LGS), is also presented because of the large amount of bike stations. In many cities, the roads have Manhattan-type directions, *i.e.*, either east-west or south-north. In order to close to the realistic scenario, this paper develops a Genetic Algorithm based Cabinet Search algorithm (GAS) to determine the locations for the cabinet deployment in the Manhattan model. The extensive numerical experiment is conducted for our strategies, which are compared to a straightforward method, the Random Placement Strategy (RPS) under the diverse parameter settings.**

***Index Terms*— Public bike system, battery cabinet placement, sensor network optimization, city voronoi diagram.**

I. Introduction

As A green traveling manner of short trip, the Public Bike Systems (PBSs) have been prevailing around the whole world, such as Hangzhou in China and Chicago in USA because it can provide the extensive bike leasing service with the omnipresent bike stations [1], [2]. It bridges the trip length gap among the long-distance transport modes such as subway and bus, and is an elegant and environment-friendly way to solve the first-and-last mile transportation problem [3], [4]. There are numerous excellent research works on PBS including user quantity prediction, bike trip planning and so on [5]–[7]. One of the main research direction focuses on the length estimation for bike trip, one key performance metric of PBS. The statistics shows that 80% of bike trips in Hangzhou are less than 3 *km* [8] while the average length of the taxi trip is 8.86 *km* according to the historical data statistic of the taxi system in Shanghai, a city quite closed to Hangzhou [9]. The primary reason causing the bike trip much shorter than taxi is the limited human physical strength so that there is a big gap between them [10]. Meanwhile, the public bike is charge-free in the first rent duration, such as the first half or one hour, and its final charge is much lower than taxis and subway either. Therefore, it can improve the service ability of the public transportation, and attract much more attention than before by extending the bike trip length.

(a) *e*-bike station (b) *e*-bike battery

Fig. 1. The cabinet with the rechargeable *e*-bike battery.

One interesting way to extend trip length is to power the bike with rechargeable battery. For example, the PBS of Hangzhou China is going to build the *e*-bike station system by adding the rechargeable battery cabinets. Fig. 1 shows a demonstration *e*-bike station in Hangzhou. Each cabinet has thirty battery slots on average for the service of the rechargeable battery rental and charging as shown in Fig. 1. The PBS station equipped with the cabinet becomes an *e*-bike station and can support the above service for the *e*-bike, which has the battery slot under its basket. User can rent *e*-bike and realize the electric-aided ridding to extend the trip length easily by relaxing the human physical strength.

It's quite expensive to build one cabinet near each bike station throughout the whole PBS. For example, there are over 1700 bike stations around the whole Hangzhou city and each cabinet costs 1 billion RMB because of the land rental and

Manuscript received February 6, 2021; revised November 18, 2021 and March 21, 2022; accepted May 9, 2022. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFC3320301, in part by the National Natural Science Foundation of China under Grant 61877015, and in part by the Science and Technology Program of Zhejiang Province under Grant 2018C04012. The Associate Editor for this article was A. Hajbabaie. *(Jianhui Zhang and Wanqing Zhang are co-first authors.) (Corresponding author: Zhigang Gao.)*

Jianhui Zhang, Wanqing Zhang, Jiacheng Wang, Jianwen Feng, Zhigang Gao are with the Department of Computer and Science, Hangzhou Dianzi University, Hangzhou 310018, China (e-mail: jh_zhang@hdu.edu.cn; wanqing@hdu.edu.cn; jcwang0717@163.com; fengjianwen@hdu.edu.cn; gaozhigang@hdu.edu.cn).

Siwen Zheng is with Huawei Technologies Co., Ltd, Hangzhou 311100, China (e-mail: siwenzheng@outlook.com).

Digital Object Identifier 10.1109/TITS.2022.3180079

hardware. It also need satisfy some demands when deploying the cabinets. 1) Cost reduction. The cabinet construction is much expensive including the land leasing fee, its manufacture and maintaining cost. 2) Quality of Service (QoS). User demand for cabinet contains triple folds: convenient rent and returning, the trip length extension and widely distributed locations to access cabinet easily.

To satisfy the above demands, this paper condenses them as the Cabinet Deployment Problem (CDP) that is how to minimize the cost of cabinet deployment while meeting the user demand, represented by QoS. Although the solution to CDP is to select a part of bike stations to deploy the cabinets, it cannot be directly formulated as the typical set cover problem [11] or its applications since each station has no clear coverage radius or area. Meanwhile, it should take the real factors, such as traffic, road geometric character and user behavior, into account. It needs new effort to solve the problem since it's a different form of the typical set cover problem.

This paper formulates the CDP as the optimization problem and presents our cabinet deployment strategies in two scenarios: the Euclidean space and Manhattan-type city. In the former, there is no geometric constraint, such as street and building, on the bike trip while the later has such constraint like the streets in Manhattan USA, where all of them are along either south-north or east-west. This paper analyzes the historical record data of Hangzhou PBS to extract some specific features on the station coverage, and specifies the CDP as the *e*lectric-bike Set Cover problem (*e*-SC) and Position Selection problem (PSP) with the goal to minimize the deployment cost while ensuring the QoS of placement strategy to satisfy the user demand. This paper first selects the candidate bike stations based on the Density based Station Clustering algorithm (DSC), and then proposes Greedy Station Coverage algorithm (GSC) and Distributed Greedy Selection algorithm (LGS) to solve the *e*-SC and PSP to determine the final cabinet deployment. In the Manhattan-type city, this paper adopts the Manhattan voronoi model to estimate the QoS so that the QoS can be more close to the real city environment than the Euclidean space, and proposes the GAS based on the genetic algorithm.

The main contributions of this paper are summarized as follows:

- This paper is the first to study the realistic CDP with the demonstration cabinet in Hangzhou as example, and analyzes the real dataset from the PBS of Hangzhou City to extract the features for the cabinet deployment.
- This paper studies the CDP in the Euclidean space and Manhattan-type city. The study for the former one allows us to illustrate the impact of the PBS features on the cabinet deployment clearly. We transform the CDP to the *e*-SC and PSP problems and propose the GSC and LGS algorithms. In the later one, we adopt the Manhattan model to measure the QoS more accurate than that in the former one, and proposes the GAS based on the genetic algorithm.
- Extensive numerical experiments are executed for the performance comparison among our strategies under the different metrics and settings.

TABLE I
SYMBOL AND MEANING

Sym.	Description	Sym.	Description
v	Station	$\mathcal{R}$	Deployment region
V	Station set	δ	Deployment cost
r	Rank	γ	Cover radius
E	Edge set	α	Resource balance deviation
G	Graph	β	Bike rental frequency
d	Distance	N	# of stations
L	# of bikes	f	QoS function
K	# of berths	S	Set of neighboring station
M	Candidate set	$\mathscr{D}$	Dominance region
c	Cluster	F	Set of gene
η	QoS demand	q	Service coverage value
θ, I	Parameters	P	Set of deployed cabinet
b	Station feature	B	Set of station features
λ	Threshold	H	Set of nodes in voronoi graph

TABLE II
INITIALISM AND FULL NAME

Initialism	Full name
PBS	Public Bike System
CDP	Cabinet Deployment Problem
e-SC	*e*lectric-bike Set Cover Problem
PSP	Position Selection Problem
QoS	Quality of Service
GSC	Greedy Station Coverage algorithm
LGS	Localized Greedy Selection algorithm
GAS	Genetic Algorithm based Cabinet Search algorithm
DSC	Density based Station Clustering algorithm
RPS	Random Placement Strategy
CG	Column Generation

A. Road Map

The rest of this paper is organized as follows: Section II reviews the related literatures. Section III shows the models and problem formulation. The strategy for the CDP in the Euclidean space is presented in Section IV while that for the Manhattan model in Section V. The numerical experiments are conducted based on the real data in Section VI. Section VII concludes this whole paper.

Most symbols used in this paper are summarized in Table I and the full name corresponding to the initialisms in Table II.

II. RELATED WORKS

This section summarizes the related works in three main topics: 1) PBS dataset analysis, 2) Electric vehicles charging station deployment and 3) Set coverage.

A. PBS Dataset Analysis

Zhang and Mi analyzed the impacts of the public bikes on energy utilization and carbon dioxide emissions in Shanghai from the spatio-temporal perspective [12]. O'Mahony and Shmoys provided a careful analysis of system utilization and gave the novel problem formulation, which was motivated by a close collaboration with the bike share system of New York City [13]. Bordagaray *et al.* took advantage of the QoS of data to analyze the bike usage casuistry within a sharing scheme and proposed an original offline data mining procedure [14]. Dabiri *et al.* examined the problem of finding the optimal

speed trajectory for a cyclist in signalised urban areas, and proposed the method to minimize the total travel time, the energy consumption and the possibility to stop at the red light [15]. There has no research works on the rechargeable battery cabinet deployment in PBS and the existing works promote us to analyze the real dataset to find solution for the CDP.

B. Electric Vehicles Charging Station Deployment

There are massive researches on the electric vehicles charging station deployment [16]–[18]. For example, Hess *et al.* presented a model for electric vehicles and their battery depletion, vehicle mobility and gave a solution for optimal placement of charging stations. Li *et al.* developed a multi-period multi-path refueling location model to capture the dynamics in the topological structure of network and determined the cost-effective stations roll out scheme on both spatial and temporal dimensions [17]. Mehar and Senouci considered in several realistic constraints and proposed a mathematical formulation of the problem [18]. He *et al.* Provides an equilibrium modeling framework that can determine the optimal allocation among metropolitan areas to maximize social welfare associated with the coupled networks [19]. Sina and Babu proposes a two-stage stochastic programming model to determine the optimal network of charging stations [20]. He and Zhou explores how to optimally locate public charging stations for electric vehicles on a road network [21]. By the optimized genetic algorithm, it calculated the best position to locate them in order to satisfy the clients demand. The electric vehicles charging station deployment has the difference with the CDP since the bike should be returned to the station PBS.

