Manuscript Click here to access/download;Manuscript;WWWJ2.pdf

Click here to view linked References

Noname manuscript No.
(will be inserted by the editor)

Constructing Dummy Query Sequences to Protect Location Privacy and Query Privacy in Location-Based Services

Zongda Wu · Guiling Li · Shigen Shen · Xinze Lian · Enhong Chen · Guandong Xu

Received: date / Accepted: date

Abstract Location-based services (LBS) have become an important part of people's daily life. However, while providing great convenience for mobile users, LBS result in a serious problem on personal privacy, i.e., location privacy and query privacy. However, existing privacy methods for LBS generally take into consideration only location privacy or query privacy, without considering the problem of protecting both of them simultaneously. In this paper, we propose to construct a group of dummy query sequences, to cover up the query locations and query attributes of mobile users and thus protect users' privacy in LBS. First, we present a client-based framework for user privacy protection in LBS, which requires not only no change to the existing LBS algorithm on the server-side, but also no compromise

The work is supported by the Zhejiang Provincial Natural Science Foundation of China (Nos. LZ18F020001 and LY19F020018), the National Natural Science Foundation of China (Nos. 61762055, 61702468 and 61962029), the National Social Science Foundation of China (No. 19BTQ056) and Open Research Project of The Hubei Key Laboratory of Intelligent Geo-Information Processing (No. KLIGIP-2018B03).

Z. Wu
Department of Computer Science and Engineering, Shaoxing University, Shaoxing, Zhejiang, China.
E-mail: zongda1983@163.com

G. Li
School of Computer Science, China University of Geosciences, Wuhan, China.
Hubei Key Laboratory of Intelligent Geo-Information Processing, China University of Geosciences, Wuhan, China.
E-mail: freay@163.com

S. Shen
Department of Computer Science and Engineering, Shaoxing University, Shaoxing, Zhejiang, China.
E-mail: shigens@126.com

X. Lian (Corresponding Author)
Oujiang College, Wenzhou University, Wenzhou, Zhejiang, China.
E-mail: xinzelian@163.com

E. Chen
School of Computer Science and Technology, University of Science and Technology of China, Hefei, China.
E-mail: cheneh@ustc.edu.cn

G. Xu
Faculty of Engineering and IT, University of Technology Sydney, Australia.
E-mail: guandong.xu@uts.edu.au

to the accuracy of a LBS query. Second, based on the framework, we introduce a privacy model to formulate the constraints that ideal dummy query sequences should satisfy: (1) the similarity of feature distribution, which measures the effectiveness of the dummy query sequences to hide a true user query sequence; and (2) the exposure degree of user privacy, which measures the effectiveness of the dummy query sequences to cover up the location privacy and query privacy of a mobile user. Finally, we present an implementation algorithm to well meet the privacy model. Besides, both theoretical analysis and experimental evaluation demonstrate the effectiveness of our proposed approach, which show that the location privacy and attribute privacy behind LBS queries can be effectively protected by the dummy queries generated by our approach.

Keywords Location-based service · Location privacy · Query privacy · Privacy protection

1 Introduction

With advances in wireless communication and mobile positioning technologies, more and more devices have been equipped with GPS receivers, which makes location-based services (LBS) become increasingly popular. LBS refers to a variety of information services provided for mobile users based on the geographical location information supplied by the GPS receivers [1,2]. For example, mobile users can send a LBS query to the server to obtain the LBS result related to some point of interests (such as Hotel, Bar and Hospital). At present, LBS has become one of the most promising mobile services, and have achieved great success in the domains of society and business [2]. It has been reported that the revenue of LBS has reached an annual global total of more than ten billion dollars. However, while providing great convenience for users, LBS result in people's serious concerns on privacy [2, 3], specifically, including location privacy and query privacy. This is because for obtaining LBS, mobile users have to report not only their current geographical locations (i.e., query locations), but also the query content that they want to know (i.e., query attributes). Obviously, the information is private, based on which an attacker can easily infer not only users' trajectory (which belongs to the category of location privacy), but also users' sensitive preferences (e.g., sensitive point of interests, which belongs to the category of query privacy). It will result in a serious threat to the privacy of a mobile user if the private information is released to an untrusted third party (e.g., the LBS server). The problem of user privacy protection in LBS is causing people's increasingly extensive concerns, i.e., it is becoming an increasingly important problem how to protect the privacy of a user in LBS [3,4].

1.1 Motivations

A number of methods have been proposed to protect user privacy in LBS, including pseudonym methods, obfuscation methods, encryption methods and dummy methods. (1) In a pseudonym method, the user identification in a query is replaced with a temporal pseudonym, to disconnect the user identity from the query [5]. However, it is difficult for this kind of method to resist the threat from data mining, i.e., the user identity can be mined from the position information of a query [6]. Also, it cannot be applied to the system that requires identity authentication [2]. (2) The basic idea of obfuscation methods is to generalize (using a cloaking region [16]) or perturb (using noises [7]) the location information in a LBS query, to make it difficult for an attacker to identify the user precise location. However, since each query has been modified before being sent to the server, sometimes, this will result in a compromise to

the query accuracy [2]. Besides, the implementation of a pseudonym or obfuscation method is generally dependent on a third-party server, resulting in a bottleneck on efficiency and privacy [2,6]. (3) The basic idea of encryption methods is to encrypt each user query, so as to make it invisible to the untrusted server, and thus achieve the goal of privacy protection (such as the privacy protection based on private information retrieval) [8]. However, this kind of method generally requires the change to the existing LBS algorithm on the server, and the support of additional hardware and algorithms, thereby, decreasing its actual usability in practice. (4) In a dummy-based method, each user query is submitted together with a group of dummy queries to the server to make it difficult for the untrusted server to infer the location or attribute related to the user query [9,32]. However, the effectiveness of this kind of method depends on the quality of dummy query construction, i.e., it is easily threatened by inference attacks based on query feature distribution [2]. In addition, the existing methods generally consider only location privacy [9] or query privacy [32], without considering both as a whole (e.g., the semantic association between the location and the attribute from the same query), as a result, decreasing the quality of dummy query construction.

From the above, we conclude that an approach that can well protect users' privacy should meet the following requirements. (1) Ensuring the **privacy** behind each LBS query sequence. Specifically, it should be difficult for an attacker (regardless of the prior knowledge that the attacker has mastered) to infer user's exact locations from the query sequence (to protect location privacy), and user's sensitive attributes (to protect query privacy). (2) Ensuring the **accuracy** of each LBS query, i.e., the query result that a user obtains finally should be the same before and after the privacy protection is introduced. (3) Ensuring the **usability** of an existing LBS, i.e., the privacy protection should not require the change to the LBS algorithm on the server-side and the support of additional hardware, and it should not lead to a significant impact on the execution efficiency of a LBS query. Actually, for the requirements (2) and (3), the privacy protection approach is required to be transparent to both the mobile users of the client-side and the LBS algorithm of the server-side.

1.2 Contributions

This paper aims to propose an effective approach to simultaneously protect users' location privacy and query privacy in LBS, which should be able to address all the problems mentioned above, i.e., compared to existing methods, the main advantage of our approach is that under the constraint of not changing the LBS algorithm, it can not only ensure the accuracy and efficiency of each user query, but also prevent the untrusted server from identifying the user locations and sensitive attributes from the query sequences. Specifically, the contributions of this paper are threefold.

Firstly, based on a client-based architecture, we present a system framework of LBS privacy protection. In the framework, for each query issued by a mobile user, the client will construct a group of dummy queries and then submits them together with the user query to the server, making it difficult for the untrusted server to identify the user query. Next, the client will filter out the LBS results that correspond to the dummy queries, and only return the result corresponding to the user query to the user, as a result, ensuring the accuracy of the result that the user obtains finally.

Secondly, based on the system framework, we introduce a privacy model to formulate the requirements that ideal dummy query sequences should meet, i.e., which should not only have similar feature distributions with the user query sequence, but also be able to cover up the query privacy and location privacy behind the user query sequence. The feature

similarity makes it difficult for an attacker to identify the user query sequence from all the query sequences. The cover-up of the dummy query sequences to the query privacy and location privacy would reduce the exposure degree of user privacy on the untrusted server-side.

Finally, based on the above system framework and privacy model, we implement an algorithm that runs on a trusted client. The algorithm can well meet the requirements of user privacy protection in LBS, i.e., which can construct a group of dummy query sequences that well meet the privacy model. In addition, we have demonstrated the effectiveness of the privacy model and its implementation algorithm by theoretical analysis and experimental evaluation.

The rest of this paper is organized as follows. Section 2 briefly reviews related work. Section 3 presents a system framework for user privacy protection in LBS, as well as a related attack model. Section 4 formulates a privacy model for LBS, presents an implementation algorithm to meet the privacy model, and analyzes the effectiveness of the privacy model theoretically. Section 5 evaluates the privacy model and its implementation algorithm by experiments. Finally, we conclude this paper in Section 6.

2 Related Work

In this section, we briefly review and analyze some privacy protection methods related to LBS, specifically, including pseudonym methods (Section 2.1), obfuscation methods (Section 2.2), encryption methods (Section 2.3) and dummy-based methods (Section 2.4).

2.1 Pseudonym Methods

The basic idea of pseudonym methods is to replace the user identification in a query with a temporal pseudonym, to disconnect the user identity from the query [5]. A pseudonym method is generally based on a centralized architecture, i.e., using a trusted third-party server to publish, change and destroy the pseudonyms. However, as pointed out in [6], a pseudonym method does not change the location and attribute information contained in a user query, making it still likely for an attacker to infer the user identity from the query content itself, i.e., the user privacy in LBS cannot be protected well by pseudonyms. In [5], a mixing zone, which is a particular area where the pseudonyms of multiple users are changed centrally, and the users are not allowed to submit queries or receive information, is used to improve the effectiveness of pseudonyms, so as to make it more difficult to trace the users. In [11, 12], a mixing zone model with k-anonymity (where the number of users who change their pseudonyms simultaneously is not less than k) is proposed, which can improve the security of location privacy to a certain degree, due to considering the staying time of each mobile user in a mixing zone. In [13], the authors design a delay-tolerant mixing zone, where the time and location in a user query are replaced by a time interval and a location area, respectively, so as to increase the probability of successfully constructing a mixing zone. However, the service quality will be reduced, since users are not allowed to communicate with servers in a mixing zone. To this end, a multiple-mixing-zone model are proposed [14, 15], to obtain a good balance between privacy protection and service quality. In summary, for a pseudonym method, it is difficult to resist the threat from data mining (i.e., the user identity can be mined from the position information), and also difficult to be applied to the systems that require identity authentication. In addition, for a pseudonym method, the

k-anonymity [11,12] (i.e., the number of users in a mixing zone) is often used as the metrics to evaluate the degree of location privacy protection.

