A Demand-Aware Location Privacy Protection Scheme in Continuous Location-based Services

Xinghua Li Lingjuan Deng Sheng Gao Jianfeng Ma Qingsong Yao
Shaanxi Key Laboratory of Network and System Security
School of Computer Science and Technology, Xidian University, Xi'an, Shaanxi, China

Abstract—**Location-based Services (LBSs) via mobile handled devices have been subject to major privacy concerns for users. Currently, most of the existing works concerning the continuous LBS queries mainly focus on users' privacy demands with little consideration of the service of quality (QoS). In this paper, we propose a Demand-Aware Location Protection scheme (DALP) for continuous LBS requests, allowing a user to customize not only location privacy but also QoS requirement. However, this results in that the privacy and QoS requirement cannot be met together in considerable query points and the location privacy protection cannot be provided for the continuous LBS queries. We point out its underlying reason is that in few LBS query regions the footprints are sparse or the privacy requirements are set unreasonably high. Therefore, a maximum Demands-Aware Query Sequence algorithm (MaxDAQS) is proposed in the scheme. Through identifying and restraining the queries in those regions, most of LBS queries are satisfied, thus, the longest LBS query sequence is obtained which can satisfy a user's specific privacy and QoS requirements simultaneously. The extensive simulations on a large dataset prove the effectiveness of our approach under various location privacy and QoS demands.**

Keywords—continuous LBS queries, location privacy, QoS.

I. Introduction

With the development of wireless communication and mobile positioning technologies, location-based services (LBSs) have emerged as potential applications in mobile computing scenarios. LBS [1] refers to a specific type of information services, which makes use of the locations of mobile devices through positioning technologies within a mobile network. Many such applications have been widely used in people's daily life, such as localization services (e.g., Google Latitude), check-in applications (e.g., Foursquare), point of interest queries (e.g., querying the nearest gas station from the current location), navigation services (e.g., finding the nearest route to a gas station) and so on. The LBS query is categorized into two classes: one is the snapshot query (e.g., the query of the nearest restaurant), and the other one is the continuous LBS queries (e.g., the continuous acquisition of the weather report according to the user's current location).

Apparently, in the LBS a user has to release his location to the application server, and there is a nature relationship between the location information and the user's identity. Consequently, exposure of the location information will result in the leakage of the identity of a user. However, a user wouldn't like others know his current location (e.g., hospital or casino). If the location privacy cannot be protected carefully, the LBS service will be affected greatly. This paper mainly focuses on the location privacy protection in the continuous LBS queries.

Currently, most continuous LBS queries anonymity schemes focus on users's privacy demands with little consideration of the QoS, leading to a poor LBS service. In order to solve this problem, we propose the Demand-Aware Location Privacy protection scheme (DALP), allowing a user to set the QoS demands besides the privacy requirement.

At the same time, in the DALP scheme the diversity of a user's privacy and QoS requirements is taken into consideration: different users have various requirements for the location privacy and QoS, and even for a same user, his requirements vary in different scenarios. For example, generally a user's privacy requirement in a hospital is higher than that in a shopping mall. In the continuous LBS queries, if the value of requirement is uniform rather than personalized for every query point (i.e., the privacy and QoS requirement for each LBS query are set same), it cannot provide a user with better service in accordance with his privacy and QoS requirements for different locations, resulting an overall poor QoS. Consequently, in the DALP scheme we allow a user to customize not only the personalized location privacy but also QoS requirements for each LBS query.

However, our experimental results indicate that the simple introduction of personalized QoS demands results in that the privacy and QoS requirements cannot be met simultaneously in considerable query points. As shown in Fig.4, more than 40% of queries cannot meet those two requirements simultaneously. Especially, when the number of continuous LBS queries gets larger (e.g., $\geq$ 30), over 93% of queries cannot meet the user's privacy and QoS demands simultaneously, and then, those queries will be refused and the location privacy protection cannot be provided for the continuous LBS queries. That is, in some queries though the size of the constructed cloaking areas in the continuous LBS queries have reached the upper bounds of a user's QoS demands, the user's privacy requirements still cannot be met. Our observation indicates that the underlying reasons are that: (1) in few LBS query regions the footprints are sparse, which limits the common user size in the continuous queries sequence; or (2) the privacy requirements at few LBS query points are set unreasonably high, and the common user set in the cloaking areas cannot meet the privacy requirements. In order to solve this problem, in the DALP scheme a maximum Demands-aware Request Sequence algorithm (MaxDAQS) is proposed. Through finding and restraining the queries in those regions, most of LBS queries are satisfied, thus, the longest LBS query sequence is obtained which can satisfy the user's specific privacy and QoS requirements. That is, most LBS queries are satisfied at the cost of constraining few ones.