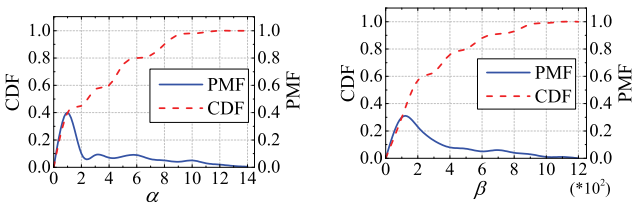
C. Coverage Problem

The coverage problem has been widely applied and studied, such as the sensor coverage problem in wireless sensor networks [22], [23]. Yadav *et al.* proposed a modified artificial bee colony algorithm for the mobile sensors deployment, with the aim of increasing the coverage area of the network in turn improving the performance of the network [22]. Alduraibi *et al.* presented three novel node placement optimization models for handling popular network design objectives [23]. In the typical application or research of the set coverage problem, the sensor range is modeled as a perfect circle or an irregular area and the coverage of the selected sensors is computed as the QoS. Different from the typical set coverage problem, the coverage range cannot be set like above in this paper, and is depends on the bike rental frequency, and city geometric structure and the balance between the bike rental and return.

From the above related works, we can find that the above deployment strategies cannot be directly applied to the cabinet deployment in the PBS.

III. System Model and Problem Formulation

This section extracts the bike station features from the real dataset, and then presents the system model and formulates the CDP problem.



(a) CDF and PMF of α (b) CDF and PMF of β

Fig. 2. Station features distribution.

A. Station Feature Extraction

This paper adopts the real dataset of the Hangzhou PBS which contains several information items: the station ID, bike ID, rental and returning time, recorded from April 1 to June 30 in 2016 of Hangzhou. From the rental and returning station ID, we find that 95% bike trips lasts less than 5 km and thus define a large cover radius γ for each station, which is enough to cover most bike trip started from themselves. For each single bike station v_i, this paper extracts the feature, denoted by $b_i(\alpha_i, \beta_i)$ from the real datasets, to depict the following information: (1) Resource balance deviation, denoted by α, *i.e.*, the ratio of the available numbers of bikes to empty berths, which reflects the ability of a bike station to support bike rental and returning. (2) User rental frequency, denoted by β obtained by the existing methods [24]–[26], which reflects the actual utilization of the station.

1) Resource Balance Deviation: We define the ratio function of the bikes to berths to calculate the deviation of the resources by the following equation.

$$\alpha_i^t = \begin{cases} \dfrac{L_i^t}{K_i^t} & \text{if } K_i^t > 0, \forall i \in V \\ L_i^m & \text{if } K_i^t = 0, \forall i \in V \end{cases}$$

where L_i^t (K_i^t) is the number of bikes (berths) at the station v_i at time duration t and L_i^m is the maximum number of bikes at v_i. V is the set of all the bike stations.

When $L_i^t = K_i^t$, the station v_i can offer equal services for the bike rental and returning. Otherwise, one of the two service is higher than the other. The resource balance deviation can measure the balance degree of the two service as the standard deviation in the following equation:

$$\alpha_i = \sqrt{\sum_{d=1}^{D}\sum_{t=1}^{T}(\overline{\alpha_i^t} - \alpha_i^t)^2/(D \cdot T)} \tag{1}$$

T is the 24 hours and divided into some equal time durations t, and D is the amount of all days. $\overline{\alpha_i^t}$ is the average of α_i^t over all time slots and days at v_i.

2) User Rental Frequency: Through the statistics for the real dataset of PBS, we get the user rental frequency β_i of station v_i by calculating the average user rental frequency in different days. β_i is calculated by the following equation:

$$\beta_i = \frac{\sum_{d=1}^{D}\sum_{t=1}^{T}\beta_i^t}{D \cdot T}$$

According to the statistics of the dataset of Hangzhou PBS, the user rental frequency of all bike stations ranges between $[0/day, 1102/day]$, and the resource balance deviation of all

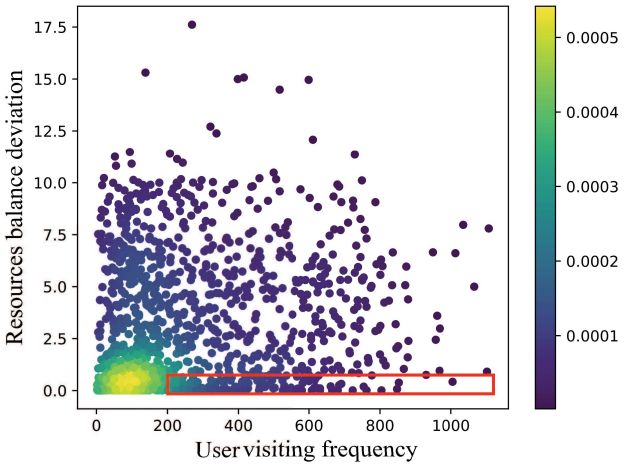


Fig. 3. Density scatter distribution of the α and β.

bike stations ranges between [0, 17]. Fig. 2 presents the Probability Mass Function (PMF) and the Cumulative Distribution Function (CDF) of two features. The stations with $\alpha < 1$ take up at most 40% as shown in Fig. 2(a). In Fig. 2(b), the stations with $\beta > 200/day$ take up less than 48%. It's better to select the bike stations with the high bike rental frequency. With the two features, we can observe that the stations with high user rental frequency, such as $\beta > 200/day$ and the low resource balance deviation, take up about 10% as shown in the red frame of Fig. 3. So the two features can be applied to design the solution to the CDP.

B. System Model

Each bike station has a fixed positions denoted by $v_i = (x_i, y_i)$. Let V denote the set of all bike stations and their positions, *i.e.*, $V = \{v_i = (x_i, y_i), i = 1 \dots N\}$, where N is the number of all the bike stations in PBS. By connecting any pair of bike stations with a undirected edge if the distance among them is no bigger than the pre-set radius γ. All edges are included in the set E so the PBS can be represented by a graph $G(V, E)$. Each edge is assigned with a weight measuring the distance between its two vertices. The distance d_{ij} between any two positions v_i and v_j can be estimated by two ways: Euclidean and Manhattan distances as the following equation [27].

$$d_{ij} = \begin{cases} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, & \text{Euclidean}, \quad (2) \\ |x_i - x_j| + |y_i - y_j|, & \text{Manhattan}. \quad (2') \end{cases}$$

This paper will select out some station positions to place cabinets. The selected position is called s-position, all of which are included in the set P ($P \subset V$). To place the cabinets at the positions in P can provide users a certain QoS on the friendly battery leasing and returning service, which is measured by the QoS function $f(\cdot)$. The specific expression of the QoS function will be given in the following context for several different cases.

C. Problem Formulation

Some factors should be taken into account to solve the CDP, *i.e.*, to select the subset P from the station set V to deploy the cabinets. Firstly, the bike stations should be selected to satisfy the required QoS, denoted by η, based on the two features: the bike rental frequency and the resource balance deviation. Secondly, the deployment cost, *i.e.*, $\sum_{v_i \in P} \delta_i$, should be decreased, where δ_i is the cost to place a cabinet at position v_i. Let $\theta_i = 1$ represent that the position v_i is selected to place a cabinet and $\theta_i = 0$ otherwise. Thus P contains all positions v_i with $\theta_i = 1$. The goal of the CDP problem is to find the set $P = \{v_i, \forall \theta_i = 1, v_i \in V\}$. Therefore, the CDP problem in this paper can be formulated as the following optimization problem:

$$\mathcal{P}_1: \quad \min \sum_{v_i \in V} \delta_i \theta_i \quad (3)$$

$$s.t. \ f(P) \geq \eta, \quad \forall P \subset V \quad (4)$$

where $f(\cdot)$ is the QoS function to measure the QoS for the s-position set P. The CDP is harder than the set cover problem, which is proved to be NP-hard [11], [28].

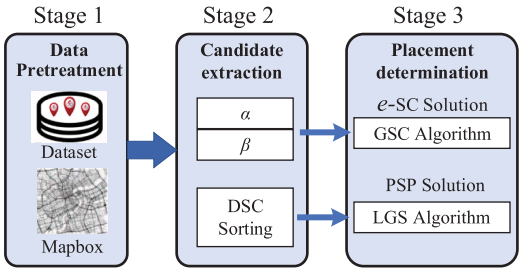


Fig. 4. Strategy overview.

IV. Strategy for Euclidean Space

A. Overview

This section presents our strategy for CDP in the Euclidean space with three stages as shown in Fig. 4:

- **Data pretreatment.** With the dataset of Hangzhou PBS and the graph $G(V, E)$, this paper implements the following two pretreatment: (1) Dataset screening. Eliminate the isolated station under the radius γ, and obtain the stations set V; (2) Map matching. Match the latitude and the longitude of each station V to those in the Mapbox [29].
- **Candidate extraction.** This stage clusters the stations according to the station features tuple $b(\alpha, \beta)$, and sorts the station positions by both the bike rental frequency and the resource balance deviation, and selects the candidate set M ($M \subset V$). Section IV-B gives the extraction process in details.
- **Placement determination.** After extracting the candidate positions with the second stage, the CDP is specified as the e-SC and the PSP respectively under the different QoS definition in Sections IV-C and IV-D.