2.2 Obfuscation Methods

The basic idea of obfuscation methods is to generalize or perturb the location information in a LBS query, to make it difficult for an attacker to identify the user location. The location generalization refers to replacing the user location with a generalized location area (called a cloaking region), generally, which is constructed by a trusted third-party combined with k-anonymity [2]. Some early methods [16] cannot ensure a predetermined privacy level for continuous queries. Recent studies try to solve the issue. For example, in [17], the relevance between users' moving locations is used to construct cloaking regions; and in [18], a location generalization method is proposed to guard against the exposure of the destination of user's moving trajectory. Besides, the privacy demands of mobile users are dynamic and diverse, so in the construction of cloaking regions, we should consider users' personalized demands. In [19], the location privacy level can be adjusted self-adaptively within a certain range, to meet users' personalized demands of location privacy in continuous queries. In [20], a user-centered location service framework is proposed, so as to balance the privacy level and utility of a mobile user in advance. However, most of the obfuscation methods depend on a third-party server, as a result, reducing the practicability of the methods [6]. For this kind of methods, the k-anonymity (e.g., the number of locations in a cloaking region), the location entropy (e.g., the area of a cloaking region) [2] or the expected estimation error (e.g., the error between adversary estimation locations and user actual locations) [34] is often used as the metrics to evaluate location privacy. The location perturbation refers to intentionally adding some errors (or noises) into each query in a controllable fashion [7]. In order to provide a better privacy guarantee, in recent studies, the differential privacy model is used to control the quantity of errors being added into continuous queries, where the geo-indistinguishability model [21] and its derivative models [22,23] are the most representative.

2.3 Encryption Methods

The basic idea of encryption methods is to encrypt users' queries to make them invisible to the untrusted server-side, to achieve the goal of privacy protection. An encryption method generally will not reveal any user location information under the precondition of ensuring the usability of LBS, thereby, achieving stricter privacy protection. Specifically, this kind of techniques can be divided into two categories: privacy protection based on private information retrieval (PIR) methods and privacy protection based on cryptographic protocol. The PIR protocols were first used to safely access outsourced data on a network [24,25], which allow users to retrieve information from a database, under the precondition that the server does not know any request from the users. PIR can also be applied to LBS queries, but it can support limited query modes due to using some complex encryption operations [6]. To this end, PIR-based methods generally need to design the solution for a particular type of spatial queries. For example, a PIR-based solution is proposed in [24] for nearest neighbor queries; and a PIR-based protocol on top of trusted hardware is designed for k-nearest neighbor queries. In addition, in [8], a solution focusing on shortest path queries, which are different with traditional spatial queries, is proposed. However, the PIR protocol is generally developed based on some cryptographic operations with high complexity, so it can only

support limited LBS data access mode [26]. In a spatial switching method, an encryption technique (e.g., space filling curves) is used to translate the location and attribute information contained in a LBS query into the data in an encryption space, and then evaluate the query in the encryption space, making that only the user himself can remap the translated data into the original space [26,27,29]. For example, the location anonymity based on the Hilbert curve is presented in [29]; and a new space switching method is then proposed to solve the issues not considered by the Hilbert-based method. However, this kind of methods requires the change to the existing LBS algorithm on the server-side and the support of additional hardware and algorithms, consequently, decreasing the actual usability of the methods [2]. In addition, this kind of methods have no metrics on location privacy, which is dependent on the security of PIR protocols or translation functions.

2.4 Dummy-based Methods

In dummy-based methods, user queries are submitted together with dummy queries to the server, so as to make it difficult for the untrusted server to infer the true locations or attributes of mobile users [9,10]. A dummy method is generally developed on a client-based architecture, independent of a third-party server, resulting in a good usability. However, for a dummy method, it is important how to ensure the construction quality of dummy queries, because randomly constructed dummies generally cannot resist inference attacks based on data feature distributions [2]. To this end, many algorithms are proposed for dummy construction. In [30], TrackMeNot for the first time proposed to hide each user query among randomly constructed dummy queries, so as to protect the user query. However, the challenge in the mechanism, as the authors pointed out, is that the dummy queries can often be ruled out easily, because they are randomly constructed and meaningless. Based on the assumption that an attacker has mastered related side information and historical query sequences, the authors in [9] propose to construct a group of dummy locations that have similar query frequency feature with user locations, so as to protect user location privacy. In the method, a location entropy that is developed based on location frequencies is used as the location privacy metrics. In addition to location privacy, some researchers also attempt to leverage dummy queries to protect textual privacy. For example, aiming at text retrieval, in [31], the authors attempted to improve the quality of dummy queries based on a semantic space derived from Wikipedia. However, the work takes into account only the textual feature of a single user query, without considering the semantic relevance between users' current queries and users' historical queries, consequently, making it still possible for an attacker of rich prior knowledge to rule out the dummies. Then, the authors proposed a similar privacy protection approach for book search service in a digital library [32]. In summary, most of the existing methods did not fully consider the data distribution characteristics for users' queries, and also did not consider the semantic associations between query locations and query attributes (e.g., it is not appropriate to query subway stations in the countryside), as a result, leading to a negative impact on the quality of constructed dummy queries, and increasing the exposure risk of user privacy [2]. Although the solution in this paper also belongs to the scope of dummy-based methods, it considers location privacy and query privacy as a whole, and constructs dummy queries by fully considering the location features, attribute features and the association features between locations and attributes, as a result, improving the construction quality of dummy queries, and then the protection of user's location privacy and query privacy.

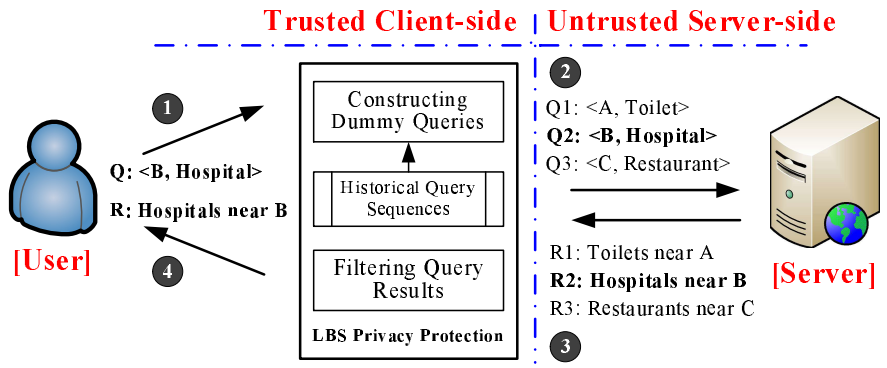


Fig. 1 The system framework used by our approach, where the historical query sequences include true user sequences and dummy sequences generated by the "constructing dummy queries" component.

3 Problem Statement

In this paper, we study an effective approach for the protection of user privacy in LBS, which can meet the following requirements: (1) ensuring the **privacy** of each user query (i.e., location privacy and query privacy); (2) ensuring the **accuracy** of each query result; and (3) ensuring the **usability** of LBS. Below, we present the system model used by our approach and then the attack model. In LBS, mobile users' privacy can be divided into location privacy and query privacy (or called attribute privacy). In Fig. 1, we show the system framework used by this paper for the protection of user privacy, as well as an example of how to protect user privacy (where A, B and C denote three locations). From Fig. 1, we see that the system model consists of an untrusted server-side and many trusted client-sides. The data processing of the system model can be briefly described as follows.

- **Step 1**. When each LBS query $q_0 = (l_0, u_0)$ (wherein, l_0 and u_0 denote a query location and a query attribute, respectively) is issued by a user, the "constructing dummy queries" component running on a client-side constructs a group of dummy LBS queries $q_1, q_2, ..., q_m$ for q_0, with the help of the historical query sequences, after taking into consideration the requirements of security and efficiency. Then, the dummy queries are submitted together with the user query to the server-side.
- **Step 2**. In the client-side, the "filtering query results" component finds out the result r_0, which corresponds to the user query q_0, from all the query results $r_0, r_1, r_2, ..., r_m$ that are returned by the LBS algorithm on the server. Then, the component returns r_0 to the user, while discarding the other query results $r_1, r_2, ..., r_m$.

Note that for a LBS query, timing information is also important (**when**, **where** and **what** query is sent to the server). In our model, the time associated to each dummy query q_i is set approximately equal to that of its corresponding user query q_0. From Fig. 1, we can see that the system framework can ensure the accuracy of each LBS result that a mobile user obtains finally, without the change to the existing LBS algorithm and the support of additional hardware. In the system framework, the privacy protection is transparent to both the LBS algorithm of the server-side and the mobile users of the client-side. Moreover, in the system framework, both location privacy and query privacy of mobile users have been taken into consideration, resulting in that users' privacy can be better protected.

From Fig. 1, we can also see that the generated dummy queries play an important role in the framework, i.e., the quality of them is the key to the LBS privacy protection, which

should be able to effectively mix up the true query locations and attributes of mobile users. However, randomly generated dummy queries are generally easy to be ruled out by an attacker who masters rich background knowledge, thus failed to protect user privacy. This is mainly caused by the following three reasons. (1) The location query sequence (or the attribute query sequence) from a mobile user has certain regularity. For example, the queries issued by the same user during a period of time often occur in some fixed location areas (e.g., near the house or company of the user), and they are often centered on some fixed attribute categories (e.g., a foodie user often like to query restaurants). In other words, the user query sequences generally show regular data feature distributions. (2) There exists some semantic association between the location and attribute from the same query (e.g., the query attribute categories that different locations can support to query are different). For example, the locations from the countryside generally cannot support to query subway stations near them. Thus, it is easy for an attacker to rule out dummy queries, based on the above two kinds of feature distributions. (3) It is also possible for the generated dummy queries themselves to reveal user privacy, e.g., the dummy locations should stay safe distances away from the user locations (to protect location privacy), and the dummy attributes should be not relevant to the sensitive categories (to protect query privacy). Otherwise, an attacker can know users' privacy directly, without ruling out the dummy queries.

