This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Protecting Privacy of Location-Based Services in Road Networks

Zheng Tan, Cheng Wang, *Senior Member, IEEE*, Chungang Yan, MengChu Zhou, *Fellow, IEEE*, and Changjun Jiang

***Abstract*— Location-Based Services (LBS), which answer users' location-dependent queries to Points of Interest (POI), have become popular along with mobile devices' widespread use. While benefiting from convenience, users may suffer privacy leak risk, due to publishing sensitive information, i.e., locations, to servers. Previous studies have proposed a number of methods for protecting LBS. Through providing provable privacy protection, Private Information Retrieval (PIR) becomes well-known for its high effectiveness in protecting privacy. However, the PIR-protected LBS must cautiously handle the consequent transmission/computation cost and service error brought by adopting PIR, as both of them impact user experience. Then an important question arises: Can the user experience of PIR-protected LBS be greatly improved? We answer it by presenting a novel interface from PIR to LBS, gaining benefits from road networks. The proposed interface is comprised of two road partition methods fully considering POI's distribution along roads, and two response-calculating methods to satisfy users' requirements on driving distance. Since Vehicular Location Based Services contain adequate road network knowledge, we accomplish our work on them. The experimental results on a real dataset validate that our interface can improve PIR-protected LBS greatly in user experience while protecting privacy.**

***Index Terms*— Internet of Vehicles, Location-Based Services (LBS), query privacy, computational private information retrieval (cPIR), Points of Interest (POI).**

I. Introduction

AMID the widespread use of mobile devices, e.g., smart phones [1], [2], people's real-time locations can be easily captured then published to information service companies. Benefiting from this, Location Based Services (LBS) emerge and rise, which allow user requests and company responses based on a user's location, for example, searching for nearby restaurants. While pursuing high accuracy and quality of LBS, researchers have to face privacy concerns [3]–[6], mainly stemming from publishing and utilizing a user's locations and personal information. For example, a GPS navigation system asks for real-time access to a vehicle's location [7] and transponders in Electronic Toll Collection (ETC) system transmit account ID and images of vehicles to servers [8]. Their related potential privacy leaking risk may stop LBS from widely spreading. Riley [9] has described this dilemma in his experiments in which users refuse to provide their location information in San Francisco.

Thus, constructing a non-privacy structure for LBS is a critical task. Without loss of generality, we concentrate on a typical kind of LBS in this paper where users publish their locations to a server expecting to receive nearby Points of Interest (POI). For simplicity, the singular and plural forms of POI and LBS are treated to be the same throughout this paper. In this scenario, disclosing locations is undesirable since the server is untrustworthy from the viewpoint of users. Many methods, including cloaking [10], anonymity [11], swapping [12], condensation [13] and differential privacy [14], have been proposed to address this problem. Although huge amounts of work and improvement have been done for these methods, an immanent vulnerability is just like a bitter pill which has to be swallowed, that the server must be aware of some location information to return nearby POI, no matter which privacy method is adopted. Since location information has to be published, these methods are very likely to be unacceptable for users, even if the processed published locations are inaccurate and incomplete. Private Information Retrieval (PIR) [15] is thus proposed to address this issue.

Generally, PIR can allow a user to retrieve information from a server but reveal nothing about what is retrieved. Its basic idea is publishing all data stored in servers to a user such that the user's querying messages need not contain any personal information. Then researchers [16], [17] devote themselves to compressing and encrypting these published data in mathematic and cryptographic methods to decrease the transmission complexity, while ensuring that users can restore their target information from the compressed data. In theory, PIR can provide complete and provable privacy protection, but incurs large cost of extra computation and transmission resources. Such cost has positive correlation with the scale of the problem by assuming the answer to any problem costs the same bits. In the assumption of PIR, the response to every

Manuscript received May 16, 2019; revised January 1, 2020 and April 3, 2020; accepted April 28, 2020. This work was supported by the National Key Research and Development Program of China under Grant 2018YFB2100801. The Associate Editor for this article was X. Cheng. *(Corresponding authors: Changjun Jiang; MengChu Zhou; Cheng Wang.)*

Zheng Tan, Cheng Wang, Chungang Yan, and Changjun Jiang are with the Key Laboratory of the Ministry of Education for Embedded System and Service Computing, Department of Computer Science, Tongji University, Shanghai 201804, China, and also with the Shanghai Electronic Transactions and Information Service Collaborative Innovation Center, Tongji University, Shanghai 201804, China (e-mail: moonmelody0418@gmail.com; chengwang@tongji.edu.cn; yanchungang@tongji.edu.cn; cjjiang@tongji.edu.cn).

MengChu Zhou is with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102 USA, and also with the Institute of Systems Engineering and Collaborative Laboratory for Intelligent Science and Systems, Macau University of Science and Technology, Macau 999078, China (e-mail: mengchu.zhou@njit.edu).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/TITS.2020.2992232

1524-9050 © 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

query should be specific and stored in servers, such that the problem scale is equal to the number of all possible queries. Thus, if tending to adopt PIR in practice with acceptable resource cost, we must select those scenarios with appropriate problem scales, or integrate some similar queries into a query to control the total query count.

Back to LBS, protecting them by PIR directly is quite hard as their queries labeled by locations are continuous and unlimited due to the characteristics of locations while PIR protocol asks for discrete and limited queries. Therefore, when employing PIR in protecting LBS, the primary mission is to design a scheme to partition locations into discrete patches, by which queries in the same patch would be integrated into one query. A supporting calculation module that can generate the nearby POI of each patch should also be presented. More formally, we call such partition scheme and calculation module an interface with which PIR can be utilized in protecting LBS. However, the interface imposes some other impacts on user experience of PIR-protected LBS, besides the service latency stemming from extra computation and transmission cost. When we utilize patches to replace precise locations, the responses, i.e., nearby POI, are made inaccurate consequently, causing the loss of service accuracy. To maintain the accuracy of LBS while protecting privacy, this kind of service error should also be kept at the low level. Thus, a well-designed interface should have acceptable computation and transmission cost and lower service error. Note that generating redundant nearby POI including all required ones and some undesirable ones is also regarded as service error in this paper.

Existing studies usually use grids to replace precise locations, and their metric of generating nearby POI is based on Euclidean distance, e.g., [19] uses a Voronoi diagram. In these studies, the resource cost and service error have negative correlation. Along with grid granularity increasing, the service error drops off while the computation and transmission cost grows rapidly. To achieve better service quality, some studies put efforts into improving PIR protocols which can decrease computation and transmission cost given certain service error. However, this kind of optimization has upper limit as this extra cost is structural and cannot be completely eliminated. Instead, we try to improve PIR-protected LBS from another viewpoint, i.e., decreasing the service error when given certain PIR protocols through improving the interface.

According to our literature survey, most kinds of POI, such as restaurants, convenient stores and hospitals, are located at roadside, especially in cities. On the other hand, in many practical scenarios, users are restricted to road networks, e.g., a user in urban districts or driving a car. Then it is obvious that if we utilize a novel partition scheme replacing a user's precise locations with the roads where s/he is, the service error can be decreased in comparison with using grids, as grids take lots of meaningless locations into account. Furthermore, in road networks, what users care about is driving distance, but not the Euclidean one. Thus, we should also present a calculation module whose 'nearby' POI depends on driving distance. In this view, replacing locations with roads can further decrease service error.

In this paper, we aim at improving the PIR-protected LBS by proposing a novel interface based on road network knowledge. With this interface, the service error of PIR-protected LBS is expected to be lower than that of traditional methods [19] when both methods cost the same computation and transmission resources. Vehicular LBS (VLBS) [22]–[26], a typical LBS for mobile vehicles, contains adequate road network knowledge, i.e., users in them are restricted to road networks and care more about road/driving distance. We thus construct our interface on it. Moreover, since a PIR-protected LBS scheme needs both a PIR protocol and an interface, we select computational PIR (cPIR) [27], a popular PIR protocol that only needs one single database, to cooperate with our interface. In a word, we show how to use cPIR to protect VLBS in this paper.

We propose a partition scheme to replace precise locations with road segments, and a calculation module to generate nearby POI for road segments. We present two partition methods, i.e., natural partition and regular one. The former adopts the natural partition of road networks, thus gains benefits on controlling service error from the positive correlation between POI density and road density. The latter imposes manually defined segment length on the basis of natural partition such that the tradeoff between service error and resource cost can be further improved. Then we present a module to calculate nearby POI for road segments, i.e, those POI whose driving distance to a road segment are less than a threshold. Its core is a path tree constructing algorithm that can traverse all eligible POI through a depth-first tree. Further, we give a more accurate method using driving time as the discrimination criterion of nearby POI since VLBS can provide vehicles' historical trajectories. Some other algorithms, such as a driving time estimation model and tensor decomposition method, are provided to realize this method.

Next, we validate the effectiveness of the proposed interface through a series of experiments. We first conduct experiments on our interface only, by changing the values of parameters in the partition schemes and calculation module. The results show that our method can provide a reliable protection for VLBS, with quite small service error, and acceptable computation and transmission cost. Then we compare our interface with the existing PIR-protected VLBS. The results validate that our method can achieve much lower service error when costing the same transmission and computation resources as existing methods. The contributions of this work are:

1. To improve PIR-protected LBS, it proposes an interface based on road network knowledge, including two road partition schemes and a nearby POI calculation module; and
2. It shows experimentally that cPIR-protected VLBS with our interface possesses lower service error than existing methods, without increasing transmission/computation cost.

Section II shows the related work on location privacy. In Section III, we elaborate the principle and function of cPIR. In Section IV, the proposed interface is introduced in detail. In Section V, we discuss how a cPIR structure works in

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

practice. Section VI discusses experimental results and their implications. Finally, we conclude this work in Section VII.

II. Related Work

When protecting location privacy in LBS, cloaking is a typical method, which makes the published locations inaccurate. However, inference attack [28], which infers precise locations from vague formats by integrating external databases, can easily disable this method. Thus, K-anonymity is proposed to address inference attack, and becomes popular in practical LBS. Sweeney shows the basic idea of K-anonymity in [11], which prefers to cloak a user among K users, by making their quasi-identifiers take the same vague values. Then an attacker only has $\frac{1}{K}$ confidence on inferring a user's sensitive information when connecting two or more databases through quasi-identifiers. Based on K-anonymity, Gedik and Liu [29] cloak the time and location simultaneously. When a user proposes a query, only with other K–1 users proposing queries in a definitive time interval can the communication with the server be constructed. Otherwise, this query is rejected. The effectiveness of K-anonymity in LBS and VLBS has been validated in [30], [31].