B. Candidate Station

Fig. 5 shows the hot map of the bike rental frequency of the Hangzhou PBS, which has different values in different regions with different area. This section applies the DSC algorithm to cluster the similar regions with one group so as to extract candidate positions in different regions throughout the city.

Fig. 5. The distribution of the bike rental frequency in Hangzhou PBS.

1) Normalization: In the feature $b(\alpha, \beta)$, the parameter β may reach a very large value while α's value can be relatively small. For example, there are three stations' features calculated by Equation (1): $b_1(10, 1000/day)$, $b_2(10, 900/day)$ and $b_3(1, 1000/day)$. For convenient calculation, we utilize the *0-1 normalization* to leave the result in the interval $[0, 1]$. After normalization, the three stations features turn to $b_1(1, 1)$, $b_2(1, 0.9)$ and $b_3(0.1, 1)$.

2) Station Clustering: After the normalization, we refer to the spatial clustering algorithm DSC to cluster stations with the most similar features into the same group [30]. Let the two parameters of the station feature b be the ordinate values respectively of a two-dimensional graph. All stations attribute points may scatter within the graph in any spatial shape. Compared to the other clustering methods (*e.g.*, k-means) [31], the DSC algorithm can find the cluster of irregular shape and specify the number of clusters by itself instead of the quantity given in advance and finding the circular clusters, such as the k-means method.

The goal of the DSC algorithm is to group stations in the set V into k clusters, where k can be determined automatically. After all of the station features are represented as point $b_i(\cdot, \cdot)$, $\forall S_i \in V$ with the above normalization process, define the set $B = \{b_i, \forall v_i \in V\}$ to store all points of the normalized station features. Let $S_{b_i}^{\gamma}$ denote the set of points in the neighboring range with the radius γ centered at b_i. Every point b_i has a density function $\rho(b_i)$ defined in the following equation:

$$\rho(b_i) = |S_{b_i}^{\gamma}|, \quad \forall b_i \in B$$

Let λ be a baseline to measure whether a point is a core point or not. There are three different types of points in the set B found by the DSC algorithm: *core point, border point* and *noise point*. A point is a core one if its density is higher than the threshold λ as given with Equation (5):

$$\rho(b_i) \geq \lambda, \quad \forall b_i \in B \tag{5}$$

All of core points are stored in the set B_c. If a point b_i is not a core point but a neighbor of the core point, *i.e.*, $B_c \cap S_{b_i}^{\gamma} \neq \varnothing, \forall b_i \notin B_c$, it's called as the border point. Let B_b be the set of all border points. If a point b_i is not in either B_c or B_b, it's called the noise point. The example in Fig. 6(a) shows the three kinds of points when $\lambda = 2$.

There are three concepts needed in the DSC algorithm. The first is *directly density-reachable*. If $b_1 \in B_c$ and $b_2 \in S_{b_1}^{\gamma}$, we call the b_2 is directly density reachable from b_1. The second concept is *density-reachable*. If $b_1, b_2, \ldots, b_n \in B (n \geq 2)$, and $b_{i+1} (v = 1, 2, \ldots n-1)$ is directly density reachable from b_i, we call the b_n is density-reachable from b_1. The last concept is *density-connected*. If $b_1, b_2, b_3 \in B$, b_2, b_3 are both density-reachable from b_1, the b_2 and b_3 are defined to be density-connected with each other.

The main idea of the DSC algorithm is to select randomly a core point as the start, and continually expand to a density-reachable area so as to obtain the maximum region containing the core and border points. The details of the idea is proposed by the following two steps.

- *Initialization*. Generate the neighborhood $S_{b_i}^{\gamma}$ for each point $b_i \in B$, and $k = 1$. The DSC algorithm assigns a cluster mark $I_i (v_i \in V)$ for each station as follows.

$$I_i = \begin{cases} k(k > 0), & v_i \in k^{th} \text{ cluster} \\ -1, & v_i \text{ is a noise station} \end{cases} \tag{6}$$

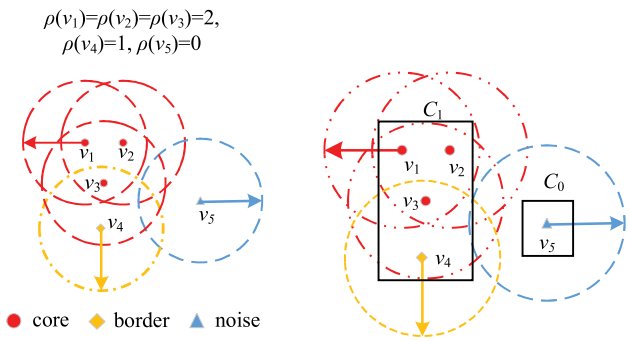
- *Clustering*. DSC runs iteratively in this phase. First, it randomly chooses a point b_i and judges if b_i is a core point or not by Equation (5). If b_i is a core point, its neighborhood is stored in a temporary set S and it continues to expand the density-connected points of b_i and then marks them belong to the k^{th} cluster. Otherwise, b_i is recorded as a noise point temporarily. b_i is confirmed as a border point if it appears in the set S under the subsequent iteration. At last, all of the remaining stations can be put into $k + 1^{th}$ clusters: c_0 (noise cluster).

Our DSC algorithm is summarized in Algorithm 1.

Algorithm 1 Density-Based Stations Clustering (DSC)

Input: V, γ and λ.
Output: Clusters $c_1, c_2, \cdots c_k$
Let $B \leftarrow V$, and generate neighborhood for each point: $S_{b_i}^{\gamma}, \forall b_i \in B$
Initialize clusters number: $k \leftarrow 1$; $I_i \leftarrow 0, \forall v_i \in V$
while $B \neq \emptyset$ **do**
  Choose b_i from B; $B \leftarrow B \backslash \{b_i\}$; Set $S \leftarrow S_{b_i}^{\gamma}$
  if $|S| < \lambda$ **then**
    b_i is a noise or a border: $I_i \leftarrow -1$
  else
    Put v_i into the k^{th} cluster: $I_i \leftarrow k$; $c_k = c_k \cup \{v_i\}$
    while $S \neq \emptyset$ **do**
      Select b_j from S; $S \leftarrow S \backslash \{b_j\}$; $B \leftarrow B \backslash \{b_j\}$;
      if $I_j = 0$ **or** -1 **then**
        Put v_j into the k^{th} cluster: $I_j \leftarrow k$; $c_k = c_k \cup \{v_j\}$;
      if b_j is a core point **then**
        $S \leftarrow S \cup S_{b_j}^{\gamma}$;
    $k \leftarrow k + 1$;
Define the noise cluster c_0 to store all of the noise points.

An example for the DSC algorithm is presented in Fig. 6 with five points $b_1, b_2, \ldots, b_5$ when $\lambda = 2$. The clustering initialization and results are presented in Fig. 6(a) and Fig. 6(b).



(a) Initialization (b) Clustering result

Algorithm progress			
Choose point	U	B	I_i
v_1 (from B)	v_2, v_3	v_2, v_3, v_4, v_5	$I_1 = 1$
v_2 (from U)	v_1, v_3	v_3, v_4, v_5	$I_2 = 1$
v_3 (from U)	v_1, v_2, v_4	v_4, v_5	$I_3 = 1$
v_4 (from U)	v_1, v_2	v_5	$I_4 = 1$
v_1 (from U)	v_2	v_5	I_1 already marked
v_2 (from U)	empty	v_5	I_2 already marked
v_5 (from B)	empty	empty	I_5 = -1
Result : c_1 = { 1,2,3,4 } c_0 = { 5 }			

(c) Detailed progress of DSC

Fig. 6. Example for DSC algorithm.

Detailed algorithm progress is showed in Fig. 6(c). We randomly choose a point b_1, a core point by Equation (5). Its neighborhood b_2, b_3 are stored in a temporary set S. It continues to expand the density-connected points of b_1 and then marks them belong to the 1^{th} cluster. Iterate this process to get the final result. The example Fig. 6(b) shows that there are two clusters, c_0 and c_1. These points are grouped into different classes and the density varies significantly between classes. Thus, stations in the same group have the similar features.

3) Candidate Set Determination: The bike station with the high bike rental frequency and the low resource balance deviation are much suitable to deploy cabinets. In this paper, the rank function $r(\cdot)$ for each station v_i is defined in Equation (7).

$$r(i) = \frac{log(\beta_i)}{\alpha_i} \tag{7}$$

where the logarithmic value of β_i is adopt to leverage the affection of the both parameters because β_i is much larger than α_i. The stations in each cluster $c \in \{c_0, \ldots, c_{|k|}\}$ are sorted in non-increasing order according to $r(\cdot)$ in Equation (7). The top φ $(0 < \varphi \leq 1)$ ranked stations are kept as the candidates for cabinets deployment, and stored in the set M. φ is an adjustable parameter. Fig. 7 shows the candidate stations with the red dots in the map when $\varphi = 0.3$.

C. Solution to e-SC

This section gives the QoS definition by taking coverage into account. The CDP is thus specified as the NP-hard e-SC problem. So this paper presents the greedy algorithm GSC to select the final s-positions P for the cabinets deployment [28].