From the above, we conclude that the dummy query sequences constructed by the privacy algorithm on the client-side for a user query sequence should meet the following requirements: (1) hiding the queries of a mobile user, i.e., having similar feature distributions (specifically, including location feature distributions, attribute feature distributions, and semantic relevance feature distributions between locations and attributes) with the user query sequence, so as to make it difficult for an attacker to rule out the dummy query sequences; and (2) covering up the location privacy and attribute privacy of a mobile user, i.e., the dummy queries should be not only semantically irrelevant to the sensitive attribute categories, but also located at safe distances away from the true query locations of a mobile user. In addition, in the system model, the dummy queries are constructed based on the user queries, which may potentially leak the user information to some extent (e.g., some features shown by the user queries), but we think that only the location or query privacy itself is sensitive and needs to be protected.

In the system framework of Fig. 1, the server-side is untrusted, which is considered as the biggest potential attacker [32], so we assume that an attacker has the following ability. (1) The attacker has obtained all the query sequences from the client-side (including the true query sequences submitted by mobile users and the dummy query sequences constructed by our approach), so he can guess the user query sequence by analyzing the feature distributions of location query sequences, the feature distributions of attribute query sequences, and the semantic associations between query locations and query attributes. (2) The attacker has mastered rich background knowledge, such as the global geographical information (including all the locations and their features) and the domain of query attributes. (3) The attacker might also know the existence of the privacy algorithm deployed on the client, and obtain a copy of the algorithm. However, the attacker should meet the following assumption. The probability $Pr(\mathcal{Q}^0|\mathcal{Q}^k, \mathcal{Q}^0)$ that a user query sequence $\mathcal{Q}^0$ can be distinguished from a dummy sequence $\mathcal{Q}^k$ by the attacker is reversely related to the feature similarity $sim(\mathcal{Q}^k, \mathcal{Q}^0)$ between them, i.e.,

$$Pr(\mathcal{Q}^0|\mathcal{Q}^k, \mathcal{Q}^0) \propto 1 - sim(\mathcal{Q}^k, \mathcal{Q}^0) \tag{1}$$

$$Pr(\mathcal{Q}^0|\mathcal{Q}^k, \mathcal{Q}^0) = 0 \leftarrow sim(\mathcal{Q}^k, \mathcal{Q}^0) \geq \theta \tag{2}$$

4 Proposed Approach

In this section, based on the system model and attack model above, we propose our approach to protect user privacy in LBS. First, based on the system model, we define a privacy model for user privacy protection called the (μ, ρ)-privacy model, which formulates the requirements that the dummy query sequences should satisfy to protect user privacy, i.e., the dummy query sequences should have similar feature distributions with the user query sequence, so as to hide the user queries; and they should be of suitable distances away from the user locations, and semantically irrelevant to the user sensitive attribute categories, so as to cover up the user location privacy and query privacy. Second, we present an implementation algorithm for the privacy model. Finally, we analyze the security of our approach.

4.1 Privacy Model

From the system model in Fig. 1, we see that when constructing dummy queries, we have to consider not only both location privacy and attribute privacy, but also the feature relevances between the current user query and the historical query sequences. Thus, a query sequence is an important data structure, which is a time-ordered sequence consisting of many queries, and can be denoted by $\mathcal{Q} = (q_i)_{i=1}^n$, where each query q_i consists of a query location l_i and a query attribute u_i, and can be further denoted by $q_i = (l_i, u_i)$. Hence, a query sequence $\mathcal{Q}$ can be represented as a location query sequence $\mathcal{L}$ and an attribute query sequence $\mathcal{U}$, denoted by $\mathcal{Q} = (\mathcal{L}, \mathcal{U})$. Below, to simplify the presentation, we use $\mathcal{Q}^0 = (q_i^0)_{i=1}^n$, $\mathcal{L}^0 = (l_i^0)_{i=1}^n$ and $\mathcal{U}^0 = (u_i^0)_{i=1}^n$ to denote the genuine query sequences from mobile users, and we use $\mathcal{Q}^k = (q_i^k)_{i=1}^n$, $\mathcal{L}^k = (l_i^k)_{i=1}^n$ and $\mathcal{U}^k = (u_i^k)_{i=1}^n$ $(k \geq 1)$ to denote the generated dummy query sequences. In Table 1, we describe key symbols used in this paper.

Table 1 Symbols and their meanings

Symbols	Meanings
$\mathcal{Q}^0 = (q_i^0)_{i=1}^n$	A user query sequence
$\mathcal{L}^0 = (l_i^0)_{i=1}^n$	A user location query sequence
$\mathcal{U}^0 = (u_i^0)_{i=1}^n$	A user attribute query sequence
$\mathcal{Q}^k = (q_i^k)_{i=1}^n$	A dummy query sequence
$\mathcal{L}^k = (l_i^k)_{i=1}^n$	A dummy location query sequence
$\mathcal{U}^k = (u_i^k)_{i=1}^n$	A dummy attribute query sequence
$\mathcal{Q} = (\mathcal{L}, \mathcal{U})$	A sequence $\mathcal{Q}$, and its location query sequence $\mathcal{L}$ and attribute query sequence $\mathcal{U}$
$sim^F(\mathcal{L}^1, \mathcal{L}^0)$	The location frequency similarity between $\mathcal{L}^1$ and $\mathcal{L}^0$
$sim^T(\mathcal{L}^1, \mathcal{L}^0)$	The location transfer similarity between $\mathcal{L}^1$ and $\mathcal{L}^0$
$sim^F(\mathcal{U}^1, \mathcal{U}^0)$	The attribute frequency similarity between $\mathcal{U}^1$ and $\mathcal{U}^0$
$sim^G(\mathcal{U}^1, \mathcal{U}^0)$	The category frequency similarity between $\mathcal{U}^1$ and $\mathcal{U}^0$
$sig(g, \mathbb{U})$	The significance of a category g related to sequences $\mathbb{U}$

First, we study the problem on location privacy protection. In a user location query sequence, the frequency of occurrence of each query location is distributed regularly, not randomly (e.g., the queries from the same user often occur in some fixed location areas), so

the frequency of occurrence of each query location is thought to be an important characteristic that reflects the move mode of a mobile user [9], which can be used to distinguish the true ones from the dummy locations. Therefore, we need to take into account the location frequency features, so as to construct the quality dummy query locations.

Definition 1 (Location Frequency) *The frequency of occurrence of a location l_i in a location query sequence $\mathcal{L}$ is defined as follows.*

$$Fr(l_i, \mathcal{L}) = |\{l \,|\, l \in \mathcal{L} \wedge l = l_i\}| \tag{3}$$

Below, we denote a subsequence consisting of the first m locations of the query sequence $\mathcal{L}$ as $\mathcal{L}_m = (l_i)_{i=1}^m$. Given a user location sequence $\mathcal{L}^0$ and a dummy location sequence $\mathcal{L}^1$, based on Definition 1, we can obtain the following two location frequency vectors: $Fr(\mathcal{L}^0) = (Fr(l_i^0, \mathcal{L}_i^0))_{i=1}^n$ and $Fr(\mathcal{L}^1) = (Fr(l_i^1, \mathcal{L}_i^1))_{i=1}^n$.

Definition 2 (Location Frequency Similarity) *The location frequency similarity between a dummy location query sequence $\mathcal{L}^1$ and a user location query sequence $\mathcal{L}^0$ can be measured by the generalized Jaccard similarity (denoted by EJ)[1] between their location frequency vectors $Fr(\mathcal{L}^1)$ and $Fr(\mathcal{L}^0)$, i.e.,*

$$sim^F(\mathcal{L}^1, \mathcal{L}^0) = EJ\left(Fr(\mathcal{L}^1), Fr(\mathcal{L}^0)\right) = \frac{Fr(\mathcal{L}^1) \cdot Fr(\mathcal{L}^0)}{\|Fr(\mathcal{L}^1)\|^2 + \|Fr(\mathcal{L}^0)\|^2 - Fr(\mathcal{L}^1) \cdot Fr(\mathcal{L}^0)} \tag{4}$$

Note that the cosine similarity[2] is a more popular measure of similarity between two non-zero vectors in a high-dimensional space, which is calculated by the cosine of the angle (i.e., the angle difference) between the vectors, and not sensitive to the element values of vectors. It may be suitable for measuring location frequency similarity. However, except the angle difference, we here should slightly consider the difference of element values between vectors, i.e., the similarity of two location query sequences is equal to 1 if and only if the angle difference and value difference between their vectors are both equal to 0. Thus, we choose the generalized Jaccard similarity in this paper.

In addition to location occurrence frequency, the transfer distance between two adjacent query locations is also an important characteristic that reflects the move mode of a mobile user. It is easy for an attacker to figure out the distance between any two locations, because he has mastered the global map information. Thus, we also need to take into account the distance transfer features to construct the dummy query locations.

Definition 3 (Location Transfer) *For two locations l_1 and l_2, the transfer distance between them can be measured by the geographical distance between them (below, we use $dist(l_1, l_2)$ to denote the distance between l_1 and l_2).*

$$Tr(l_1, l_2) = \begin{cases} 0, & l_2 \text{ is null} \\ dist(l_1, l_2), & \text{otherwise} \end{cases} \tag{5}$$

Given a user location sequence $\mathcal{L}^0$ and a dummy location sequence $\mathcal{L}^1$, based on Definition 3, we obtain the following two location transfer vectors: $Tr(\mathcal{L}^0) = (Tr(l_i^0, l_{i-1}^0))_{i=1}^n$ and $Tr(\mathcal{L}^1) = (Tr(l_i^1, l_{i-1}^1))_{i=1}^n$.

[1] https://en.wikipedia.org/wiki/Jaccard_index

[2] https://en.wikipedia.org/wiki/Cosine_similarity

Definition 4 (Location Transfer Similarity) *The location transfer similarity between a dummy location query sequence $\mathcal{L}^1$ and a user location query sequence $\mathcal{L}^0$ can be measured by the generalized Jaccard similarity between their corresponding location transfer vectors $Tr(\mathcal{L}^1)$ and $Tr(\mathcal{L}^0)$, i.e.,*

$$sim^T(\mathcal{L}^1, \mathcal{L}^0) = EJ\left(Tr(\mathcal{L}^1), Tr(\mathcal{L}^0)\right) \tag{6}$$

Now, we capture the most important two feature distributions of a location query sequence (Definitions 1 and 3). In this paper, we mainly take into account the two location feature distributions to construct the dummy locations that are highly similar to the user locations (Definitions 2 and 4), thereby, making it difficult for an attacker to rule out the dummy locations, i.e., ensuring that the user locations can be well hidden by the dummy locations. However, besides the location feature similarity, the dummy locations should also keep safe distances from the user locations (i.e., the dummy locations are harmful to the location privacy). Below, we define the privacy of user location.