While K-anonymity is popular with LBS because of implementation simplicity, its vulnerabilities on protecting privacy are obvious. First, it is conducted on databases, thereby a third trusted anonymizer is essential to capture and store users' data. Protecting locations locally is impossible due to the lack of global features. However, entrusting the whole protection system on a single point is unadvisable. Once attackers gain access to it, all users' data are in danger, not to mention how to ensure that it is trusted [19]. On the other hand, protecting an isolated location cannot resist correlation attack, which infers precise locations from moving trajectories through a Hidden Markov model [32], [33].

To address these problems, some other methods are presented. One classical method is the transformation approach proposed in [34]. As its name indicates, this method tends to conduct LBS in an encrypted space. The elements in a map, such as roads, locations and POI, are embedded into a high-dimensional space, with their features in an actual space, such as length and directions, being preserved. As the dimensions of a new space have no actual meaning, the whole querying process is encrypted for the server such that the location privacy is protected. It is validated that both users and the server can conduct their tasks efficiently in the encrypted space. Another kind of methods is the cryptographic-based ones. In [35], [36], the secure multi-party computation is utilized to obtain strong privacy guarantees. In [37], Popa *et al.* propose a practical system called VPriv that utilizes cryptographic protocols. They prove that it can run efficiently in real-time VLBS. Yeh and Lin [38] propose a proxy-based authentication and billing scheme for vehicular networks. In [39], Lu *et al.* propose a dynamic privacy-preserving key management scheme for VLBS. No matter which transformation approaches or cryptographic-based methods are carried out on practical applications, large computation complexity represents a serious challenge.

Apart from the above ones, some other methods have also been utilized in LBS and VLBS. Yiu *et al.* [40] provide a method called SpaceTwist in which users may propose a series of queries which include only a real one, while other fake ones are generated from fake locations around a real point. To resist correlation attack, fake locations are given in ascending distance order. Differential privacy is also used for protecting LBS. Andrés *et al.* [41] present a method to make location indistinguishable by adding random noise.

The existing work adopting PIR contains two categories, i.e., hardware-based PIR [42], [43] and cPIR protocols [18]–[21]. The former realizes PIR through hardware, while the latter relies on computational number theory. Both of them have to face huge computation and transmission complexity.

III. cPIR Framework in LBS

To establish an interface between cPIR and VLBS, we should first have a deep insight into cPIR. Thus, in this section we firstly introduce the structure of cPIR. Afterwards, we evaluate the privacy level of cPIR through differential privacy. At last, we discuss the adjustment of a cPIR structure when applying it to practice.

A. cPIR Structure

cPIR is a practical method evolving from theoretical PIR, by gaining benefits from the computational bound providing by Quadratic Residuosity Assumption (QRA). The quadratic residuosity problem is to decide whether a is a quadratic residue modulo S, where S is a k-bit large composite number by multiplying two $\frac{k}{2}$-bit primes p_1 and p_2, and a is a residue of S. QRA, initially used as a cryptographic setting by Goldwasser and Micali [44], states that it is computationally hard to find the solution of a quadratic residuosity problem, i.e., the quadratic residues of a large composite number. The privacy guarantee of cPIR just originates from this computational hardness. Next we show how QRA works in protecting privacy and introduce the cPIR structure in practical scenarios.

We illustrate a typical information retrieval scenario in Fig. 1, in which a user selects a query q_h from the query set $\{q_1, q_2, \ldots\}$ and publishes it to a server, then the server extracts corresponding returned data $R(q_h)$ from the returned data set $\{R(q_1), R(q_2), \ldots\}$ and transmits it to the user. When utilizing cPIR to protect privacy, the server should construct an $m \times n$ PIR matrix M, whose element $M_{i,j}$ stores the returned data $R(q_{(j-1)\cdot m+i})$. Its scale, $m \times n$, is provided to users before retrieval starts.

Then the PIR-protected retrieval begins. Firstly, a client encodes queries based on the scale of PIR matrix. For query q_h, the client can calculate the location of its returned data in PIR matrix, $M_{a,b}$, through $a = h \bmod m$ and $b = \lceil h/m \rceil$. Then the client randomly generates a k-bit composite number $S = p_1 \cdot p_2$ as a modulus and a residue set $Y = \{y_1, y_2, .., y_n\}$, in which $y_b \in \overline{Q}(S)$ and $\forall x \neq b, y_x \in Q(S)$. $\overline{Q}(S)$ denotes the quadratic non-residue set of S while $Q(S)$ is the quadratic residue set. The PIR query, containing both the modulus and residue set, is transmitted to the server,

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

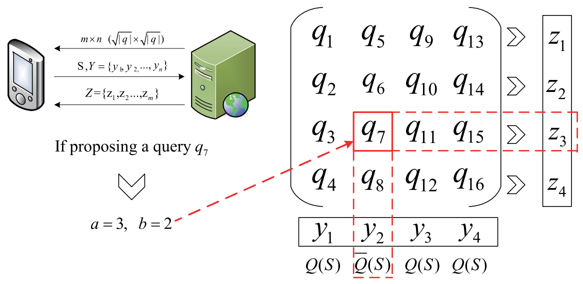

$m \times n$ $(\sqrt{|q|} \times \sqrt{|q|})$
$S, Y = \{y_1, y_2, \ldots, y_n\}$
$Z = \{z_1, z_2 \ldots, z_m\}$
If proposing a query q_7
$a = 3, \ b = 2$
q_1 q_5 q_9 q_{13}
q_2 q_6 q_{10} q_{14}
q_3 q_7 q_{11} q_{15}
q_4 q_8 q_{12} q_{16}
z_1 z_2 z_3 z_4
y_1 y_2 y_3 y_4
$Q(S)$ $\overline{Q}(S)$ $Q(S)$ $Q(S)$

Fig. 1. System structure and cPIR process.

while the factorization of S is kept secret. According to QRA, the server cannot distinguish quadratic non-residues and quadratic residues of S, and thus cannot konw which element is queried. This query is $(n+1)*k$ bit in total.

After receiving a PIR query, the server computes a number $z_i \in Z_S^*$ for each row as follows: It first computes

$$\omega_{i,j} = \begin{cases} 1 & \text{if } M_{i,j} = 0 \\ y_j & \text{if } M_{i,j} = 1 \end{cases} \tag{1}$$

then computes

$$z_i = \prod_{j=1}^{n} \omega_{i,j}. \tag{2}$$

Note that each z_i can be simplified to a k-bit integer. All calculated z_i's constitute the returned data of query q_h, denoted by $Z = \{z_1, z_2, \ldots, z_m\}$, along with the transmission cost mk bits.

At last, the client can decode the value of $M_{a,b}$ since it knows the factorization of S. It chooses the ath element in Z and then decodes the value of $M_{a,b}$ via Legendre Symbol:

$$M_{a,b} = \begin{cases} 1 & \text{if } z_a \in \overline{Q}(N) \\ 0 & \text{if } z_a \in Q(N) \end{cases} \tag{3}$$

Then the process of retrieving information ends with the client obtaining expected information and the server gaining nothing. The proof of QRA and a more detailed description on PIR process are given in Appendix A.

Note that the communications between a server and a client cost $(m+n+1)*k$ bit in total. As the matrix scale $m \times n$, i.e., the number of all possible queries, is definite, the transmission cost can achieve the minimum when $m = n$. Thus, assuming the query count is $|q|$, the PIR matrix should be set as a square matrix, with a scale $\sqrt{|q|} \times \sqrt{|q|}$ (if $|q|$ is not a square number then the server would fill extra data to M). In a word, cPIR can protect privacy at the price of transmission complexity of $O(\sqrt{|q|})$ and computation complexity of $O(|q|)$. Consequently, the transmission and computation cost are two critical criteria in evaluating the practicability of a cPIR structure. Moreover, it is obvious that the transmission and computation cost increases along with the query count $|q|$ growing. Thus, when applying cPIR in practice, we must control the query count, i.e., the problem scale.

B. Privacy Evaluation

Besides transmission and computation cost, we should make the privacy-preserving level of a cPIR structure clear. Although there exist various metrics to evaluate privacy, we choose the strictest and general one, ϵ-differential privacy [45]. According to its definition, a privacy-preserving method $\mathcal{A}()$ satisfies ϵ-differential privacy if any two neighboring values x_1 and x_2 in a database are subject to the following equation after being protected by $\mathcal{A}()$:

$$\frac{Pr(\mathcal{A}(x_1) \in Y)}{Pr(\mathcal{A}(x_2) \in Y)} \le e^{\epsilon}, \tag{4}$$

where Y denotes any subset satisfying $Y \in \mathbb{Y}$ and $\mathbb{Y}$ is the range of $\mathcal{A}()$. Along ϵ axes, it becomes more and more difficult to distinguish different original values (i.e., x_1 and x_2) for a server who can only get access to the protected values (i.e., $\mathcal{A}(x_1)$ and $\mathcal{A}(x_2)$), and therefore the privacy information gets better and better protection. Intuitively, this parameter indicates the probability of inferring actual data from the protected ones. Note that in PIR, x_1 and x_2 should belong to two different columns, o_a and o_b, of a PIR matrix to satisfy the definition of neighboring values.

Next, we utilize this metric to measure cPIR. cPIR is regarded as a complete method and does not disclose any information when it is not cracked. For example, if it is utilized in VLBS, a server can only know the querying user is in a road network, but has no idea about his/her accurate location. In other words, no matter where two querying users are, their location information is always the same in the view of a server, that they are both distributed uniformly in the road network. Thus, as long as cPIR works, the protected values accessed by a server can always constitute the same, independent of the original values. More formally, the probability of the protected values belonging to any column o_i is:

$$Pr(\mathcal{I}(x_1) = o_i | \mathcal{I} = \mathcal{I}_1) = Pr(\mathcal{I}(x_2) = o_i | \mathcal{I} = \mathcal{I}_1) = \frac{1}{|o|} \tag{5}$$

where $\mathcal{I}$ denotes a cPIR method, $\mathcal{I}_1$ is a working cPIR structure and $|o|$ (i.e., n) is the column count.

