FROST: Movement History–Conscious Facility Relocation

MENG WANG, Xi'an Polytechnic University
HUI LI, Xidian University
JIANGTAO CUI, Xidian University
SOURAV S. BHOWMICK, Nanyang Technological University
PING LIU, Xidian University

The facility relocation (FR) problem, which aims to optimize the placement of facilities to accommodate the changes of users' locations, has a broad spectrum of applications. Despite the significant progress made by existing solutions to the FR problem, they all assume each user is stationary and represented as a single point. Unfortunately, in reality, objects (e.g., people, animals) are mobile. For example, a car-sharing user picks up a vehicle from a station close to where he or she is currently located. Consequently, these efforts may fail to identify a superior solution to the FR problem. In this article, for the first time, we take into account the movement history of users and introduce a novel FR problem, called MOTION-FR, to address the preceding limitation. Specifically, we present a framework called FROST to address it. FROST comprises two exact algorithms: *index based* and *index free*. The former is designed to address the scenario when facilities and objects are known *a priori*, whereas the latter solves the MOTION-FR problem by jettisoning this assumption. Further, we extend the index-based algorithm to solve the general k-MOTION-FR problem, which aims to relocate k inferior facilities. We devise an approximate solution due to NP-hardness of the problem. Experimental study over both real-world and synthetic datasets demonstrates the superiority of our framework in comparison to state-of-the-art FR techniques in efficiency and effectiveness.

CCS Concepts: • **Human-centered computing** → **Ubiquitous and mobile computing**; • **Information systems** → *Decision support systems*; Spatial-temporal systems;

Additional Key Words and Phrases: Facility relocation, movement history, spatial database

ACM Reference format:
Meng Wang, Hui Li, Jiangtao Cui, Sourav S. Bhowmick, and Ping Liu. 2020. FROST: Movement History–Conscious Facility Relocation. *ACM Trans. Intell. Syst. Technol.* 11, 1, Article 9 (January 2020), 26 pages.
https://doi.org/10.1145/3361740

The work was partially completed when M. Wang was a Ph.D. student at Xidian University, Xi'an, China.

This work was supported by the National Natural Science Foundation of China (nos. 61672408, 61976168, 61972309), Key Research and Development Plan of Jiangxi (no. 20181ACE50029) and Shaanxi (no. 2019ZDLGY13-09), and the Ph.D. Fund of Xi'an Polytechnic University (no. BS201919).

Authors' addresses: M. Wang, School of Computer Science, Xi'an Polytechnic University, No. 19 Jinhua South Road, Xi'an, Shaanxi 710048, China; email: wamengit@sina.com; H. Li, School of Cyber Engineering and State Key Laboratory of Integrated Services Networks, Xidian University, 266 Xinglong Section of Xifeng Road, Xi'an, Shaanxi 710126, China; email: hli@xidian.edu.cn; J. Cui (corresponding author) and P. Liu, School of Computer Science and Technology, P.O. Box 160, Xidian University, No. 2 South Taibai Road, Xi'an, Shaanxi 710071, China; emails: cuijt@xidian.edu.cn, pliu_11@stu.xidian.edu.cn; S. S. Bhowmick, School of Computer Science and Engineering, Nanyang Technological University, Blk N4-2a-32, 50 Nanyang Ave, Singapore 639698, email: assourav@ntu.edu.sg.

2157-6904/2020/01-ART9 $15.00
https://doi.org/10.1145/3361740

1 INTRODUCTION

The *facility relocation* (FR) problem aims to reallocate facilities in light of change in users' locations. It is useful in many applications, ranging from urban planning to marketing to logistics. For instance, when a new subway line is launched, many people may resettle to different locations. As a result, facilities may need to be reallocated. Several studies [9, 12, 16, 17, 23] have been undertaken to solve the FR problem under the *Min-dist* criterion with a variety of constraints. Specifically, given a set of users U, existing facility locations F, and a set of new locations C, assume that each user is associated with a nearest facility in F. It is rational to expect that the distance between each user and his or her nearest facility is minimized. Suppose now that we intend to replace an arbitrary facility $f \in F$ with a new location $c \in C$ so that f can be relocated to a more attractive location c. Observe that there are $|F| \cdot |C|$ potential choices. The *Min-dist-based FR problem* [16] aims to find an optimal $\langle f, c\rangle$ pair over all of these potential choices such that the average distance between all users and their nearest facilities is minimized.

Existing solutions to the FR problem assume that each user is stationary and is represented as a single point. However, with increasing mobility of end users and proliferation of mobile applications, it is realistic to assume that a user (i.e., customer) is mobile and can be represented by a series of spatial positions. For example, suppose a car-sharing company (e.g., *Autolib*, *Car2Go*) wishes to improve customer experiences by making its vehicles more accessible to users. To this end, the company needs to optimize the existing service network (i.e., parking and charging stations) by relocating an "inferior" station (i.e., location) by substituting it from a collection of potential new candidates (locations) such that the average pickup distance for users can be minimized. Naturally, if a user is mobile, then he or she would probably pick up a car from a station close to where he or she is currently located. Consequently, for a more effective solution, evaluation of the average distance for the FR problem cannot be solely attributed to a single location of a user. It is important to consider the set of locations associated with a user's *movement history* to identify a suitable new location. In addition, there exist two other limitations attributed to the assumptions in most traditional FR techniques: the numbers of users and facilities are limited (e.g., tens of them), and the distances between users and facilities are known in advance. For a small number of objects, efficiency is not a critical issue; however, it cannot be ignored for massive car-sharing users. For the latter, it is not always feasible to pre-determine the distance matrix for mobile people. It is not difficult to comprehend that there are several other potential applications similar to the aforementioned scenario. For instance, the FR problem is useful for a more efficient network of delivery terminals for a logistics company or for branches of a franchise chain or bank to be closer to attract more customers, and so forth.

Motivated by these scenarios, in this article we present a novel Min-dist-based FR problem, called MOTION-FR (*mo*vemen*t* h*i*story–c*on*scious *f*acility *r*elocation), that takes into account the movement history of users to relocate a facility. Since the majority of users' movements between locations in the real world are usually confined to roads, we focus on the road network distance and assume that both facilities and users are located on a road network denoted as $G(V, E)$. Additionally, as stated in Qi et al. [15, 16], in many real applications, companies can only choose from a finite number of candidate places for rent or sale in a region or on a road. In this article, we also follow this setting. Consequently, the MOTION-FR problem can be intuitively defined as follows. Given a set of mobile objects (e.g., users) U, a set of existing locations F for a specific kind of facility (e.g., stations, service branches), a candidate location set C, and a road network $G(V, E)$, the goal of the MOTION-FR problem is to find the optimal facility-candidate pair such that if we relocate the facility from its current location to a substitute location

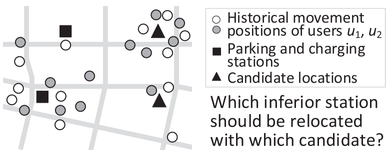


Fig. 1. A motivating example.

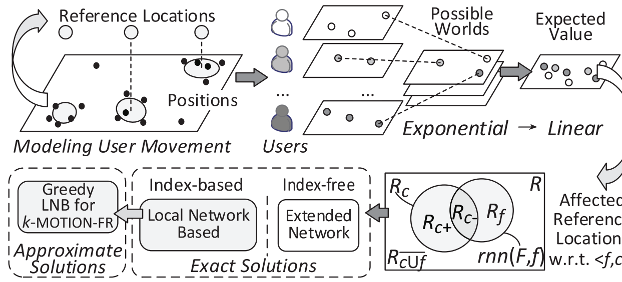


Fig. 2. Framework of FROST.

chosen from C, the average network distance between all objects and their nearest facilities is minimized.[1]

Figure 1 illustrates the MOTION-FR problem. The goal of the FR problem is to relocate an existing facility (in solid squares) to a new location (candidate locations are shown in solid triangles) such that the average distance for each user and its nearest facility is minimized. Existing FR solutions address this by considering the users (e.g., u_1, u_2) as stationary objects. Consequently, each user is represented by a single static position. In contrast, in our MOTION-FR problem, we exploit the movement history of users (white and gray circles) to determine FR.

Compared to existing FR solutions under the Min-dist criterion, the MOTION-FR problem brings in three novel challenges. First, a mobile object is described by a set of spatial positions instead of a single one [21]. Consequently, this leads to large amount of positions, which may demand costly computation. Furthermore, considering every position into account may generate inferior solutions, as some positions may be *erroneous* or *insignificant* to the relocation problem. For instance, there may be positions with GPS errors or visited by a user only occasionally. Hence, it is vital to identify and eliminate these noisy positions during computation of average distance. Second, even if we have garnered knowledge of relevant positions for each user's movement history, it is nontrivial to determine which facility a specific user will visit for service. Consequently, the average distance is difficult to evaluate. Finally, the computational overhead can be prohibitively expensive due to massive network distance computations over large number of users, existing facilities, and candidate locations.

In this work, we propose a systematic framework called FROST (*f*acility *r*elocation using m*o*vement hi*st*ory) to address the preceding challenges. Figure 2 depicts the architecture of FROST to address the MOTION-FR problem and consists of three components as follows. In the first component, we model user movement by utilizing the concept of *reference locations* [13]. Each reference location is one of the activity places to which a user frequently shows up every day and thus is worth considering. We can identify for each user multiple reference locations with different presence probabilities by exploiting the *kernel method* [24].

The second component formalizes the MOTION-FR problem. In light of the uncertainty of movements, users who are probably present at any one of the reference locations can be modeled following the *possible world* semantics [2]. Based on the criterion of the Min-dist FR problem (i.e., stationary objects), the optimal answer concerning a specific possible world can be obtained. As any possible world may occur, all possible worlds influence the result. Consequently, the average

[1]In this work, we focus on optimizing the generic network distance and ignore other application-dependent factors (e.g., policy, open/close cost, relocate cost) for various scenarios. In fact, these factors can be easily incorporated as a postprocessing step by a top-k version of our proposed solution.

distance is, in fact, a random variable. Hence, all facility-substitute pairs can be ranked according to the expected average distance over all possible worlds.

In the last component, we systematically solve the proposed MOTION-FR problem. As the number of possible worlds is huge due to massive number of users, it is impractical to explore all of them due to its exponential complexity. Hence, to avoid exhaustive enumeration, we design a *reference location–based policy*, which can be effectively leveraged to reduce the ranking process from exponential complexity to a linear one. Furthermore, city road networks usually have large volumes of vertices and edges, which may result in costly network traversals in finding the optimal answer for users. To address this problem, we propose an approach with the help of *network locality* that provides superior efficiency. In addition, we present an alternative solution based on a *network extension* scheme for addressing the MOTION-FR problem from scratch, in the case that existing facilities and users are unknown in advance.

We further generalize the problem to relocating k FR pairs that collectively achieve the minimum average distance. As the problem is NP-hard [23], we devise an approximate solution to the problem. In summary, the major contributions of this article are as follows:

- We present a novel framework called FROST to address the problem of MOTION-FR. To the best of our knowledge, this is the first effort to consider the movement history of objects to address the FR problem.
- We present two novel solutions to address the MOTION-FR problem efficiently under different scenarios and devise an approximate solution to address the general k-MOTION-FR problem.
- Comprehensive experiments are conducted on real and synthetic data over two real road networks. The results demonstrate that, compared to state-of-the-art FR techniques, our proposed solutions significantly improve the efficiency by orders of magnitude and provide more effective answers.

The rest of this article is organized as follows. We give an overview of related work in Section 2. In Section 3, we discuss user movement history and model mobile objects using reference locations. Section 4 formally gives the definition of the MOTION-FR problem. In Section 5, two methods are proposed to solve MOTION-FR offline and online, respectively. An approximate solution for the general k-MOTION-FR problem is designed in Section 6. Section 7 analyzes the costs of the solutions. Section 8 reports the experimental study, and Section 9 concludes this article. The set of frequently used notations in this work is given in Table 1 for easy reference.