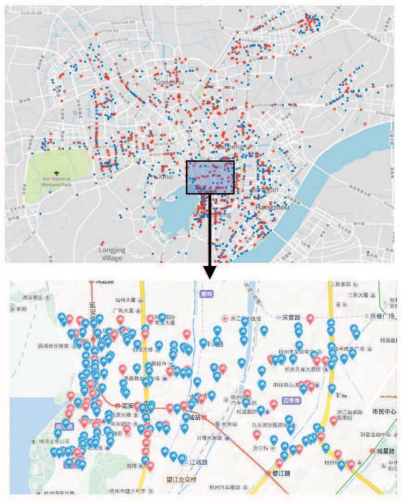


Fig. 7. Candidate positions distribution.

1) QoS Definition Under e-SC : It's quite expensive and oversupplying to build cabinet by each bike station. An advisable strategy allows each user to have at least one station to lease or return battery in their acceptable range within the distance γ. Let S_i^{γ} denote the set of positions which are covered by the cabinet position v_i as the following equation:

$$S_i^{\gamma} = \{v_j : d(v_i, v_j) \leq \gamma, v_j \in V\}$$

$|S_i^{\gamma}|$ value reflects the service ability of the position v_i for the near bike stations. So the QoS function $f(\cdot)$ can be specified as the following equation:

$$f(P) = \sum_{v_i \in V} \sum_{v_j \in P} q(v_i, v_j) \tag{8}$$

where $q(v_i, v_j)$ represents whether the station position v_i is covered by the cabinet position v_j as the following equation:

$$q(v_i, v_j) = \begin{cases} 1 & if\ v_j \in S_i^{\gamma} \\ 0 & otherwise \end{cases}$$

When setting $f(P) \geq 1$ in Equation (4), the CDP problem can be translated into the typical set cover problem, which is how to select a result set P from candidate positions set M so as to satisfy that almost all bike stations can be covered by at least one cabinet as Algorithm 2.

2) GSC Algorithm: The GSC algorithm is presented to solve the e-SC based on the candidate selection. Its main idea is to select the position with the maximal $|S_i^{\gamma}|$ value at each selection round. Suppose there are $|M|$ candidate sites, and the maximum number of domain sites for each site does not exceed n. Then the time complexity of GSC is $O(|M|^2 * log|M|)$, and the space complexity is $O(n|M|)$.

D. Solution to PSP

The PBS may cover the big city area so this section designs distributed way, the LGS algorithm, to decide the set P of

Algorithm 2 GSC Algorithm for e-SC

Input: The set M of candidate positions, cabinet cost $\delta_i(\forall v_i \in M)$, and cover radius γ;
Output: The set P of cabinet deployment positions.
1: Generate the set S_i^γ for each $v_i \in M$
2: Initialize: $P \leftarrow \emptyset$;
3: **while** $\bigcup S_i^\gamma \neq V(v_i \in P)$ **do**
4: Sort all station positions by their $|S_i^\gamma|$;
5: Find the station position v_j with the maximum value of $|S_j^\gamma|$, and set $\theta_j = 1$;
6: Remove position v_j and those covered by it from the set M;
7: Add station position v_j to the set P with $\theta_j = 1$;

s-positions for cabinets placement. It's proved that the result achieved by LGS has the lower bound $(1 - 1/e)f_{opt}$. f_{opt} represents the theoretical optimal QoS value.

1) QoS Definition Under PSP : This paper adopts another QoS definition based on the Euclidean distance:

$$f(P) = \sum_{v_i \in v} log(\sum_{v_j \in P} q(v_i, v_j)) \tag{9}$$

where $q(v_i, v_j)$ represents the QoS provided by the cabinet at the position v_j to the user at position v_i. $q(v_i, v_j)$ is measured by the Euclidean distance $d(\cdot)$ in Equation (2).

$$q(v_i, v_j) = \begin{cases} \epsilon e^{-d(v_i, v_j)} & d(v_i, v_j) \leq \gamma \\ 0 & d(v_i, v_j) > \gamma \end{cases}$$

ϵ is a previously given constant. The over-crowded cabinet deployment can cause the battery profligacy. So, this paper utilizes the logarithmic function in Equation (9) to simulate the QoS growth. The CDP becomes the PSP problem, which is how to select a subset P of positions from M so as to ensure the QoS satisfying the user demand η.

2) LGS Algorithm Design: The LGS algorithm is designed to solve the PSP effectively. According to the QoS definition in Equation (9), we condense that the position v_i has no impact on the QoS value of the user at position v_j if the distance between them is greater than the cover radius γ. Therefore, this paper divides the area of PBS stations into adjacent and independent square sub-areas, denoted by $\mathcal{R} = \{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_n\}$ [32]. The set of candidate positions in the region $\mathcal{R}_i$ is denoted by M_i. Then, the LGS algorithm is executed in each region in parallel. In addition, a server is used to collect the selected positions, to find the result set P, and then feedback to each region with the final set P of s-positions. The detailed algorithm description is summarized in Algorithms 3 and 4.

Algorithm 3 LGS Algorithm for Region $\mathcal{R}_i \in \mathcal{R}$

Input: Graph $G(V, E)$;
1: Initialize the candidate set M_i for region $\mathcal{R}_i$.
2: **if** Receive the updated set P from the server **then**
3: Choose position $v^* = \text{argmax}_{v_j \in M_i \setminus P}[f(P \bigcup \{v_j\})]$.
4: Send the position v^* to the server.

Algorithm 4 LGS Algorithm for the Server

Output: The set P of s-positions
1: Initialize result set: $P \leftarrow \emptyset$
2: **while** $f(P) < \eta$ **do**
3: Send the updated set P of s-positions to regions.
4: **if** Receive a selected position v^* from a region **then**
5: Add position v^* to the set P: $\theta^* = 1$, $P \leftarrow P \cup \{v^*\}$.

The LGS algorithm runs distributively since each region of the PBS in $\mathcal{R}$ only receives the information about the updated set P of s-positions from the server and then chooses the position for cabinet deployment in parallel. The LGS algorithm consists of two sub-algorithms: the LGS algorithm for the region $\mathcal{R}_i$ in Algorithm 3 and the LGS algorithm for the server in Algorithm 4. As Algorithm 3 shows, the LGS algorithm for the region $\mathcal{R}_i$ has two phases: the first is to initialize the candidate set inside the region $\mathcal{R}_i$. If the updated set P of s-positions is received, the second is to choose the position v_j with the maximal $f(P \bigcup \{v_j\})$ value from candidate set M_i and then send the position v_j to the server. As Algorithm 4 shows, the LGS algorithm for the server has the following phases. In the first phase, it generates the empty set P. In each selection round, the server sends the updated set P to all regions if the QoS of the set P does not satisfy the user demand η yet. Then, the server adds the position v^* to the set P if it receives a position v^* from someone region. The algorithm outputs the set P as the end till its QoS satisfies the user demand η. Suppose there are $|M|$ candidate sites, which are divided into $|k|$ sub-areas to execute algorithm clients respectively. Then the time complexity of LGS is $O(|M| * |k|)$, and the space complexity is $O(n|M|)$.

E. Theoretical Analysis

Lemma 1: Given the set P ($P \subset M$) and $f(P) = \sum_{v_i \in V} log(\sum_{v_j \in P} q(v_i, v_j))$, $f(P)$ is a sub-modular set function.

Proof: According to the QoS definition in Equation (9), we have:

$$\begin{aligned} & f(P) + f(\{v_l\}) \\ &= \sum_{v_i \in V} \log(\sum_{v_j \in P} q(v_i, v_j)) + \sum_{v_i \in V} \log(q(v_i, v_l)) \\ &= \sum_{v_i \in V} [\log(\sum_{v_j \in P} q(v_i, v_j)) + \log(q(v_i, v_l))] \\ &\geq \sum_{v_i \in V} \log[\sum_{v_j \in P} q(v_i, v_j) + q(v_i, v_l)] \\ &\geq f(P \cup \{v_l\}) \end{aligned}$$

we easily get $f(P) + f(\{v_l\}) \geq f(P \cup \{v_l\})$, *i.e.*, $f(P \cup \{v_l\}) - f(P) \leq f(\{v_l\})$. Given the set P, $f(P \cup \{v_l\}) - f(P)$ is the marginal profit of the v_l and has the decreasing gain property. So, for an arbitrary subset $P'(P' \subseteq P)$, we have $f(P \cup \{v_l\} - P') \geq f(P \cup \{v_l\}) - f(P')$. Therefore, $f(B)$ is a sub-modular set function. This completes the proof. □

Theorem 2: *Let f_{opt} be the optimal QoS value, and f_{greedy} be the one achieved by greedy selection progress in Algorithm 3. It thus has that f_{greedy} is at least $(1-1/e)f_{opt}$.*

Proof: As shown in a fundamental result in Golovin *et al.* work [33], f is a monotonically increasing sub-modular function according to the lemma 1. The LGS algorithm in Algorithms 3 and 4, which starts with the empty set and adds the position v_j with the maximal $f(P \cup \{v_l\})$ can maximally improve the QoS value in each round and obtains a near-optimal solution, *i.e.*, $f_{greedy} \geq (1-1/e)f_{opt}$. This completes the proof. □

V. Strategy for Manhattan-Type City

In the real city, the bike trip is restricted by the street or building, and this section proposes the cabinet deployment strategy based on the Manhattan model.