However, we should also recognize that cracking and disabling a cPIR structure is possible. Recalling (A.10), when the factorization of S is unknown, the probability of distinguishing quadratic residue and non-residue is composed of two components. The first component, namely, the probability of $\frac{1}{2}$, originates from the fact [44] that in a normal situation, the quadratic residues and non-residues take up half and half in the set $\{a\}$ when $\left(\frac{a}{S}\right) = 1$. The latter one, the probability of $\frac{1}{k^c}$, is relevant to cPIR structure failure. For simplicity, we only consider complete failure as the deduction of considering partial failure is similar. If this nearly negligible event happens to arise, then a server can directly obtain the column where an original query locates. It is formulated as follows:

$$\begin{cases} Pr(\mathcal{I}(x_1) \in Y_1 | \mathcal{I} = \mathcal{I}_2) = 1 \\ Pr(\mathcal{I}(x_2) \in Y_2 | \mathcal{I} = \mathcal{I}_2) = 1 \end{cases} \tag{6}$$

in which $\mathcal{I}_2$ represents a disabled cPIR method, $Y_1 = \{o_a\}$ and $Y_2 = \{o_b\}$. Moreover, the probability distribution of a working and disabled cPIR structure can also be formulated as follows

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

according to (A.10):

$$\begin{cases} Pr(\mathcal{I} = \mathcal{I}_1) > 1 - p_f \\ Pr(\mathcal{I} = \mathcal{I}_2) < p_f \end{cases} \tag{7}$$

At last we should take overall consideration of the above two situations. Without loss of generality, we set $Y = Y_1 = \{o_a\}$. Then through combining (5) - (7), we have:

$$\begin{aligned} &\frac{Pr(\mathcal{I}(x_1) \in Y)}{Pr(\mathcal{I}(x_2) \in Y)} \\ &= \frac{Pr(\mathcal{I}(x_1) \in Y, \mathcal{I} = \mathcal{I}_1) + Pr(\mathcal{I}(x_1) \in Y, \mathcal{I} = \mathcal{I}_2)}{Pr(\mathcal{I}(x_2) \in Y, \mathcal{I} = \mathcal{I}_1) + Pr(\mathcal{I}(x_2) \in Y, \mathcal{I} = \mathcal{I}_2)} \\ &= \frac{Pr(\mathcal{I}(x_1) \in Y|\mathcal{I}=\mathcal{I}_1) \cdot Pr(\mathcal{I}=\mathcal{I}_1) + Pr(\mathcal{I}(x_1) \in Y, \mathcal{I}=\mathcal{I}_2)}{Pr(\mathcal{I}(x_2) \in Y|\mathcal{I}=\mathcal{I}_1) \cdot Pr(\mathcal{I}=\mathcal{I}_1) + Pr(\mathcal{I}(x_2) \in Y, \mathcal{I}=\mathcal{I}_2)} \\ &< \frac{1 + |o| \cdot p_f - p_f}{1 - p_f} = e^{\epsilon_c}. \end{aligned} \tag{8}$$

Based on this equation, we can obtain $\epsilon_c = \log \frac{1 + |o| \cdot p_f - p_f}{1 - p_f}$. Then the cPIR structure satisfies ϵ_c-differential privacy.

C. cPIR for Multiple-Bit Data

In practical applications, the information to be queried always contains multiple bits, bringing possibility to further decrease on transmission cost by changing the shape of a PIR matrix. Assuming that the returned data of a query have w bits, then w PIR matrices should be constructed in the server. When proposing a query, a user only needs to send one residue set $Y = \{y_1, y_2, \ldots, y_n\}$, the same as the 1-bit situation, while the server should return w groups of Z. If the scale of PIR matrix is $m \times n$, the transmission cost can be expressed as $C_t = k \cdot n + w \cdot k \cdot m$. Note that the computation cost does not vary with the shape since the definitive product $|q| = m \times n$ leads to fixed multiplication count. On the other hand, the transmission cost achieves its minimal value when $m = \sqrt{\frac{|q|}{w}}$ and $n = \frac{|q|}{m}$. Considering that both m and n should be nonzero integers, we make a judgement on which approximate value of m, rounding up or down, is the optimal solution. Then the column size n can be computed as $n = \lceil \frac{|q|}{m} \rceil$. These w matrices also constitute a PIR tensor with a scale of $m \times n \times w$.

IV. Interface From cPIR to VLBS

In this section we construct an interface between cPIR and VLBS on the basis of road network. As said above, an interface should contain a partition scheme and a nearby POI calculation module. Thus, we first propose two location partition methods, which partition a road network into segments and then record them. Next we demonstrate how to generate the response for a segment through constructing a path tree. At last we give the exact definition of service error.

A. Location Partition Scheme

Since we tend to use cPIR to protect VLBS, we have to overcome its difficulties, i.e., the queries must be discrete to satisfy its structural requirements, and the query count should be controlled within a relatively small range to make the transmission/computation cost acceptable. Existing studies address them by utilizing girds to partition continuous location, then replacing precise locations with their corresponding grids. Although their experiments validate that this solution makes the resource cost acceptable, the service error stemming from using approximate locations is a bit too large. Thus, we aim to present a novel partition scheme which can decrease the service error.

Before introducing our method, we first make a general introduction to service error. A typical service in VLBS is calculating and returning nearby POI based on users' locations. Since the precise locations are replaced by approximate grids, the calculated POI are inaccurate. We define service error as the difference between precise and inaccurate POI.

Then we disclose the disadvantages of using grids. In theory, if both users and POI are distributed uniformly in a region, then partitioning this region with grids may be the optimal solution as the service error cannot be decreased further. However, according to our survey, users in LBS are always restricted by road networks, i.e., they usually appear on roads. In actual maps, POI are mainly located at the sides of roads. Obviously, using grids does not take full use of the above road information, thus causing large service error. Moreover, in road networks, users usually care more about the driving distance between POI and themselves, but not the Euclidean distance used in traditional methods. Then an intuitive idea to overcome these disadvantages is replacing users' precise locations with the road segments where they are.

Thus, in this part, we introduce how to partition the road network. As our goal is to decrease service error, we first discuss which factors would influence it. Generally, if the POI distribution is definite, then the smaller the length of a road segment is, the smaller service error is. However, shortening road segments has negative impact since the segment count, as well as resource cost, increase rapidly consequently. Then we look at another fact: if the road partition is definite, road segments in high POI density regions, as well as high user density regions, affect overall service accuracy more. Based on the two facts, a compromised method considering both service error and resource cost is setting small segment length in high POI density regions and large length in low density regions. Fortunately, the natural partition of actual road networks can satisfy these settings if we regard the roads between two adjacent crossings as a road segment. In urban regions, POI density is high and length between two adjacent crossings is small, while these in suburb regions are just the opposite. Besides, this partition scheme brings much more benefits to decreasing service error through separating crossings from road segments. This point will be explained in the next subsection. Thus, we adopt the natural partition as a partition method in our work.

More formally, in natural partition, the road between two adjacent crossings is treated as a road segment, named by a natural road segment (abbreviated as NRS). For a particular region G, the ith crossing in it is denoted by c_i, constituting a crossing set C. Then the NRS set is defined as $\mathbb{S} = \{n_i | \ \forall c_a, c_b \in C, n_i = \langle c_a, c_b \rangle\}$, in which n_i is the NRS between crossing c_a and c_b. Moreover, if n_i is a two-way road,

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

this segment should be refined into two NRSs. For simplicity, we adopt a superscript, l or r, to differentiate the two directions of one road segment, denoted by n_i^l and n_i^r. The NRS count is expressed by $|\mathbb{S}|$.

Next we present another partition method. Although natural partition is effective in theory, the service error of those long road segments still tends to be large. Conducting further partitions on them can achieve a better tradeoff between service error and transmission/computation cost. Thus, we define a partitioning length L to accomplish this task. Those NRSs whose length is less than L remain the same while others would be partitioned into several segments. This partition scheme is named as regular partition, and the road segments in it are called regular road segments, abbreviated as RRS. Taking NRS n_i as an example, if its length, $L(n_i)$, is less than a given L, then this entire NRS is defined as an RRS. On the contrary, if $L(n_i) \geq L$, n_i is partitioned into $\lfloor \frac{L(n_i)}{L} \rfloor$ L-length segments from one endpoint. Apart from these full segments, the remainder of n_i, whose length is $L_r = L(n_i) - L \cdot \lfloor \frac{L(n_i)}{L} \rfloor$, needs special handling. If $L_r < \frac{L}{2}$, then this remanent part should be integrated into the last RRS in n_i; otherwise it should be regarded as an independent RRS. The RRSs generated through the above method are to be labeled by r_j, constituting an RRS set $\mathbb{R} = \{r_j\}$, whose count is $|\mathbb{R}|$. The samples of NRS and RRS are illustrated in Fig. 2. Note that query q_i mentioned in Section III means the query on n_i or r_i in the following discussions.

B. Generating Returned Data

After partitioning road networks, we next discuss how to generate nearby POI for every road segment. Generally, nearby POI, abbreviated as NPOI, denote these ones which can be reached within a definitive driving time. For a specific NRS n_i or RRS r_j, we utilize an NPOI set to record all their NPOI, denoted by $\mathbb{N}(n_i)$ or $\mathbb{N}(r_j)$.

In practice, an alternative criterion, driving distance, is also acceptable for users if we lack historical trajectory records as they are indispensable for predicting driving time. In what follows, we propose a series of algorithms calculating NPOI sets based on non-trajectory and trajectory-aware conditions. When designing them, we mainly comply with two design metrics: (i) finding all NPOI without omission; and (ii) simulating the practical VLBS as much as possible. They guarantee the reliability and practicality of our methods. Moreover, as whether road segments are RRSs or NRSs makes no difference on calculating NPOI, we take the natural partition as an instance in this section.

1) POI in VLBS: Before generating NPOI, we prefer to perform approximate treatment on POI to simplify the model. Although there exist spaces from roads to POI in actual scenarios such that interacting with POI requires drivers to park and walk, this space can be omitted as most POI are just located at roadsides. And omitting this space can help us converge the NPOI generation model to road networks. Thus, we map every POI to a road segment. By making a vertical line to its nearest road, each POI has a mapping point on a road. Then reaching these mapping points can be regarded as reaching corresponding POI. The POI mentioned in the following sections are these mapping points. Moreover, we assume that no POI can locate at crossings considering traffic rules.

2) Non-Trajectory Condition: When lacking historical trajectories, we adopt a boundary distance, denoted by D, as the discrimination criterion of NPOI in non-trajectory conditions. In order to explain how D works, we introduce a concept of a nearby road set that contains the nearby roads of an NRS. Setting a given location l_i as a starting point, its nearby road set is composed of those roads within D distance from the starting point. Then its NPOI set $\mathbb{N}(l_i)$ is composed of all POI locating at its nearby roads.