2 RELATED WORK

Min-dist. The Min-dist problem, originated from the k-median problem [1], aims at minimizing the average/sum distance between users and their nearest facilities. Zhang et al. [30] studied a typical Min-dist problem with existing facilities in continuous Euclidean space. Using lower-bound estimators for pruning, they progressively divided a given region until the answer was identified. Xiao et al. [26] took into account road distance in the Min-dist problem. They proposed a divide-and-conquer–based framework to split road network into sub-networks, among whose local optima the global answer was derived. Chen et al. [3] presented a pruning technique based on *nearest location component* (an alias of *local network* [7]) to solve the problem and two extensions in continuous search space. However, as discussed in Qi et al. [15, 16], we can only choose from some locations for rent or sale rather than anywhere in real applications. To this end, Qi et al. [15] studied the problem with an extra set of candidate locations for establishing a new facility. Index-based methods were developed to solve the problem on an L_2 distance metric. Cui et al. [4] made an effort toward the Min-dist problem with candidates on a road network. Utilizing spatial

Table 1. Notations

Notation	*Definition*
$d(\ell_1, \ell_2)$	Network distance from location ℓ_1 to ℓ_2
$d_\varepsilon(\ell_1, \ell_2)$	Euclidean distance between ℓ_1 and ℓ_2
$nn(F, \ell)$, $2nn(F, \ell)$	Network nearest and sub-nearest facilities of ℓ chosen from F
$d_{nn}(F, \ell)$, $d_{2nn}(F, \ell)$	Network distances from ℓ to $nn(F, \ell)$ and $2nn(F, \ell)$
$rnn(F, f)$	Set of users (reference locations) whose nearest facilities are f among F
$Pr[l(u) = r]$, $p(r)$	Probability that u is present at r
$E[d_{nn}(F, L(u))]$	Expected nearest facility distance of user u
$\langle f, c\rangle$	An FR pair
$\Delta(\langle f, c\rangle)$	Expected change of total distance (*ED*) when $\langle f, c\rangle$ is carried out
$\delta_r(\langle f, c\rangle)$	Expected change of nearest facility distance of r when $\langle f, c\rangle$ is carried out
R_c, R_f	Reference locations whose nearest facilities are c or $2nn(F, \ell)$ for $\langle f, c\rangle$
R_{c+}, R_{c-}	Sets of reference locations where $\delta_r(\langle f, c\rangle) \geq 0$ and $\delta_r(\langle f, c\rangle) < 0$, respectively
$\Delta^+(v_i v_j)$	ED upper bound on $v_i v_j$
$v_i v_j, r\overrightarrow{v_i v_j}$	Edge with endpoints v_i and v_j, and a directed edge with respect to r, respectively
$\sigma^+(r\overrightarrow{v_i v_j}), \sigma^-(r\overrightarrow{v_i v_j})$	Maximum and minimum reductions of distance on $r\overrightarrow{v_i v_j}$ for any candidate

locality–based indices in both Euclidean space and road network, the problem can be addressed efficiently. Nevertheless, these approaches cannot be directly applied to our problem due to the following limitations. First, the distance calculation in the road network is more complex than in Euclidean space, and the techniques developed for Euclidean distance are not efficient for road network, which can be indicated by the experiments on the state of the art [16] (detailed in Section 8.2). Second, in general cases, results derived under continuous search space may not even be contained in the given candidates, and thus the approaches may not provide correct answers to our problem. Third, as discussed in Qi et al. [16], the distance change depends on the relative positions of users, the obsolete facility, and a substitute in the FR problem, which is more complex than the Min-dist criterion.

Papadias et al. [14] focused on a variation of the Min-dist problem, which finds an optimal facility among candidates to serve all users with the minimum average distance. Their method was for both memory and disk-resident queries in Euclidean space. Yiu et al. [29] solved the problem on the road network utilizing network connectivity information and spatial locality. Yan et al. [27] presented a two-phase convex-hull-based pruning technique for both exact and approximate solutions. There also exist research works [1, 10] taking movement into consideration. Bespamyatnikh et al. [1] focused on the 1-median problem. Based on kinetic data structures, they presented lower bounds and algorithms for exact and approximate solutions. Khan et al. [10] studied the problem with users' paths, where the distance from a facility to a path was defined as the minimum distance to any of its line segments. The proposed quadtree-based solution hierarchically pruned irrelevant facilities and line segments of paths. None of these techniques can address the problem we propose due to the following fundamental differences. First, it is a set of facilities that collectively serve all users in our problem, whereas these works focused on only a single one. Second, the problems either treated movement as static points in every time interval [1] or only considered the nearest sample point of a path [10]. These models cannot be used to represent the entire movement history.

Facility relocation. FR problems aim to minimize the total distance between users and facilities by replacing some existing facilities with substitutes, under a variety of constraints. Despite the same (Min-dist-based) criterion, the FR problem is more complicated than the Min-dist problem, as the obsolete facility f and picked candidate c in an $\langle f, c\rangle$ pair play their roles together in distance evaluation (detailed in Section 5.1). Wang et al. [23] first introduced the FR problem and clarified the difference compared to the k-median problem. Three approximate solutions were proposed for solving the FR problem. Sonmez and Lim [17] took into account weight changes of users. Based on an integer programming model, a decomposition algorithm was designed for near-optimal solutions. Turanoglu and Akkaya [20] studied an extension called the *dynamic facility layout problem* (DFLP) where weights of users change over time. A hybrid heuristic was proposed that was a simulated annealing algorithm based on bacterial foraging optimization. Goranci et al. [8] further extended the DFLP in metric space where users may change their positions. Based on the coverage radii of facilities, they designed a tree-based hierarchical decomposition method. These efforts cannot be available to MOTION-FR for the following reasons. First, the distance matrix between the user and facility was given as a known parameter, whereas in our problem it is unknown and the distance evaluation is a critical issue that needs to be settled. Second, in these studies, as the number of users and facilities were limited (e.g., tens of them) and the efficiency was not the focus, their algorithms cannot guarantee the efficiency for a large scale of users. Third, they took into account not only the transportation distance/cost between users and facilities but also the relocation cost of facilities.

Li et al. [12] studied another kind of FR problem in which all facilities were relocated. They proposed a fast PAM-based refinement for the problem. Halper et al. [9] extended the problem by taking the distances/costs traveled by facilities into account. Integer programming formulation and local search heuristics were utilized. In Farahani et al. [6], there was only one facility, and the resettlement could be conducted several times over time. As relocating all facilities deviates from our problem setting (similar to the 1-median problem), these approaches cannot be applied to solve MOTION-FR.

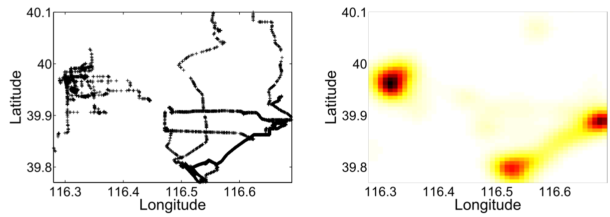
Qi et al. [16] studied an FR problem in Euclidean space with the same optimization objective as ours. The authors designed methods based on spatial locality to restrict the search space. Unlike in Euclidean space, where the aim is to decrease the number of facility-candidate pairs to be calculated, the key challenge in the road network is to reduce the number of network traversals, due to the expensive computation cost. As empirical studies shown in Section 8, their proposed techniques are adapted for solving our problem by replacing with network distance; however, they are not as efficient as for Euclidean space. Hence, techniques for the road network are required.

3 PRELIMINARIES

In this section, we first discuss the representation of mobile objects and then introduce the *kernel method*, which is employed to model movements of users in our framework.

3.1 Mobile Objects

In general, mobile objects (e.g., people, animals) are ubiquitous in real-life applications. The movement of a user is commonly represented in two ways: *continuous* (e.g., trajectory [31]) and *discrete* (e.g., check-ins [28] and lifelogs [22]). The former explicitly records every move of an object, whereas the latter works in an implicit way, which reflects activities at locations and movements between them. For both cases, a moving user is modeled as a set of positions [21] (e.g., sample points of a trajectory or check-ins/lifelogs at locations). Nevertheless, in addition to the costly computation, taking every position into account is inappropriate due to three kinds of valueless points, namely *noisy*, *passing-by*, and *outlier points*. Noisy points result from data or GPS errors, whereas the latter two are relevant to mobility. Compared to places where daily activities are conducted, passing-by points are those where a user does not spend any time except passing by. In



(a) Positions of a user. (b) Kernel distribution.

Fig. 3. Capturing reference locations.

addition, outlier points are those visited by users occasionally. These points are either not contributive or poorly contributive to our goal to determine which facility is inferior and where is optimal to build a substitute. Hence, it is more reasonable to focus on *representative points*, which are more effective in describing the frequent activity areas of a user.

On the other hand, as remarked in Li et al. [13], moving users are intimately associated with two major behaviors: frequently appearing at some places and staying for a duration. If we incorporate Tobler's first law of geography ("near things are more related than distant things") [19], then we can characterize the mobility behavior of users as follows. First, a user usually correlates to some reference locations [13]. Second, a user tends to conduct his or her activities nearby the reference locations [28].

In summary, it makes sense to identify reference locations from raw movement history data, which enables us to eliminate the aforementioned limitations and pave the way to effectively capture daily activity places for handling the MOTION-FR problem.

3.2 Capturing Reference Locations

We employ the *kernel method* [24] to capture reference locations of each user. The kernel method has been widely used in a variety of domains, including the detection of frequent activity places for humans (the accuracy is up to 92.3%) [18]. In this study, we use standard bivariate normal density kernel, which is $f(\langle x, y\rangle) = \frac{1}{nh^2}\sum_{i=1}^{n}\frac{1}{2\pi}\exp(-\frac{d_\varepsilon(\langle x,y\rangle,\langle x_i,y_i\rangle)}{2h^2})$, where $f(\langle x, y\rangle)$ is the estimated probability density function, $\langle x_i, y_i\rangle$ is a sample position ($i \in [1, n]$) of an object, $d_\varepsilon(\langle x,y\rangle,\langle x_i,y_i\rangle)$ denotes the Euclidean distance, and h is the smoothing parameter. We employ a popular method introduced by Worton [24] to choose h: $h = \frac{1}{2}(\varsigma_x^2 + \varsigma_y^2)^{\frac{1}{2}} n^{-\frac{1}{6}}$, where ς_x and ς_y are the standard deviations of x_i and y_i.

In line with Li et al. [13], we capture reference locations of each user as follows. We discretize the continuous space into grids and evaluate the density for each of them. The top-p% (5% in this article) grids with the highest density values are selected. Inspired by the notion of a *reference spot* [13], in this work we aggregate the adjacent grids among the selected ones together to form a series of separate grid groups. In each group, the peak grid with the highest density is intuitively regarded as a reference location. Moreover, the density accumulation of each group is normalized and viewed as the presence probability that a user appears nearby the corresponding reference location. Figure 3(a) and (b) illustrate positions of a user and the probabilities of grids calculated using the kernel method. Based on the kernel method and grid group strategy, the valueless points are avoided and three of his or her reference locations are captured.

In practice, reference locations usually can be reached by a road network. This indicates that they are located on edges of a given network $G(V, E)$. If this is not the case, then we assume that a reference location is located at its nearest point in G. This assumption is reasonable, as residential and office buildings may not exactly be located on roads in G.

4 PROBLEM DEFINITION

In this section, we formally define the MOTION-FR problem studied in this article. We begin by introducing some terminology that is necessary to facilitate exposition.

A *location* ℓ in this work is a planar position on an edge in a given directed road network $G(V, E)$, with a geographical coordinate (i.e., latitude and longitude). Each directed edge in E between a pair of vertices in V is associated with a positive weight, which represents travel distance or time cost, and so forth. Given any two locations ℓ_1 and ℓ_2, the *directed network distance* from ℓ_1 to ℓ_2 is denoted by $d(\ell_1, \ell_2)$, which may not be equal to $d(\ell_2, \ell_1)$. Since the locations of existing facilities (respectively, candidates to deploy a substitute) can usually be obtained precisely, we denote *facilities* (respectively, *candidates*) as a set of locations $F = \{f_1, f_2, \ldots, f_{|F|}\}$ (respectively, $C = \{c_1, c_2, \ldots, c_{|C|}\}$), where $|F|$ (respectively, $|C|$) is the cardinality of F (respectively, C). For a user at location ℓ, the network nearest and sub-nearest (i.e., second nearest) facilities with respect to F are denoted as $nn(F, \ell)$ and $2nn(F, \ell)$, and the network distances from ℓ to them are defined as $d_{nn}(F, \ell) = d(\ell, nn(F, \ell))$ and $d_{2nn}(F, \ell) = d(\ell, 2nn(F, \ell))$, respectively.[2] Conversely, we denote the set of users whose nearest facilities are f among F as $rnn(F, f)$. An *FR pair* that consists of an obsolete facility $f \in F$ and a candidate $c \in C$ for substitution is defined as $\langle f, c\rangle$. For a user located at ℓ, if $\langle f, c\rangle \in F \times C$ is carried out, the distance to his or her nearest facility will become $d_{nn}(F \backslash \{f\} \cup \{c\}, \ell)$.

To minimize the average distance between users and their respective nearest facilities, we have to identify their locations first. However, as discussed in Section 3, a mobile object u may be present at a set of n_u reference locations $L(u) = \{r_1, r_2, \ldots, r_{n_u}\}$. Let $l(\cdot)$ denote a reference location(s) where "·" is(are) present, then $l(u) \in L(u)$ and $\sum_{i=1}^{n_u} Pr[l(u) = r_i] = 1$. Observe that this differs from the classical Min-dist criterion [15], where each object has only a single location. How can we select the Min-dist FR pair given that objects are mobile?

Since the movements of users are uncertain, they can be described by following the *possible world* semantics. Given a set of m users $U = \{u_1, u_2, \ldots, u_m\}$, each user u_i is associated with reference locations $L(u_i)$, then a *possible world* $w = (r_1^w, r_2^w, \ldots, r_m^w)$ is a list of location instances with one instance for each user, where $r_i^w \in L(u_i)$. Assume that the reference locations of users are independent from each other, then $Pr[l(U) = w] = \prod_{i=1}^{m} Pr[l(u_i) = r_i^w]$. Let W be all possible worlds, then $|W| = \binom{|L(u_1)|}{1} \ldots \binom{|L(u_m)|}{1} = \prod_{i=1}^{m} |L(u_i)|$. Obviously, $\sum_{w \in W} Pr[l(U) = w] = 1$.