A. City Voronoi Construction

Generalized voronoi diagram is widely used in the coverage problem [34]. Let $V = \{v_1, v_2, \ldots, v_M\}$ be the set of points in a two-dimensional space $\mathcal{R}^2$. For any point $v_i \in \mathcal{R}^2$, $d\{v_i, v_j\}$ denotes the distance from v_i to v_j. The distance metric can be of the Euclidean, Manhattan, Chessboard or another metric [35]. The dominance region of v_i over v_j can be defined as:

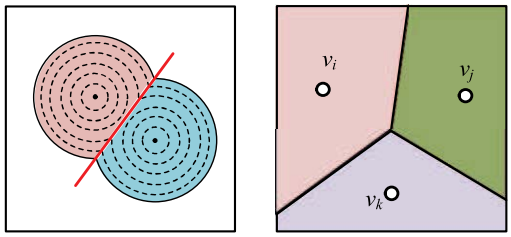
$$\mathscr{D}\{v_i, v_j\} = \{v_k | d\{v_i, v_k\} \leq d\{v_j, v_k\}\}$$

For the generator point v_i, the voronoi region of v_i can be defined by:

$$H(v_i) = \{\cap_{j \neq i} \mathscr{D}\{v_i, v_j\}\} \tag{10}$$

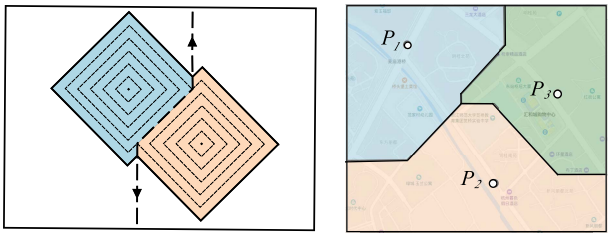
where $H(v_i)$ is the set of all points that are not closer to any other point than v_i. Subsequent voronoi regions $H(v_1), H(v_2), \ldots, H(v_M)$ are the partitions of the generalized voronoi diagram $H(V)$. The existing algorithm constructs the Euclidean voronoi diagram $H(V)$ by simulating the expansion of wavefronts of every point starting at $t = 0$ as a constant speed under the Euclidean space [36]. At time $t \geq 0$ the wavefront is the set of all points whose distance from the source point is equal. The border is formed when the wavefronts of two points merge. The borders of the $H(V)$ partition consist of all points that can be reached equally quickly from at least two points. The examples of the wavefront expansion and voronoi diagram under Euclidean metric are presented in Fig. 8(a) and 8(b).

While the Euclidean distance offers high precision in the free space, the Manhattan distance metrics are known to better approximate city geography [35]. Riding distance between two station positions v_i and v_j can be estimated by Manhattan distance in Equation (2′) [27]. As shown in Fig. 9(a) and 9(b), the city voronoi diagram $H(V)$ can be constructed in a similar wavefront expansion way. Due to the different distance metric, the dissimilarity is that the wavefront here is a square instead of a circle.



(a) European wavefront expansion (b) European voronoi

Fig. 8. Voronoi diagram in the Euclidean space.



(a) Wavefront expansion (b) Manhattan voronoi

Fig. 9. Voronoi diagram in the Manhattan space.

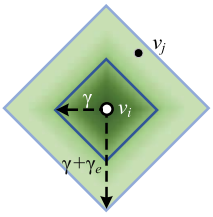


Fig. 10. Binary gradient model.

B. QoS Definition Under Manhattan Space

This section defines the QoS model by considering the factors closer to the real traffic scene. Some stations scatter in the city and each one can be modelled as the point to expand the wavefront in Manhattan space. For an arbitrary position v_i, its QoS $q(v_i, v_j)$ of the station v_i to any road coordinate v_j can be defined as a binary gradual change model [37] as shown in Fig. 10. Its QoS $q(v_i, v_j)$ is defined as:

$$q(v_i, v_j) = \begin{cases} 1, & d(v_i, v_j) \leq \gamma \\ \varphi' e^{1-\frac{d(v_i,v_j)}{\gamma}}, & \gamma < d(v_i, v_j) < \gamma + \gamma_e \\ 0, & d(v_i, v_j) \geq \gamma + \gamma_e \end{cases} \tag{11}$$

where γ_e is the error radius and φ' is the adjustable parameter of between 0 to 1. $d(v_i, v_j)$ is the Manhattan distance between v_i and v_j.

In addition, Fig. 11 shows the path coverage map obtained by randomly selecting a number of users' riding track data. Based on the analysis of a large number of bicycle users' historical travel data, we find that about 90% of the roads have experienced users' riding tracks. Notice that the bike station is established along the roadside, and their position is the road coordinate. Let $\mathcal{R}$ denote the set of all possible coordinates in

(a) 10% user cycling track (b) 30% user cycling track (c) 50% user cycling track (d) 100% user cycling track

Fig. 11. User riding path coverage map.

the grid road map. The QoS of road coordinate v_i is calculated by Equation (12).

$$q(P, v_i) = \sum_{v_j \in P} q(v_j, v_i), \qquad \forall v_i \in \mathcal{R} \tag{12}$$

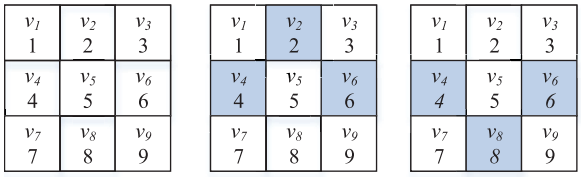
C. GAS Algorithm Design

This section presents the GAS algorithm based on the genetic algorithm, which is an artificial optimization algorithm with a stochastic search heuristic based on the natural selection and evaluation [38]. The genetic algorithm includes some parts or operations: population generation, gene, fitness function, crossover, selection and mutation.

1) Population Generation: The primary population is a collection of individuals in the genetic algorithm, and corresponds to the number of candidates positions in this paper. The set M of candidate positions is generated by the method in Section IV-B, where the distance is measured by the Manhattan model in Equation (2′) and the QoS function is given in Equation (11). Based on the set M, the GAS algorithm generates a set of first generation population. Denote the set by F_0 and the population by positive integer ϵ_1.

2) Gene: In the genetic algorithm each individual has a unique gene, which can be represented by a binary sequence. This paper represents each station deployment scheme with a unique gene by the binary sequence, which has N_g bits and $N_g \triangleq |M|$. Define the variable θ_i for each bit and its value is 0 and 1 to represent the information quantity of two states. $\theta_i = 1$ indicates that the station $v_i \in M$ is selected as the deployment cabinet, and otherwise $\theta_i = 0$. For example, suppose that there are 9 candidate stations as shown in Fig. 12. The number in the grid represents the subscript of the variable θ_i, and the colored coordinates represent that the station v_i is selected to deploy one cabinet. The cabinet deployment scheme on the map can be represented by a 9-bit binary sequence. As shown in the figure, the binary sequence of deployment scheme of cabinet shown in Fig. 12 is 010101000, and that in Fig. 12 is 000101010.

3) Fitness Function: In the selection process, the genetic algorithm eliminates the chromosome with low adaptability and only retains the chromosome with high adaptability. After several generations of iterative optimization, the quality of offspring individuals will be increased. Since the goal of the CDP is to minimize the cost, the fitness function of GAS algorithm is formally formulated as the overall cost in Equation (3). The fitness of individual deployment scheme in each generation is calculated by Equation (3).



(a) Gene (b) Gene code one (c) Gene code one

Fig. 12. Coding scheme of GAS algorithm.

4) Crossover: With the arithmetic crossover in the genetic algorithm, two individuals exchange a part of their genes in the quits location so as to create two new individuals. In the GAS algorithm, the genes in F_0 are randomly paired and each pair of genes find the random quits location in gene and exchange their genes in the location.

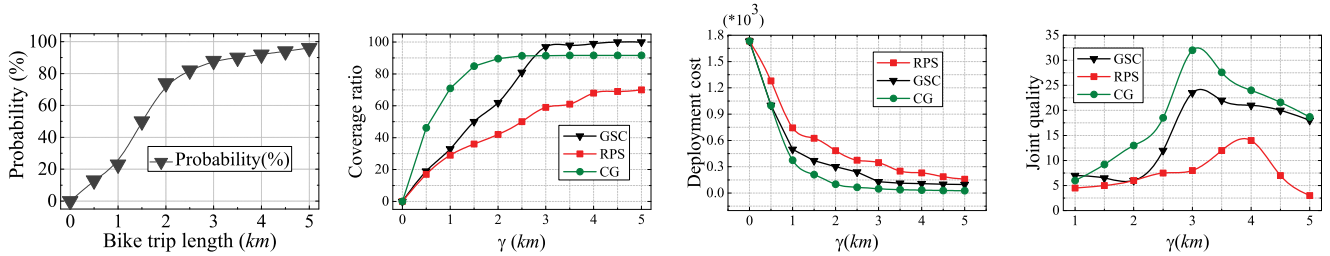
5) Selection: In order to save the most adaptive individual structure to the next generation population, the GAS algorithm uses the best preserved selection method. Since the CDP requires the threshold QoS, the fitness value of each individual in F_0 is calculated respectively by Equation (12). By the fitness value, all individuals in F_0 are sorted into an non-decreasing order according to the fitness value. The individuals in the order whose QoS satisfies the constraint (4) are reserved in a new set F_1, and delete the last half parts of individuals in F_0 and obtain an updated set F_0.