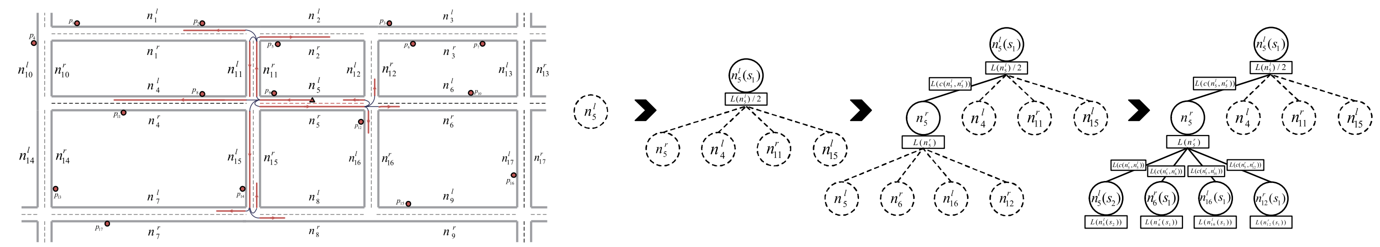
Next we introduce how to calculate the nearby road set, i.e., the NPOI set, of an NRS through constructing a path tree. First we should determine starting point of an NRS as NRSs are not precise points. According to our survey on real datasets, people do not show preference on any specific area of an NRS when proposing queries. The querying locations usually distribute uniformly in an NRS. Hence, we choose the mid-point of an NRS as its starting point. This point is named as an approximate location of users querying in this NRS. Obviously, the service error just originates from this approximating treatment. In this setting, a point's NPOI set is regarded as the NOPI set of an NRS. We can also generate a redundant NPOI set for an NRS, which contains the NPOI of every point in this NRS. To calculate it, we just need to regard an NRS's endpoint along the driving direction as the approximate location and calculate all its NPOI. Then the NPOI of the endpoint and those POI located at this NRS compose the redundant NPOI set. Both measures incur service error. And both of them can validate the benefits of using road network knowledge when compared to existing works. We show the first one in this paper. Then we construct a path tree to traverse its nearby roads. In the tree, nodes represent NRSs weighted by their length and edges are crossings connecting two NRSs. Each edge is also assigned a weight that measures the distance between two NRSs it connects. A path tree is only subject to one criterion, that the sum of all weights, including those for nodes and edges, on one path from the root node to any leaf node, should be equal to D.

We elaborate a depth-first tree-constructing process through taking NRS n_5^l as an instance. Intuitively, the guideline is to construct all D-length pathes, incorporating several NRSs and crossings. Thus, it is necessary to give a function that can compute the distance between any two points in one NRS or crossing. Herein we introduce a distance function $f_d(\cdot)$. Since the linear distance between two points can be easily calculated according to their longitudes and latitudes, partitioning these nonlinear roads and crossings into numerous linear segments then summing all their length is the basic idea of $f_d(\cdot)$. The detailed discussions combining actual scenes are to be given in Section VI-A.

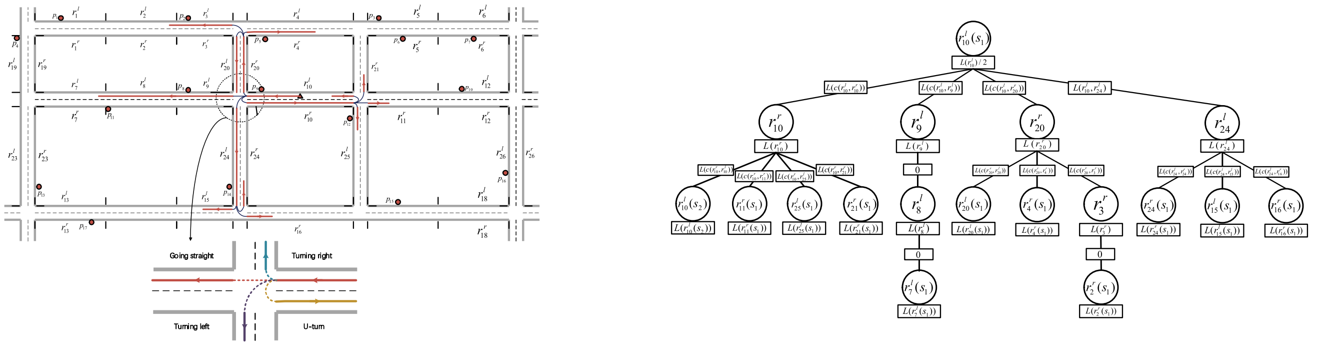
As shown in Fig. 2(a), length accumulation starts at the mid-point of n_5^l, thereby the root node is temporarily written as $n_5^l(s_1)$. Note that the format $n_i(s_j)$ signifies the jth sub-segment of NRS n_i, in which j has no practical significance but assists people to distinguish different sub-segments.

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



(a)



(b)

Fig. 2. (a) shows the natural partition on a road network and a sample of constructing a path tree, in which these nodes depicted by dotted circles denote candidate ones; and (b) shows the regular partition on a road network and a whole path tree of RRS r^l_{10}. These roads labeled by red lines denote the nearby roads of n^l_5 in (a) and r^l_{10} in (b).

Moreover, the starting and ending points of this sub-segment, expressed as latitudes and longitudes, are to be determined and recorded in this node. While the starting point of $n^l_5(s_1)$ is given, its ending point depends on actual situations. If $\frac{L(n^l_5)}{2} > D$, it is obvious that the discrimination criterion D is satisfied within NRS n^l_5. Then node $n^l_5(s_1)$ denotes a D-length road segment that starts from the mid-point and spreads along driving direction. The latitude and longitude of its ending point can be determined based on the starting point and length. Its weight $L(n^l_5(s_1))$ is D. So far the tree-construction process has finished.

Otherwise, the construction process goes on with half n^l_5 recorded in node $n^l_5(s_1)$ and assigns weight $\frac{L(n^l_5)}{2}$. The next steps give full expressions to the idea of *depth-first* construction. We maintain a parameter L^* to record the travelled length, e.g., $L^* = \frac{L(n^l_5)}{2}$ currently, and generate an adjacent set $A(n^l_5)$ recording all adjacent roads of n^l_5 towards driving directions, i.e., $A(n^l_5) = \{n^r_5, n^l_4, n^r_{11}, n^l_{15}\}$. This adjacent set signifies drivers' possible choices when they meet the crossing at the end of NRS n^l_5. Then we temporarily set these elements in $A(n^l_5)$ as child nodes of node $n^l_5(s_1)$ such that corresponding edges can be created.

Next we should determine whether these candidate nodes deserve their places, child nodes of $n^l_5(s_1)$, or not. We first take n^r_5 into account. The edge between nodes n^r_5 and n^l_5 signifies the crossing connecting them, denoted by $c(n^l_5, n^r_5)$. The weight of this edge, representing the length of the u-turn from n^l_5 to n^r_5, is expressed as $L(c(n^l_5, n^r_5))$. After updating the value of L^* with $L^* = \frac{L(n^l_5)}{2} + L(c(n^l_5, n^r_5))$, a determining process on n^r_5 is conducted as follows: If $L^* \geq D$, a null symbol, Ø, takes this place. Otherwise, we should judge whether $L^* + L(n^r_5)$ is larger than D or not. If so, node n^r_5 is replaced by node $n^r_5(s_1)$ whose weight is $D - L^*$. This node represents a sub-segment on n^r_5 starting at crossing $c(n^l_5, n^r_5)$. Obviously, a D-length path, from node $n^l_5(s_1)$ to node $n^r_5(s_1)$, has been constructed, in which $n^r_5(s_1)$ is a leaf node. Then we need to backtrack to its parent node along with $L^* = L^* - L(c(n^l_5, n^r_5))$ and process the next child node, i.e., node n^l_4. If not, the candidate node n^r_5 becomes a regular member, with a weight $L(n^r_5)$. The construction continues to go deeply by generating child nodes $A(n^r_5) = \{n^l_5, n^r_6, n^l_{16}, n^r_{12}\}$ and updating $L^* = \frac{L(n^l_5)}{2} + L(c(n^l_5, n^r_5)) + L(n^r_5)$. The above steps are implemented recursively until all leaf nodes are generated, which also indicates the completion of a path tree. Along with the construction process, we maintain an array recording traversed ancestor nodes. When a new candidate node has already existed in this array, no more additional implementation is required and we skip to the next node. We illustrate this tree-construction algorithm in Fig. 2(a) by showing the detail of generating a length-D path.

After constructing a path tree of an NRS, we can easily obtain its NPOI by traversing all nodes, for example, $\mathbb{N}(n^l_5) = \{p_2, p_5, p_8, p_9, p_{12}, p_{14}\}$. We should construct a path tree for each NRS in preprocessing. Although the interface of natural partition has been established, there still exists a problem in regular partition. When we compare the road partitions in Figs. 2(a) and 2(b), one difference stands out that any two NRSs are connected by a crossing, while regular partition is not subject to this pattern. For example, there does not exist a crossing, or even any gap, between RRS r^l_8 and r^l_9. Thus, for regular partition, the definition of an edge in a path tree is not restricted to a crossing, but represents a transfer from one RRS to another. Then if a transfer meets a crossing,

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

its weight is set as the crossing's length; and otherwise its weight is 0. We illustrate this idea as well as a whole path tree of regular partition in Fig. 2(b).

The above method is a basic one embodying the idea of transforming road networks into directed and weighted graphs, and can help explain experimental results. Based on this, the Dijkstra algorithm can be adopted in actual calculation. Moreover, the tree-construction process also explains the reason for which crossings must not be included in road segments. If two locations are separated by a crossing, their NPOI sets are likely to be quite different even if they are close. Then ensuring a road segment does not cover any crossing can make the difference among the NPOI sets of locations in one road segment smaller, thus decreases service error. This validates that our two partition schemes are reasonable compared to partitioning roads only based on length but neglecting crossings.

3) Trajectory-Aware Condition: Although less driving distance is useful in finding NPOI, some complex road conditions under which travelling on different roads with equal length may differ a lot over driving time, thus making the above structure approximate. Hence, a precise structure, directly employing driving time as a discrimination criterion, is in demand. Benefiting from historical driving trajectories, we propose an improved algorithm given a trajectory-aware condition.

Analogous to a non-trajectory condition, a path tree is also essential with a boundary time T and a time estimating function $f_t(\cdot)$. T is a hyper-parameter representing drivers' expectations on NPOI. Function $f_t(\cdot)$ is used to estimate the driving time between any two points on one NRS. Obviously, almost all these assumptions are nothing new, except the time estimating function. Thus, the main task of this part is to elaborate how $f_t(\cdot)$ works.