Definition 5 (Location Privacy) *Given a privacy parameter $\mu \geq 1$, a user location query sequence $\mathcal{L}^0$ and a group of dummy location query sequences $\mathbb{L}$, if $|\mathbb{L}| \geq \mu$ and satisfy the following two requirements, then it is deemed that they can effectively ensure the μ-location privacy of $\mathcal{L}^0$.*

- ***Ensuring the location feature similarity***. *The location frequency feature and location transfer feature of the user location sequence $\mathcal{L}^0$ should be similar to those of each dummy location sequence $\mathcal{L}^k \in \mathbb{L}$, i.e.,*

$$\forall \mathcal{L}^k \in \mathbb{L} \rightarrow sim^F(\mathcal{L}^0, \mathcal{L}^k)\, sim^T(\mathcal{L}^0, \mathcal{L}^k) \geq \theta_1 \tag{7}$$

- ***Ensuring the user location security***. *Each location l_i^k in a dummy query sequence $\mathcal{L}^k$ should be of a safe distance away from the corresponding location l_i^0 in the user query sequence $\mathcal{L}^0$, i.e.,*

$$\forall \mathcal{L}^k \in \mathbb{L} \wedge \forall l_i^k \in \mathcal{L}^k \rightarrow dist(l_i^k, l_i^0) \geq \theta_2 d^* \tag{8}$$

In Equations 7 and 8, d^* denotes the farthest geographical distance, and θ_1 and θ_2 are two thresholds whose values are preset (in the experiment, we empirically set $\theta_1 = 0.1$ and $\theta_2 = 0.01$). Here, for a dummy location, if its distance from a user location is greater than $\theta_2 d^*$, then it has no impact on the location privacy; and for a dummy location sequence $\mathcal{L}^k$, if its similarity to a user sequence $\mathcal{L}^0$ is greater than θ_1, then it cannot be distinguished from the user sequence, i.e., $Pr(\mathcal{L}^k | \mathcal{L}^k, \mathcal{L}^0) = 0$.

Next, we study the problem of attribute privacy protection. Similarly, the attribute occurrence frequency is also an important characteristic of users' queries. For example, the queries from the same user during a period of time are often centered on some fixed or related attributes (e.g., travel enthusiasts like to query nearby attractions). More importantly, because the attacker has known the entire attribute query sequences, he/she can easily figure out the frequency value of occurrence of each attribute in a query sequence. To this end, we need to consider the attribute frequency feature to construct the quality dummy attributes.

Definition 6 (Attribute Frequency) *The frequency of occurrence of an attribute u_i in an attribute query sequence $\mathcal{U}$ is defined as follows.*

$$Fr(u_i, \mathcal{U}) = |\{u \,|\, u \in \mathcal{U} \wedge u = u_i\}| \tag{9}$$

A subsequence consisting of the first m attributes of the query sequence $\mathcal{U}$ can be denoted by $\mathcal{U}_m = (u_i)_{i=1}^m$. Given a user attribute sequence $\mathcal{U}^0$ and a dummy attribute sequence $\mathcal{U}^1$, based on Definition 6, we can obtain the following two attribute frequency vectors: $Fr(\mathcal{U}^0) = (Fr(u_i^0, \mathcal{U}_i^0))_{i=1}^n$ and $Fr(\mathcal{U}^1) = (Fr(u_i^1, \mathcal{U}_i^1))_{i=1}^n$.

Definition 7 (Attribute Frequency Similarity) *The attribute frequency similarity between a dummy attribute sequence $\mathcal{U}^1$ and a user attribute sequence $\mathcal{U}^0$ can be measured by the generalized Jaccard similarity between their attribute frequency vectors $Fr(\mathcal{U}^1)$ and $Fr(\mathcal{U}^0)$, i.e.,*

$$sim^F(\mathcal{U}^1, \mathcal{U}^0) = EJ\left(Fr(\mathcal{U}^1), Fr(\mathcal{U}^0)\right) \tag{10}$$

In fact, categories are more generalized concepts than attributes, e.g., "Home Inn" (a well-known hotel company) is an attribute, and "Express Hotel" and "Hotel" are categories, whose occurrence frequency values can also well reflect the query mode of a mobile user. It is easy for the attacker who masters the rich background knowledge to further figure out the frequency of occurrence of a category in a query sequence, based on the attribute frequency values. To this end, we also need to consider the category frequency feature to construct the dummy query attributes.

Definition 8 (Category Frequency) *A category indicates a set of attributes, which consists of all the attributes belonging to the category. The frequency of occurrence of a category g in an attribute query sequence $\mathcal{U}$ is defined as follows.*

$$Fr(g, \mathcal{U}) = \sum\nolimits_{u_i \in g} Fr(u_i, \mathcal{U}) \tag{11}$$

Let g_i represent the category, which an attribute u_i belongs to. Given a user attribute sequence $\mathcal{U}^0$ and a dummy attribute sequence $\mathcal{U}^1$, based on Definition 8, we obtain two category frequency vectors as: $Gr(\mathcal{U}^0) = (Fr(g_i^0, \mathcal{U}_i^0))_{i=1}^n$ and $Gr(\mathcal{U}^1) = (Fr(g_i^1, \mathcal{U}_i^1))_{i=1}^n$.

Definition 9 (Category Frequency Similarity) *The category frequency similarity between a dummy attribute sequence $\mathcal{U}^1$ and a user attribute sequence $\mathcal{U}^0$ can be measured by the generalized Jaccard similarity between their category frequency vectors $Gr(\mathcal{U}^1)$ and $Gr(\mathcal{U}^0)$, i.e.,*

$$sim^G(\mathcal{U}^1, \mathcal{U}^0) = EJ\left(Gr(\mathcal{U}^1), Gr(\mathcal{U}^0)\right) \tag{12}$$

Now, based on Definitions 8 and 10, we can construct dummy attribute query sequences highly similar to the user attribute query sequence, making it difficult the attacker to rule out the dummy attributes, and as a result ensuring that the user attributes can be hidden effectively. However, in addition to the attribute feature similarity, the dummy attributes should be able to effectively reduce the exposure degree of each user sensitive attribute on the untrusted server-side, so as to cover up the user attribute privacy. Different from the query locations, not all the query attributes are sensitive and need to be protected, and different users often have different sensitive attributes. Thus, we introduce a concept of sensitive attribute categories, to allow a user to assign in advance the sensitive attribute categories that need to be protected (i.e., all the attributes belonging to these categories are sensitive). Below, the user sensitive attribute categories are denoted by $\mathcal{G}^*$. Moreover, for the non-sensitive attributes in a user query sequence, we no longer construct the dummy query attributes for them. Below, we first define the significance of an attribute category, and then the privacy of user attribute.

Definition 10 (Category Significance) *The significance of an attribute category g related to a group of attribute sequences $\mathbb{U}$ is defined as follows.*

$$sig(g, \mathbb{U}) = \frac{\sum_{\mathcal{U} \in \mathbb{U}} Fr(g, \mathcal{U})}{\sum_{\mathcal{U} \in \mathbb{U}} \sum_{u_i \in \mathcal{U}} Fr(u_i, \mathcal{U})} \tag{13}$$

Definition 11 (Attribute Privacy) *Given a privacy parameter $\rho \geq 1$, a user attribute query sequence $\mathcal{U}^0$ and a group of dummy attribute query sequences $\mathbb{U}$, if $\mathbb{U}$ satisfy the following two requirements, then it is deemed that they can effectively ensure the ρ-attribute privacy of $\mathcal{U}^0$.*

- ***Ensuring the attribute feature similarity***. *The attribute frequency feature and category frequency feature of the user attribute sequence $\mathcal{U}^0$ should be similar to those of each dummy attribute sequence $\mathcal{U}^k \in \mathbb{U}$, i.e.,*

$$\forall \mathcal{U}^k \in \mathbb{U} \rightarrow sim^F(\mathcal{U}^k, \mathcal{U}^0)\, sim^G(\mathcal{U}^k, \mathcal{U}^0) \geq \theta_3 \tag{14}$$

- ***Ensuring the user attribute security***. *Based on the dummy attribute sequences $\mathbb{U}$, the significance of each sensitive category $g^* \in \mathcal{G}^*$ can be reduced effectively, i.e.,*

$$\forall g^* \in \mathcal{G}^* \rightarrow \frac{sig(g^*, \{\mathcal{U}^0\})}{sig(g^*, \{\mathcal{U}^0\} \cup \mathbb{U})} \geq \rho \tag{15}$$

In Equation 14, θ_3 is a threshold similar to θ_1, whose value is empirically set to 0.1 in the experiment. In Definitions 5 and 11, we formulate the requirements of user location privacy and user attribute privacy, respectively. However, a location and an attribute from the same user query are not independent of each other, actually, between which there exists some semantic association, i.e., for different locations, the attribute categories that they can support to query may be different to each other. For example, it is suitable to query nearby subway stations in Beijing, but it is not suitable in the countryside. Thus, such semantic association is an important characteristic to distinguish the dummy queries from the user queries. Based on Definitions 5 and 11, after considering the semantic associations between query locations and attributes, we further define the LBS privacy.