6) Mutation: Mutation is an essential way to produce new genes in nature. It changes some gene values of individuals according to a small probability, so as to produce new individuals or new genes, which may have the great increase of survival probability and adaptability. In the GAS algorithm, each individual in the updated F_0 randomly selects locations to mute in its gene, and generate one new gene, which is added into F_0.

With the above procedures, the details of our is summarized in Algorithm 5 as follows.

VI. Experiment Evaluation

This section conducts our numerical experiment to evaluate the performance of four algorithms: GSC, LGS, GAS and CG. A straightforward strategy, called Random Placement Strategy (RPS), is proposed to compare with our four algorithms. The idea of RPS is to select a set of stations randomly as the set of locations to deploy the cabinets, and the number of candidate sites in the RPS algorithm can be increased. The column generation is an efficient algorithm for solving



(a) Bike trip length distribution (b) γ vs. coverage ratio (c) γ vs. deployment cost (d) γ vs. joint quality

Fig. 13. Performance evaluation of the GSC algorithm.

Algorithm 5 Mutation of GAS Algorithm

Input: The set V of stations, a positive constant integer $\epsilon_2 > 0$;

Output: The set P of stations to deploy cabinets;

1: Obtain the clusters by Algorithm 1 with the distance model in Equation $(2')$;
2: Obtain the set M of candidates by the method in Section IV-B;
3: Create the first generation F_0 of individuals by randomly creating ϵ_1 binary sequences, each of which has N_g bits;
4: Define an index $Index = 0$ and a temporary set F_2;
5: **while** $Index \leq \epsilon_2$ **do**
6: Crossover;
7: $F_2 \leftarrow F_1$;
8: Selection to obtain an updated F_1;
9: Sort all individual in F_1 and F_2 into a non-decreasing order according to their fitness values, and keep the former half part of the order into F_1;
10: Mutation;
11: $Index++$;
12: Find the individual in F_1 with the highest fitness value and assign its gene to P.

large-scale linear optimization problems, and is proposed to compare with our four algorithms [39]. Firstly, the experiment setting are provided. After that, a series of different parameters comparison among the GSC, LGS, GAS, CG algorithms and the RPS are presented in Sections VI-B, VI-C and VI-D.

A. Experiment Setting

This paper applies Python to connect the database to process the dataset collected from *MySql*. Python program for the experiment is divided into three stages. Firstly, we import the *PyMySql Model* and scientific computing package *NumPy* to process the dataset and generate two station features [40], [41]. Fig. 3 in the previous section is drew by python program to illustrate the station features. Secondly, this paper implements the LGS algorithm and indicates the candidate positions on the map as shown in Fig. 7 by the python *Plotly* model [42]. All the points in Fig. 7 represent the bike station positions while the red points are the candidate positions selected by the LGS algorithm. Stations distribution in the square of Fig. 7 are enlarged so as to observe more clearly. Finally, we execute the GSC or the LGS algorithm to get the s-position set P. In the experiment, the deployment cost is the number of s-positions.

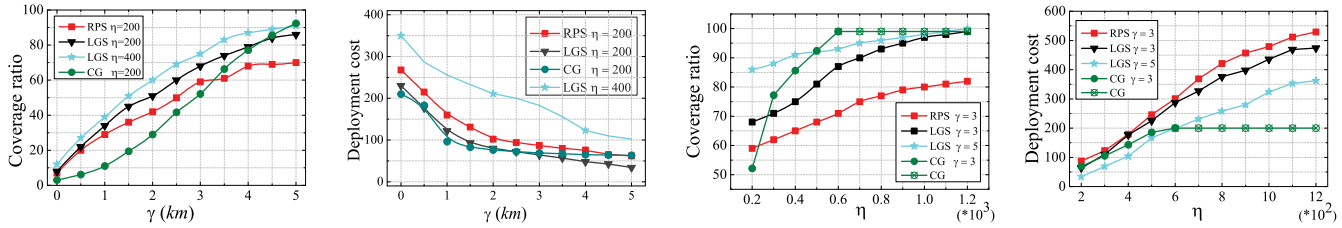
B. Evaluation for GSC Algorithm

In this experiment, the cabinet cost for each position is set to the same ($\delta_i = \delta_j, \forall v_i, v_j \in M$). GSC algorithm selects 129 positions from the total 1773 positions for cabinets placement. We analyze the performance of the GSC algorithm in different cover radius values γ, which is compared to the RPS algorithm.

Fig. 13(a) summarizes the trip lengths distribution of the Hangzhou PBS. More than 80% of the trips are shorter than 3 km, which is consistent with the assumption that PBS is troubled by "3-km curse". The experiment is evaluated under the cover radius ranges between $[0, 5]$ km because almost 95% of the bike trips are shorter than 5 km. Three parameters below are measured to evaluate the performance under two methods.

1) Coverage Ratio: It's the ratio of the number of the stations covered by s-positions over the overall stations, *i.e.*, N. The coverage ratios of the GSC algorithm and the RPS in different cover radius are shown in Fig. 13(b). If the cover radius is set to 3 km, every s-position has the ability to cover the stations within 3 km centered at itself. Some stations have no neighbor within 1 km and these isolated stations need to be removed. After the remove operation, only about 60% of remaining stations can be covered by s-positions by the RPS algorithm. The coverage ratio of the Algorithm 2 reaches 98.3% the as cover radius is set to 3 km. The increase of the coverage ratio reaches to the bottleneck as the cover radius ranges in $[4, 5]$ km. When the GSC algorithm and the RPS has the same cover radius, the coverage ratio of the GSC is always higher than the RPS. The coverage ratio of the CG algorithm reaches the bottleneck as cover radius is set to be 2 km, which is earlier than the GSC and RPS algorithms. But the coverage rate of the CG algorithm is lower than that of the GSC algorithm.

2) Deployment Cost: The cabinet deployment cost of the GSC and the RPS under different cover radius are shown in Fig. 13(c). We set the same deployment cost for each station in the experiment ($\delta_i = \delta_j, \forall v_i, v_j \in M$). Thus, the deployment cost in the experiment is replaced by the s-positions number. If the cover radius γ is small, we need place more cabinets and thus have a high cost. With the GSC algorithm, the deployment cost declines obviously, *i.e.*, the number of selected positions decrease from 1688 to 129, when the cover radius value



(a) γ vs. coverage ratio (b) γ vs. deployment cost (c) η vs. coverage ratio (d) η vs. deployment cost

Fig. 14. Performance evaluation of the LGS algorithm.

change from 0 km to 3 km. However, when the cover radius is set to 4 km or 5 km, the total cost is decreased slightly. Compared to the GSC algorithm, RPS selects the stations with the same coverage ratio. As shown in Fig. 13(c), the RPS would cost more than the GSC algorithm under the same γ value. The CG algorithm costs less than the GSC algorithm under the same γ value.

3) Joint Quality: Both coverage ratio and total deployment cost are important reference indicators that reflect the experiment effectiveness. In order to observe the effect of these two parameters intuitively, this section defines the joint QoS $\mathrm{QoS}(s_i) = \dfrac{|S_i^{\gamma}|}{\gamma\,\delta_i}$ to integrate these two parameters: the coverage ratio and the deployment cost. With the GSC algorithm, the lower deployment cost and cover radius value result in the higher coverage ratio value. Fig. 13(d) illustrates the joint QoS of all s-positions selected out in different cover radius γ under the GSC, the CG and the RPS algorithms. The simulation shows that under the same cover radius, the CG algorithm always has better performance than others. In addition, when the cover radius is 3 km, the GSC and the CG algorithms has the better performance than RPS.

C. Evaluation for LGS Algorithm

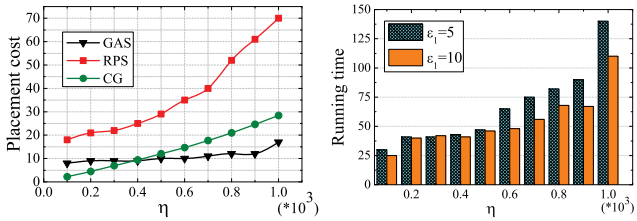
This subsection sets $\alpha = 1$, $\beta = 200/400$ to execute our algorithm. The coverage ratio and deployment cost are considered as metrics for evaluation.

1) Impact of Cover Radius: Fig. 14(a) and Fig. 14(b) indicate that the coverage ratio and deployment cost variation under different cover radius values. As Fig. 14(a) shows that the coverage ratio of the CG algorithm increases from 3% to 89.6% and the coverage ratio of the LGS algorithm is increased from 8% to 86% under the user demand $\eta = 200$ when the cover radius γ increases from 0 to 5 km. In contrast, the coverage ratio of the RPS only reaches to 70% under the user demand $\eta = 200$. In addition, the coverage ratio of the LGS algorithm under the user demand $\eta = 400$ is larger than the coverage ratio under the user demand $\eta = 200$ and reaches to 91% when the cover radius $\gamma = 5$. The coverage ratio of the CG algorithm under the user demand $\eta = 200$ is lower than that of the LGS algorithm when the cover radius $\gamma = [1, 4]$. As Fig. 14(b) shows that the deployment cost of the LGS algorithm is decreased from 228 cabinets to 34 and the deployment cost of the CG algorithm decreases from 205 cabinets to 87 under the user demand $\eta = 200$ when the cover radius γ increases from 0 to 5 km. In contrast, the deployment cost of RPS is decreased to 62 cabinets under the user demand $\eta = 200$. In addition, the deployment cost of the LGS algorithm under the user demand $\eta = 400$ is larger than the deployment cost under the user demand $\eta = 200$ and reaches to 104 cabinets when the cover radius $\gamma = 5$. The deployment cost of the CG algorithm under the user demand $\eta = 200$ is less than that of the LGS algorithm when $\gamma = [0, 2.5]$.