Actually, there are many studies addressing the problem of estimating travel time through utilizing statistical or learning models, such as markov model and neural network. Some of them train models on realtime data which contain traffic flow speeds and road messages, while others rely on historical data, which record the historical trajectories under various environmental conditions, such as weather, time and congestion conditions. Although the realtime model should be more accurate in theory, the history-based model also achieves pretty acceptable accuracy through detailed distinguishment on environmental conditions. Both kinds have widespread use in practical applications [46], [47].

In this paper, we select history-based models as our estimating method due to the restriction of cPIR. As shown in Section III, the server must generate a PIR matrix which contains NPOI of all road segments before implementing PIR process. The running mode of realtime models [48], [49] is realtime response that the server calculates returned values after receiving queries. If integrating cPIR method to realtime models, the server has to run NPOI-calculation program on all roads every time it receives a query. It brings too large computation burden and results in too high latency. On the contrary, the history-based models can provide PIR matrices offline. These models [50], [51] usually regard the average time of historical trajectories under a specific environmental condition as the estimation of this condition. By utilizing history-based models, the server can generate and store corresponding PIR matrices for each specific environmental condition (e.g., 12:00 on Monday, sunny and moderate congestion) in advance. According to our survey on real datasets, the environmental condition count is limited at the appropriate level of granularity. Thus, the server just needs to store finite PIR matrices. When a query arrives, the server selects a PIR matrix matching current environmental condition and implements PIR process on it. Obviously, cPIR structure with history-based models is more applicable in practice.

As our trajectory dataset only contains vehicles' locations and corresponding time slots, we simplify the environmental condition as congestion condition in this work. Based on our survey on dataset, we partition a day into several time quanta with different congestion conditions. In a time quantum, we employ the average of all user's driving time on a road segment except outliers as its estimated driving time. More formally, by assuming the driving time of the ith driver on the jth NRS or crossing at the kth time quantum is $t(i, j, k)$, we have the estimated time as:

$$t^*(j,k) = \frac{\sum_{j=1}^{n} t(i,j,k)}{n}, \tag{9}$$

in which n is driver count. However, there exist two problems to be addressed: (i) The historical trajectories are very likely to be sparse. For example, some NRSs and crossings have been travelled by only a few drivers, which may cause errors on estimating driving time; and (ii) As shown in a path tree, lots of nodes are not entire NRSs. What we actually need is a flexible function estimating the driving time of any two points, not restricted to an entire NRS.

Next we solve these problems one by one. Motivated by the method in [52], we adopt tensor decomposition to fill sparse data. First we construct a trajectory tensor A_r with three dimensions, user id i, NRS or crossing id j and time quantum k, whose element is $t(i, j, k)$. We prefer to decompose it into the multiplication of a core tensor $S \in \mathbb{R}^{d_U \times d_R \times d_W}$ and three matrices, $U \in \mathbb{R}^{E \times d_U}$, $R \in \mathbb{R}^{F \times d_R}$ and $T \in \mathbb{R}^{G \times d_T}$, in which E, F and G represent the user, NRS/crossing, and time quantum count, and d_U, d_R and d_T are latent factor count, usually self-defined. Then an objective function is defined as:

$$\begin{aligned} L(S,R,U,T) = \frac{1}{2}\|A_r - S \times_U U \times_R R \times_T T\|^2 \\ + \frac{\sigma}{2}(\|S\|^2 + \|U\|^2 + \|R\|^2 + \|T\|^2), \end{aligned} \tag{10}$$

where the format $||\cdot||^2$ denotes l_2 norm, $\frac{\sigma}{2}(\|S\|^2 + \|U\|^2 + \|R\|^2 + \|T\|^2)$ is the regularization factor to avoid over-fitting, and σ is used to control the contributions of this factor. After a training process with gradient descent, we can obtain a full tensor as $A_f = S \times_U U \times_R R \times_T T$.

Afterwards, we concentrate on how to estimate driving time between any two points in one NRS. Herein we introduce the idea of sectors in motor sports, for example, Formula 1. Generally, a circuit in Formula 1 is partitioned into 3 or 4 sectors according to their characters, such as more chicanes, more high-speed corners, or more straights. The time statistic of one sector is relatively independent from others', thereby the travel in one sector can be considered as uniform motion, while

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

the averaged speed differs in different sectors. Back to our problem, one NRS can be approximately regarded as a sector such that the driving time between any two points can be easily calculated. However, this analogy is imperfect stemming from neglecting one critical point, that only with similar characters can roads be included into one sector. Considering actual scenes, it is common that road and traffic conditions vary along one NRS, due to all sorts of reasons, such as adjacent to schools, lane count changing, or even some segments under maintenance. We discuss how to partition sectors more detailedly in Appendix C, combining practical applications. Then each NRS in the trajectory tensor is replaced by several sectors. After a tensor decomposition process, we can give the time estimating function $f_t(\cdot)$.

C. Service Error

After elaborating the partition scheme and calculation module, we can finally give the formal definition on service error:

$$\lambda = 1 - \frac{|\mathbb{N}(q_i) \cap \overline{\mathbb{N}}(q_i)|}{max\{|\mathbb{N}(q_i)|, |\overline{\mathbb{N}}(q_i)|\}}, \tag{11}$$

in which $\overline{\mathbb{N}}(q_i)$ is the actual NPOI set of query q_i and $\mathbb{N}(q_i)$ denotes the NPOI set of NRS n_i, at which query q_i locates. Note that this equation only gives the service error of one query. Averaging as many practical queries as possible reveals actual error of the proposed interface.

V. VLBS With cPIR

Since the interface has been established, we can elaborate an entire retrieving process in cPIR-protected VLBS. We take regular partition in Fig. 2 as an instance to facilitate a reader's understanding.

As the size of returned data w is unclear, we regard this instance as a 1-bit situation. The PIR matrix scale is set as 7×8 since $|\mathbb{R}| = 52$. Afterwards, the NPOI set of each RRS $\mathbb{N}(r_i)$, i.e., the value on each element $M_{a,b}$, can be determined and stored based on the algorithm in Section IV, for example, $\mathbb{N}(r_1^l) = M_{1,1} = \{p_1, p_2, p_4, p_5\}$ and $\mathbb{N}(r_2^l) = M_{2,1} = \{p_1, p_2, p_4\}$. If users propose queries on r_{10}^l, their clients transmit query messages containing a large composite number S and a set $Y = \{y_1, y_2, \ldots, y_7\}$, in which y_3 is a quadratic non-residue of S. Then the server calculates and returns a set $Z = \{z_1, z_2, \ldots, z_8\}$. The clients can decode the value on $M_{3,3}$ by determining whether z_3 belongs to the quadratic residue set or not. Assuming that a POI can be expressed by k_p bits data, transmitting all information stored in $\mathbb{N}(n_{10}^l)$ asks for $6 \cdot k_p$ PIR matrices.

Although everything seems to go well, one practical problem may incur the failure of all this privacy-protecting structure. As shown in Fig. 3, NPOI count differs in different RRSs, for example, r_1^l has 4 NPOI while r_1^r has 2 only. Then querying on r_1^l requires computation and transmission on $4k_p$ PIR matrices, while only $2k_p$ ones are essential for r_1^r. If this assumption comes true, the server can get some knowledge about querying locations from detecting operation times and transmission scales. Within these NPOI sets listed in Fig. 3, a user querying on r_1^r can be easily located in the

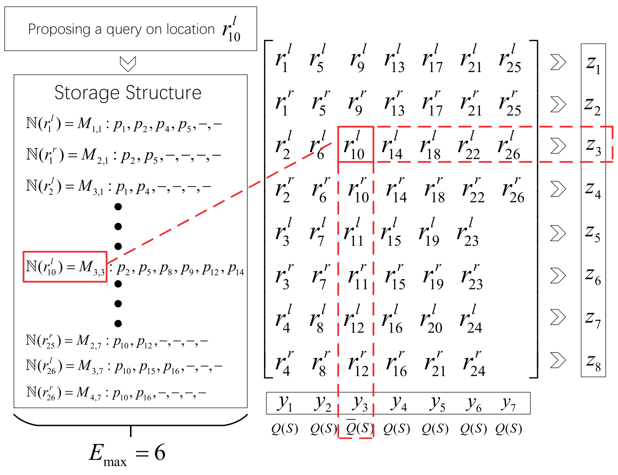

Proposing a query on location r_{10}^l
Storage Structure
$\mathbb{N}(r_1^l) = M_{1,1} : p_1, p_2, p_4, p_5, -, -$
$\mathbb{N}(r_1^r) = M_{2,1} : p_2, p_5, -, -, -, -$
$\mathbb{N}(r_2^l) = M_{3,1} : p_1, p_4, -, -, -, -$
$\mathbb{N}(r_{10}^l) = M_{3,3} : p_2, p_5, p_8, p_9, p_{12}, p_{14}$
$\mathbb{N}(r_{25}^r) = M_{2,7} : p_{10}, p_{12}, -, -, -, -$
$\mathbb{N}(r_{26}^l) = M_{3,7} : p_{10}, p_{15}, p_{16}, -, -, -$
$\mathbb{N}(r_{26}^r) = M_{4,7} : p_{10}, p_{16}, -, -, -, -$
$E_{\max} = 6$
y_1 y_2 y_3 y_4 y_5 y_6 y_7
$Q(S)$ $Q(S)$ $\overline{Q}(S)$ $Q(S)$ $Q(S)$ $Q(S)$ $Q(S)$
z_1 z_2 z_3 z_4 z_5 z_6 z_7 z_8

Fig. 3. Road-based PIR process.

range $\{r_1^r, r_{25}^r, r_{26}^r\}$, which indeed causes privacy leak. Thus, no matter which location is queried, its PIR matrix count should be subject to the largest requirement among all road segments, i.e., $6k_p$ PIR matrices are necessary in our example. More formally, assuming that the NPOI count of r_i is E_i (i.e., $|\mathbb{N}(q_i)|$), the maximum NPOI count among all RRSs is named as matrix restriction, denoted by E_{max}. To protect privacy, some dummy data should be padded into those NPOI sets whose NPOI count is less than E_{max}, until achieving E_{max}. These dummy data are denoted by '$-$' in Fig. 3. Moreover, the PIR matrix count w is determined as $w = E_{max} \cdot k_p$.

To measure this extra cost, we define a redundancy rate, γ. Note that except the POI distribution, query frequencies on road segments also affect γ. For example, if all queries occur on the road segment with the largest NPOI count, then no transmission is a waste even if NPOI count on each segment is extremely uneven. Thus, the redundancy rate is:

$$\gamma = 1 - \frac{\sum_i E_i \cdot F_i}{E_{max} \cdot \sum_i F_i}, \tag{12}$$

in which F_i denotes the query time on the ith road segment.