The rest of this paper is organized as follows. The related work is introduced in Section II. Section III presents the system structure model and storage structure. The detailed schemes and algorithms are given in Section IV. The experiments are made and the results are analyzed in Section V. Section VI concludes this paper.

II. Related Work

In most of existing works, location privacy in continuous LBS queries has been considered rarely. Chow et al. [2] first proposed to solve the location privacy problem in continuous LBS queries. By distinguishing between location privacy and query privacy, they analyzed that existing approaches for snapshot queries were not suitable for continuous queries and pointed out that those constructed cloaking areas in location sparse area were vulnerable to both *query sampling attacks* and *query tracking attacks*. Finally, they proposed to construct the anonymity region with two properties in terms of k*-sharing* and *memorization* to prevent these two types of attacks. However, because each query exploits the anonymity set constructed in the first query, it is likely to cause large anonymity regions. Pingley et al. [3] proposed to protect a user's query privacy in continuous LBSs based on multiple dummy queries with the same location. However, they did not consider the location privacy in a user's query. Taking a user's current environment into consideration, Gao et al. [4] proposed to construct different equivalence classes without any intersection for location privacy protections both in snapshot queries and continuous queries through the help of the user's surrounding partners in participatory sensing. Specially in continuous LBSs, the user and his/her partners in each equivalence class only upload their different locations and queries without any identity information to resist *query tracking attacks*. The effectiveness of these schemes would rely largely on users distribution.

The most related works to our work are [5–7], which exploit the historical footprints to construct the cloaking area with common users. To be specific, Xu et al. [5] proposed to use the size of anonymity set k to measure the location privacy in continuous LBS queries. They constructed the minimum boundary anonymity region that included k users based on the user's current footprint and the historical footprints of the other users. However, it may be hard for a user to assign the privacy level and should learn the locations of all the queries in advance. Thus, in their following work [6] , they let the user express the demand of location privacy via public region and proposed feeling-based privacy model. They used *entropy* to measure the popularity of public region based on users' historical footprints, and then protected the user's location privacy in continuous LBS by ensuring the popularity of the user's current location is no less than that of the public region. However, those two schemes assume that the user has the unified demand of location privacy. Obviously, it does not accord with the fact. For the diverse demands of location privacy, Wang et al. [7] presented *L2P2* that are suitable for both the metrics of k*-anonymity* and *entropy*. They proposed two schemes, namely *basic L2P2* and *enhanced L2P2* to construct the anonymity region. To be specific, the first does not consider the relationship among queries and construct the anonymity region only based on historical footprints, while the later construct the anonymity region with common users based on the historical footprints. However, this scheme cannot solve the rapid growth of anonymity regions, especially in the case that the historical footprints are rare, resulting in a poor QoS. Besides, the above schemes do not consider the user's personalized QoS demands at different query points, thus, the query results may be inaccurate when the anonymity regions are too large.

III. Preliminaries

This section begins with an overview of the system architecture for the protection of the location privacy in continuous LBS queries. Subsequently, we will introduce the data storage structure used. Finally, we discuss the privacy metrics used in this paper.

A. System architecture

Similar to existing works [5–10], in this paper, we also employ the classical C/S system structure depicted by Fig. 1, which consists of four modules: positioning systems, the mobile device, the proxy server and location-based services provider (LSP). Zeimpekis et al. [11] summarized outdoor and indoor positioning technology.