2) Impact of QoS Demand: Fig. 14(c) and Fig. 14(d) indicate the coverage ratio and the deployment cost variation under different user demand values. As Fig. 14(c) shows that the coverage ratio of the LGS algorithm is increased from 68% to 97% under the cover radius $\gamma = 3$ when the user demand η increases from 200 to 1200. In contrast, the coverage ratio of the RPS reaches to 82% under $\gamma = 3$. In addition, the coverage ratio of the LGS algorithm under $\gamma = 5$ is larger than the coverage ratio under the cover radius $\gamma = 3$, and reaches to 100% when the user demand $\eta = 1200$. As Fig. 14(d) shows that the deployment cost of the LGS algorithm is increased from 64 cabinets to 474 under the cover radius $\gamma = 3$ when the user demand η increases from 200 to 1200. In contrast, the deployment cost of the RPS reaches to 529 cabinets under the cover radius $\gamma = 3$ and the user demand $\eta = 1200$. In addition, the deployment cost of the LGS algorithm under the cover radius $\gamma = 5$ is lower than the deployment cost under the cover radius $\gamma = 3$ and reaches to 362 cabinets when the user demand $\eta = 1200$. When the user demand $\eta = 600$, the coverage ratio reaches 98% in Fig. 14(c), and the deployment cost of the CG algorithm reaches 200 in Fig. 14(d). When $\eta = [600, 1200]$, there is a different line from the front. It means that the user demand cannot be satisfied by the 200 cabinets no matter how CG selects their sites. CG requires the amount of cabinets to be fixed in advance while RPS and LGS do not. For short, the LGS algorithm always has a higher coverage ratio and a lower deployment cost than the RPS under the same setup.

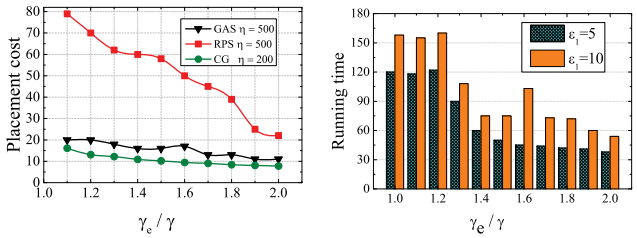
D. Evaluation for GAS Algorithm

This section shows the numerical experimental performance of GAS algorithm, and compares the site deployment cost and algorithm running time index for different QoS demand η, coverage radius ratio γ_e/γ, population size ϵ_1 and other parameters. The deployment cost of each station is the same ($\delta_i = \delta_j, \forall v_i, v_j \in M$) so the number of sites can also reflect the deployment cost.



(a) η vs. deployment cost (b) η vs. running time

Fig. 15. Influence of QoS demand.



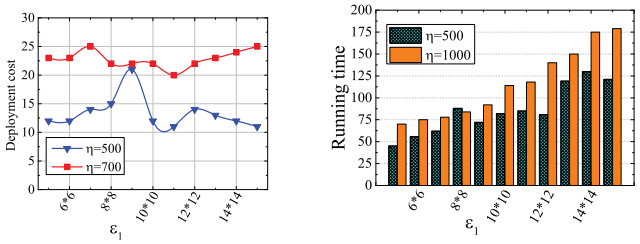
(a) γ_e/γ vs. Deployment cost (b) γ_e/γ vs. running time

Fig. 16. Influence of coverage radius ratio.

1) QoS Demand η: Fig. 15 shows the change of deployment cost and running time of GAS algorithm under different QoS demands. In Fig. 15(a), when the coverage radius ratio $\gamma_e/\gamma = 1.5$, and population size $\epsilon_1 = 10$ and the QoS demand η increases from 0 to 1000, the number of sites obtained by GAS algorithm increases from 7 to 16 and CG algorithm increases from 1 to 29 while that of RPS strategy increases to about 70. In addition, when $\eta = 800$, the curve trend in Fig. 15(a) illustrates that the number of sites obtained by GAS algorithm increases dramatically. CG algorithm basically increases at the constant rate.

As shown in Fig. 15(b), when the coverage radius ratio $\gamma_e/\gamma = 1.5$ is set and the population size is $\epsilon_1 = 5$, and the QoS demand η increases from 0 to 1000, the running time of GAS algorithm increases from 32.13 s to 143.56 s. When the population size $\epsilon_1 = 10$, the running time of GAS algorithm increases from 24.24 s to 107.65 s. It can be observed that the more the number of population is, the quicker the GAS algorithm converges. The running time of GAS algorithm increases sharply when $\eta = 1000$ occurs in the Fig. 15(b).

2) Coverage Radius Ratio γ_e/γ: Fig. 16 shows the deployment cost and running time of GAS algorithm under different coverage radius ratio γ_e/γ. As shown in Fig. 16(a), the coverage radius ratio increases from 1.1 to 2 when the QoS demand is set as $\eta = 500, \epsilon_1 = 10$. The number of s-positions obtained by GAS is reduced from 20 to 11, while the number of s-positions obtained by RPS is reduced from 79 to 22. Two curves in Fig. 16(a) show that the deployment cost of GAS algorithm is always significantly less than that of RPS. When setting the $\eta = 500$, the user demand cannot meet the constraint of $\eta = 500$ by CG algorithm, because the 200 candidate sites are too few. So we set $\eta = 200$ for CG to compare with other algorithms. Fig. 16(a) shows that the deployment cost of the CG algorithm is gradually decreasing when $\eta = 200$, and is lower than the cost of the GAS algorithm



(a) ϵ_1 vs. deployment cost (b) ϵ_1 vs. running time

Fig. 17. Influence of population size.

when $\eta = 500$. As shown in Fig. 16(b), under the QoS demand $\eta = 500$ and population size parameter $\epsilon_1 = 5$, the coverage radius γ_e/γ increases from 1 to 2, and the running time of GAS algorithm is reduced from 120.54 s to 38.45 s. When the population size parameter is $\epsilon_1 = 10$, the running time of GAS is reduced from 155.01 s to 54.43 s. When the ratio of coverage radius reaches 1.5, the running time of GAS algorithm is significantly reduced.

3) Population Size ϵ_1: Fig. 17 shows the change of deployment cost and running time of GAS algorithm under different population sizes. As shown in Fig. 17(a), when the QoS demand is $\eta = 500, \gamma_e/\gamma = 1.5$ and the population ϵ_1 increases from 5 to 15, the number of stations obtained by GAS algorithm floats around 12. The number of stations obtained by GAS floats around 25 when the QoS demand is $\eta = 700$. The change of populations has no significant impact on the number of s-positions generated by GAS. In addition, Fig. 17(a) shows that the number of s-positions obtained by GAS increases with the increase of η parameter. As shown in Fig. 17(b), when $\eta = 500$ and the population sizes increases from 5 to 15, the running time of GAS increases from 45.58 s to 120.7 s. When $\eta = 1000$, the running time of GAS increases from 69.43 s to 178.34 s. In Fig. 17(b), the running time of GAS algorithm increases linearly with the growth of population size.

VII. Conclusion

Traditional PBS is characterized by the short-distance trip. Some cities are trying to develop a new kind of cabinet by adding battery cabinets to the existing PBS for mitigating the pressure on the long-distance transportation modes. So we should handle with the CDP, that is: which positions should be selected out to place the battery cabinets so as to minimize the deployment cost and ensure that the QoS provided by the cabinet satisfies the user demand. In order to evaluate the performance of GSC, LGS, CG and GAS algorithms, this paper proposes a random deployment strategy as a comparison. Experiments show that the algorithm used in this paper is better than the random deployment strategy.

References

[1] P. DeMaio, "Bike-sharing: History, impacts, models of provision, and future," *J. Public Transp.*, vol. 12, no. 4, pp. 41–56, 2009.

[2] D. Zhang, X. Xu, and X. Yang, "User satisfaction and its impacts on the use of a public bicycle system: Empirical studies from Hangzhou, China," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2512, no. 1, pp. 56–65, Jan. 2015.

[3] C. Bullock, F. Brereton, and S. Bailey, "The economic contribution of public bike-share to the sustainability and efficient functioning of cities," *Sustain. Cities Soc.*, vol. 28, pp. 76–87, Jan. 2017.

[4] S. Shaheen and N. Chan, "Mobility and the sharing economy: Potential to facilitate the first- and last-mile public transit connections," *Built Environ.*, vol. 42, no. 4, pp. 573–588, Dec. 2016.

[5] J. Zhang, P. Lu, Z. Li, and J. Gan, "Distributed trip selection game for public bike system with crowdsourcing," in *Proc. IEEE Conf. Comput. Commun.*, Honolulu, HI, USA, Apr. 2018, pp. 2717–2725.

[6] Z. Li, J. Zhang, J. Gan, P. Lu, and F. Lin, "Large-scale trip planning for bike-sharing systems," in *Proc. IEEE 14th Int. Conf. Mobile Ad Hoc Sensor Syst. (MASS)*, Orlando, FL, USA, Oct. 2017, pp. 328–332.

[7] Z. Li, J. Zhang, J. Gan, P. Lu, Z. Gao, and W. Kong, "Large-scale trip planning for bike-sharing systems," *Pervasive Mobile Comput.*, vol. 54, pp. 16–28, Mar. 2019.