VI. Experiments

In this section, we experimentally validate the improvement of PIR-protected LBS brought by utilizing our interface. We first introduce the utilized datasets, including road networks and trajectories, and the preprocessing which constructs a PIR tensor. Then we evaluate the proposed interface on four criteria, service error, redundancy rate, transmission cost and server time. At last we compare our interface with existing methods to validate our contribution on improving PIR-protected LBS. We also give a further discussion on how users select privacy-protection methods in practice.

A. Datasets and Preprocessing

In this part we elaborate the utilized datasets and data preprocessing. A part of Shenzhen's road network downloaded from OpenStreetMap is adopted as the experimental base, in which each road is identified by several datum nodes with their longitudes and latitudes. After eliminating some

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

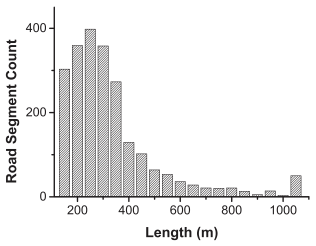


Fig. 4. The length distribution of all NRSs.

unsatisfactory roads and transforming all roads into two-way roads, we get a total of 1539 crossings and 2430 NRSs, whose length distribution is depicted in Fig 4. To avoid the boundary error originating from selecting sub-maps, we pad the boundary with mirror road networks through JOSM as these mirrors' driving time is computable in trajectory-aware conditions. POI are also extracted from OpenStreetMap with the aid of Google map and Baidu map. Five kinds of venues, including convenient stores, hospitals, restaurants, cinemas and residential areas, constitute the POI set whose number is 5936. Considering our computing power, we transmit 13-bits POI identifiers to represent the 64-bits POI coordinates.

The dataset prepared for trajectory-aware condition contains travelling trajectories of about 10000 vehicles in 8 days. Each trajectory is composed of a set of sampling nodes, which are labeled by vehicle ID, time stamps and locations (longitudes and latitudes). Unfortunately, their locations might be inaccurate, sometimes even outside of roads, stemming from the error of GPS equipment. Thus, we should match drivers' locations to a road network. Herein, we adopt the method proposed by Greenfeld [53] which takes full advantage of spatial-temporal relativity in trajectory records. Afterwards, we can easily obtain the travelling time of a vehicle on a road segment through time stamps.

Next, we discuss the distance function $f_d(\cdot)$. Calculating the distance between any two points is to obtain the length of a connected path, composed of several roads. As mentioned above, each road is identified by a series of datum nodes, and these segments between two adjacent datum nodes are considered straight. The length of a straight segment can be easily calculated by the Euclidean distance equation, based on the latitudes and longitudes of its endpoints. Thus $f_d(\cdot)$ is expressed as a length summation of several straight segments.

B. Evaluation

In this part we aim at evaluating how cPIR performs in VLBS with our interface. In general, we protect VLBS through cPIR at the price of inaccurate service, and extra transmission/computation. Thus the evaluation criteria consist of two categories, accuracy loss and resource consumption. A service error is adopted to measure the accuracy loss due to replacing users' precise locations with approximate ones. On the resource axis, three sub-criteria, transmission cost, server time and redundancy rate, are taken into consideration. The transmission cost and server time are the most direct indices to evaluate user experience on VLBS, especially the impact of integrating cPIR, which reveal practical transmission and computation complexity. Actually, their values depend on three factors, the original cost of answering a query, the extra cost brought by integrating a cPIR structure into the system, and the redundant cost originating from padding dummy data. The former two factors highly rely on actual scenes, while the redundant cost varies with different partition schemes. Thus, we introduce a redundancy rate to evaluate it. The following experimental results are all based on 5000 times queries originating at those places randomly selected from the trajectory data on a daytime period. Besides, as the overall redundancy rate trend in our experiments is a little disorderly and it varies a lot among different road networks and POI distribution, the following redundancy rates are calculated from some manually selected points which can reveal road networks' property in our view.

We first conduct experiments to show the influence of modulus size k. As these contents are only complementary experiments of our interface, their detailed descriptions are put in Appendix B. According to the results, we adopt $k = 256$ in the following experiments considering our computing power.

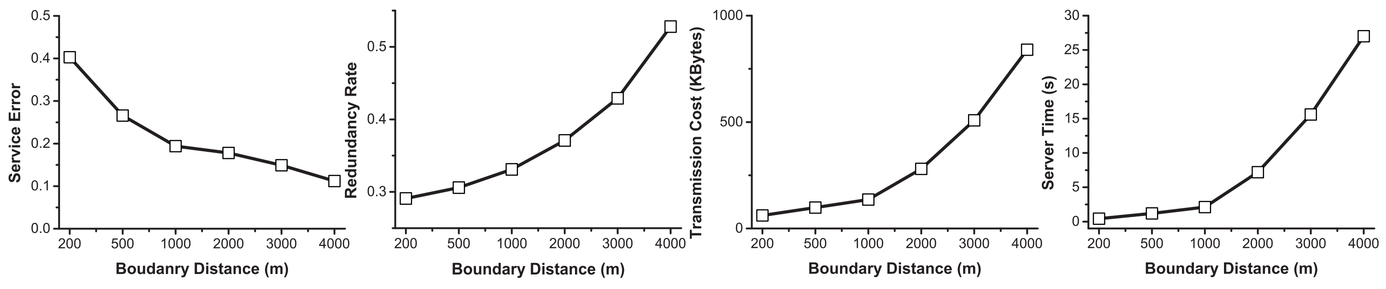
Next we carry out a series of control experiments to evaluate the practicability of cPIR and the proposed interface in VLBS. This part is to find out the optimal partition methods in our interface, and then we can use it to compare with existing methods. Since existing methods cannot adopt driving time as determining criterion, herein we only demonstrate the experiments on non-trajectory condition. Please see the experiments and results about trajectory-aware condition in Appendix C.

For a non-trajectory condition, we employ 6 grades of boundary distance, i.e., $200, 500, 1000, 2000, 3000$ and 4000 meters. Moreover, we prepare 6 values, i.e., $10, 50, 100, 200, 400$ and 600 meters, for partitioning length L in regular partition. Then the results on natural partition are illustrated in Fig. 5 and those on regular partition in Fig. 6. Based on them, we can conclude that:

(i) Service error decreases as boundary distance or time grows, while the other three criteria increase. Actually, the trends of transmission cost and server time are intelligible. Thus, we aim at explaining how the curves of service error and redundancy rate shape. Since direct comprehension on these statistical diagrams is rather difficult, we seek answers from detailed results, assisted by the graphic tools of OpenStreetMap. Regarding the service error, it is reasonable to infer that its downtrend originates from the connectivity of a road network. Generally, a location can be reached through multiple pathes, which is also embodied in the constructing process of path trees. Then it is common that the error area of a path is covered by the legitimate area of another path, thereby this error is eliminated. This kind of events mostly occurs in circuit pathes, such as a two-way road allowing u-turn, and a ring road of a block. Along with the rise of boundary distance or time, the ratio of circuit pathes in a tree significantly increases such that the service error drops down. On the redundancy rate axis, its uptrend probably stems from the length difference of NRSs/RRSs. In the above descriptions, we pin the redundancy problem on uneven POI distributions. Actually, another factor, that NRSs/RRSs differ in the covered region size of their

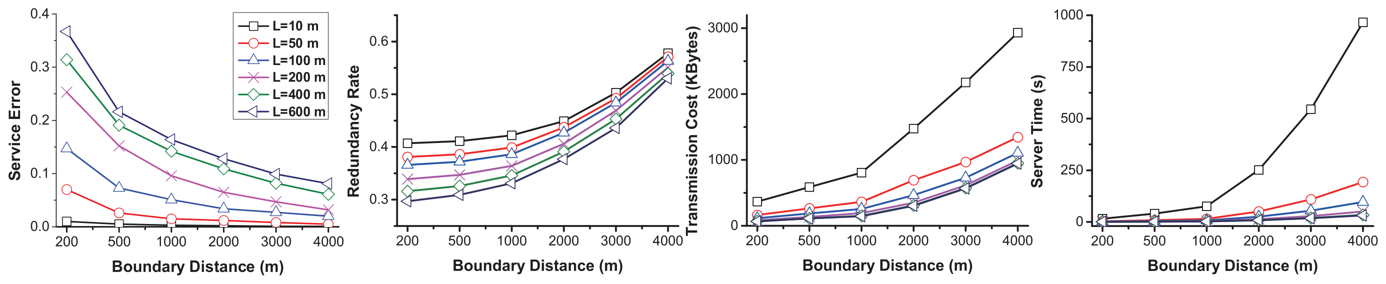
Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



(a)

Fig. 5. The experimental results on a natural partition scheme in non-trajectory condition.



(a)

Fig. 6. The experimental results on regular partition schemes in non-trajectory condition.

nearby road sets, also plays an important role. We explain it with the aid of path trees, as a nearby road set of an NRS/RRS is equal to its path tree. We have stated that a road segment may have several child nodes when meeting a crossing in a path tree. Each child node indicates a new sub-path and the sub-path count tends to have exponential correlation with the tree layer count. Thus, for a given boundary distance or time, shorter length of NRSs/RRSs means more layers, more available sub-pathes, as well as larger regions covered by path trees. A larger covered region is very likely to bring more NPOI. This phenomenon aggravates the redundancy problem and indicates that users in road-sparse areas are likely to suffer large redundancy.

(ii) When taking the same boundary distance, regular partition achieves lower service error than natural partition, but suffers larger transmission cost and server time, no matter what the value of regular length L is. As RRS count is always larger than NRS count for any L according to Section IV-A, it is easy to embrace that regular partition is a high-accuracy but large-cost scheme.

(iii) These experimental results validate that cPIR can be utilized to protect VLBS in practice with a regular partition scheme taking regular length from 200 to 400 m. Intuitively, the practicality asks for lower service error, and less transmission cost and server time. Yet the accuracy and resource requirements are mutually exclusive. Thus we have to make a compromise. Referring to Fig. 6, for a given boundary distance or time, transmission cost and server time grow explosively when L shrinks from 200 m, especially when $L < 100$ m. On contrary, drops are quite inconspicuous in the other direction, even if L increases twice. It signifies that enlarging L when $L > 200$ m cannot achieve significant effects on decreasing transmission and computation cost. The absolute values of transmission cost and server time are also affordable when $L \in [200, 400]$, which are about $309 - 350$ KBytes and $8.8-13.0$ s with $D = 2000$ m. On the other hand, service error in this zone is also sustainable, which is about $0.065 - 0.109$ with $D = 2000$ m. Thus, we define $[200, 400]$ as the optimal length zone of our experiments, which is also recommended for practical use. Moreover, when D takes large values (e.g., 3000 and 4000 meters), the returned NPOI are too many, e.g., thousands of POI, for users. Then users tend to pay more attention on the accuracy of those closer NPOI. Thus we make minor adjustments when calculating service error. We partition D into several levels (e.g, $D_1 = 1000$, $D_2 = 2000$ and $D_3 = D = 3000$) and calculate service error of each level. Then we assign each level a weight and add their weighted service error up as the total one. The weight has positive correlation with POI count, and negative correlation with D_i. In this method, the service error's values are larger than original ones.