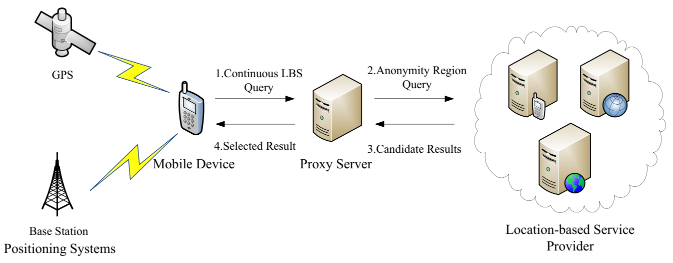


Fig. 1: System architecture model for the location privacy protection in the continuous LBS queries

The mobile device communicates with LSP through a proxy server, which is considered as a trusted entity. The proxy server is used to receive and cloak the locations in the continuous queries, then transmit the cloaking areas to the LSP. The typical process for the location privacy protection in the continuous LBS queries is described as follows. A mobile user u sends a LBS query formalized as $Q_i = \{u, t, l(x, y), r\}$ to the proxy server, where $l(x, y)$ represents the current location of $Q_i.u$ at time stamp t and $Q_i.r$ is his/her demand. It should be noted that $Q_i.r$ can represent both privacy and QoS demand. And here, the location k-anonymity method is used for privacy protection, and in order to prevent an attacker diminishing the possible user set through observing the common users in a sequence of queries, a common user set (or common users' historical footprints) is used in the continuous LBS queries. The QoS is measured by the area size of the constructed cloaking area, and its reasonability lies in that the cloaking area size will directly affect the query latency, server's celling and the accuracy of query results.

After receiving the user's continuous LBS queries $\mathcal{Q} = \{Q_1, Q_2, \cdots, Q_n\}$, according to the historical footprints database, the proxy server computes the longest sequence of LBS queries where all the demands of a user's privacy and QoS are satisfied and then constructs cloaking area set $\mathcal{C} = \{C_i, i = 1, 2, \cdots, n\}$ to provide location privacy protection. Then, it sends the cloaking area set $\mathcal{C}$ to LSP for candidate queries. LSP processes a sequence of cloaking area queries and then returns the candidate results to the proxy server. Note that only the common users in the continuous LBS queries are considered in the construction of the cloaking area set $\mathcal{C}$. In the above process, how to find the longest LBS queries sequence where at each query point the user's demands $R = \{Q_i.r, i = 1, 2, \cdots, n\}$ are satisfied is our goal. Therefore, our proposal mainly works on the proxy server.

B. Data storage structure

Similar to the work [5, 6, 9], based on the simple grid-based approach (shown in Fig.2), we construct the footprint database for efficient retrieval of users' location data . Each cell at the grid stores a set of user who have appeared in it, which is represented as $\langle cid, \mathcal{U} \rangle$ in the grid table G, wherein cid is the cell identifier and $\mathcal{U}$ is the set of user identifiers that have appeared in the cell. As shown in Fig.2, $\{u_2, u_3\}$ have historical footprints in the cell c_4, while $\{u_1, u_3, u_4\}$ have ever appeared in the cell c_8. In addition, we keep a footprint table F that has one entry for each mobile user with the form $F = \{\cdots, f_i, \cdots, f_j \cdots\}$, where f_i represents $\langle t, l(x, y) \rangle$ that records the temporal-spatial information. Thus, given a user's identifier, we can efficiently retrieve all his historical footprints from G and F.

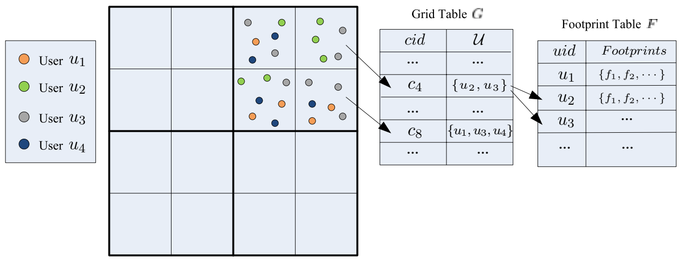


Fig. 2: Storage structure for users's historical footprints

In addition, given a cell identifier, we can easily find the historical footprints that have passed through this cell from the grid table G and footprint table F. The grid cells can be easily figured out according to the respective cloaking areas constructed based on the user's demands in the continuous LBS queries. Then, the common user set in these cloaking areas can be easily identified, based on which an eventual cloaking area with common footprints is formed for each corresponding query.

C. Privacy metrics

At present, there are two kinds of metrics to measure location privacy,they are *k-anonymity* and *entropy-based*. Same as [7], we also use $P(c)$ to represent the privacy level for the cloaking area c.

Definition 1 (k-anonymity privacy): Let c represent a cloaking area and $\mathcal{U} = \{u_1, u_2, \cdots, u_k\}$ represent the set of users whose footprints are in c at time t. The k-anonymity privacy of c can be expressed as $P(c) = |\mathcal{U}|$.