Note that for a particular possible world w, each user is associated with only a single reference location. This is in line with the setting of the Min-dist FR problem.

Definition 4.1. Given a facility set F, a possible world w, and a FR pair $\langle f, c\rangle$, the *change of total distance* between all users U and their respective nearest facilities is defined[3] as

$$\Delta_w(\langle f, c\rangle) = \sum_{i=1}^{m} (d_{nn}(F, r_i^w) - d_{nn}(F \backslash \{f\} \cup \{c\}, r_i^w)).$$

$\Delta_w(\langle f, c\rangle)$ can be an alternative to evaluate the distance change on average (i.e., $\Delta_w(\langle f, c\rangle)/m$), as the denominator m is consistent for every pair $\langle f, c\rangle \in F \times C$. Hence, the FR pair with the maximal $\Delta_w(\langle f, c\rangle)$ is the optimum in w with respect to the Min-dist criterion.

Figure 4(a) depicts three existing facilities $F = \{f_1, f_2, f_3\}$, two candidates $C = \{c_1, c_2\}$, and three users $U = \{u_1, u_2, u_3\}$, and each of them has two reference locations. The numbers on the edge segments represent the travel costs. Figure 4(b) shows a possible world $w = (r_{11}, r_{22}, r_{32})$ with FR

[2]Without ambiguity, in this article the directed network distance and network nearest (respectively, sub-nearest) facility are referred to as distance and nearest (respectively, sub-nearest) facility for simplicity.
[3]For clarity and brevity, we omit F and U in the representation, as they are regarded as default settings in this work.

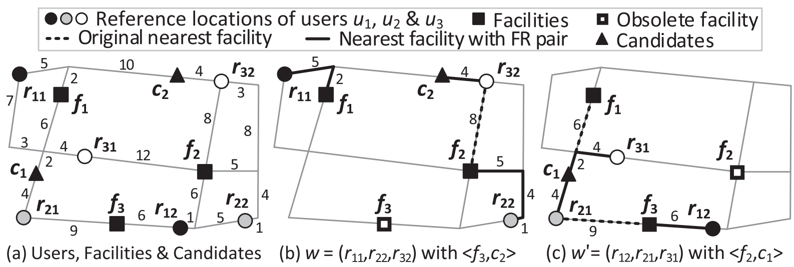


Fig. 4. Changes to total distance in possible worlds.

pair $\langle f_3, c_2 \rangle$. According to Definition 4.1, $\langle f_3, c_2 \rangle$ is the optimal solution in w among all FR pairs ($\Delta_w(\langle f_3, c_2 \rangle) = 4$). Unfortunately, there are seven other possible worlds, and $\langle f_3, c_2 \rangle$ may not always be the optimum. For example, in $w' = (r_{12}, r_{21}, r_{31})$, users fail to benefit from $\langle f_3, c_2 \rangle$. Instead, another FR pair $\langle f_2, c_1 \rangle$ is optimal ($\Delta_{w'}(\langle f_2, c_1 \rangle) = 9$). Accordingly, we need a mechanism to rank FR pairs taking all possible worlds into account. As the distribution of Δ_w for any FR pair is a random variable, it is reasonable to evaluate the expected value over all possible worlds.

Definition 4.2. Given a set of facilities F and a set of moving users U for all possible worlds W, the *expected change of total distance* (*ED*) with respect to a FR pair $\langle f, c \rangle$ is defined as

$$\Delta(\langle f, c \rangle) = \sum_{w \in W} (\Delta_w(\langle f, c \rangle) \times Pr[l(U) = w]).$$

We are now ready to define the MOTION-FR problem addressed in this article that incorporates the concepts of reference location and the expected Min-dist criterion following possible world semantics.

Definition 4.3. Given a directed road network G and a set of users U, each of whose movement can be modeled as a set of reference locations, the *movement history–conscious facility relocation* (MOTION-FR) problem aims to find an FR pair $\langle f, c \rangle$ among a set of existing facilities F and a set of candidate locations C such that $\langle f, c \rangle_{OPT} = \text{argmax}_{\langle f, c \rangle \in F \times C} \Delta(\langle f, c \rangle)$.

To find the optimal FR pair with the highest *ED*, we need to calculate $\Delta_w(\langle f, c \rangle)$ for every $w \in W$, which requires enumerating all possible worlds. Unfortunately, since $|W|$ increases exponentially with $|U|$, it is impractical to directly compute *ED* using Definition 4.2. To address this challenge, next we show how the *ED* computation can be transformed from the aspect of reference locations and can be completed in polynomial time.

Definition 4.4. Let r^w be the reference location of a mobile user u in a possible world w. Then the *expected nearest facility distance* of u with respect to F in all possible worlds W is defined as

$$E[d_{nn}(F, L(u))] = \sum_{w \in W} (d_{nn}(F, r^w) \times Pr[l(U) = w]).$$

LEMMA 4.5. $E[d_{nn}(F, L(u))] = \sum_{r \in L(u)} (d_{nn}(F, r) \times p(r))$.[4]

THEOREM 4.6. *Given a set of facilities F, a set of users U, and a FR pair $\langle f, c \rangle$, the expected change of total distance in Definition 4.2 can be computed as* $\Delta(\langle f, c \rangle) = \sum_{i=1}^{m} (E[d_{nn}(F, L(u_i))] - E[d_{nn}(F \backslash \{f\} \cup \{c\}, L(u_i))])$.

In light of the assumption that reference locations of users are independent with each other, Theorem 4.6 can be derived, which means that we no longer need to enumerate all possible worlds and

[4]In the following, for brevity we shall use $p(r)$ to stand for $Pr[l(u) = r]$.

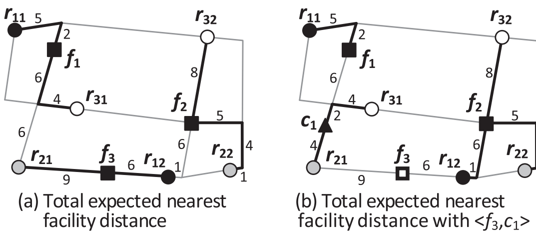


Fig. 5. ED with $\langle f_3, c_1 \rangle$.

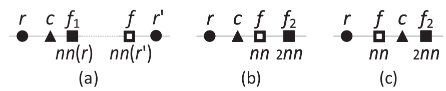


Fig. 6. Relationships between f and c.

the complexity of *ED* significantly drops to linearity. Suppose that each reference location in Figure 4(a) occurs with the same probability, say 0.5. As illustrated in Figure 5(a), $\sum_{i=1}^{3} E[d_{nn}(F, L(u_i))]$ can be easily obtained as 25; in Figure 5(b), we can compute $\sum_{i=1}^{3} E[d_{nn}(F\backslash\{f_3\} \cup \{c_1\}, L(u_i))] = 21$, where a substitute is built at c_1 to replace f_3. In this case, $\Delta(\langle f_3, c_1 \rangle) = 4$. Similarly, we evaluate Δ for each of the FR pairs, and the one with highest expected value will be selected as the optimal solution (i.e., $\langle f, c \rangle_{OPT}$) for the MOTION-FR problem.

Extensions to the MOTION-FR problem. The MOTION-FR problem can be further extended by appending context information to reference locations, which could be useful in practice. First, facilities have obvious temporal character in some scenarios (e.g., people usually go to cinemas or bars in the evening). By filtering out the positions that are temporally irrelevant, we can extract *time-dependent reference locations* for time-related facilities. Another extension is based on behavior patterns of users. For example, office workers are more likely to go to restaurants for lunch from their offices rather than from homes. That means that the probabilities to visit some specific type of facilities vary with the places at which a user is present. Given a facility type T, the transition probability to visit facilities of type T from reference location r_i, denoted by $Pr_i(T)$, can be simply evaluated[5] as $Pr_i(T) = \frac{\text{\# of visits to facilities of type } T \text{ from } r_i}{\text{\# of total visits from } r_i}$. Based on the Markov model, we can normalize $\frac{Pr[l(u)=r_i]\cdot Pr_i(T)}{\sum_{i=1}^{n} Pr[l(u)=r_i]\cdot Pr_i(T)}$ as the *weighted presence probabilities* for the specific facility type T.

In the following section, we will present solutions to address the MOTION-FR problem, which are generalized and can directly be applied for such extensions.

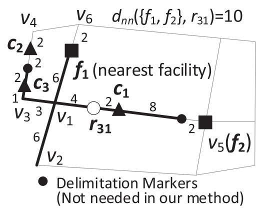
5 INDEX-BASED AND INDEX-FREE SOLUTIONS

A straightforward solution to the MOTION-FR problem based on Theorem 4.6 and Definition 4.3 is to check all FR pairs exhaustively. Specifically, for each $\langle f, c \rangle \in F \times C$, we compute its $\Delta(\langle f, c \rangle)$. Then the FR pair with the greatest *ED* is the optimal answer. Despite the avoidance of enumerating all possible worlds, this method is still expensive due to the massive network traversals that occur due to repeatedly finding the nearest facility of each reference location for every FR pair. In this section, we present two novel techniques for solving the MOTION-FR problem with superior efficiency. We begin by first discussing the relationship between f and c in terms of reference locations.

5.1 Relationship between f and c

As discussed in Section 4, if the reference location r for user u shifts to $nn(F\backslash\{f\} \cup \{c\}, r)$ as its new nearest facility with respect to $\langle f, c \rangle$, then we denote *expected change of nearest facility distance of* r as $\delta_r(\langle f, c \rangle)$. Note that this is computed as $\delta_r(\langle f, c \rangle) = (d_{nn}(F, r) - d_{nn}(F\backslash\{f\} \cup \{c\}, r)) \times p(r)$. Then, for a specific FR pair $\langle f, c \rangle$, we can divide all reference locations $R = \bigcup_{u \in U} L(u)$ into three groups as follows based on their changes to nearest facilities:

[5]The probability calculation is beyond the scope of this work. We only give a straightforward idea to evaluate.



(a) Local Network of r_{31}

v_iv_j	$\Delta^+(v_iv_j)$	$Pr[l(u)=r]$	σ^+	σ^-	$d(\ell_c, v_j)$
v_5v_1	5 + ...	$p(r_{31})$=0.5	10	6	0
		...	...	...	...
v_1v_5	5 + ...	$p(r_{31})$=0.5	10	-2	0
		...	...	...	...
v_1v_6	3 + ...	$p(r_{31})$=0.5	6	-2	0
		...	...	...	...
v_1v_2	3 + ...	$p(r_{31})$=0.5	6	0	0
		...	...	...	...
v_1v_3	3 + ...	$p(r_{31})$=0.5	6	3	0
		...	...	...	...
v_3v_4	1.5 + ...	$p(r_{31})$=0.5	3	-4	0
		...	...	...	...

(b) Local Network Table

Fig. 7. LNB method.

- $r \in R_c = rnn(F \backslash \{f\} \cup \{c\}, c)$: c will be the new nearest facility and $\delta_r(\langle f, c\rangle) = (d_{nn}(F, r) - d(r, c)) \times p(r)$.
- $r \in R_f = rnn(F, f) \backslash rnn(F \backslash \{f\} \cup \{c\}, c)$: Once f is removed, $2nn(F, r)$ will be the new nearest facility. Thus, $\delta_r(\langle f, c\rangle) = (d_{nn}(F, r) - d_{2nn}(F, r)) \times p(r)$.
- $r \in R_{\overline{f \cup c}} = R \backslash (rnn(F, f) \cup rnn(F \backslash \{f\} \cup \{c\}, c))$: The reference locations are unaffected by f or c, and therefore their nearest facilities are unchanged.

Note that R_f is not simply defined as $rnn(F, f)$ because $rnn(F, f) \cap R_c \neq \emptyset$ if f is exactly $nn(F, r)$ and $d(r, c) < d_{2nn}(F, r)$. This indicates that c will be the new nearest facility rather than $2nn(F, r)$ after removing f. Figure 6 illustrates three different relationships between f and c when $\langle f, c\rangle$ is carried out: (a) f and c respectively affect different sets of reference locations (i.e., $rnn(F, f) \cap rnn(F \cup \{c\}, c) = \emptyset$), (b) f is $nn(F, r)$ and $d(r, c) \leq d_{nn}(F, r)$, and (c) f is $nn(F, r)$ and $d_{nn}(F, r) < d(r, c) \leq d_{2nn}(F, r)$. By considering the value ranges of these cases, we further partition R_c into R_{c+} and R_{c-}. The former consists of cases (a) and (b) where $\delta_r(\langle f, c\rangle) \geq 0$. For $r \in R_{c+}$, c must be the new nearest facility, whether $nn(F, r)$ is f or not, then the value of $\delta_r(\langle f, c\rangle)$ is only relevant to c regardless of f. For the latter, R_{c-} is for case (c) where $\delta_r(\langle f, c\rangle)$ is negative.