[8] Y. Wei. *Xinhua Paper*. Accessed: May 16, 2017. [Online]. Available: http://mini.eastday.com/mobile/170516223138603.html

[9] T. Xu *et al.*, "Taxi driving behavior analysis in latent vehicle-to-vehicle networks: A social influence perspective," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, San Francisco, CA, USA, Aug. 2016, pp. 1285–1294.

[10] A. Goodman, S. Sahlqvist, and D. Ogilvie, "New walking and cycling routes and increased physical activity: One-and 2-year findings from the U.K. iConnect study," *Amer. J. Public Health*, vol. 104, no. 9, pp. 38–46, Aug. 2014.

[11] A. Caprara, M. Fischetti, and P. Toth, "A heuristic method for the set covering problem," *Oper. Res.*, vol. 47, no. 5, pp. 730–743, Oct. 1999.

[12] Y. Zhang and Z. Mi, "Environmental benefits of bike sharing: A big data-based analysis," *Appl. Energy*, vol. 220, pp. 296–301, Jun. 2018.

[13] E. O'Mahony and D. B. Shmoys, "Data analysis and optimization for (citi) bike sharing," in *Proc. AAAI*, 2015, pp. 687–694.

[14] M. Bordagaray, L. dell'Olio, A. Fonzone, and Á. Ibeas, "Capturing the conditions that introduce systematic variation in bike-sharing travel behavior using data mining techniques," *Transp. Res. C, Emerg. Technol.*, vol. 71, pp. 231–248, Oct. 2016.

[15] A. Dabiri, A. Hegyi, and S. Hoogendoorn, "Optimized speed trajectories for cyclists, based on personal preferences and traffic light information-a stochastic dynamic programming approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 2, pp. 777–793, Feb. 2020.

[16] A. Hess, F. Malandrino, M. B. Reinhardt, C. Casetti, K. A. Hummel, and J. M. Barceló-Ordinas, "Optimal deployment of charging stations for electric vehicular networks," in *Proc. 1st Workshop Urban Netw. (UrbaNe)*, 2012, pp. 1–6.

[17] S. Li, Y. Huang, and S. J. Mason, "A multi-period optimization model for the deployment of public electric vehicle charging stations on network," *Transp. Res. C, Technol.*, vol. 65, pp. 128–143, Apr. 2016.

[18] S. Mehar and S. M. Senouci, "An optimization location scheme for electric charging stations," in *Proc. Int. Conf. Smart Commun. Netw. Technol. (SaCoNeT)*, Jun. 2013, pp. 1–5.

[19] F. He, D. Wu, Y. Yin, and Y. Guan, "Optimal deployment of public charging stations for plug-in hybrid electric vehicles," *Transp. Res. B, Methodol.*, vol. 47, pp. 87–101, Jan. 2013.

[20] S. Faridimehr, S. Venkatachalam, and R. B. Chinnam, "A stochastic programming approach for electric vehicle charging network design," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 5, pp. 1870–1882, May 2019.

[21] F. He, Y. Yin, and J. Zhou, "Deploying public charging stations for electric vehicles on urban road networks," *Transp. Res. C, Emerg. Technol.*, vol. 60, pp. 227–240, Nov. 2015.

[22] R. K. Yadav, D. Gupta, and D. Lobiyal, "Dynamic positioning of mobile sensors using modified artificial bee colony algorithm in wireless sensor networks," *Int. J. Control Theory Appl.*, vol. 10, no. 18, pp. 167–176, 2017.

[23] F. Alduraibi, N. Lasla, and M. Younis, "Coverage-based node placement optimization in wireless sensor network with linear topology," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–6.

[24] X. Kong, M. Li, T. Tang, K. Tian, L. Moreira-Matias, and F. Xia, "Shared subway shuttle bus route planning based on transport data analytics," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 4, pp. 1507–1520, Oct. 2018.

[25] X. Kong, M. Li, J. Li, K. Tian, X. Hu, and F. Xia, "CoPFun: An urban co-occurrence pattern mining scheme based on regional function discovery," in *Proc. World Wide Web*, May 2018, pp. 1–26.

[26] F. Xia, J. Wang, X. Kong, Z. Wang, J. Li, and C. Liu, "Exploring human mobility patterns in urban scenarios: A trajectory data perspective," *IEEE Commun. Mag.*, vol. 56, no. 3, pp. 142–149, Mar. 2018.

[27] V. Perlibakas, "Distance measures for PCA-based face recognition," *Pattern Recognit. Lett.*, vol. 25, no. 6, pp. 711–724, Apr. 2004.

[28] V. Chvatal, "A greedy heuristic for the set-covering problem," *Math. Oper. Res.*, vol. 4, no. 3, pp. 233–235, Aug. 1979.

[29] *Mapbox*. Accessed: Jul. 12, 2017. [Online]. Available: https://www.mapbox.com/mapbox-studio/

[30] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. KDD*, vol. 96, 1996, pp. 226–231.

[31] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A k-means clustering algorithm," *J. Roy. Stat. Soc. C, Appl. Statist.*, vol. 28, no. 1, pp. 100–108, 1979.

[32] J. Gan, J. Zhang, and S. Zheng, "Where you really are: User trip based city functional zone ascertainment," in *Proc. IEEE 37th Int. Perform. Comput. Commun. Conf. (IPCCC)*, Orlando, FL, USA, Nov. 2018, pp. 17–19.

[33] D. Golovin, M. Faulkner, and A. Krause, "Online distributed sensor selection," in *Proc. 9th ACM/IEEE Int. Conf. Inf. Process. Sensor Netw.*, New York, NY, USA, 2010, pp. 220–231.

[34] X.-Y. Li, P.-J. Wan, and O. Frieder, "Coverage in wireless ad hoc sensor networks," *IEEE Trans. Comput.*, vol. 52, no. 6, pp. 753–763, Jun. 2003.

[35] C. Torpelund-Bruin and I. Lee, "Generalized Voronoi diagrams with obstacles for use in geospatial market analysis and strategy decisions," in *Proc. Int. Workshop Educ. Technol. Training Int. Workshop Geosci. Remote Sens.*, Dec. 2008, pp. 287–290.

[36] R. Görke, C.-S. Shin, and A. Wolff, "Constructing the city Voronoi diagram faster," *Int. J. Comput. Geometry Appl.*, vol. 18, no. 4, pp. 275–294, Aug. 2008.

[37] M. Abo-Zahhad, N. Sabor, S. Sasaki, and S. M. Ahmed, "A centralized immune-Voronoi deployment algorithm for coverage maximization and energy conservation in mobile wireless sensor networks," *Inf. Fusion*, vol. 30, pp. 36–51, Jul. 2016.

[38] M. Gen and L. Lin, *Genetic Algorithms*. Hoboken, NJ, USA: Wiley, 2008.

[39] M. Albareda-Sambola, E. Fernández, Y. Hinojosa, and J. Puerto, "The single period coverage facility location problem: Lagrangean heuristic and column generation approaches," *TOP*, vol. 18, no. 1, pp. 43–61, Jul. 2010.

[40] *PyMySQL's Documentation*. Accessed: 2016. [Online]. Available: http://pymysql.readthedocs.io/en/latest/

[41] Numpy Developers. *Numpy's Documentation*. Accessed: 2017. [Online]. Available: http://www.numpy.org/

[42] *Plotly's Documentation*. Accessed: 2017. [Online]. Available: https://pypi.python.org/pypi/plotly/

Jianhui Zhang (Member, IEEE) received the B.S. degree in mechanotronics and the M.S. degree in fluid mechanics from Northwestern Polytechnical University, China, in 2000 and 2003, respectively, and the Ph.D. degree in control theory and engineering from Zhejiang University, Hangzhou, China, in 2008. He is currently working as a Full Professor with the School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou. He is also the Leader of the Group of Internet of Things, including seven faculties and more than 50 graduated students. His research interests include the Internet of Things, machine learning, wireless mobile sensing, and city computing.

Wanqing Zhang received the B.S. degree in electronic information science from Nankai University Binhai College, Tianjin, China, in 2019. She is currently pursuing the M.S. degree in computer technology with Hangzhou Dianzi University, Hangzhou, China. Her current research interests include battery cabinet deployment and city computing.

Jiacheng Wang received the B.S. degree in electronic information science and technology from the School of Electronics and Electrical Engineering, Wenzhou University, Wenzhou, China, in 2020. He is currently pursuing the M.S. degree in computer technology with Hangzhou Dianzi University, Hangzhou, China. His research interests include swarm learning, edge computing, and security.

Jianwen Feng received the B.S. degree in computer and application from Hangzhou Dianzi University, Hangzhou, China, in 1993, and the M.S. degree in measurement technology and instruments from Zhejiang University, Hangzhou, in 1999. She is currently working as a Professor with the School of Computer Science and Technology, Hangzhou Dianzi University. Her research interests include embedded systems and system integration.

Zhigang Gao received the B.S. degree in physics and the M.S. degree in computer application from Lanzhou University, China, in 1996 and 2000, respectively, and the Ph.D. degree in computer science and technology from Zhejiang University, Hangzhou, China, in 2008. He is currently working as a Full Associate Professor with the School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou. His research interests include the Internet of Things, machine learning, and mobile computing.

Siwen Zheng received the M.S. degree from Hangzhou Dianzi University, Hangzhou, China, in 2019. She is currently working with Huawei Technologies Co., Ltd, Hangzhou. Her research interests include city computing and crowdsourcing.