Definition 2 (entropy-based privacy): Let c represent a cloaking area and $\mathcal{U} = \{u_1, u_2, \cdots, u_k\}$ denote the set of users whose footprints are in c at time t and n_i represent the number of u_i's historical footprints in c. We can compute the total number of footprints of $\mathcal{U}$ in c as $N = \sum_{i=1}^{k} n_i$. The entropy of c is $H(c) = -\sum_{i=1}^{k} \frac{n_i}{N} log \frac{n_i}{N}$, and the privacy of c is defined as $P(c) = 2^{H(c)}$.

In the following, we mainly focus on how to find and constrain inappropriate queries to maximize the number of DAQS in the continuous LBS queries when the user's demands cannot be met.

IV. The Proposed DALP Scheme

The proposed DALP scheme mainly consists of the following two steps:

1) Check Original Queries: For each query Q_i in the continuous LBS queries $\mathcal{Q} = \{Q_1, Q_2, \cdots, Q_n\}$, we need to check whether it satisfies $P(A_i) \geq Q_i.p$. If not, the user needs to adjust his demands $Q_i.r(p, (q_x, q_y))$ to a reasonable level.
2) Maximize Demand-aware Query Sequence: If there is any query Q_i in the continuous LBS queries $\mathcal{Q} = \{Q_1, Q_2, \cdots, Q_n\}$ that the privacy and QoS demands cannot be satisfied simultaneously with respect to the common user set $\mathcal{U}$, i.e., $P_{\mathcal{U}}(A_i) < Q_i.p$, the maximize DAQS algorithms (MaxDAQS) is executed, through finding and restraining the queries that most affect privacy protection in the continuous LBS queries,we make user's queries demands met as much as possible. All the queries in the sequence $\mathcal{Q}'$ generated by maximize DAQS algorithms can satisfy the user's privacy and QoS demands.

A. Check Original Queries

According to the contexts and preferences, a user can specify location privacy and QoS demands for each query. After the proxy server gets the user's continuous LBS queries, it will firstly traverse each query and check whether the privacy level provided by the QoS constrained region can meet the corresponding privacy demand, without considering the constraint of common users. If there are such queries, the user needs to adjust his privacy or QoS demand to a reasonable level. This is because that if a query cannot meet the demands, the settings of privacy and QoS demands are obviously unreasonable and the continuous queries with stricter constraint (i.e., common user set) can't meet either.

B. Maximize DAQS

In the step above, each query is considered independent from each other. And after the first step, we ensure that the privacy level provided by the QoS constrained area can meet the user's privacy demand at each query point. However, the eventual privacy level that the continuous LBS queries can provide depends on the common users in all the queries. In other words, the privacy level of a query not only relies on the footprints in its cloaking area formed by this query, but on the number of common users' historical footprints in all the queries. And then, it will lead to such a possibility that the privacy level provided by the continuous LBS queries with respect to the common users' footprints cannot meet the privacy demands of some queries. And our experiments have proved this point. Fig.4 shows that under the restriction of common users's footprints, more than 40% of queries cannot meet their privacy demands, and especially, when the number of continuous LBS queries gets larger (e.g., $\geq$ 30), over 93% cannot meet their privacy demands. That is, although the cloaking area C_i has reached the upper bound of the QoS constrained region (i.e., the maximum size of the QoS constraint region), the privacy provided by the cloaking area still cannot satisfy the user's privacy demand $Q_i.p$ under the restriction of common user set. In this case, if we simply refuse to answer those queries without any processing, the success rate of queries would be very low.

In order to maximize the number of successful queries , we analyze the parameter settings and the relationship among the continuous queries, and find that the essential reasons leading to the query failure at a high proportion are that: (1) in few LBS query regions the footprints are sparse, which limits the common user size in the continuous queries sequence; or (2) the privacy requirements are set unreasonably high at few LBS query points, so the common user set in the cloaking areas cannot meet the privacy requirements. Therefore, we propose a maximum DAQS algorithm(*MaxDAQS*), through identifying and constraining those two kinds of queries, we can obtain a longest LBS queries sequence where the user's privacy and QoS demands can be met simultaneously at each query point.