Definition 12 (LBS Privacy) *Given a user query sequence $\mathcal{Q}^0 = (\mathcal{L}^0, \mathcal{U}^0)$, and a group of dummy query sequences $\mathbb{Q}$, if there is a subset $\mathbb{Q}' = (\mathbb{L}', \mathbb{U}')$ of $\mathbb{Q}$, which satisfy the following three requirements, then it is deemed that the query sequences $\mathbb{Q}$ can ensure the (μ, ρ)-privacy of $\mathcal{Q}^0$.*

- ***Ensuring the μ-location privacy***. *The dummy location sequences $\mathbb{L}'$ corresponding to $\mathbb{Q}'$ can ensure the μ-location privacy of the user location sequence $\mathcal{L}^0$.*
- ***Ensuring the ρ-attribute privacy***. *The dummy attribute sequences $\mathbb{U}'$ corresponding to $\mathbb{Q}'$ can ensure the ρ-attribute privacy of the user attribute sequence $\mathcal{U}^0$.*
- ***Ensuring the semantic association between locations and attributes***. *For each query q_i^k (we denote it as $q_i^k = (l_i^k, u_i^k)$) belonging to any query sequence $\mathcal{Q}^k \in \mathbb{Q}'$, the category g_i^k of the attribute u_i^k should be well matched with the location l_i^k, i.e., $g_i^k \in \mathcal{G}(l_i^k)$, where $\mathcal{G}(l_i^k)$ denotes all the categories that the location l_i^k can support to query.*

Algorithm 1: Constructing a new dummy query for a non-empty query sequence

Input: (1) A new user query $q_i^0 = (l_i^0, u_i^0)$; (2) a non-empty user query sequence $\mathcal{Q}^0 = (\mathcal{L}^0, \mathcal{U}^0)$; (3) a dummy query sequence $\mathcal{Q}^k = (\mathcal{L}^k, \mathcal{U}^k)$; and (4) related thresholds (e.g., $\theta_1, \theta_2, \theta_3$).
Output: A new dummy query $q_i^k = (l_i^k, u_i^k)$ associated with the dummy query sequence $\mathcal{Q}^k$.

begin
  set $(d_1, d_2) \leftarrow$ **estimate**$(\mathcal{L}^k, \mathcal{L}^0, \theta_1)$, $(d_3, d_4) \leftarrow$ **estimate**$(\mathcal{U}^k, \mathcal{U}^0, \theta_3)$; set $\mathcal{Q} \leftarrow \oslash$, $\mathcal{L} \leftarrow \oslash$; `/* initialization */`
  while $\mathcal{Q} = \oslash$ **do**
    set $\mathcal{L} \rightarrow \left\{ l \,|\, l \notin \mathcal{L} \wedge |Tr(l, l_{i-1}^k) - Tr(l_i^0, l_{i-1}^0)| \leq d_1 \right\}$; `/* obtain a set of dummy location candidates, which have similar distance transfer features with the user location */`
    foreach $l \in \mathcal{L}$ **do** `/* remove the dummy locations with dissimilar frequency features, or with unsafe distances */`
      if $\left(d_2 \leq |Fr(l, \mathcal{L}_i^k) - Fr(l_i^0, \mathcal{L}_i^0)|\right) \vee \left(dist(l, l_i^0) < \theta_2 \cdot d^*\right)$ **then** set $\mathcal{L} \leftarrow \mathcal{L} - \{l\}$
    foreach $l \in \mathcal{L}$ **do** `/* remove the dummy locations only related to the sensitive categories */`
      if $|\mathcal{G}(l) - \mathcal{G}^*| = 0$ **then** set $\mathcal{L} \leftarrow \mathcal{L} - \{l\}$; `/* $\mathcal{G}(l)$ denotes the attributes $l$ supports to query */`
    if $g_i^0 \notin \mathcal{G}^*$ **then** `/* if the current query attribute isn't sensitive */`
      set $\mathcal{Q} \leftarrow \left\{(l, u_i^0) \,|\, l \in \mathcal{L}\right\}$; **continue**; `/* stop the current loop ($g_i^0$ is the category of $u_i^0$) */`
    foreach $l \in \mathcal{L}$ **do**
      set $\mathcal{U} \leftarrow \{\, u \,|\, u \in g \wedge g \in \mathcal{G}(l) - \mathcal{G}^* \wedge |Gr(g, \mathcal{U}_i^k) - Gr(g_i^0, \mathcal{U}_i^0)| \leq d_3 \wedge |Fr(u, \mathcal{U}_i^k) - Fr(u_i^0, \mathcal{U}_i^0)| \leq d_4 \,\}$;
      from $\mathcal{U}$, randomly select an attribute u, and then set $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(l, u)\}$; `/* generate a dummy candidate for the current user query */`
  from $\mathcal{Q}$, randomly select a query as a new dummy query q_i^k associated with the dummy query sequence $\mathcal{Q}^k$.

4.2 Implementation Algorithm

In this subsection, we discuss the algorithm implementation for the LBS privacy model. Specifically, we discuss: (1) how to construct a dummy query (including a dummy location and a dummy attribute) according to the user historical query sequence and the dummy historical query sequences; (2) how to construct a dummy query when the historical query sequences are empty; and (3) based on Steps 1 and 2, how to construct a group of dummy queries for a user query, such that the generated dummy query sequences can well ensure the (μ, ρ)-privacy of the user query sequence.

In Step 1, we construct a dummy query that meets the following requirements as much as possible: (1) the dummy query has a similar feature (i.e., location frequency, location transfer, attribute frequency and category frequency) with the user query; and (2) the dummy query is not only of a safe distance away from the user location, but also irrelevant to the user sensitive categories. It can be seen that on the one hand, there may be many dummy queries that can meet the above requirements (i.e., the solution is non-unique); on the other hand, there also may be no dummy queries that meet the requirements, when the thresholds θ_1, θ_2 and θ_3 (see Definitions 5 and 11) are set to stricter values. In Step 1, we only attempt to search one solution that can well meet the above requirements, instead of the optimal

Algorithm 2: Constructing a new dummy query for an empty query sequence.

Input: A user query $q_1^0 = (l_1^0, u_1^0)$, and related threshold parameters.
Output: A new dummy query $q_1^k = (l_1^k, u_1^k)$.

1 **begin**
2 set $\mathcal{Q} \leftarrow \oslash$; set $\mathcal{L}^{\#} \leftarrow \mathcal{L}^{\#} - \{l_1^0\}$; /* initialization ($\mathcal{L}^{\#}$ denotes a set of all the locations) */
3 **while** $\mathcal{Q} = \oslash$ **do**
4 obtain a smaller subset $\mathcal{L}$ of $\mathcal{L}^{\#}$, satisfying $\forall l \in \mathcal{L} \rightarrow dist(l, l_i^0) \geq \theta_2 \cdot d^*$;
5 **foreach** $l \in \mathcal{L}$ **do** /* remove the dummy locations only related to the sensitive categories */
6 **if** $|\mathcal{G}(l) - \mathcal{G}^*| = 0$ **then** set $\mathcal{L} \leftarrow \mathcal{L} - \{l\}$
7 **if** $g_i^0 \notin \mathcal{G}^*$ **then** $\mathcal{Q} \leftarrow \{(l, u_i^0) \,|\, l \in \mathcal{L}\}$; **continue** ; /* stop the current loop */
8 **foreach** $l \in \mathcal{L}$ **do**
9 $\mathcal{U} \leftarrow \{u \,|\, u \in g \wedge g \in \mathcal{G}(l) - \mathcal{G}^*\}$;
10 from the set $\mathcal{U}$, randomly select an attribute u, and then set $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(l, u)\}$; /* generate a dummy query candidate. */
11 from the candidate set $\mathcal{Q}$, randomly select a query as the dummy query q_1^k corresponding to the user query q_1^0.

solution. Algorithm 1 details the implementation of Step 1. From Algorithm 1, it can be seen that we use a greedy strategy to construct a dummy query q_i^k for the user query q_i^0, i.e., we do not take into account the dummy query construction for the subsequent user queries, when constructing a dummy query for the current user query. In addition, in the privacy model, the similarity thresholds θ_1 and θ_3 are designed for query sequences, and the algorithm is designed for single queries, thus the algorithm introduces four new similarity thresholds d_1, d_2, d_3 and d_4 (to replace θ_1 and θ_3), which can be estimated based on θ_1 and θ_3.

In Algorithm 1, we first obtain a set of dummy location candidates $\mathcal{L}$ that have similar location transfer features with the user location l_i^0 (Line 4). Second, from the candidate set $\mathcal{L}$, we remove the dummies whose frequency features are not similar with that of l_i^0, or whose distances away from l_i^0 are unsafe (Lines 5-6). Third, we search a dummy attribute u for each dummy location $l \in \mathcal{L}$, so as to construct a set of dummy query candidates $\mathcal{Q}$, where the dummy attribute u is not only required to have similar attribute frequency and category frequency features with the user attribute u_i^0, but also required to be semantically-irrelevant with the sensitive attribute categories and well-matched with the dummy location l (Line 12). Finally, from the candidate set $\mathcal{Q}$, we randomly select a query as a dummy query q_i^k of the user query q_i^0. Moreover, for the user query q_i^0, if its attribute u_i^0 is not sensitive, we will no longer construct its corresponding dummy attribute (i.e., the current loop will be terminated at Line 10). To simplify the algorithm presentation, we assume that in Algorithm 1 there exists at least one solution (i.e., the condition at Line 3 will not always be true). It can be seen that the output of Algorithm 1 is uncertain, i.e., the same input will lead to different output, because at Lines 13 and 14, the output dummy query q_i^k is selected randomly from two larger sets $\mathcal{U}$ and $\mathcal{Q}$, so as to better ensure the security (see Section 4.3). Moreover, the amount of uncertainty introduced by the algorithm is determined by the sizes of $\mathcal{U}$ and $\mathcal{Q}$, and the sizes of $\mathcal{U}$ and $\mathcal{Q}$ are indirectly controlled by θ_1 and θ_3, so the amount of uncertainty is positively related to the values of θ_1 and θ_3. In addition, at the worst case (i.e., at Line 4, we need to obtain all the locations in the map), the time complexity of Algorithm 1 is equal to $O(|\mathcal{L}^{\#}|)$, where $\mathcal{L}^{\#}$ denotes all the locations in the map.

In Step 2, we study how to construct a dummy query when the historical sequences are empty (i.e., $\mathcal{Q}^k$ and $\mathcal{Q}^0$ are null). At this time, we no longer need to consider the features of location frequency, location transfer, attribute frequency and category frequency, and only need to consider how to construct a dummy query for the current user query, which is of a safe distance from the user location, and irrelevant to the user sensitive categories. Algorithm 2 details the implementation of Step 2. It can be seen that a greedy strategy is also used, because we cannot know the subsequent user queries when processing the current user query, and it is difficult to establish an accurate prediction model to predict the subsequent query locations and query attributes that will be issued by the user. In addition, it can be seen that the output of Algorithm 2 is also uncertain so as to better ensure the security. At the worst case, the time complexity of Algorithm 2 is equal to $O(|\mathcal{L}^{\#}|)$.

In Step 3, we focus on how to construct a set of dummy queries for a user query, actually, which can be solved by running Algorithm 1 or 2 several times. From Definitions 5 and 11, we know that the privacy parameter μ denotes the number of the dummy location sequences that a user wants to construct, and the parameter ρ denotes the number of the constructed attribute sequences if $sig(g^*,\mathcal{U}^k) = 0$. Thus, the running time of Algorithm 1 or 2 for the construction of a dummy query set should be approximately equal to $\max(\mu, \rho)$. In addition, from Algorithms 1 and 2, we see that it is somewhat possible that the candidate set $\mathcal{Q}$ cannot meet the requirements in Definition 12. However, in the two algorithms, for the construction of dummy query candidates, we try our best to make the candidates in accordance not only with the requirements of location feature similarity and location privacy (Definition 5), but also with the requirements of attribute feature similarity and attribute privacy (Definition 11). The experimental results presented in Section 5 also demonstrate that the dummy query sequences constructed by our approach can well ensure the (μ, ρ)-privacy, and the running efficiency of the algorithms is reasonable.