C. Comparison With Existing Studies

We aim to improve PIR-protected LBS through decreasing service error. Thus, in this part, we give service error comparison between our interface and existing ones at the same level of resource cost. As computation and transmission cost have positive correlation, we only select the former as a controlled variable in our experiments.

We first implement the interface in [19] on our datasets as a control group, where the locations are partitioned by grids and nearby POI are defined as those with less Euclidean distance. Then we select appropriate partition and calculation

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

methods in our interface for comparison. As the distance in the control group cannot be measured by driving time, we select the non-trajectory condition in which distance is the metric of generating NPOI. Besides, we select the regular partition method since we have validated its effectiveness. According to Section VI-B, for a non-trajectory condition and regular partition scheme, the optimal length zone is 200m to 400m. Thus, in experiments, we calculate the transmission cost of our interface when $L \in [200, 400]$, and then adjust grid granularity in the control group to achieve the same transmission cost. After that, we can compare the service error of both methods.

Next we accomplish two control experiments. As mentioned before, presenting an interface based on road network knowledge is motivated by two facts, i.e., users and POI are usually located at roads, and users care more about driving distance. Thus, we conduct two experiments to demonstrate their benefits. First, we validate that our interface gains benefits from considering POI and user distribution. To this end, we make a modification on the definition of NPOI, that whether a POI belongs to NPOI depends on its Euclidean distance to a user. In this assumption, the metric of generating both precise and approximate NPOI is Euclidean distance, and the value of Euclidean boundary distance should be adjusted to return similar POI count to that using the prior-defined boundary (driving) distance. Note that a grid's NPOI set is composed of the NPOI of its midpoint. And the PIR matrix stores two same NPOI sets for every grid to simulate the characteristics of two-way road segments. The result is depicted in Fig. 7. The curves are smoothed as adjusting the settings to make the transmission cost of both partition methods identical is quite difficult. This result preliminarily proves the superiority of using road network knowledge. We give another intuitive experiment in Appendix D to further prove this. However, the result may reverse when roads are dense in the selected map. Thus Appendix D also includes an explanation for this.

Then we take driving distance into consideration. The precise NPOI are defined and calculated by driving distance, and the complete version of our interface can be implemented. On contrary, the control group still uses Euclidean distance to calculate NPOI. Then the comparison of service error is given in Fig. 8. It decreases 67.1% when $D = 2000$, and 66.2% when $D = 1000$. This result further shows that our interface can satisfy users' requirements better. In closing, our interface brings significant improvement to PIR-protected LBS by utilizing road network knowledge.

D. Further Discussion

Now we make a further discussion on protecting the privacy in LBS. Although the above experiments have validated that our interface can provide provable protection for VLBS with acceptable service loss, and performs better than those existing methods, users can still question that the transmission/computation cost is too large for practical applications, especially in those scenarios asking for low-latency. Actually, there exist two branches of methods in protecting LBS, i.e., cryptography-based methods containing PIR, and probability-based ones. The former usually provides

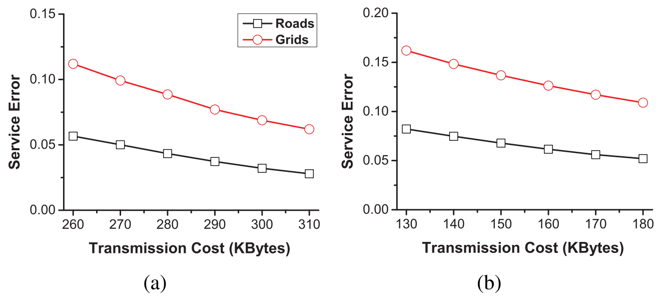


Fig. 7. Service error under Euclidean distance when our method and traditional one cost the same resources: (a) $D = 2000$ and (b) 1000.

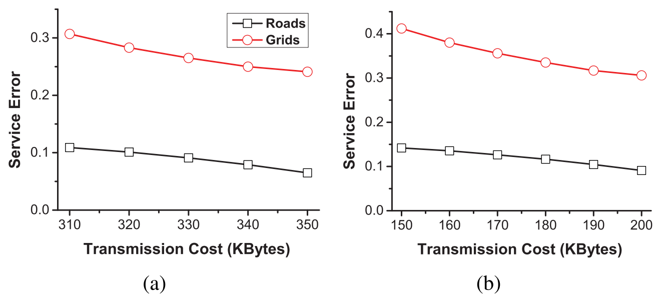


Fig. 8. Service error under driving distance when our method and traditional one cost the same resources: (a) $D = 2000$ and (b) 1000.

strong privacy guarantee but costs large extra resources, while the latter has lower privacy level but causes no extra cost. Herein we aim to reveal the difference of their performance in practical applications. Note that no matter which method is adopted, the service accuracy should be guaranteed, i.e., the service error should be kept at the low-level. Then we compare their transmission/computation cost and privacy level at the same level of service error.

As K-anonymity and differential privacy are the two most popular probability-based methods, we select two novel and highly cited schemes derived from them, i.e., XStar [54] and PIM [55] as our baselines. The results show that our method performs several hundreds times better than probability-based ones in terms of privacy protection, but costs one hundred times more computation and transmission resources. Thus, if tending to protect users' privacy, practical LBS can provide two modes for users to select, i.e., strong privacy, and low latency. When users have high demands for realtime response, e.g., searching nearby exits of freeways, they can select the low latency mode. When they demand high privacy, e.g., searching nearby hospitals, they can select the strong privacy mode.

VII. Conclusion and Future Work

In this paper, we optimize the PIR-protected LBS through proposing a novel interface based on road network knowledge. Two partition schemes and a path tree construction method constitute the proposed interface. Considering actual scenes, we also provide two application conditions differing in background knowledge. The experimental results validate that the service error of PIR-protected LBS decreases significantly

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

through adopting our interface. This also proves the effectiveness and practicality of the proposed interface on LBS.

An immediate next step in the line of this work is to explore how to generate the optimal length zone adaptively. It is a totally experimental parameter. It is also a global one as we extract the zone from the average results of all users' experimental data. However, in the view of one query, its value differs in different querying locations. According to our experiments, to provide a better tradeoff between service error and resource cost, smaller regular length is more effective in downtown areas, in which both roads and POI are more dense. This circumstance reveals that the optimal length zone of a specific location highly depends on the distribution of its surrounding roads and POI. Moreover, some other features, such as congestion conditions and time, may also have impacts on it. Thus, we intend to adopt machine learning [56] in training local optimal length zone for every road segment or area. The potential difficulties mainly contain feature engineering [57] on road and trajectory data, and selecting proper models and objective functions to achieve a desired tradeoff.

References

[1] M. Ghahramani, M. Zhou, and C. T. Hon, "Mobile phone data analysis: A spatial exploration toward hotspot detection," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 1, pp. 351–362, Jan. 2019.

[2] M. Ghahramani, M. Zhou, and C. T. Hon, "Extracting significant mobile phone interaction patterns based on community structures," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 3, pp. 1031–1041, Mar. 2019.

[3] S. Gisdakis, V. Manolopoulos, S. Tao, A. Rusu, and P. Papadimitratos, "Secure and privacy-preserving smartphone-based traffic information systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 3, pp. 1428–1438, Jun. 2015.

[4] M. Frustaci, P. Pace, G. Aloi, and G. Fortino, "Evaluating critical security issues of the IoT world: Present and future challenges," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2483–2495, Aug. 2018.

[5] N. Wu, Z. Li, K. Barkaoui, X. Li, T. Murata, and M. Zhou, "IoT-based smart and complex systems: A guest editorial report," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 1, pp. 69–73, Jan. 2018.

[6] P. Zhang, M. Zhou, and G. Fortino, "Security and trust issues in fog computing: A survey," *Future Gener. Comput. Syst.*, vol. 88, pp. 16–27, Nov. 2018.

[7] M. Handte, S. Foell, S. Wagner, G. Kortuem, and P. J. Marron, "An Internet-of-Things enabled connected navigation system for urban bus riders," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 735–744, Oct. 2016.

[8] F. Kelly, "Road pricing: Addressing congestion, pollution and the financing of Britain's roads," *Ignenia*, vol. 29, no. 1, pp. 34–40, 2006.

[9] P. F. Riley, "The tolls of privacy: An underestimated roadblock for electronic toll collection usage," *Comput. Law Secur. Rev.*, vol. 24, no. 6, pp. 521–528, Jan. 2008.

[10] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proc. 1st Int. Conf. Mobile Syst., Appl. Services (MobiSys)*, 2003, pp. 31–42.

[11] L. Sweeney, "K-anonymity: A model for protecting privacy," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, Oct. 2002.

[12] S. E. Fienberg and J. McIntyre, "Data swapping: Variations on a theme by Dalenius and Reiss," *J. Off. Statist.*, vol. 21, no. 2, p. 309, 2005.

[13] C. C. Aggarwal and S. Y. Philip, "A condensation approach to privacy preserving data mining," in *Proc. Int. Conf. Extending Database Technol.* Berlin, Germany: Springer, 2004, pp. 183–199.

[14] Y. Wang, M. Gu, J. Ma, and Q. Jin, "DNN-DP: Differential privacy enabled deep neural network learning framework for sensitive crowdsourcing data," *IEEE Trans. Computat. Social Syst.*, vol. 7, no. 1, pp. 215–224, Feb. 2020.

[15] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Proc. IEEE 36th Annu. Found. Comput. Sci.*, Oct. 1995, pp. 41–50.

[16] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *J. ACM*, vol. 45, no. 6, pp. 965–981, 1998.

[17] C. Cachin, S. Micali, and M. Stadler, "Computationally private information retrieval with polylogarithmic communication," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1999, pp. 402–414.

[18] P. Indyk and D. Woodruff, "Polylogarithmic private approximations and efficient matching," in *Proc. Theory Cryptogr. Conf.* Berlin, Germany: Springer, 2006, pp. 245–264.

[19] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location based services: Anonymizers are not necessary," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2008, pp. 121–132.