Lemma 5.1. $\Delta(\langle f, c\rangle) = \sum_{r \in R_{c+} \cup R_{c-} \cup R_f} \delta_r(\langle f, c\rangle)$.

Reconsider Figure 5(b). If $\langle f_3, c_1\rangle$ is carried out, only r_{12}, r_{21}, and r_{31} change their nearest facilities: the sub-nearest facility f_2 for r_{12}, and candidate c_1 for r_{31} (case (a)) and r_{21} (case (b)), i.e., $R_{f_3} = \{r_{12}\}$, $R_{c_1+} = \{r_{21}, r_{31}\}$ and $R_{c_1-} = \emptyset$. Based on Lemma 5.1, $\Delta(\langle f_3, c_1\rangle) = \sum_{r \in \{r_{12}, r_{21}, r_{31}\}} \delta_r(\langle f_3, c_1\rangle) = 4$.

5.2 Local Network–Based Method

To avoid the repeat network traversals for every FR pair, we adapt the *local network* [7] for computing *ED* efficiently, which was proposed to model the network locality of a user with respect to his or her nearest facility. Specifically, it is a sub-network expanded from user location $l(u) = r$ with a distance less than or equal to $d_{nn}(F, r)$, denoted by $LN(r)$. For instance, the bold edge segments in Figure 7(a) represent $LN(r_{31})$. We present a *local network-based* (LNB) method that addresses the motion-fr problem by extending with local sub-networks and incorporating the distance factor to the network locality. Our approach differs from Ghaemi et al. [7] in the following ways. First, only existing facilities, rather than users, are settled on the road network to reduce network complexity. Second, we extend the concept of *local network* to *sub-nearest facility* for dealing with the reference locations in $R_{c-} \cup R_f$, which is beyond the problem setting in Ghaemi et al. [7]. Third, we introduce the notion of *expanding direction* of an edge to replace delimitation markers (as shown in Figure 7(a)), which are used to delimit the local networks [7]. Combining with expanding direction, we design two similar data structures to index local networks and local sub-networks, based on which LNB method can retrieve the optimal FR pair without any network traversal.

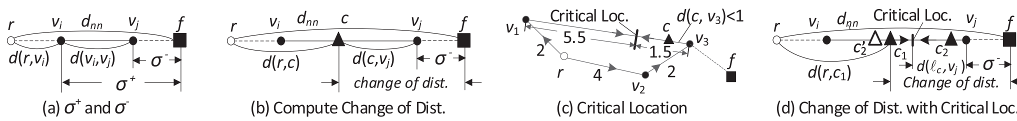


Fig. 8. Compute change of distance via σ^+ and σ^-.

According to Section 5.1, the positive return of *ED* is only brought by R_{c+} when $\langle f, c\rangle$ is launched, whereas R_{c-} and R_f lead to negative ones. In the following, we first focus on R_{c+} for the positive return.

Edge with expanding direction. As cases (a) and (b) shown in Figure 6, candidate c is nearer than original nearest facilities, then reference locations in R_{c+} can be obtained whose local networks enclose c. Furthermore, based on the expanding direction from r, edges covered by $LN(r)$ have lower and upper distances from r. Take c_3 in Figure 7(a) as an example. With the help of $LN(r_{31})$, we can easily derive that r_{31} is contained in R_{c_3+}. To evaluate the positive return brought by r_{31}, we must have the distance to c_3. When finding the shortest path by network traversal (e.g., the Dijkstra algorithm [5]), all edges in $LN(r_{31})$ have to be explored before c_3 is encountered. To completely avoid repeated traversals from r_{31} to candidates inside $LN(r_{31})$, we can make use of distance information in $LN(r_{31})$. Specifically, given the query candidate c_3 and its location on edge v_3v_4 (e.g., $d(v_3, c_3)$), if we are able to directly fetch the lower distance of v_3v_4 expanding from r_{31} (i.e., $d(r_{31}, v_3)$) $\delta_{r_{31}}(\langle f, c_3\rangle)$ can be immediately calculated as $(d(r_{31}, v_3) + d(v_3, c_3)) \times p(r_{31})$. To this end, we extend the local network with direction and distance information.

Given a reference location r and an undirected edge v_iv_j, if a network path expanded from r traverses the edge or part of it from v_i toward v_j, we denote the *directed edge with respect to r* as $r\overrightarrow{v_iv_j}$. For example, v_1v_3 and v_1v_5 are two edges in Figure 7(a). Expanding the local network of r_{31}, v_1v_3 is traversed from v_1 toward v_3, and thus the directed edge with respect to r_{31} is $r_{31}\overrightarrow{v_1v_3}$. The case of v_1v_5 is complicated. The network path expanded from r_{31} to f_1 traversing v_1v_5 is from v_5 toward v_1. Hence, we denote it as $r_{31}\overrightarrow{v_5v_1}$. In contrast, the directed edge is $r_{31}\overrightarrow{v_1v_5}$ from r_{31} to f_2.

Given $r\overrightarrow{v_iv_j}$ on which a candidate is built, we denote the *maximum and minimum changes of distance on* $r\overrightarrow{v_iv_j}$ as $\sigma^+(r\overrightarrow{v_iv_j})$ and $\sigma^-(r\overrightarrow{v_iv_j})$, respectively. In Figure 8(a), r accesses the nearest facility f traversing $r\overrightarrow{v_iv_j}$ (the general case that r expands its local network along $r\overrightarrow{v_iv_j}$ also holds), then $\sigma^+(r\overrightarrow{v_iv_j})$ and $\sigma^-(r\overrightarrow{v_iv_j})$ can be easily computed as $d_{nn}(F, r) - d(r, v_i)$ and $d_{nn}(F, r) - d(r, v_j)$. Recall that only R_{c+} contributes to a positive return and $\delta_{r\in R_{c+}}(\langle f, c\rangle)$ is only relevant to c regardless of f, which means that the *ED* upper bound of any FR pair only depends on c. Utilizing $\sigma^+(r\overrightarrow{v_iv_j})$, we define the *ED upper bound on* v_iv_j, denoted by $\Delta^+(v_iv_j)$, as the maximum positive return that can be achieved by any FR pair, if the candidate c is built on v_iv_j. Specifically, $\Delta^+(v_iv_j)$ is computed as $\sum_{r\in R_{c+}}(\sigma \times p(r))$, where R_{c+} consists of reference locations whose local networks overlap v_iv_j and

$$\sigma = \begin{cases} \sigma^+(r\overrightarrow{v_iv_j}) & \text{if } r\overrightarrow{v_iv_j}, \\ \sigma^+(r\overrightarrow{v_jv_i}) & \text{if } r\overrightarrow{v_jv_i}. \end{cases} \tag{1}$$

As there is no road of retrogression when expanding a local network, either of the cases in Equation (1) holds. On the other hand, based on $\sigma^-(r\overrightarrow{v_iv_j})$, if a candidate c is built on $r\overrightarrow{v_iv_j}$ within the local network (Figure 8(b)), we can calculate $\delta_r(\langle f, c\rangle)$ regardless of any obsolete facility as

$$\begin{aligned} \delta_r(\langle f, c\rangle) &= (d_{nn}(F, r) - d(r, c)) \times p(r) \\ &= (d_{nn}(F, r) - (d(r, v_j) - d(c, v_j))) \times p(r) \\ &= (\sigma^-(r\overrightarrow{v_iv_j}) + d(c, v_j)) \times p(r). \end{aligned} \tag{2}$$

Equation (2) has three special cases. First, if $\delta_r(\langle f, c\rangle) < 0$, which means that c is located outside the local network of r (i.e., $r \notin R_{c+}$), then we set $\delta_r(\langle f, c\rangle)$ to zero regardless of obsolete facilities. Second, if $\delta_r(\langle f, c\rangle) > \sigma^+(r\overrightarrow{v_i v_j})$, it means that $d(c, v_j) > d(v_i, v_j)$, which is impossible. Consequently, c must not be accessed from v_i to v_j. This indicates that the reverse direction $r\overrightarrow{v_j v_i}$ exists and r must locate on $v_i v_j$. Hence, c takes effect on the direction from v_j to v_i, and $\delta_r(\langle f, c\rangle) = (\sigma^-(r\overrightarrow{v_j v_i}) + d(c, v_i)) \times p(r)$ instead. In this case, $v_i v_j$ is called a *bidirectional* directed edge with respect to r. If a candidate exactly overlaps r, $\sigma^+(r\overrightarrow{v_i v_j}) = \sigma^+(r\overrightarrow{v_j v_i}) = d_{nn}(F, r)$. Accordingly, $\sigma = d_{nn}(F, r_k)$ (in Equation (1)).

For example, consider c_2 in Figure 7(a). In this case, $\sigma^-(r_{31}\overrightarrow{v_3 v_4}) + d(c_2, v_4) = -4 + 2 < 0$. It indicates that c_2 is outside $LN(r_{31})$, then $\delta_{r_{31}}(\langle f, c_2\rangle)$ is 0 regardless of obsolete facilities. The situation of c_1 is more complex as both $r_{31}\overrightarrow{v_1 v_5}$ and $r_{31}\overrightarrow{v_5 v_1}$ exist. As $\sigma^-(r_{31}\overrightarrow{v_5 v_1}) + d(c_1, v_1) = 6 + 6 > \sigma^+(r_{31}\overrightarrow{v_5 v_1}) = 10$, c_1 leads to the change of distance from the reverse direction $r_{31}\overrightarrow{v_1 v_5}$ instead of $r_{31}\overrightarrow{v_5 v_1}$. Hence, regardless of f, $\delta_{r_{31}}(\langle f, c_1\rangle) = (\sigma^-(r_{31}\overrightarrow{v_1 v_5}) + d(c_1, v_5)) \times 0.5 = (-2 + 10) \times 0.5 = 4$.

The third special case can be described using Figure 8(c). Suppose that $v_1 v_3$ is a bidirectional road. Then the shortest path from r to c is $r \to v_2 \to v_3 \to c$ instead of $r \to v_1 \to c$. This is another kind of *bidirectional* directed edge with respect to r. This is because except for the shortest path (via $v_2 v_3$), the other paths from r to v_3 (e.g., via $v_1 v_3$) have longer distances (e.g., $d(r, v_3) < d(r, v_1) + d(v_1, v_3)$). Then c is reached via the shortest path from v_3 (i.e., $r\overrightarrow{v_3 v_1}$) instead of v_1 (i.e., $r\overrightarrow{v_1 v_3}$). Obviously, there exists one and only one location ℓ_c, called *critical location*, on $v_1 v_3$ such that $d(r, v_1) + d(v_1, \ell_c) = d(r, v_3) + d(v_3, \ell_c)$. If an edge $v_i v_j$ has a critical location ℓ_c, the farthest distance from r to any location of $v_i v_j$ is onto ℓ_c instead of its endpoints. Thus, Equation (2) needs to be adjusted if c locates on $v_i v_j$ following the lemma presented next.

LEMMA 5.2. *Let c be on a bidirectional $v_i v_j$ with ℓ_c regardless of obsolete facilities and $d(r, v_i) < d(r, v_j) < d(r, v_i) + d(v_i, v_j)$. If $d(c, v_j) \geq d(\ell_c, v_j)$, then Equation (2) is applicable; otherwise, it is adjusted to $\delta_r(\langle f, c\rangle) = (\sigma^-(r\overrightarrow{v_i v_j}) + 2 \times d(\ell_c, v_j) - d(c, v_j)) \times p(r)$.*

Local network/sub-network data structures. In this part, we further extend the local network concept to the *local sub-network* to handle the negative returns associated with reference locations in $R_{c-} \cup R_f$. Then we devise data structures to index local networks and local sub-networks.

Definition 5.3. Given $G(V, E)$, a set of facilities F, and a reference location r, the *local sub-network* of r, denoted by $L_2N(r)$, is a sub-network expanded from r with a distance greater than $d_{nn}(F, r)$ and less than or equal to $d_{2nn}(F, r)$.

We can leverage on the notions of local network and local sub-network to obtain R_{c-} and R_f with respect to $\langle f, c\rangle$ as follows: $R_{c-} = rnn(F, f) \cap \{r | c \in v_i v_j \wedge v_i v_j \in L_2N(r)\}$ and $R_f = rnn(F, f) \backslash \{r | c \in v_i v_j \wedge v_i v_j \in LN(r) \cup L_2N(r)\}$.

Based on the preceding discussion, we design the data structures *Local Network Table* (LNT) and *Local Sub-Network Table* (L_2NT) to organize edges of local networks and sub-networks. Each entry of LNT is associated with a directed edge and consists of its $\Delta^+(v_i v_j)$ and a set of reference location entries, each of which is in the form of $\langle p(r), \sigma^+(r\overrightarrow{v_i v_j}), \sigma^-(r\overrightarrow{v_i v_j}), d(\ell_c, v_j)\rangle$, where $d(\ell_c, v_j)$ is set to zero if ℓ_c does not exist. Figure 7(b) illustrates part of LNT that is correlated with r_{31} for simplicity. L_2NT is similar to LNT except $\Delta^+(v_i v_j)$. We construct the two index tables in four steps. First, from a reference location r, we expand network paths via the Dijkstra algorithm. For each $r\overrightarrow{v_i v_j}$ traversed, we record $d(r, v_i)$ and $d(r, v_i) + d(v_i, v_j)$. Second, after the nearest facility of r is found, we compute for each recorded $r\overrightarrow{v_i v_j}$ the σ^+, σ^- and $d(\ell_c, v_j)$ into LNT. For $r\overrightarrow{v_i v_j}$, if $d(\ell_c, v_j) > 0$, then it means a critical location and $r\overrightarrow{v_j v_i}$ exist. Hence, we insert the reverse edge into LNT. Third,

ALGORITHM 1: Local Network Based.