4.3 Security Analysis

From the system model, we see that in a LBS query process, the order that the user query q_i^0 occurs in the query set $\{q_i^k\}_{k=0}^m$ is random. However, the attacker can classify each query to know which queries belong to the same sequence (i.e., rearrange all the queries to form several independent sequences), according to the analysis of the location and attribute features of the historical query sequences recorded by the server. Below, we discuss what can an attacker deduce about the user locations or sensitive attributes, according to the collected LBS query sequences $\mathbb{Q} = (\mathbb{L}, \mathbb{U})$?

Definition 13 (Level I Privacy) *A dummy-based system has Level I privacy protection, if the user query sequence $\mathcal{Q}^0$ can be effectively **hidden** in a group of dummy query sequences $\mathbb{Q}$, i.e., the probability that an attacker can distinguish the user query sequence $\mathcal{Q}^0$ from $\{\mathcal{Q}^0\} \cup \mathbb{Q}$ is equal to $1/(1 + |\mathbb{Q}|)$.*

Remark 1 The LBS system developed based on our approach has Level I privacy protection.

Rationale. According to the attack model in Section 3.2, the attacker can guess based on the prior knowledge that the feature distributions of a user query sequence are particular. However, each dummy sequence $\mathcal{Q}^k \in \mathbb{Q}$ constructed by our approach has highly similar location and attribute feature distributions with the user sequence $\mathcal{Q}^0$, i.e., the feature similarity between $\mathcal{Q}^k$ and $\mathcal{Q}^0$ is greater than θ_1 (for $\mathcal{L}^k$ and $\mathcal{L}^0$) or θ_3 (for $\mathcal{U}^k$ and $\mathcal{U}^0$). Thus, based on the implications of θ_1 and θ_3, we know that $\mathcal{Q}^k$ cannot be distinguished from $\mathcal{Q}^0$

according to the location or attribute features. In addition, due to the rich prior knowledge, the attacker can guess based on the semantic associations between locations and attributes. However, for each dummy sequence $\mathcal{Q}^k$, the dummy location and dummy attribute from the same query are well matched with each other, i.e., the attacker cannot identify out $\mathcal{Q}^k$ according to the semantic associations between locations and attributes. From the above, we conclude that the probability that the attacker can distinguish the user sequence $\mathcal{Q}^0$ from $\mathbb{Q}$ is equal to $1/(1+|\mathbb{Q}|)$. □

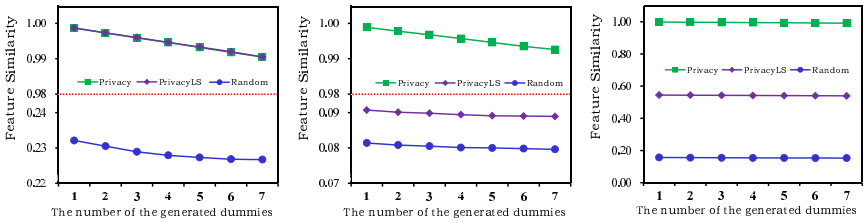
Based on Remark 1 and the attack model, we can conclude that our approach can resist from some existing location privacy attacks, such as colluding attacks and inference attacks. (**Case 1**) The attacker can collude with some users to predict the location privacy and query privacy of other users. However, the attacker, who has almost controlled the server-side, has mastered rich background knowledge, thus the ability of the attacker cannot be further enhanced by colluding with some users, or increasing the number of colluding users, i.e., our approach is colluding attack resistant. (**Case 2**) The attacker can infer a user query sequence according to the mastered background knowledge. However, due the high feature similarity between each dummy sequence $\mathcal{Q}^k$ and the user sequence $\mathcal{Q}^0$, it is difficult for the attacker to distinguish $\mathcal{Q}^k$ from $\mathcal{Q}^0$, i.e., the probability that $\mathcal{Q}^k$ is inferred as the user sequence is equal to that of $\mathcal{Q}^0$, so our approach is inference attack resistant. (**Case 3**) The attacker might also obtain a copy of the privacy algorithm. At this time, the attacker can in turn input each query q_i^t in the set $\{q_i^k\}_{k=0}^m$, and then test whether the privacy algorithm outputs the others $\{q_i^k\}_{k=0}^m - \{q_i^t\}$. If successfully, then it indicates that q_i^t is a user query. However, such an attempt will not succeed, because all the dummy locations and attributes in our algorithms are randomly selected from larger sets (see Lines 13 and 14 in Algorithm 1, and Lines 10 and 11 in Algorithm 2), i.e., the same data input will lead to different output. Thus, the attacker cannot infer the user queries by running our algorithm several times with different submitted queries.

Definition 14 (Level II Privacy) *A dummy-based system has Level II privacy protection, if it has Level I privacy, and the location privacy and query privacy behind each user query sequence $\mathcal{Q}^0$ can be effectively **covered up** by a group of dummy query sequences $\mathbb{Q}$, i.e., the exposure degree of the location privacy and query privacy on $\{\mathcal{Q}^0\} \cup \mathbb{Q}$ is not greater than $1/(1+|\mathbb{Q}|)$ of the original on $\mathcal{Q}^0$.*

Remark 2 The LBS system developed based on our approach has Level II privacy protection.

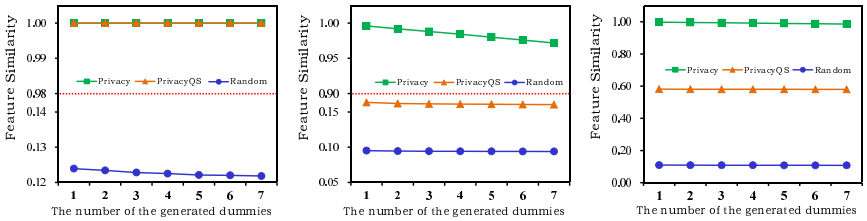
Rationale. According to the privacy model and its algorithms given in Section 4.2, each dummy location $l_i^k \in \mathcal{Q}^k$ has a safe distance (greater than $\theta_2 d^*$) from its user location $l_i^0 \in \mathcal{Q}^0$ (i.e., it has no impact on the location privacy on l_i^0), so each dummy location sequence $\mathcal{Q}^k$ has no impact on the location privacy on $\mathcal{Q}^0$, i.e., the exposure degree of the location privacy on $\{\mathcal{Q}^0\} \cup \mathbb{Q}$ is reduced to $1/(1+|\mathbb{Q}|)$ of the original on $\mathcal{Q}^0$. In addition, for each sensitive category g^*, because $\sum_{u_i^k \in \mathcal{U}^k} Fr(u_i^k, \mathcal{U}^k) = \sum_{u_i^0 \in \mathcal{U}^0} Fr(u_i^0, \mathcal{U}^0)$ and $Fr(g^*, \mathcal{U}^k) = 0$, based on Definition 10, we can easily prove that $sig(g^*, \{\mathcal{U}^0\}) = (1+|\mathbb{Q}|) \cdot sig(g^*, \{\{\mathcal{U}^0\} \cup \mathbb{U}\}$. Thus, we conclude that the exposure degree of the query privacy on $\{\mathcal{Q}^0\} \cup \mathbb{Q}$ is reduced to $1/(1+|\mathbb{Q}|)$ of the original on $\mathcal{Q}^0$. □

It can be observed that compared to Definition 13, Definition 14 presents higher security. If a LBS system meets Level II privacy, then it first means that the attacker cannot know which one in $\{\mathcal{Q}^0\} \cup \mathbb{Q}$ is the user query sequence. At this time, based on the prior knowledge that the queries from the same user over a period of time often occur in some



(a) **simFr** (b) **simTr** (c) **simLoc**

Fig. 2 The experimental results for the location feature similarity



(a) **simFu** (b) **simGr** (c) **simAtt**

Fig. 3 The experimental results for the attribute feature similarity

fixed locations, the attacker can still guess the user locations by computing the frequency of occurrence of each location in $\mathbb{L}$. However, for each high frequency location obtained by the attacker, the probability that it is a true user location is only equal to $1/(1+|\mathbb{L}|)$, because in $\mathbb{L}$ each user location has been covered up by $|\mathbb{L}|$ dummy locations with safe distances away from it. Likewise, for each high frequency attribute (or category) in $\mathbb{U}$, either it is not sensitive, or the probability that it is a sensitive attribute (or category) is only equal to $1/(1+|\mathbb{U}|)$. In summary, it is difficult for the attacker to guess the user location privacy and attribute privacy in a LBS system with Level II privacy.

5 Experiment Evaluation

From the security analysis in Section 4.3, it can be seen that the security of our approach is dependent on the quality of constructed dummy query sequences, i.e., whether the dummy query sequences can effectively hide the user query sequence (Level I privacy), and whether the dummy query sequences can effectively cover up the location privacy and query privacy behind the user query sequence (Level II privacy). In this section, we evaluate the effectiveness of the dummy query sequences by experiments. The experiments consists of two parts: (1) the first part evaluates the feature distribution similarity of the dummy query sequences to the user query sequence; and (2) the second part evaluates the effectiveness of the dummy query sequences to cover up users' location privacy and query privacy.

5.1 Experimental Setup

First, we briefly describe the experimental setup, including the reference dataset, user query sequences and algorithm candidates.

(1) Reference dataset. In the experiments, we adopted the real check-in data from Gowalla[3] supplied by the SNAP repository [33]. Gowalla is an online location-based social network application, where users share their current locations by checking-in the application, so each check-in record consists of userid, time, location (latitude and longitude) etc. Firstly, we selected 1000 users and selected 1000 check-in records for each user. Secondly, we divided the geographic region (its upper left coordinate is about (20, -11) and right lower coordinate is about (60, 20)) covered by all the check-in records into 100×100 location subregions and 1000×1000 location cells (i.e., locations). Thirdly, we divided all the location subregions into 10 categories, and assigned the category for each subregion according to the density (which is computed by the number of check-in records located at the subregion). Finally, from the category system supplied by a Baidu points-of-interest application[4], we selected 25 attribute categories in advance, and for each location category (10 in total), we randomly assigned the attribute categories that it can support to query, making that given any location l_i^k, based on the location category it belongs to, we can know the attribute categories $\mathcal{G}(l_i^k)$ that it can support to query.