In the *MaxDAQS* algorithm, we will further check whether the privacy provided by the continuous LBS queries with respect to the common user's footprints can meet the user's demands. If there are any query points where the privacy demands cannot be met, we need to find those two kinds of queries mentioned above and then constrain them.

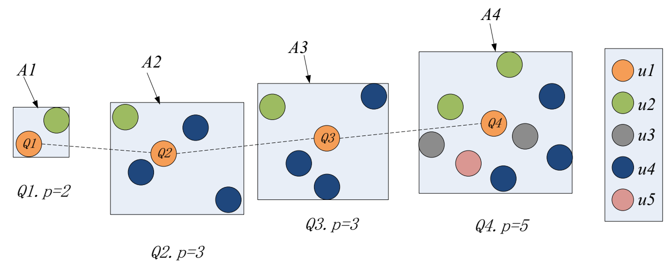


Fig. 3: An example of continuous LBS queries

1) ***Finding the queries which the footprints are sparse in the QoS constrained regions :*** In the continuous LBS queries $\mathcal{Q} = \{Q_1, Q_2, \cdots, Q_n\}$, if there is a query point Q_i where the distribution of footprints in its QoS constrained region A_i is sparse, it may lead the number of common users between A_i and $\mathcal{A} - A_i$ to decrease. And then, the privacy level that the continuous LBS queries can provide will decrease, resulting in that the privacy demands at most query points cannot be satisfied. As shown in Fig.3, there are just two users' footprints at query point Q_1, which will greatly limit the privacy levels of other queries under k-anonymity model.

To find and constrain those queries, we give the Footprint-sparse Query Region Search algorithm. Based on the analysis

Algorithm 1 Footprint-sparse Region Query Search

```
Input: The Continuous LBS Queries $\mathcal{Q} = \{Q_1, Q_2, \cdots, Q_n\}$,
    Anonymity Regions after Check Original Queries $\mathcal{A} = \{A_1, A_2, \cdots, A_n\}$, Grid Table $G$, Footprint Table $F$.
Output: the footprint-sparse query $Q_s$, a sequence of queries
    $\mathcal{S}_1$ whose demands are still unsatisfied after the constraint of $Q_s$.
 1: $temp[\cdot] \leftarrow 0, \mathcal{S}_1 \leftarrow \Phi$;
 2: for each $Q_i \in \mathcal{Q}$ do
 3:     $\mathcal{U} \leftarrow \bigcap_{j=1 \wedge\ j \neq i}^{|\mathcal{Q}|} G(A_j)$;
 4:     $temp[i] \leftarrow \sum_{j=1 \wedge j \neq i}^{|\mathcal{Q}|} P_{\mathcal{U}}(A_j)$;
 5: end for
 6: $Q_s \leftarrow max\{temp[\cdot]\}$;
 7: for each $Q_i \in \mathcal{Q} - Q_s$ do
 8:     if $P_{\mathcal{U}}(A_i) < Q_i.p$ then
 9:         $\mathcal{S}_1 \leftarrow \mathcal{S}_1 + Q_i$;
10:     end if
11: end for
12: Return $[Q_s,\ \mathcal{S}_1]$;
```

above, we know that constraining such kind of queries may significantly increase the number of common users in $\mathcal{A} - A_i$, and thus the privacy levels of the rest queries will be increased saliently. In the algorithm 1, the query in the original LBS queries sequence $\mathcal{Q}$ will be constrained one by one, and the summed privacy level of other queries is calculated. During this process,we select a query Q_s as the constraint point if the summed privacy of the rest queries can reaches the maximum value after constraining it. At the same time, the query set $\mathcal{S}_1$ that still cannot meet the privacy demands after the constraint of Q_s is returned. Alg. 1 depicts the process in detail.

2) ***Finding the queries that the location privacy demands are set too high:*** In the continuous LBS queries, if the user's privacy demand $Q_i.p$ of some a query is set too high, the privacy provided by the cloaking area formed by the common users's footprints hardly meets the privacy demand, that is, $P_{\mathcal{U}}(A_i) < Q_i.p$, resulting in that not all the queries in the continuous LBS can be satisfied. As shown in Fig.3, the privacy demand is 5 at query point Q_4 which is the highest in the continuous LBS queries, and the privacy level based on the common users' footprints hardly meets its demands under k-anonymity model. Obviously, to get the longest DAQS, we also need to find and constrain those queries.