[20] S. Papadopoulos, S. Bakiras, and D. Papadias, "Nearest neighbor search with strong location privacy," *Proc. VLDB Endowment*, vol. 3, nos. 1–2, pp. 619–629, Sep. 2010.

[21] S. Wang, D. Agrawal, and A. El Abbadi, "Generalizing PIR for practical private retrieval of public data," in *Proc. IFIP Annu. Conf. Data Appl. Secur. Privacy*. Berlin, Germany: Springer, 2010, pp. 1–16.

[22] Z. Ning *et al.*, "A cooperative quality-aware service access system for social Internet of vehicles," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2506–2517, Aug. 2018.

[23] X. Cheng, L. Fang, and L. Yang, "Mobile big data based network intelligence," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4365–4379, Dec. 2018.

[24] Z. Ding, X. Li, C. Jiang, and M. Zhou, "Objectives and state-of-the-art of location-based social network recommender systems," *ACM Comput. Surv.*, vol. 51, no. 1, pp. 1–28, Apr. 2018.

[25] J. Cheng, J. Cheng, M. Zhou, F. Liu, S. Gao, and C. Liu, "Routing in Internet of vehicles: A review," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 5, pp. 2339–2352, Oct. 2015.

[26] X. Cheng, C. Chen, W. Zhang, and Y. Yang, "5G-enabled cooperative intelligent vehicular (5GenCIV) framework: When benz meets marconi," *IEEE Intell. Syst.*, vol. 32, no. 3, pp. 53–59, May 2017.

[27] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: Single database, computationally-private information retrieval," in *Proc. 38th Annu. Symp. Found. Comput. Sci.*, 1997, p. 364.

[28] M. Naveed, S. Kamara, and C. V. Wright, "Inference attacks on property-preserving encrypted databases," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*. New York, NY, USA: ACM, 2015, pp. 644–655.

[29] B. Gedik and L. Liu, "Location privacy in mobile systems: A personalized anonymization model," in *Proc. 25th IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2005, pp. 620–629.

[30] B. Hoh *et al.*, "Virtual trip lines for distributed privacy-preserving traffic monitoring," in *Proc. 6th Int. Conf. Mobile Syst., Appl., Services (MobiSys)*, 2008, pp. 15–28.

[31] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Preserving privacy in GPS traces via uncertainty-aware path cloaking," in *Proc. 14th ACM Conf. Comput. Commun. Secur. (CCS)*, 2007, pp. 161–171.

[32] M. Götz, S. Nath, and J. Gehrke, "MaskIt: Privately releasing user context streams for personalized mobile applications," in *Proc. Int. Conf. Manage. Data (SIGMOD)*, 2012, pp. 289–300.

[33] C.-Y. Chow and M. F. Mokbel, "Enabling private continuous queries for revealed user locations," in *Advances in Spatial and Temporal Databases*. Berlin, Germany: Springer, 2007, pp. 258–275.

[34] A. Khoshgozaran and C. Shahabi, "Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy," in *Advances in Spatial and Temporal Databases*. Berlin, Germany: Springer, 2007, pp. 239–257.

[35] G. Zhong, I. Goldberg, and U. Hengartner, "Louis, lester and pierre: Three protocols for location privacy," in *Privacy Enhancing Technologies*. Berlin, Germany: Springer, 2007, pp. 62–76.

[36] S. Zhong, L. Li, Y. G. Liu, and Y. R. Yang, "Privacy-preserving location-based services for mobile users in wireless networks," Dept. Comput. Sci., Yale Univ., New Haven, CT, USA, Tech. Rep. ALEU/DCS/TR-1297, 2004.

[37] R. A. Popa, H. Balakrishnan, and A. J. Blumberg, "VPriv: Protecting privacy in location-based vehicular services," in *Proc. USENIX Secur. Symp.*, 2009, pp. 335–350.

[38] L.-Y. Yeh and Y.-C. Lin, "A proxy-based authentication and billing scheme with incentive-aware multihop forwarding for vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 4, pp. 1607–1621, Aug. 2014.

[39] R. Lu, X. Lin, X. Liang, and X. Shen, "A dynamic privacy-preserving key management scheme for location-based services in VANETs," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 1, pp. 127–139, Mar. 2012.

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[40] M. L. Yiu, C. S. Jensen, X. Huang, and H. Lu, "SpaceTwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services," in *Proc. IEEE 24th Int. Conf. Data Eng.*, Apr. 2008, pp. 366–375.

[41] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," 2012, *arXiv:1212.1984*. [Online]. Available: http://arxiv.org/abs/1212.1984

[42] A. Khoshgozaran, C. Shahabi, and H. Shirani-Mehr, "Location privacy: Going beyond K-anonymity, cloaking and anonymizers," *Knowl. Inf. Syst.*, vol. 26, no. 3, pp. 435–465, Mar. 2011.

[43] A. Khoshgozaran, H. Shirani-Mehr, and C. Shahabi, "SPIRAL: A scalable private information retrieval approach to location privacy," in *Proc. 9th Int. Conf. Mobile Data Manage. Workshops (MDMW)*, Apr. 2008, pp. 55–62.

[44] S. Goldwasser and S. Micali, "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, Apr. 1984.

[45] C. Dwork, "Differential privacy," in *Encyclopedia of Cryptography and Security*, H. C. A. van Tilborg and S. Jajodia, Eds. Boston, MA, USA: Springer, 2011, pp. 338–340, doi: 10.1007/978-1-4419-5906-5_752.

[46] P. Amirian, A. Basiri, and J. Morley, "Predictive analytics for enhancing travel time estimation in navigation apps of apple, Google, and Microsoft," in *Proc. 9th ACM SIGSPATIAL Int. Workshop Comput. Transp. Sci. (IWCTS)*. New York, NY, USA: ACM, 2016, pp. 31–36.

[47] Google Developers. *Google Maps Distance Matrix API*. Accessed: Nov. 12, 2019. [Online]. Available: https://developers.google.com/maps/documentation/distance-matrix

[48] J. Rice and E. vanZwet, "A simple and effective method for predicting travel times on freeways," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 3, pp. 200–207, Sep. 2004.

[49] I. Sanaullah, M. Quddus, and M. Enoch, "Developing travel time estimation methods using sparse GPS data," *J. Intell. Transp. Syst.*, vol. 20, no. 6, pp. 532–544, Nov. 2016.

[50] W. Luo, H. Tan, L. Chen, and L. M. Ni, "Finding time period-based most frequent path in big trajectory data," in *Proc. Int. Conf. Manage. Data (SIGMOD)*. New York, NY, USA: ACM, 2013, pp. 713–724.

[51] M. Rahmani, E. Jenelius, and H. N. Koutsopoulos, "Route travel time estimation using low-frequency floating car data," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 2292–2297.

[52] Y. Wang, Y. Zheng, and Y. Xue, "Travel time estimation of a path using sparse trajectories," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2014, pp. 25–34.

[53] J. S. Greenfeld, "Matching GPS observations to locations on a digital map," in *Proc. 81st Annu. Meeting Transp. Res. Board*, 2002, vol. 1. no. 3, pp. 164–173.

[54] T. Wang and L. Liu, "Privacy-aware mobile services over road networks," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 1042–1053, Aug. 2009.

[55] Y. Xiao and L. Xiong, "Protecting locations with differential privacy under temporal correlations," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*. New York, NY, USA: ACM, 2015, pp. 1298–1309.

[56] S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, and J. Wang, "Dendritic neuron model with effective learning algorithms for classification, approximation and prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 2, pp. 601–614, Feb. 2019.

[57] H. Liu, M. Zhou, and Q. Liu, "An embedded feature selection method for imbalanced data classification," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 3, pp. 703–715, May 2019.

Zheng Tan received the B.S. degree from the Department of Computer Science, Tongji University, Shanghai, China, where he is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering. His research interests include network security and privacy and wireless security and privacy.

Cheng Wang (Senior Member, IEEE) received the M.S. degree from the Department of Applied Mathematics, Tongji University, in 2006, and the Ph.D. degree from the Department of Computer Science, Tongji University, in 2011. He is currently a Professor at the Department of Computer Science, Tongji University. His research interests include wireless communications and networking, wireless privacy, and intelligent transportation systems.

Chungang Yan received the Ph.D. degree from Tongji University, Shanghai, China, in 2006. She is currently a Professor with the Department of Computer Science and Technology, Tongji University, Shanghai. She has published more than 100 articles in domestic and international academic journals and conference proceedings. Her current research interests include concurrent model and algorithm, Petri net theory, formal verification of software, and trusty theory on software process.

MengChu Zhou (Fellow, IEEE) received the B.S. degree in control engineering from the Nanjing University of Science and Technology, Nanjing, China, in 1983, the M.S. degree in automatic control from the Beijing Institute of Technology, Beijing, China, in 1986, and the Ph.D. degree in computer and systems engineering from the Rensselaer Polytechnic Institute, Troy, NY, USA, in 1990. He joined the New Jersey Institute of Technology (NJIT), Newark, NJ, USA, in 1990, where he is currently a Distinguished Professor of electrical and computer engineering. His research interests are in Petri nets, intelligent automation, the Internet of Things, big data, web services, and intelligent transportation. He has over 800 publications including 12 books, 500 journal articles (over 400 in the IEEE TRANSACTIONS), and 29 book-chapters and holds 26 patents. He is a Life Member of Chinese Association for Science and Technology, USA, and served as its President, in 1999. He is a fellow of International Federation of Automatic Control (IFAC) and American Association for the Advancement of Science (AAAS). He was a recipient of the Humboldt Research Award for U.S. Senior Scientists from Humboldt Foundation, the Franklin V. Taylor Memorial Award, and the Norbert Wiener Award from the IEEE Systems, Man, and Cybernetics Society. He is the Founding Editor of the *IEEE Press Series on Systems Science and Engineering*.

Changjun Jiang received the Ph.D. degree from the Institute of Automation, Chinese Academy of Science, Beijing, China, in 1995. He is currently the leader of the Key Laboratory of the Ministry of Education for Embedded System and Service Computing, Tongji University, Shanghai, China. He is a fellow of IET and an Honorary Professor with Brunel University London. He has published more than 300 articles in journals and conference proceedings. He has led over 30 projects. His research interests include concurrency theory, Petri nets, formal verification of software, cluster, grid technology, intelligent transportation systems, and service-oriented computing. He was a recipient of the one international prize and seven prizes in the field of science and technology.

Authorized licensed use limited to: University of Exeter. Downloaded on July 16,2020 at 03:50:26 UTC from IEEE Xplore. Restrictions apply.