(2) User query sequences. For generating a user query sequence $\mathcal{Q}^0$, we need to construct a location sequence $\mathcal{L}^0$ and an attribute sequence $\mathcal{U}^0$. Here, each location sequence $\mathcal{L}^0$ was chosen from the check-in data from Gowalla. Below, we show how to construct the attribute sequence $\mathcal{U}^0$. First, we in advance randomly assigned the attributes for each attribute category (averaging 40 attributes per category). Second, to construct each attribute u_i^0 for $\mathcal{U}^0$, we randomly selected the attribute category g_i^0 of u_i^0 from all the preset categories according to a standard normal distribution, and then randomly selected the attribute value of u_i^0 from all the preset attributes belonging to g_i^0 according to a uniform distribution. In the above process, the number of the sensitive categories contained in each user attribute sequence is an experimental parameter and can be adjusted dynamically.

(3) Algorithm candidates. In the experiments, we used the following four dummy-based algorithm candidates: (1) *Privacy*, i.e., the approach proposed in this paper; (2) *PrivacyLS* [9], which constructs dummy queries to protect the location privacy by considering the location frequency feature; (3) *PrivacyQS* [31], which constructs dummy queries to protect the query privacy by considering the query context; and (4) *Random* (used as the baseline), which uses a random way to construct dummy locations and dummy attributes. In the experiments, we did not compare against other algorithms mentioned in the related work section, since they are designed under different privacy models (i.e., pseudonym, obfuscation or encryption), so they are incomparable to our approach.

5.2 Feature Distribution Similarity

In the first group of experiments, we aim to evaluate the effectiveness of the dummy query sequences produced by our approach to hide the user query sequences (i.e., the feature distribution similarity between the user query sequences and the dummy query sequences). Here, we use the metrics developed based on the privacy model in Section 4.1, combined with the location entropy proposed in [9]. For an algorithm candidate A (*Privacy*, *PrivacyLS*, *PrivacyQS* or *Random*) and a user query sequence $\mathcal{Q}^0 = (\mathcal{L}^0, \mathcal{U}^0)$, let $\mathbb{Q} = (\mathbb{L}, \mathbb{U})$ denote a group of dummy query sequences generated by the candidate A for the user sequence $\mathcal{Q}^0$. Then, the location feature similarity metrics for the candidate A can be formulated as

[3] http://snap.stanford.edu/data/loc-gowalla.html

[4] http://map.baidu.com

follows:

$$\textbf{simFr}(A) = \min_{\mathcal{L}^k \in \mathbb{L}} sim^{Fr}(\mathcal{L}^k, \mathcal{L}^0) \tag{16}$$

$$\textbf{simTr}(A) = \min_{\mathcal{L}^k \in \mathbb{L}} sim^{Tr}(\mathcal{L}^k, \mathcal{L}^0) \tag{17}$$

$$\textbf{simLoc}(A) = \frac{1}{2}(\textbf{simTr}(A) + \textbf{simFr}(A)) \tag{18}$$

Similarly, based on Definitions 7 and 9, we define the attribute feature similarity metrics for the candidate A, respectively denoted by $\textbf{simFu}(A)$, $\textbf{simGr}(A)$ and $\textbf{simAtt}(A)$. Finally, we define the location and attribute relevance metric for the candidate A as follows (where g_i^k denotes the category of the attribute u_i^k of the query q_i^k, and $\mathcal{G}(l_i^k)$ denotes the attribute categories the location l_i^k of q_i^k supports to query).

$$\textbf{revLA}(A) = \min_{\mathcal{Q}^k \in \mathbb{Q}} \frac{|\{q_i^k \,|\, q_i^k \in \mathcal{Q}^k \wedge g_i^k \in \mathcal{G}(l_i^k)\}|}{|\mathcal{Q}^k|} \tag{19}$$

It is obvious that for each of the above metrics, a higher value is better, which means that the dummy query sequences have more similar feature distributions with the user query sequence, making it difficult for an attacker to identify the user query sequence from the set $\mathbb{Q} \cup \{\mathcal{Q}^0\}$. In the experiments, the length of each user query sequence is fixed to 1000 (i.e., $|\mathcal{L}^0| = 1000$). The experiment results are shown in Figs. 2 to 4, where the value of each point is the average of 50 running results, and the caption of each subfigure denotes the related metric used in the experiments. In addition, the X axis denotes the number of generated dummy query sequences; and the Y axis denotes the metric value between the user query sequence and the dummy query sequences. In Fig. 2, the experimental results of *PrivacyQS* are not presented, because the location privacy issue is not considered in the algorithm. Similarly, the experimental results of *PrivacyLS* are not presented in Fig. 3, because it does not consider the attribute privacy.

From Fig. 2, it can be seen that compared to those of the baseline *Random* approach, the dummy locations constructed by *Privacy* or *PrivacyLS* exhibit a much better feature distribution similarity with the user query locations. Specifically, the overall similarity from *Random* is less than 0.2, and the overall similarity from *Privacy* or *PrivacyLS* is greater than 0.6. Also, it can be seen that compared to *PrivacyLS*, the dummy locations constructed by our recommended *Privacy* have a better overall feature similarity (close to 1.0). This is because *PrivacyLS* only takes into account the location frequency feature, without considering the location transfer feature (see the result in the subfigure (b)). From Fig. 3, we see that the dummy attributes constructed by *Privacy* or *PrivacyQS* exhibit a much better feature distribution similarity, compared to those from *Random*, and the dummy attributes constructed by *Privacy* further exhibit much better overall feature similarity than those from *PrivacyQS* (because the category frequency feature is not considered by *PrivacyQS*). Specifically, the feature similarity of the dummy attributes constructed by *Privacy* is close to 1.0, and the similarity almost remains unchanged, with the changing of the number of dummy query sequences. Finally, from Fig. 4, we also see that the dummy query sequences constructed by *Privacy* or *PrivacyQS* exhibit good relevance between the locations and the attributes (all close to 1.0); and those from *Random* or *PrivacyLS* exhibit worse relevance, because the location and attribute relevance feature is not considered by them.

From the above, we conclude that the dummy query sequences constructed by our approach have a highly similar feature distribution with the user query sequence (close to 1.0), thereby, making it difficult for an attacker to rule out the dummy query sequences, i.e., the user query sequences can be effectively hidden by the dummy query sequences constructed by our approach.

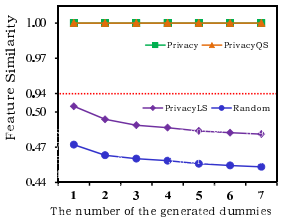
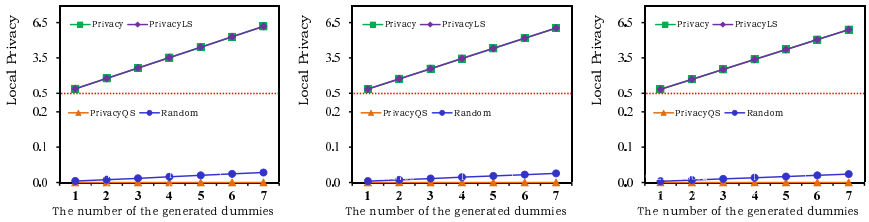
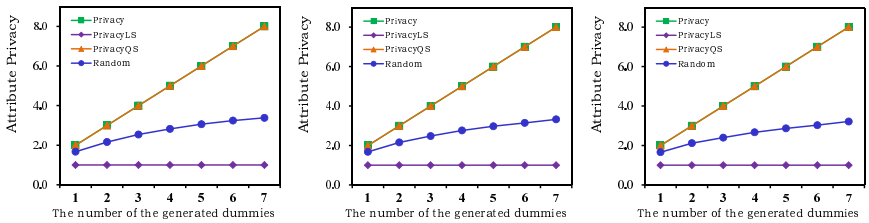


Fig. 4 The experimental results for the location and attribute relevance



(a) **priLoc** (200) (b) **priLoc** (600) (c) **priLoc** (1000)

Fig. 5 The experimental results for the user location privacy protection



(a) **priAtt** $(1; 200)$ (b) **priAtt** $(2; 600)$ (c) **priAtt** $(3; 1000)$

Fig. 6 The experimental results for the user attribute privacy protection

5.3 Privacy Exposure Degree

In the second group of experiments, we aim to evaluate the effectiveness of the dummy query sequences produced by our approach to cover up the location privacy and attribute privacy behind the user query sequences (so as to reduce the exposure degree of the user privacy). Here, we use privacy metrics developed based on Definitions 5 and 11. Given an algorithm A and a user query sequence $\mathcal{Q}^0 = (\mathcal{L}^0, \mathcal{U}^0)$, let $\mathbb{Q} = (\mathbb{L}, \mathbb{U})$ denote a group of dummy query sequences generated by the candidate A for the user sequence $\mathcal{Q}^0$, and $\mathcal{G}^*$ denote the user sensitive attribute categories. Then, the location privacy metric and the attribute privacy metric can be formulated as follows (where $\{l_i^k\}$ denotes all the dummy locations in $\mathbb{L}$ corresponding to the user location l_i^0):

$$\textbf{priLoc}(A) = \min_{l_i^0 \in \mathcal{L}^0} |\{l | l \in \{l_i^k\}, dist(l, l_i^0) \geq \theta_2 d^*\}| \tag{20}$$

$$\textbf{priAtt}(A) = \min_{g \in \mathcal{G}^*} sig(g^*, \{\mathcal{U}^0\}) / sig(g^*, \{\mathcal{U}^0\} \cup \mathbb{U}) \tag{21}$$

Note that Equation 20 (location privacy) combined with Equation 18 (location feature similarity) can be used to count the number of the dummy locations that have not only indis-

tinguishable feature distributions but also privacy-lossless distances with the user locations, thus the location privacy metric is accordant to the k-anonymity (a widely-used privacy metric) to a certain extent. Also, Equation 21 is similar. For the two metrics, a higher value is better, which means a smaller exposure degree of the user locations or sensitive attributes, resulting in better effectiveness to cover up the location privacy or query privacy behind the user query sequence. The experimental results are shown in Figs. 5 and 6, where the value of each point is the average of 50 running results. In Fig. 5, the caption of each subfigure denotes the length of each user location sequence (i.e., $|\mathcal{L}^0|$, which is set to 200, 600 or 1000). In Fig. 6, the caption of each subfigure denotes the number of the sensitive categories related to each user attribute sequence (i.e., $|\mathcal{G}^*|$, which is set to 1, 3 or 5) and the length of each user location sequence (i.e., $|\mathcal{L}^0|$). In addition, the X axis denotes the number of dummy query sequences constructed for each user query sequence; and the Y axis denotes the location privacy metric or the attribute privacy metric.