An algorithm of Excessive Privacy-demand Query Search is devised to realize this function. Obviously, such kind of queries are certain to be included in the query set (denoted by $\mathcal{D}$) where the user's privacy demands cannot be met. To get the longest DAQS, the queries in the $\mathcal{D}$ will be constrained one by one, and the number of the queries that cannot meet the user's demands after the constraint is calculated. During this process we select a query Q_l as the constraint point if $|\mathcal{D}|$ reaches the minimum value after constraining it. At the same time, the query set $\mathcal{S}_2$ that still cannot meet the privacy demands after the constraint of Q_l is returned. Alg. 2 depicts the process in detail.

3) ***MaxDAQS Algorithm:*** According to the analysis above, in the continuous LBS queries, we propose the *MaxDAQS* algorithm to maximize the success rate of queries when the

Algorithm 2 Excessive Privacy-demand Query Search

```
Input: The Continuous LBS Queries $\mathcal{Q} = \{Q_1, Q_2, \cdots, Q_n\}$, Anonymity Regions after Check Original Queries $\mathcal{A} = \{A_1, A_2, \cdots, A_n\}$, Grid Table $G$, Footprint Table $F$
Output: the excessive privacy-demand query $Q_l$, a sequence of queries $\mathcal{S}_2$ whose demands are still unsatisfied after the constraint of $Q_l$.
1: $temp[\cdot] \leftarrow 0, \mathcal{S}_2 \leftarrow \Phi$;
2: Compute common user set $\mathcal{U} \leftarrow \{u_1, u_2, \cdots\}$ in $\mathcal{A}$;
3: for each $A_i \in A$ do
4:     if $P_{\mathcal{U}}(A_i) < Q_i.p$ then
5:         $\mathcal{D} \leftarrow \mathcal{D} + Q_i$;
6:     end if
7: end for
8: for each $Q_i \in \mathcal{D}$ do
9:     $\mathcal{U} \leftarrow \bigcap_{j=1 \wedge j \neq i}^{|\mathcal{Q}|} G(A_j)$;
10:    for each $Q_j \in \mathcal{D} - Q_i$ do
11:        if $P_{\mathcal{U}}(A_j) < Q_j.p$ then
12:            $temp[i] \leftarrow temp[i] + 1$;
13:        end if
14:    end for
15: end for
16: $Q_l \leftarrow min\{temp[\cdot]\}$
17: for each $Q_i \in \mathcal{Q} - Q_l$ do
18:     if $P_{\mathcal{U}}(A_i) < Q_i.p$ then
19:         $\mathcal{S}_2 \leftarrow \mathcal{S}_2 + Q_i$;
20:     end if
21: end for
22: Return $[Q_l,\ \mathcal{S}_2]$
```

user's location privacy and QoS demand cannot be met simultaneously. By calling *Footprint-sparse Region Query Search* and *Excessive Privacy-demand Query Search* algorithms, the *MaxDAQS* first finds those two kinds of queries Q_s and Q_l, and gets the corresponding query set $\mathcal{S}_1$ and $\mathcal{S}_2$ that the queries still cannot meet the demands after constraining them respectively. Then, by comparing the size of $\mathcal{S}_1$ and $\mathcal{S}_2$, the *MaxDAQS* selects Q_s or Q_l to constrain, until all the demands of the rest queries can be satisfied. The detailed *MaxDAQS* is described by Alg.3.

V. Evaluation and Discussion

We experimentally evaluate the proposed model and algorithms. The comparison of the anonymity service success rate before and after the $MaxDAQS$ is made.

A. Dataset

All algorithms are implemented in C++ programming language, and the experiments are performed in a PC with a Intel(R) Core(TM) 2 Quad 3.0GHz processor, 2GB RAM and Windows XP operation system. All the experiment data are generated by the Network-based Generator of Moving Objects [12] proposed and implemented by Brinkhoff. The generator is used to generate mobile users and simulate their movement on the real road map of Oldenberg, Germany, a city about $16km \times 16km$ which is splited into 25600 grids (each of them is $100m \times 100m$). About 35K records of historical footprints are generated by the generator, and they are used as the historical footprint dataset in our experiments. A set of