Input: LNT, L_2NT, F, C and $\Delta(f)$ $(f \in F)$
Output: optimal FR pair $\langle f, c\rangle_{OPT}$
1 initialize LNH of C based on LNT ordered by $\Delta^+(v_i v_j)$;
2 **while** $LNH \neq \emptyset$ **do**
3   $\langle c, \Delta^+(v_i v_j), v_i v_j\rangle = Pop(LNH)$;
4   **if** $\Delta(\langle f, c\rangle_{OPT}) > \Delta^+(v_i v_j)$ **then return** $\langle f, c\rangle_{OPT}$;
5   $\sum_{r \in R_{c+}} \delta_r(\langle f, c\rangle) := 0$; `// without regard to any facility`
6   $\Delta(f)' := \emptyset$; `// initial offset value of any facility is 0`
7   **foreach** $r \in R_{c+}$ *(r associated with $v_i v_j$ in LNT)* **do**
8   **if** $d(\ell_c, v_j) > d(c, v_j)$ **then**
9   compute $\delta_r(\langle f, c\rangle)$ based on Lemma 5.2;
10   add $\delta_r(\langle f, c\rangle)$ to $\sum_{r \in R_{c+}} \delta_r(\langle f, c\rangle)$;
11       $\Delta(nn(F, r))' := \Delta(nn(F, r))' - \delta_r(\langle f, c\rangle)$;
12   **continue**;
13     $d := \sigma^-(r\overrightarrow{v_i v_j}) + d(c, v_j)$;
14   **if** $0 < d \leq \sigma^+(r\overrightarrow{v_i v_j})$ **then**
15   add $d \times p(r)$ to $\sum_{r \in R_{c+}} \delta_r(\langle f, c\rangle)$;
16       $\Delta(nn(F, r))' := \Delta(nn(F, r))' - d \times p(r)$;
17   **else if** $d > \sigma^+(r\overrightarrow{v_i v_j})$ **then**
18   **if** *$v_j v_i$ exists* **then**
19         $d := \sigma^-(r\overrightarrow{v_j v_i}) + d(c, v_i)$;
20   **if** $0 < d$ **then**
21   add $d \times p(r)$ to $\sum_{r \in R_{c+}} \delta_r(\langle f, c\rangle)$;
22           $\Delta(nn(F, r))' := \Delta(nn(F, r))' - d \times p(r)$;
23   **foreach** $r \in R_{c-}$ *(r associated with $v_i v_j$ in L_2NT)* **do**
24   compute $\Delta(nn(F, r))' := \Delta(nn(F, r))' + (\sigma^-(r\overrightarrow{v_i v_j}) + d(c, v_j) + d_{2nn}(F, r) - d_{nn}(F, r)) \times p(r)$; `// the process is similar to lines 9—22`
25   $Max_{\Delta(f)} := \max\{\Delta(f_i)' | f_i \in \Delta(f)', \Delta(f_j) | f_j \in F\}$;
26   $\Delta(\langle f, c\rangle) := \sum_{r \in R_{c+}} \delta_r(\langle f, c\rangle) + Max_{\Delta(f)}$;
27   **if** $\Delta(\langle f, c\rangle) > \Delta(\langle f, c\rangle_{OPT})$ **then** $\langle f, c\rangle_{OPT} := \langle f, c\rangle$ and $\Delta(\langle f, c\rangle_{OPT}) := \Delta(\langle f, c\rangle)$;
28 **return** $\langle f, c\rangle_{OPT}$;

we continue the traversal until the sub-nearest facility is found. Each edge of $L_2N(r)$ and the corresponding $p(r)$, σ^+, σ^- and $d(\ell_c, v_j)$ are stored into L_2NT. Notably, if $d(r, c) > d_{2nn}(F, r)$ (for some frontier edges), which means that c is outside $L_2N(r)$, we set $d(r, c) = d_{2nn}(F, r)$. Following the preceding three steps, all reference locations are iteratively scanned, and their d_{nn}s and d_{2nn}s are recorded. Finally, we accumulate $\Delta^+(v_i v_j)$ of each $v_i v_j$ in LNT with corresponding reference locations.

The index-based LNB algorithm. Observe that LNT and L_2NT enable us to evaluate the optimum from all FR pairs without undertaking network traversal. Algorithm 1 outlines this procedure.

As discussed in Section 5.2, candidate c determines the ED upper bound of any FR pair $\langle f, c\rangle$ with c, which means that we can utilize the ED upper bound to tighten the search space by ignoring inferior FR pairs. Based on LNT, we design and initialize a max-heap LNH ordering C by $\Delta^+(v_i v_j)$ to this end. According to the definition of $\Delta^+(v_i v_j)$, candidates in LNH can be ranked based on the edges they are located on. Each entry of LNH is a 3-tuple $\langle c, \Delta^+(v_i v_j), v_i v_j\rangle$. For $v_i v_j$ on which c is located, if the reverse directed edge $v_j v_i$ exists, we use $\Delta^+(v_i v_j) + \Delta^+(v_j v_i)$ for ordering instead (line 1). Then we iteratively check the top candidate in LNH. If the ED of current optimal FR pair is greater than $\Delta^+(v_i v_j)$ related to the popped candidate, the validation is finished (line 4). Otherwise, we first evaluate $\Delta(\langle f, c\rangle)$ taking R_{c+} into account regardless of f. In this process, we use a hash map $\Delta(f)'$ to record offset values of $\Delta(f)$s for corresponding facilities with respect to $r \in R_{c+}$, where $f = nn(F, r)$ (lines 6, 11, 16, and 22). Specifically, $\Delta(f)$ is the maximum negative effect brought by removing f and computed as $\Delta(f) = \sum_{r \in rnn(F,f)}[(d_{nn}(F,r) - d_{2nn}(F,r)) \times p(r)]$. Note that σ^+ is not checked in line 20, as it is impossible that $\sigma^-(r\overrightarrow{v_i v_j}) + d(c, v_j)$ is invalid in both directions. Similar to lines 9 through 22, we evaluate $\Delta(f)'$ values with respect to $r \in R_{c-}$ (lines 23 and 24). Afterward, we evaluate the maximum negative effect with respect to c. If a superior FR pair is found, we update the current optimal one (lines 25 through 27). Next, we probe the unchecked candidates in LNH until the optimal FR pair is finally retrieved.

5.3 Extended Network Method

In the preceding method, existing facilities and movements of users are assumed to be known *a priori*. Consequently, we can pre-compute and index them offline so that the MOTION-FR problem with an arbitrary group of candidates can be answered efficiently. Nevertheless, in some cases, we may not have any idea about facilities or users in advance. How can we address the MOTION-FR problem in this scenario? To this end, we propose a method inspired by the *network extension* scheme so that the MOTION-FR problem can be answered from scratch. The main idea of the index-free method is to accumulate $\delta_r(\langle f, c\rangle)$s progressively for each obsolete facility and candidate substitute as nearest and sub-nearest facilities are discovered for reference locations.

The proposed method is index free and consists of four steps that are detailed in Algorithm 2. First, we extend network facility vertices from F to $F \cup C$. Second, for each reference location r, we explore its nearest and sub-nearest facilities in $F \cup C$. In this process, if a candidate vertex $c \in C$ is encountered before (respectively, after) $nn(F, r)$ is found, we temporarily record the distance in a vector V_n (respectively, V_2) (lines 5 through 8). Third, once $2nn(F, r)$ is found, we compute and accumulate the corresponding $\delta_r(\langle f, c\rangle)$s for all encountered candidates in V_n and V_2 (lines 11 through 19). We employ a *Candidate Accumulation Table* (CAT) to store the accumulated ED of every candidate regardless of obsolete facilities (i.e., R_{c+}). Each entry of CAT is in the form of $\langle c, \sum_{r \in R_{c+}} \delta_r(\langle f, c\rangle)\rangle$. Similarly, a *Facility Accumulation Table* (FAT) is designed for evaluating $\Delta(f)$s. Its entry is in the form of $\langle\langle f, c\rangle, value\rangle$, where $value$ related to $\langle f, c\rangle$ has two aspects: If $c = null$, it means removing f without a substitute (i.e., $value = \Delta(f)$); otherwise, $value$ is an accumulative offset value for $\sum_{r \in R_{c-} \cup R_f} \delta_r(\langle f, c\rangle)$ (by adding up to $\Delta(f)$). Finally, after all reference locations are scanned, we compute EDs of all FR pairs based on CAT and FAT, which leads us to the final answer.

6 EXTENSION TO k-MOTION-FR

In the preceding section, we develop methods to solve the MOTION-FR problem in offline and online manners, respectively. In certain applications, however, it may be desirable to relocate k inferior facilities with substitutes that collectively achieve the most reduction of expected distance. For instance, a car-sharing company may have sufficient budget to relocate five inferior stations. Consequently, we need to generalize the MOTION-FR problem. We refer to this problem as the k-MOTION-FR problem (i.e., the multi-FR problem), which is defined based on Definition 4.3.

Definition 6.1. Given a set of existing facilities F, a set of candidate locations C, a set of m users U, and a positive integer k ($k \leq |F| \wedge k \leq |C|$), let $\langle F_k, C_k\rangle$ be an FR set pair where $F_k \subseteq F$, $C_k \subseteq C$ and $|F_k| = |C_k| = k$, and the k-MOTION-FR problem aims to find an optimal FR set pair $\langle F_k, C_k\rangle_{OPT}$ such that $\langle F_k, C_k\rangle_{OPT} = \text{argmax} \sum_{i=1}^{m}(E[d_{nn}(F, L(u_i))] - E[d_{nn}(F \backslash F_k \cup C_k, L(u_i))])$.

Observe that the k-median problem, which is NP-hard [23], is a special case of the k-MOTION-FR problem when $F = \emptyset$. Hence, the k-MOTION-FR problem is NP-hard. As the methods proposed in Section 5 obtain the optimal result accurately, the greedy heuristic can be applied to solving k-MOTION-FR. The EN method requires network traversals when querying, which will result in expensive time cost in each greedy step, and then we extend the aforementioned LNB

ALGORITHM 2: Extended Network.

extend $G(V, E)$ with candidate vertices in C;
foreach $r_j \in L(u_i) \wedge u_i \in U$ **do**
    $nn := null$; // temporary nearest facility for each r_j
    while *traverse* G *from* r_j *for each vertex* v **do**
        if $v \in C$ *and* $nn = null$ **then**
            record $\langle v, d(r_j, v)\rangle$ in a 2D Vector V_n;
        else if $v \in C$ *and* $nn \neq null$ **then**
            record $\langle v, d(r_j, v)\rangle$ in a 2D Vector V_2;
        else if $v \in F$ *and* $nn = null$ **then**
            $nn := v, d_{nn} := d(r_j, v)$;
        else if $v \in F$ *and* $nn \neq null$ **then**
            $d_{2nn} := d(r_j, v)$;
            add $(d_{nn} - d_{2nn}) \times p(r_j)$ to $FAT.\langle nn, null\rangle$;
            foreach $v_k \in V_n$ **do**
                add $(d_{nn} - d(r_j, v_k)) \times p(r_j)$ to $CAT.v_k$;
                add $(d_{2nn} - d_{nn}) \times p(r_j)$ to $FAT.\langle nn, v_k\rangle$;
            foreach $v_k \in V_2$ **do**
                add $(d_{2nn} - d(r_j, v_k)) \times p(r_j)$ to $FAT.\langle nn, v_k\rangle$;
            release V_n, V_2 and **break**;
compute $\Delta(\langle f, c\rangle)$ $\langle f, c\rangle \in F \times C$ and **return** $\langle f, c\rangle_{OPT}$;

algorithm for an approximate solution to the k-MOTION-FR problem. The basic idea of the greedy algorithm for k-MOTION-FR is to perform a sequence of steps as follows. In the n-th step, an FR pair $\langle f_n, c_n\rangle$ is selected greedily from $F_n \times C_n$ by executing the LNB algorithm (Algorithm 1), where $F_n = F_{n-1} \cup \{c_{n-1}\} \backslash \{f_{n-1}\}$ and $C_n = C_{n-1} \backslash \{c_{n-1}\}$. The process terminates until $n = k$ or $\Delta(\langle f_n, c_n\rangle) \leq 0$. The latter condition means that the global optimum has been achieved before the k-th step. Between the successive (n-*1*)-th and n-th steps, we need to update LNT, L_2NT, $\Delta(f_{n-1})$, and $\Delta(c_{n-1})$ according to $\langle f_{n-1}, c_{n-1}\rangle$ selected in the (n-*1*)-th step. The update procedure is as follows. First, we find reference locations whose $LN(r)$s overlap $v_s v_e$, on which c_{n-1} is located. For each of these reference locations r, we compute $d_{nn}(F, r) - d(r, c_{n-1})$ based on Equation (2) or Lemma 5.2. Second, LNT and L_2NT entries associated with r, as well as $\Delta(f_{n-1})$ and $\Delta(c_{n-1})$, are updated depending on whether $nn(F, r)$ is exactly f_{n-1} or not. Specifically, if $nn(F, r) = f_{n-1}$, we update the $\sigma^+(r\overrightarrow{v_i v_j})$ and $\sigma^-(r\overrightarrow{v_i v_j})$ values of edges $v_i v_j \in LN(r) \cup L_2N(r)$, which is based on the following lemma, and $v_i v_j$ entries are moved from LNT to L_2NT if $\sigma^+(r\overrightarrow{v_i v_j}) \leq 0$.