From Fig. 5, it can be seen that compared to those of *Random* or *PrivacyQS*, the dummy location sequences constructed by *Privacy* or *PrivacyLS* exhibit much better effectiveness on covering up the user location privacy. Specifically, the effectiveness to cover up the user location privacy is almost positively relevant to the number of dummy query sequences constructed for each user query sequence, independently of the length of each user location sequence (i.e., $|\mathcal{L}^0|$). From Fig. 5, it can be also seen that *PrivacyQS* has the worst performance in terms of location privacy security (equal to 0), because the location privacy problem is not considered by the approach at all (i.e., each dummy location sequence is the same to its corresponding user location query sequence). From Fig. 6, it can be seen that the dummy attribute sequences constructed by the *Privacy* approach have good effectiveness to cover up the user query privacy (i.e., which can effectively reduce the exposure degree of the sensitive attribute categories), and the effectiveness is almost positively relevant to the number of constructed dummy query sequences, independently of the length of each user query sequence and the number of the sensitive attribute categories. Also, it can be seen that the dummy query sequences constructed by *PrivacyQS* or *Random* can also reduce the exposure degree of the sensitive attribute categories, but the performance stability is worse than the *Privacy* approach. This is because the two algorithms do not select dummy attributes from the non-sensitive categories when constructing dummy attribute sequences. In addition, *PrivacyLS* has the worst effectiveness on attribute privacy protection, because the attribute privacy issue is not considered by the approach.

From the above, it can be concluded that the dummy query sequences from our approach can not only reduce the exposure degree of the user sensitive attributes but also the exposure degree of the user locations, making it difficult for an attacker to obtain the user query locations or query attributes under the precondition of not identifying out the user query sequence, i.e., the location privacy and query privacy can be effectively covered up by the dummy sequences constructed by our approach.

6 Conclusion

Location-based services (LBS) have become an important part of people's daily life. However, while providing great convenience for users, LBS result in a serious problem on personal privacy, i.e., location privacy and query privacy. To this end, in this paper, we proposed an approach for protecting user personal privacy in location-based services (LBS), whose basic idea is to construct dummy query sequences to cover up the user locations and attributes, and in turn protect user personal privacy in LBS. First, we used a client-based system framework

that requires not only no change to the existing LBS algorithms, but also no compromise to the accuracy of a LBS query. Second, based on the framework, we introduced a privacy model to formulate the constraints that ideal dummy query sequences should satisfy. Third, we present an implementation algorithm to construct dummy query sequences that can well meet the privacy model.

Finally, both theoretical analysis and experimental evaluation have demonstrated the effectiveness of our approach: (1) the dummy query sequences constructed by the approach can effectively hide the user queries, i.e., having highly-similar feature distributions with the user query sequence, including the features of location sequences, the features of attribute sequences, and the relevance features between query locations and query attributes, thereby, making it difficult for an attacker to rule out the dummy queries; (2) the dummy query sequences constructed by the approach can effectively cover up users' query privacy and location privacy, i.e., they are not only semantically irrelevant to the user sensitive attribute categories, but also far distant from the user locations; and (3) it does not cause serious performance overheads on the running efficiency. Thus, we conclude that our approach can be used to effectively protect user privacy in LBS.

In summary, this paper presents a valuable study attempt to the protection of user privacy in LBS. The main theoretical and practical implications of our study is to propose an effective approach for the protection of users' location privacy and query privacy in LBS, and compared with other existing works, the proposed approach can ensure the security of users' LBS privacy on the untrusted server-side, without jeopardizing the usability, accuracy and efficiency of each LBS query. As a result, it is easy for our approach to be integrated with an existing LBS application, i.e., our approach has a positive impact on the construction of a privacy-preserving LBS application. However, for the practical application of the proposed approach, there are still some limitations that we need to further study and solve, e.g., since LBS applications are of various forms (e.g., Mobile Terminal), we need to study how to implement a seamless connection between our approach and each kind of application interface.

References

1. T. Peng, Q. Liu, G. Wang. Enhanced location privacy preserving scheme in location-based services. IEEE Systems Journal, 2017, 11 (1): 219–230
2. S. Zeng, Y. Mu, M. He et al. New approach for privacy-aware location-based service communications. Wireless Personal Communications, 2018, 11 (2): 1057–1073
3. Z. Li, Q. Pei, I. Markwood et al. Location privacy violation via GPS-agnostic smart phone car tracking. IEEE Transactions on Vehicular Technology, 2018, 67 (6): 5042–5053
4. M. Ghaffari, N. Ghadiri N, M. H. Manshaei et al. P4QS: A peer to peer privacy preserving query service for location-based mobile applications. IEEE Transactions on Vehicular Technology, 2017, 66(10): 9458–9469
5. C. Kalaiarasy, N. Sreenath, A. Amuthan. Location privacy preservation in VANET using mix zones - A survey. Proc. of ICCCI, 2019: 1–5.
6. X. Ding, W. Yang, R. Choo et al. Privacy-preserving similarity joins using MapReduce. Information Sciences, 2019, 493: 20–33
7. R. Dewri, R. Thurimella. Mobile local search with noisy locations. Pervasive and Mobile Computing, 2016, 32: 78–92
8. L. Zhang, J. Li, S. Yang et al. Privacy preserving in cloud environment for obstructed shortest path query. Wireless Personal Communications, 2017, 96(2): 2305–2322
9. B. Niu, Q. Li, Q. Zhu et al. Achieving k-anonymity in privacy-aware location-based services. Proc. of INFOCOM, 2014: 754–762
10. Z. Wu, G. Li, Q. Liu, G. Xu and E. Chen. Covering the sensitive subjects to protect personal privacy in personalized pecommendation. IEEE Transactions on Services Computing, 2018, 11 (3): 493–506

11. F. Li, C. Zhang, B. Niu et al. Efficient scheme for user's trajectory privacy. Chinese Journal on Communications, 2015, 36(12): 114–123
12. N. Ravi, C. Krishna, I. Koren. Enhancing vehicular anonymity in ITS: A new scheme for mix zones and their placement. IEEE Transactions on Vehicular Technology, 2019, 68(11): 10372–10381
13. B. Palanisamy, L. Liu, K. Lee et al. Anonymizing continuous queries with delay-tolerant mix-zones over road networks. Distributed and Parallel Databases, 2014, 32(1): 91–118
14. I. Memon, Q. Ali, A. Zubedi et al. DPMM: Dynamic pseudonym-based multiple mix-zones generation for mobile traveler. Multimedia Tools and Applications, 2016: 1–30
15. D. Zhao, Y Jin, K. Zhang et al. EPLA: efficient personal location anonymity. GeoInformatica, 2018, 22 (1): 29-47.
16. B. Gedik, L. Liu. Protecting location privacy with personalized k-anonymity: Architecture and algorithms. IEEE Transactions on Mobile Computing, 2008, 7(1): 1–18
17. D. Xue, L. Wu, H. Li et al. A novel destination prediction attack and corresponding location privacy protection method in geo-social networks. International Journal of Distributed Sensor Networks, 2017, 13(1): 1–16
18. S. Soma, T. Hashem, M. Cheema et al. Trip planning queries with location privacy in spatial databases. World Wide Web, 2017, 20: 205–236
19. B. Agir, T. Papaioannou, R. Narendula et al. User-side adaptive protection of location privacy in participatory sensing. GeoInformatica, 2014, 18(1): 165–191
20. R. Dewri, R. Thurimella. Exploiting service similarity for privacy in location-based search queries. IEEE Transactions on Parallel and Distributed Systems, 2014, 25(2): 374–383
21. M. E. Andrs, N. E. Bordenabe, K. Chatzikokolakis et al. Geo-indistinguishability: Differential privacy for location-based systems. Proc. of CCS, 2013: 901–914
22. R. Mendes, M. Cunha, J. Vilela. Impact of frequency of location reports on the privacy level of geo-indistinguishability. Proceedings on Privacy Enhancing Technologies, 2020, 2: 379–396
23. X. Dong, T. Zhang, D. Lu et al. Preserving geo-indistinguishability of the primary user in dynamic spectrum sharing. IEEE Transactions on Vehicular Technology, 2019, 68(9): 8881–8892
24. J. Lai, Y. Mu, F. Guo F , et al. Privacy-enhanced attribute-based private information retrieval. Information Sciences, 2018, 454: 275–291
25. Z. Mei, H. Zhu, Z. Cui et al. Executing multi-dimensional range query efficiently and flexibly over outsourced ciphertexts in the cloud. Information Sciences, 2018, 432: 79C96
26. L. Zhang, S. Tang, J. Chen et al. Two-factor remote authentication protocol with user anonymity based on elliptic curve cryptography. Wireless Personal Communications, 2015, 81(1): 53–75
27. X. Ding, P. Liu, H. Jin. Privacy-preserving multi-keyword top-k similarity search over encrypted Data. IEEE Transactions on Dependable and Secure Computing, 2019, 16(2): 344–357
28. I. Salman, L. Miss, D. Babak et al. On cloud security attacks: A taxonomy and intrusion detection and prevention as a service. Journal of Network and Computer Applications, 2016 74: 98–120
29. A. Khoshgozaran, H. Shirani-Mehr, C. Shahabi. Blind evaluation of location based queries using space transformation to preserve location privacy. GeoInformatica, 2013, 17(4): 599–634
30. W. Meng, B. Lee, X. Xing et al. Trackmeornot: Enabling flexible control on web tracking. Proc. of WWW, 2016: 99–109
31. Z. Wu, J. Shi, C. Lu et al. Constructing plausible innocuous pseudo queries to protect user query intention. Information Sciences, 2015, 325: 215–226
32. Z. Wu, R. Li, J. Xie et al. A user sensitive subject protection approach for book search service. Journal of the Association for Information Science and Technology, 2020, 71(2): 183–195
33. J. Leskovec, R. Sosic. SNAP: A general-purpose network analysis and graph-mining library. ACM Transactions on Intelligent Systems and Technology, 2016, 8(1): 1
34. M. Haus, M. Waqas M, A. Ding et al. Security and privacy in device-to-device (D2D) communication: A review. IEEE Communications Surveys & Tutorials, 2017, 19(2): 1054–1079