Algorithm 3 MaxDAQS

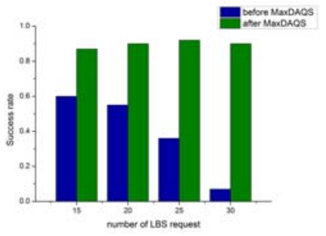
```
Input: Cloaking Areas $\mathcal{A}$ after Pre-processing, LBS Queries Sequence $\mathcal{Q} = \{Q_1, Q_2, \cdots, Q_n\}$, Grid Table $G$, Footprint Table $F$.
Output: DAQS $\mathcal{Q}'$
1: $\mathcal{U} \leftarrow \Phi, \mathcal{D} \leftarrow \Phi$;
2: Compute common user set $\mathcal{U} \leftarrow \{u_1, u_2, \cdots\}$ in $\mathcal{A}$;
3: for each $A_i \in \mathcal{A}$ do
4:     if $P_{\mathcal{U}}(A_i) < Q_i.p$ then
5:         $\mathcal{D} \leftarrow \mathcal{D} + Q_i$; // If the privacy provided by the cloaking area cannot meet the demand, add this query to the set $\mathcal{D}$.
6:     end if
7: end for
8: while $\sim IsEmpty(\mathcal{D})$ do
9:     $[Q_s, \mathcal{S}_1] \leftarrow Footprint - sparse\ Area\ Search(\mathcal{Q}, \mathcal{A}, G, F)$;
10:    $[Q_l, \mathcal{S}_2] \leftarrow Excessive\ Privacy - demand\ Area\ Search(\mathcal{Q}, \mathcal{A}, G, F)$
11:    if $|\mathcal{S}_1| \leq |\mathcal{S}_2|$ then
12:        $\mathcal{Q} \leftarrow \mathcal{Q} - Q_s$; $\mathcal{D} \leftarrow \mathcal{S}_1$;
13:    else
14:        $\mathcal{Q} \leftarrow \mathcal{Q} - Q_l$; $\mathcal{D} \leftarrow \mathcal{S}_2$;
15:    end if
16: end while
17: Return $\mathcal{Q}' \leftarrow \mathcal{Q}$
```

continuous LBS queries are generated On this dataset and each record contains a user's ID, a timestamp and a location.

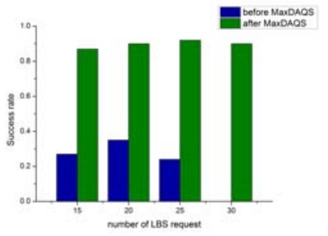
B. Comparison of anonymity service success rate before and after MaxDAQS

In this experiment, for each query, the QoS demand q_x and q_y are selected randomly from the range [0.625, 10]% of the space ($16km \times 16km$) (that is, from 160 grids to 2560 grids). The privacy demands are randomly chosen from a average value from 10 to 50 for *k-anonymity model* and from 3 to 10 for *entropy-based model*. To test the effectiveness of $MaxDAQS$, the anonymity service success rate before and after $MaxDAQS$ are compared with the number of LBS queries seting as 15, 20, 25 and 30 respectively.

After the introduction of the QoS in the continuous LBS queries, the cloaking areas are initially set to the upper bound of the QoS constraint regions (i.e., the maximum size of QoS constraint regions). If we do not use the $MaxDAQS$, when the privacy level provided by the common users's historical footprints in all the QoS constraint regions cannot meet the user's privacy demand at some a query point, this query will be refused simply. And the anonymity service success rate is defined as the ratio of the number of queries that can be provided anonymity service to the number of all the queries . The results are shown in Fig.4(a) and 4(b), from which it can be seen that more than 40% of queries cannot be provided service and will be restrained if $MaxDAQS$ is not used. While with $MaxDAQS$ algorithm, over 87% of the queries can be provided anonymity service. Especially, when the number of continuous LBS queries gets larger (e.g., 30), relatively big proportion (93% in Fig.4(a)) and even all the queries (100% in Fig.4(b) cannot meet the user's privacy and



(a) Success rate(k-anonymity)



(b) Success rate((entropy))

Fig. 4: Comparison of success rate before and after $MaxDAQS$

QoS demands simultaneously, that is, those queries will be denied. While even in this situation, our scheme can guarantee 90% anonymity service success rate. From the results, we know that: (1) The introduction of the QoS demand will result in that in the continuous LBS queries relatively large proportion queries cannot meet the user's privacy and QoS demands simultaneously. (2) Through finding and refusing a small part of queries, $MaxDAQS$ can effectively improve the anonymity service success rate. That is, with $MaxDAQS$ a large proportion of queries can be provided anonymity service in the continuous LBS queries.