LEMMA 6.2. *Given $\langle f, c\rangle$ and r such that $nn(F \backslash \{f\} \cup \{c\}, r) = c$, for $v_i v_j \in LN(r) \cup L_2N(r)$, if $v_i v_j$ is still inside the new local and sub-local networks of r, $\sigma^+(r\overrightarrow{v_i v_j})$ and $\sigma^-(r\overrightarrow{v_i v_j})$ are updated as $\sigma^+(r\overrightarrow{v_i v_j}) - (\sigma^-(r\overrightarrow{v_s v_e}) + d(c, v_e))$ and $\sigma^-(r\overrightarrow{v_i v_j}) - (\sigma^-(r\overrightarrow{v_s v_e}) + d(c, v_e))$, where c is on $v_s v_e$.*

Furthermore, if $nn(F, r) \neq f_{n-1}$, we remove $v_i v_j \in L_2N(r)$ from L_2NT, as edges between c and $nn(F, r)$ are as new $L_2N(r)$. The σ^+ and σ^- values in $LN(r)$ are updated, and the edges whose $\sigma^+ \leq 0$ are moved from LNT to L_2NT. Third, similar to the first two steps, we update $LN(r)$ and $L_2N(r)$ of each r whose $L_2N(r)$ overlaps $v_s v_e$. Fourth, we update $G(V, E)$ by dropping f and inserting c as a facility. Finally, for $r \in rnn(F, f)$, we expand network paths from $2nn(F, r)$ toward the new sub-nearest facility to update $LN(r)$ and $L_2N(r)$.

7 COST ANALYSIS

This section conducts theoretical study of the proposed methods: straightforward, LNB, and EN.

The straightforward method described in the very beginning of Section 5 iteratively checks FR pairs based on Theorem 4.6. The complexity is $O(|F| \cdot |C| \cdot |R| \cdot |V| \log |V|)$.

Table 2. Real Facilities of CA and BJ

CA	$\|F\|$	BJ	$\|F\|$
Post Offices(PO)	800	*Cafes(CF)*	1,500
Gas Stations(GA)	**800**	***Gas Stations(GS)***	**1,500**
Subway Restaurants(SU)	1,600	*Logistics(LO)*	2,500
ATMs(AT)	1,600	*Parking Stations(PK)*	2,500

LNB pre-computes $d_{nn}(F, r)$ and $d_{2nn}(F, r)$ for reference locations, and the cost is $O(|R| \cdot |V| \log |V|)$. The best case occurs when each reference location has its sub-nearest facility on the same edge, then both LNT and L_2NT contain $|R|$ records. In the worst case, if there are only two facilities, each reference location needs to traverse massive edges, then the two tables are with $|E| \cdot |R|$ records. On the other hand, the running cost of LNB only depends on the number of entries, say γ, visited in *LNH*. In practice, only a limit number of edges are checked—that is, $\gamma \ll |E|$. Because the evaluation of $\Delta(\langle f, c\rangle)$ is only based on the floating-point operation, its running cost can be regarded as $O(1)$. Hence, the complexity of LNB is $O(\gamma)$.

As the network is extended by candidates as extra vertices via the Dijkstra algorithm, the time complexity of EN is simply $O(|R| \cdot (|V| + |C|) \log(|V| + |C|))$. For space complexity, CAT and FAT are $O(|C|)$ and $O(|F| \cdot |C|)$. Note that only the candidates $c \in C'$ whose related $\langle f, c\rangle$ pairs have *offset* relationship are stored in FAT, then actually $|C'| \ll |C|$. Hence, the space cost of FAT is acceptable.

In summary, as far as query efficiency is concerned, we have LNB $\gg$ EN $\gg$ straightforward. Our experimental study shall validate this.

8 PERFORMANCE STUDY

In this section, we investigate the performance of our proposed solutions from a variety of aspects.

8.1 Experimental Setup

Datasets. We use two real road networks: California (CA) [11] and Beijing City (BJ).[6] CA contains 21,693 *bidirectional* edges and 21,047 vertices. BJ consists of 433,391 *unidirectional* edges and 171,504 vertices. Table 2 shows the real datasets of facilities for CA[7] and BJ. Facilities in each group are randomly chosen from respective datasets. GA and GS are used for CA and BJ, respectively, in the following experiments unless explicitly stated otherwise.

We conduct experiments on real and synthetic datasets of users. The real users data of BJ is available from Zheng et al. [32]. There are 136,686 sample points for each user on average. For CA, we use the discrete check-in data[8] made by 26,619 users. By filtering out those who have fewer than 20 check-ins, we pick 9,656 users and the average number of check-ins is about 125. The presence probabilities for reference locations of each user are skewed, where the differences between the maximum and minimum presence probabilities are 0.278 (CA) and 0.367 (BJ) on average. We generate synthetic data with larger cardinalities following the feature of real data to investigate the scalability of our proposed solutions. Specifically, we synthesize 1 to 6 reference locations with proportions 5%, 20%, 35%, 25%, 10%, and 5% for each user, the number of which ranges from 20k to 100k. Moreover, we use two schemes, *Random* (R) and *Uniform* (U), to generate presence

[6]Road network and facilities of BJ are available from Cui et al. [4].
[7]Post offices are available from Li et al. [11] and others are downloaded from http://www.poi-factory.com/search/pfc/california.
[8]Check-in data in California are obtained from http://snap.stanford.edu/data/.

probabilities of reference locations. For the former, probabilities are distributed randomly. For *Uniform*, if a user has n reference locations, the presence probability at each location is uniformly set as $1/n$. Considering daily activity areas of a user are unlikely to be far apart (e.g., as discussed in Xia et al. [25], trip radii are mostly between about 10 and 20 km), reference locations of a user are confined and randomly distributed in a 30 km range. We choose 50, 100, 200, 400, and 800 locations within the geographical domain for candidates at random.[9] Unless specified otherwise, the default values of $|C|$, $|U|$ and the distribution of presence probability are 200, 60k, and *Random*, respectively.

Algorithms. We evaluate the following algorithms and semantics in our experiments; all are implemented in C++ and tested on a 3.2-GHz quad-core machine with 32 GB of RAM[10]

- EN: The extended network method described in Algorithm 2.
- LNB: The local network–based method described in Algorithm 1. LNT and L_2NT are loaded in main memory.
- RID: The state-of-the-art technique for solving the FR problem in Euclidean space [16], which is called the *replacement influence distance* (RID). We adapt the technique to solve MOTION-FR in a road network for comparison. Given F, the nearest (respectively, sub-nearest) facility circle of a user u at r is a circle centered at r with radius $d_{nn}(F, r)$ (respectively, $d_{2nn}(F, r)$). Obviously, as $d(\ell_2, \ell_1)$ is the lower bound of $d_\varepsilon(\ell_2, \ell_1)$, any candidate that lies outside the (sub-)nearest facility circle of r cannot affect r. In other words, only r whose (sub-)nearest facility circle encloses the substitute candidate may contribute to *ED*; otherwise, it can be pruned. Additionally, the *replacement influence circle* (RIC) is a circle centered at f with radius $\max_{r \in rnn(F,f)}\{d_{nn}(F, r) + d_{2nn}(F, r)\}$. Once a candidate c is outside the RIC of f, c and f respectively affect different sets of reference locations. We regard reference locations with presence probabilities as static objects with weights. Similar to Qi et al. [16], C, RICs, and (sub-)nearest facility circles are indexed by respective R-trees (in main memory), and the *spatial join* scheme is used. The index structures are also adapted on the basis of a greedy heuristic such that it can update when an FR pair is conducted.
- RID-Q: Only the query part of the RID method (without the index part).
- M-FR: The MOTION-FR semantics, which is proposed in this article to evaluate *ED*.
- S-FR: The static Min-dist-based FR semantics [16] to evaluate average distance. As static Min-dist-based FR requires that a user be represented by a single position, we pick and only consider the reference location with the highest probability for each user.

As discussed in Section 7, the cost of the straightforward method on network traversal is proportional to the product of cardinalities of facilities and candidates. For instance, when we run it with 50 candidates and a PO facility group on the CA road network, it takes nearly 4 days to complete. In contrast, EN takes only tens of seconds. Hence, the straightforward method is impractical, and we omit it in our experiments. In the following, we will compare the methods listed previously (i.e., EN, LNB, RID, and RID-Q) mainly on execution time, and the changing trend of *ED* is also evaluated. The impacts of different parameters (e.g., $|C|$, $|F|$, $|U|$, spatial distribution) on efficiency are examined respectively. For the two FR semantics, we investigate the differences on selected FR pairs and *ED* values. The reasons behind the phenomena in all experiments are analyzed in detail.

[9]For the synthetic objects that are not exactly located on roads, we shift them to the closest point of on the road network.
[10]Source codes are available from https://github.com/lihuixidian/frost-tist/.

Table 3. Comparison between MOTION-FR and Static Min-dist-Based FR Semantics

	F	Real							Synthetic						
		d_f	%	d_c	%	$ED_{\text{M-FR}}$	$ED_{\text{S-FR}}$	gap %	d_f	%	d_c	%	$ED_{\text{M-FR}}$	$ED_{\text{S-FR}}$	gap %
CA	PO	412	29.8	145	10.5	12,967	7,065	45.5	173	12.5	459	33.2	15,469	12,887	16.7
	GA	245	17.7	117	8.4	7,334	5,092	30.6	805	58.3	74	5.4	37,051	15,515	58.1
	SU	702	50.8	346	25.1	1,259	−5,074	n/a	218	15.8	98	7.1	44,042	−3,256	n/a
	AT	354	25.6	265	19.2	2,351	−4,966	n/a	311	22.5	460	33.3	70,074	13,224	81.1
BJ	CF	62	23.8	42	16.4	131	106	19.2	62	23.8	24	9.4	9,731	5,085	47.7
	GS	79	30.7	21	8.2	21	18	11.2	89	34.4	8	3.2	749	695	7.2
	LO	90	34.6	24	9.1	41	35	13.8	84	32.3	7	2.7	1,747	1,014	41.9
	PK	39	14.9	5	2.1	108	97	9.5	28	10.7	8	3.2	10,424	9,815	5.8

8.2 Experimental Results

Comparison with static Min-dist-based FR. We first study the results of optimal FR pairs that are mined under M-FR and S-FR semantics. We compare the Euclidean distances (more spatially intuitive than network distances) between the obsolete facilities (respectively, selected candidates) chosen by M-FR and S-FR, denoted by d_f (respectively, d_c), as well as the proportions with respect to the whole ranges of CA (1,380 km) and BJ (259 km). Moreover, we also evaluate *ED*s of the optimal FR pairs for the two semantics, denoted respectively by $ED_{\text{M-FR}}$ and $ED_{\text{S-FR}}$. The gaps between $ED_{\text{M-FR}}$ and $ED_{\text{S-FR}}$ are also given in ratio as $\frac{ED_{\text{M-FR}} - ED_{\text{S-FR}}}{ED_{\text{M-FR}}}$. To eliminate the influence of cardinality on results, we make the same cardinality for facility sets in CA (800) and BJ (1,500). We use 20k synthetic and real users for comparison. The results are reported as the averages for 20 different candidate groups, each of which contains 100 randomly generated candidates.

As reported in Table 3, the optimal FR pairs differ greatly in both distances and *ED*s, which means that users' movements really have an impact on results. The d_f values are mostly further than those for d_c, which may have reasons from two aspects. On one hand, the distribution of existing facilities makes superior candidates be within a relatively fixed area and more profitable than in other regions. On the other hand, as facility sets are with larger cardinalities than candidate sets, inferior facilities might be with a relatively higher proportion and scattered in several regions. We also observe great differences of d_c values between real and synthetic users, which means that the different distributions of users (i.e., synthetic users are distributed more uniformly than real ones) significantly affect the results. From aspect of *ED*, MOTION-FR makes significantly better decisions than static FR, although the margins vary widely due to different distributions of facility groups. Comparing the *ED* gaps for real and synthetic users, it reveals that the more similar the distributions of users and facilities are, the smaller gaps there exist. Note that for Subway restaurants in CA, the results selected by static FR even lead to negative returns for both real and synthetic users. That implies that movements of users are determinant in certain application scenarios.

In addition, we outline the efficiency comparison between the proposed solutions and RID (RID-Q). Following the default experiment settings, Table 4 gives an overview of execution time for the methods. Based on index structures, the querying time of LNB is reduced by orders of magnitude compared to RID-Q. For online methods, EN mostly outperforms RID.

In the following experiments, we will study the impacts of different parameters on performance.

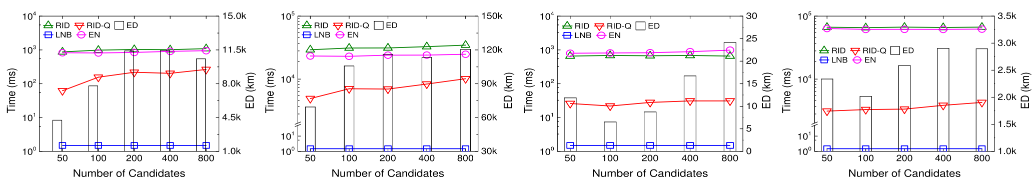
Effect of $|C|$. In this part, we investigate the performance by varying the number of candidates. As shown in Figure 9, LNB shows the best scalability, and it is superior to RID-Q with offline index structures. The running time of LNB is within milliseconds regardless of $|C|$ and road datasets,

Table 4. Comparison on Running Time among Algorithms (ms)

		CA		BJ	
		Real	Syn.	Real	Syn.
Index Based	LNB	1	2	1	2
	RID-Q	218	6,951	15	6,029
Index Free	EN	843	23,963	827	61,423
	RID	1,030	31,226	671	68,545

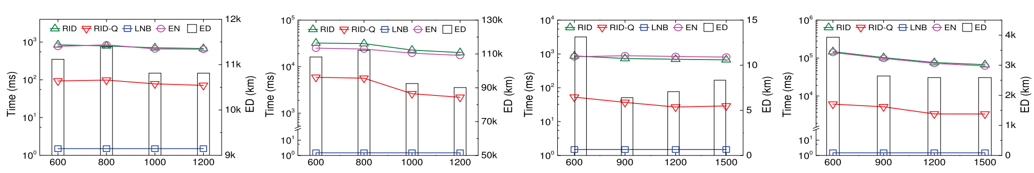
Table 5. Expected Change of Total Distance (km)

	$\|C\|$=50	100	200	400	800
UNIFORM(CA)	75,096	104,007	116,337	113,530	119,183
RANDOM(CA)	70,632	97,082	109,876	106,983	112,352
diff.(CA) %	6.32	6.53	5.88	6.12	6.08
UNIFORM(BJ)	2,292	2,803	2,566	2,810	2,492
RANDOM(BJ)	2,182	2,662	2,450	2,672	2,376
diff.(BJ) %	5.06	5.30	4.73	5.15	4.87



(a) CA (real) (b) CA (synthetic) (c) BJ (real) (d) BJ (synthetic)

Fig. 9. Effect of $|C|$.



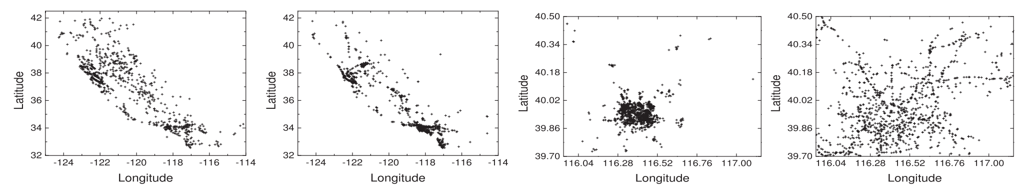
(a) CA (real) (b) CA (synthetic) (c) BJ (real) (d) BJ (synthetic)

Fig. 10. Effect of $|F|$.

as the cost is only relevant to the lookup process and numeric computation of LNT and L_2NT without network retrieval. The reason RID (RID-Q) is not as efficient as in Euclidean space is twofold. First, the pruning power of the (sub-)nearest facility circle and RIC declines for network distance. Second, the efficiency is mainly determined by the shortest path evaluation that is more costly compared to Euclidean distance computation. In online scenarios, EN is better than RID in most cases, except for the case of real users in BJ. The reason is, for a relatively small number of users, that the overhead of FAT significantly offsets the query efficiency of EN. That is why the gaps between EN and RID become wider as $|C|$ grows in Figure 9(c).

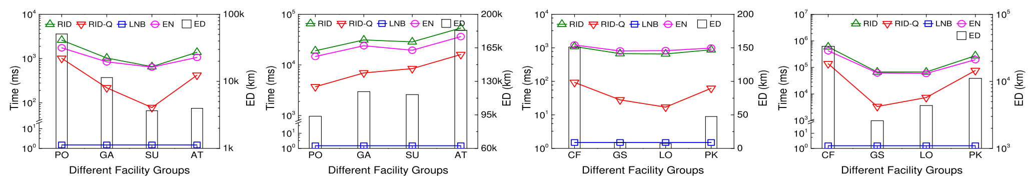
ED values mostly grow with the increase of $|C|$, which means that the more widely candidates are distributed, the higher probability a superior choice exists in C. This indicates that the geographical distributions of users' movements and facilities determine the optimal candidate location. The cases that candidate sets with lower cardinalities lead to larger *ED*s occur sporadically on account of the randomness of generated candidates.

Effect of $|F|$. In this set of experiments, we study the effect of the number of facilities. To keep consistent geographical distribution, we randomly choose four groups from GA (CA) and GS (BJ). The cardinalities are 600, 800, 1,000, 1,200 (GA) and 600, 900, 1,200, 1,500 (GS), respectively. As shown in Figure 10, except for LNB, the running cost slightly drops with the increase of $|F|$ for the other methods. This is because LNB only depends on float-point computation, whereas the others require network traversal. As more facilities lead to shorter network paths on average, the



(a) Post Offices (CA) (b) Gas Stations (CA) (c) Cafes (BJ) (d) Gas Stations (BJ)

Fig. 11. Geographical distribution of facilities.



(a) CA (real) (b) CA (synthetic) (c) BJ (real) (d) BJ (synthetic)

Fig. 12. Effect of geographical distribution.

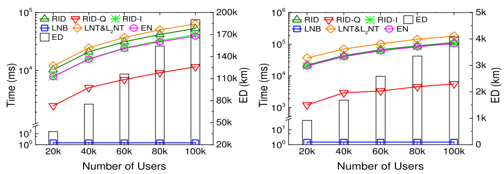
(sub-)nearest facility circles are smaller. Hence, more FR pairs can be pruned, and fewer network traversals are needed.

From the *ED* results, both in CA and BJ, we find an interesting phenomenon: There seems to be a cardinality threshold of F. When $|F|$ is above the threshold (about 1k and 900 for CA and BJ, respectively), *ED* values drop dramatically. As facility sets that are chosen from the corresponding datasets with different cardinalities follow the same geographical distribution, the reason may be as follows: Once $|F|$ is below the threshold, many users have to travel significantly farther. In contrast (i.e., above the threshold), a large proportion of users cannot benefit from more facilities. That means, for given users, that it is unnecessary and even wasted to operate too many facilities simultaneously.

Effect of geographical distribution in F. We also explore the effect of the geographical distribution of facilities. The two facility groups in BJ as shown in Figure 11(c) and (d) have the same cardinality, whereas their distributions are entirely distinct: Gas Stations are relatively uniform, whereas Cafes are skewed. This results in significant difference in both the costs and *ED* values of EN, RID, and RID-Q. For LO and PK in Figure 12(c), EN involves a similar quantity of network traversal. However, due to geographical distribution, the nearest facility circle and replacement influence circle policies have different pruning powers. When the difference in distribution increases (e.g., CF and GS in BJ), the effect is enlarged, which results in the performance degradation of RID-Q.

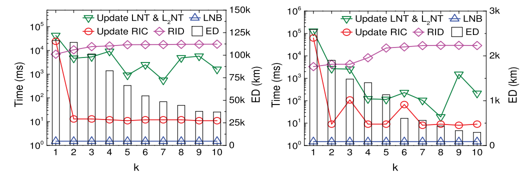
Similarly, in Figures 11(a) and (b), GA is more skewed compared to PO. Due to the different distributions, there exist huge gaps in time costs and *ED* values. As shown in Figure 12(a) and (b), the results for real and synthetic datasets are opposite, and the difference in distribution of real and synthetic users is the reason. The geographical distribution of real users is similar to that of GA but differs from PO. That means that if the deployment of facilities is less in line with users' movements, it is more needed to launch FR, and then users will benefit from shorter distances to access facilities. In our opinion, this is the scenario where FR problems should be applied.

Effect of $|U|$. We investigate the scalability with respect to the cardinality of users. Figure 13 reports that the results are qualitatively similar to Figure 9. LNB exhibits the best performance, followed by RID-Q, EN, and RID. The cost mostly increases when the number of users (i.e.,



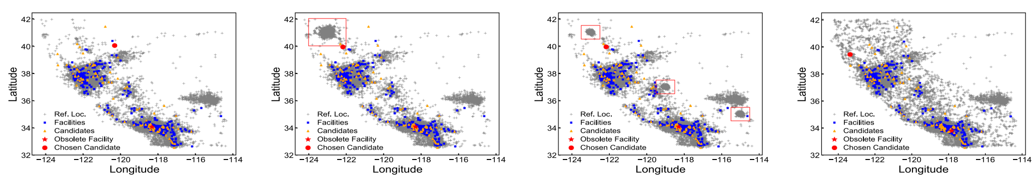
(a) CA (b) BJ

Fig. 13. Effect of $|U|$.



(a) CA (b) BJ

Fig. 14. Relocating k facilities.



(a) Original ref. loc. of real users in CA. (b) Redeploy 10% ref. loc. with 1 center. (c) Redeploy 10% ref. loc. with 3 centers. (d) Redeploy 10% ref. loc. uniformly.

Fig. 15. Effect of change in users' locations.

reference locations) grows. LNB demonstrates stable performance and still finishes execution within milliseconds.

Next, we study the index costs of LNB and RID. As RID needs to pre-compute $d_{nn}(F,r)$, $d_{2nn}(F,r)$ for reference locations, $\Delta(f)$s for facilities, and then to construct R-trees for C, RICs, and (sub-)nearest facility circles, its index cost (RID-I in Figure 13) is more than EN. Constructing LNT and L_2NT (LNT&L_2NT in Figure 13) takes around 40% extra time compared to EN. The reason is twofold. First, the standard C++ hash map template class, utilized in implementation, repeatedly re-allocates memory during expanding edges. Second, as the number of edges is large, the computation of σ^+, σ^-, *offset*, and hash keys consumes some time. In short, LNB has a much better query efficiency, whereas RID has a shorter index time, and RID is worse than EN for online scenarios.

Effect of change in users' locations. Next, we study whether changes in partial reference locations of some users actually have impact on selected FR results. To this end, we choose 500 gas stations in CA as existing facilities that serve users. Following the geographical distribution of the real datasets of users, we randomly pick 100 gas stations as candidates, which are not overlapped with the selected facilities. We use two schemes to simulate the change. First, we choose one and three center locations in CA as new central business districts or entertainment districts that are attractive to users. As shown in Figure 15(b) and (c), the red rectangles indicate the centers. Then we randomly pick 10%, 30%, and 50% reference locations and redeploy them around the centers following Gaussian distribution. Second, we uniformly redeploy 10%, 30%, and 50% reference locations in CA.

Figure 15 illustrates the optimal FR pairs for original reference locations of real users and those for the aforementioned schemes. Compared to Figure 15(a), the locations of both obsolete facilities and candidates in Figure 15(b) through (d) are different, which means that even if a small part of users (10%) change some of their frequent activity areas, the results will still be influenced. This observation is exactly in line with our motivation that movement history should be taken into account for FR. Since the experiment results on 30% and 50%, as well as other facility datasets, are qualitatively similar, due to space constraints, we do not report them in detail.

Effect of the presence probabilities. Here we study the effect on resulting FR pairs in terms of the presence probability distribution of reference locations. We compare for both CA and BJ the *ED* differences between *Uniform*(U) and *Random*(R) presence probabilities on average by varying $|C|$. As reported in Table 5, the optimal FR pair in CA gains larger reduction of average distance than in BJ due to the larger geographical area of CA. Moreover, the proportion differences of *ED*s between *Uniform* and *Random* are around 6% (CA) and 5% (BJ). In other words, the movements of users, namely the skewed presence probabilities at reference locations, influence the result.

Effect of network scale. In this part, we briefly report the effect of network scale. As demonstrated in the preceding experiments, except LNB, the running time in BJ is nearly one order of magnitude longer than the counterpart in CA with respect to similar geographical distribution. As the scale of BJ has a larger magnitude than that of CA, the scalability is in line with the theoretical analysis. Note that if the geographical distribution of facilities is quite different (e.g., the skewed CF versus the uniform GS in BJ), the time cost might deviate from the network scale.

Performance of k-MOTION-FR. We compare the greedy LNB and adapted greedy RID to solve the k-MOTION-FR problem. We set $k = 10$. Figure 14 reports the *ED* value and the time of querying and updating index structures. For greedy LNB, the time to update LNT and L_2NT is more than one order of magnitude shorter compared to the construction time. For RID, the update time is tens of milliseconds, whereas the decrease of query time is quite limited. As greedy LNB has a significantly shorter total time than greedy RID, it is a feasible and superior solution to k-MOTION-FR.