C. Discussion

Performance: In our proposal, in order to ensure the security of continuous LBS queries, only the common users are considered for the construction of cloaking areas. To provide real location privacy protection for the continuous LBS queries, the proxy server choose i users near to the querier at the first query point, and all the cloaking areas of the subsequent LBS queries should include those i users. Since the i users may move in different directions, future cloaking areas will become increasingly large, and eventually unacceptable for any LBS services. In order to ensure the QoS of continuous LBS queries, the user's original route should be got first to find the common users with similar route for the construction of the cloaking areas. Same as *L2P2* [7], the proposed model takes the historical footprints and n LBS queries as the inputs. The user sends all his possible LBS queries along the route to the proxy server before his itinerary which processes the queries in offline fashion and gets the cloaking areas satisfying the user's privacy and QoS demands. Because the process is pretreated, its latency will not affect the queries during the user travels.

VI. Conclusions

The existing location privacy protection schemes for continuous LBS queries does not take into consideration of a user's personalized QoS demands, and while we find that the simple introduction of QoS demands results in that a considerable proportion of LBS queries cannot be provided service. To solve this issue, we propose demands-aware location privacy protection model, allowing the user to customize his personalized location privacy and QoS demands for each query in the continuous LBS queries. Observations indicate that there are two reasons leading that the formed cloaking areas based on the common users' historical footprints cannot meet the user's personalized requirements. The first and the most important one is that there are some queries where the footprints are too sparse; and the second one is that in some queries the the privacy requirements are set too high. Based on these observations, we propose the $MaxDAQS$ algorithm, through constraining those two types of queries, the longest LBS queries sequence where each query can meet the user's location privacy and QoS demands is obtained. In such a method, the LBS service that can meet the user's demands is provided as much as possible. Extensive experiments indicate that the proposed DALP model can achieve good QoS with high anonymity service success rate in the condition of various location privacy and QoS demands.

References

[1] K. Virrantaus, J. Markkula, A. Garmash, V. Terziyan, J. Veijalainen, A. Katanosov, and H. Tirri, "Developing gis-supported location-based services," in *IEEE WISE'01*, 2001, pp. 66–75.
[2] C. Y. Chow and M. Mokbel, "Enabling private continuous queries for revealed user locations," in *SSTD'07*, 2007, pp. 258–275.
[3] A. Pingley, N. Zhang, X. Fu, H.-A. Choi, S. Subramaniam, and W. Zhao., "Protection of query privacy for continuous location based services," in *INFOCOM'11*, 2011, pp. 1710–1718.
[4] S. Gao, J. Ma, W. Shi, and G. Zhan, "Towards location and trajectory privacy protection in participatory sensing," in *MobiCASE'11*. Springer, 2011, pp. 381–386.
[5] T. Xu and Y. Cai, "Exploring historical location data for anonymity preservation in location-based services," in *IEEE INFOCOM'08*, 2008, pp. 547–555.
[6] T. Xu and Y. Cai, "Feeling-based location privacy protection for location-based services," in *ACM CCS'09*, 2009, pp. 348–357.
[7] Y. Wang, D. Xu, X. He, C. Zhang, F. Li, and B. Xu, "L2p2: Location-aware location privacy protection for location-based services," in *INFOCOM'12*, 2012, pp. 1996–2004.
[8] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *ACM MobiSys'03*, 2003, pp. 31–42.
[9] M. F. Mokbel, C. Y. Chow, and W. G. Aref, "The new casper: query processing for location services without compromising privacy," in *VLDB'06*, 2006, pp. 763–774.
[10] B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity: Architecture and algorithms," *IEEE Trans on Mobile Computing*, vol. 7, no. 1, pp. 1–18, 2008.
[11] V. Zeimpekis, G. M. Giaglis, and G. Lekakos, "A taxonomy of indoor and outdoor positioning techniques for mobile location services," *ACM SIGecom Exch.*, vol. 3, no. 4, pp. 19–27, 2002.
[12] T. Brinkhoff, "A framework for generating network-based moving objects," *GeoInformatica*, vol. 6, no. 2, pp. 153–180, 2002.