9 CONCLUSION

In this article, we introduce a novel FR problem called MOTION-FR that exploits, for the first time, movement history of users for relocating facilities. We utilize reference locations to model users' movement history and present a framework called FROST to address it. FROST comprises two algorithms: *index based* and *index free*. The first algorithm LNB is designed to address the problem efficiently when facilities and users are known *a priori*. The second algorithm, namely EN, solves the problem from scratch without assuming any knowledge of facilities and users ahead of time. We further generalize the problem to k-MOTION-FR, which is NP-hard, and devise an approximate solution by extending LNB. Extensive experiments on both real and synthetic datasets demonstrate the superiority of our proposed approaches compared to state-of-the-art FR techniques in efficiency and effectiveness. As part of future work, we plan to study FR under competition with other service providers, which might further improve the availability of our research.

APPENDIX

A PROOFS

A.1 Proof of Lemma 4.5

Observe that a user u is present at $r \in L(u)$ with $p(r)$, and for all $|W|$ possible worlds, r exactly occurs in $p(r) \times |W|$ possible worlds. For the possible worlds $W' \subseteq W$ in which a specific r occurs, namely $r^w = r$ if $r^w \in W'$, $\sum_{r^w \in W'} d_{nn}(F, r^w) \times Pr[l(U) = w] = d_{nn}(F, r) \times p(r)$. Taking into account all $r \in L(u)$, $E[d_{nn}(F, L(u))] = \sum_{r \in L(u)} (d_{nn}(F, r) \times p(r))$.

A.2 Proof of Theorem 4.6

If we replace an obsolete facility f with a substitute at candidate location c, $E[d_{nn}(F \backslash \{f\} \cup \{c\}, L(u))] = \sum_{r \in L(u)} (d_{nn}(F \backslash \{f\} \cup \{c\}, r) \times p(r))$. According to Lemma 4.5, we can transform the computation of ED from the perspective of possible worlds to reference locations as follows.

$$\begin{aligned}\Delta(\langle f,c\rangle) &= \sum_{j=1}^{|W|}\Delta_{w_j}(\langle f,c\rangle)\times Pr[l(U)=w_j]\\ &= \sum_{j=1}^{|W|}\sum_{i=1}^{m}(d_{nn}(F,r_i^{w_j})\times Pr[l(U)=w_j]-d_{nn}(F\backslash\{f\}\cup\{c\},r_i^{w_j})\times Pr[l(U)=w_j])\\ &= \sum_{i=1}^{m}(E[d_{nn}(F,L(u_i))]-E[d_{nn}(F\backslash\{f\}\cup\{c\},L(u_i))]).\end{aligned}$$

A.3 Proof of Lemma 5.1

With the substitution of R upon Lemma 4.5 and Theorem 4.6, $\Delta(\langle f,c\rangle)$ can be rewritten as $\sum_{r\in R}\delta_r(\langle f,c\rangle)$. According to the definitions, it is self-evident that R_{c+}, R_{c-}, R_f and $R_{\overline{f\cup c}}$ are mutually exclusive and $R=R_{c+}\cup R_{c-}\cup R_f\cup R_{\overline{f\cup c}}$. Hence, we can compute their EDs separately and then accumulate them for the total. As $\sum_{r\in R_{\overline{f\cup c}}}\delta_r(\langle f,c\rangle)=0$, $R_{\overline{f\cup c}}$ can be ignored, and thus $\Delta(\langle f,c\rangle)=\sum_{r\in R_{c+}\cup R_{c-}\cup R_f}\delta_r(\langle f,c\rangle)$.

A.4 Proof of Lemma 5.2

According to the definition of critical location, we have $d(r,v_i)+d(v_i,\ell_c)=d(r,v_j)+d(\ell_c,v_j)$ and $d(v_i,\ell_c)=d(v_i,v_j)-d(\ell_c,v_j)$, then $d(\ell_c,v_j)=\frac{1}{2}(d(r,v_i)+d(v_i,v_j)-d(r,v_j))$. When $d(c,v_j)\geq d(\ell_c,v_j)$, as c_1 in Figure 8(d), it locates on $r\overrightarrow{v_iv_j}$, then Equation (2) holds. If $d(c,v_j)<d(\ell_c,v_j)$, as c_2 in Figure 8(d), it is reached from v_j, namely $r\overrightarrow{v_jv_i}$. As the distances from r to locations on v_iv_j are symmetrical with ℓ_c, we can find the symmetry location c_2' of c_2 such that $d(r,c_2')=d(r,c_2)$. As Equation (2) holds for c_2', we can compute $\delta_r(\langle f,c_2\rangle)=\delta_r(\langle f,c_2'\rangle)=(\sigma^-(r\overrightarrow{v_iv_j})+2\times d(\ell_c,v_j)-d(c,v_j))\times p(r)$.

A.5 Proof of Lemma 6.2

As $nn(F\backslash\{f\}\cup\{c\},r)=c$, thus $d_{nn}(F\backslash\{f\}\cup\{c\},r)=d(r,c)$. According to the definition of $\sigma^+(r\overrightarrow{v_iv_j})$ and Equation (2), we have $\sigma^+(r\overrightarrow{v_iv_j})'=d(r,c)-d(r,v_i)=d_{nn}(F,r)-d(r,v_i)-(d_{nn}(F,r)-d(r,c))=\sigma^+(r\overrightarrow{v_iv_j})-(\sigma^-(r\overrightarrow{v_sv_e})+d(c,v_e))$, where $\sigma^+(r\overrightarrow{v_iv_j})'$ is the updated value of $\sigma^+(r\overrightarrow{v_iv_j})$ after $\langle f,c\rangle$ is conducted. Similarly, $\sigma^-(r\overrightarrow{v_iv_j})'=\sigma^-(r\overrightarrow{v_iv_j})-(\sigma^-(r\overrightarrow{v_sv_e})+d(c,v_e))$.

REFERENCES

[1] Sergei Bespamyatnikh, Binay K. Bhattacharya, David G. Kirkpatrick, and Michael Segal. 2000. Mobile facility location. In *Proceedings of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM'00)*. 46–53.

[2] Muhammad Aamir Cheema, Xuemin Lin, Wei Wang, Wenjie Zhang, and Jian Pei. 2010. Probabilistic reverse nearest neighbor queries on uncertain data. *IEEE Transactions on Knowledge and Data Engineering* 22, 4 (2010), 550–564.

[3] Zitong Chen, Yubao Liu, Raymond Chi-Wing Wong, Jiamin Xiong, Ganglin Mai, and Cheng Long. 2014. Efficient algorithms for optimal location queries in road networks. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data (SIGMOD'14)*. 123–134.

[4] Jiangtao Cui, Meng Wang, Hui Li, and Yang Cai. 2018. Place your next branch with MILE-RUN: Min-dist location selection over user movement. *Information Sciences* 463–464 (2018), 1–20.

[5] E. W. Dijkstra. 1959. A note on two problems in connection with graphs. *Numerische Mathematics* 1 (1959), 269–271.

[6] Reza Zanjirani Farahani, Wai Yuen Szeto, and Saeed Ghadimi. 2015. The single facility location problem with time-dependent weights and relocation cost over a continuous time horizon. *Journal of the Operational Research Society* 66, 2 (2015), 265–277.

[7] Parisa Ghaemi, Kaveh Shahabi, John P. Wilson, and Farnoush Banaei Kashani. 2010. Optimal network location queries. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems (CIS'10)*. 478–481.

[8] Gramoz Goranci, Monika Henzinger, and Dariusz Leniowski. 2018. A tree structure for dynamic facility location. In *Proceedings of the 2018 European Symposium on Algorithms*. Article 39, 13 pages.

[9] Russell D. Halper, S. Raghavan, and Mustafa Sahin. 2015. Local search heuristics for the mobile facility location problem. *Computers & Operations Research* 62 (2015), 210–223.

[10] A. K. M. Mustafizur Rahman Khan, Lars Kulik, Egemen Tanin, Hua Hua, and Tanzima Hashem. 2018. Efficient computation of the optimal accessible location for a group of mobile agents. *ACM Transactions on Spatial Algorithms and Systems* 4, 4 (2018), Article 10, 32 pages.

[11] Feifei Li, Dihan Cheng, Marios Hadjieleftheriou, George Kollios, and Shang-Hua Teng. 2005. On trip planning queries in spatial databases. In *Proceedings of the International Symposium on Spatial and Temporal Databases (SSTD'05)*. 273–290.

[12] Yuhong Li, Yu Zheng, Shenggong Ji, Wenjun Wang, Leong Hou U, and Zhiguo Gong. 2015. Location selection for ambulance stations: A data-driven approach. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL'15)*. Article 85.

[13] Zhenhui Li, Bolin Ding, Jiawei Han, Roland Kays, and Peter Nye. 2010. Mining periodic behaviors for moving objects. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'10)*. 1099–1108.

[14] Dimitris Papadias, Yufei Tao, Kyriakos Mouratidis, and Chun Kit Hui. 2005. Aggregate nearest neighbor queries in spatial databases. *ACM Transactions on Database Systems* 30, 2 (2005), 529–576.

[15] Jianzhong Qi, Rui Zhang, Lars Kulik, Dan Lin, and Yuan Xue. 2012. The min-dist location selection query. In *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering (ICDE'12)*. 366–377.

[16] Jianzhong Qi, Rui Zhang, Yanqiu Wang, Andy Yuan Xue, Ge Yu, and Lars Kulik. 2014. The min-dist location selection and facility replacement queries. *World Wide Web* 17, 6 (2014), 1261–1293.

[17] Ayse Durukan Sonmez and Gino J. Lim. 2012. A decomposition approach for facility location and relocation problem with uncertain number of future facilities. *European Journal of Operational Research* 218, 2 (2012), 327–338.

[18] Benoit Thierry, Basile Chaix, and Yan Kestens. 2013. Detecting activity locations from raw GPS data: A novel kernel-based algorithm. *International Journal of Health Geographics* 12, 11 (2013), 213.

[19] Waldo R. Tobler. 1970. A computer movie simulating urban growth in the detroit region. *Economic Geography* 46 (1970), 234–240.

[20] Betül Turanoglu and Gökay Akkaya. 2018. A new hybrid heuristic algorithm based on bacterial foraging optimization for the dynamic facility layout problem. *Expert Systems with Applications* 98 (2018), 93–104.

[21] Meng Wang, Hui Li, Jiangtao Cui, Ke Deng, Sourav S. Bhowmick, and Zhenhua Dong. 2016. PINOCCHIO: Probabilistic influence-based location selection over moving objects. *IEEE Transactions on Knowledge and Data Engineering* 28, 11 (2016), 3068–3082.

[22] Peng Wang, Lifeng Sun, Shiqiang Yang, and Alan F. Smeaton. 2015. Improving the classification of quantified self activities and behaviour using a Fisher kernel. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 International Symposium on Wearable Computers (UbiComp/ISWC'15 Adjunct)*. 979–984.

[23] Qian Wang, Rajan Batta, Joyendu Bhadury, and Christopher M. Rump. 2003. Budget constrained location problem with opening and closing of facilities. *Computers & Operations Research* 30, 13 (2003), 2047–2069.

[24] B. J. Worton. 1989. Kernel methods for estimating the utilization distribution in home-range studies. *Ecology* 70, 1 (1989), 164–168.

[25] Feng Xia, Jinzhong Wang, Xiangjie Kong, Zhibo Wang, Jianxin Li, and Chengfei Liu. 2018. Exploring human mobility patterns in urban scenarios: A trajectory data perspective. *IEEE Communications Magazine* 56, 3 (2018), 142–149.

[26] Xiaokui Xiao, Bin Yao, and Feifei Li. 2011. Optimal location queries in road network databases. In *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering (ICDE'11)*. 804–815.

[27] Da Yan, Zhou Zhao, and Wilfred Ng. 2011. Efficient algorithms for finding optimal meeting point on road networks. *Proceedings of the VLDB Endowment* 4, 11 (2011), 968–979.

[28] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik Lun Lee. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'11)*. 325–334.

[29] Man Lung Yiu, Nikos Mamoulis, and Dimitris Papadias. 2005. Aggregate nearest neighbor queries in road networks. *IEEE Transactions on Knowledge and Data Engineering* 17, 6 (2005), 820–833.

[30] Donghui Zhang, Yang Du, Tian Xia, and Yufei Tao. 2006. Progressive computation of the min-dist optimal-location query. In *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB'06)*. 643–654.

[31] Yu Zheng, Yanchi Liu, Jing Yuan, and Xing Xie. 2011. Urban computing with taxicabs. In *Proceedings of the 13th International Conference on Ubiquitous Computing (UbiComp'11)*. 89–98.

[32] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. 2009. Mining interesting locations and travel sequences from GPS trajectories. In *Proceedings of the 18th International Conference on World Wide Web (WWW'09)*. 791–800.

Received May 2019; revised August 2019; accepted September 2019