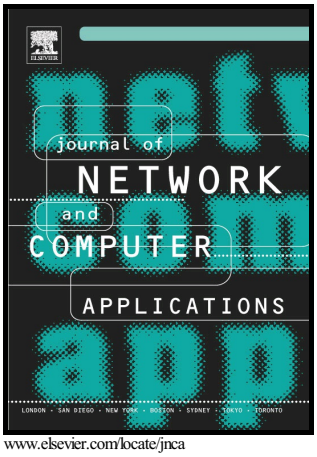
Efficient Location Privacy Algorithm for Internet of Things (IoT) Services and Applications

Gang Sun, Victor Chang, Muthu Ramachandran, Zhili Sun, Gangmin Li, Hongfang Yu, Dan Liao



www.elsevier.com/locate/jnca

PII: S1084-8045(16)30242-9
DOI: http://dx.doi.org/10.1016/j.jnca.2016.10.011
Reference: YJNCA1738

To appear in: *Journal of Network and Computer Applications*

Received date: 29 September 2016
Accepted date: 18 October 2016

Cite this article as: Gang Sun, Victor Chang, Muthu Ramachandran, Zhili Sun Gangmin Li, Hongfang Yu and Dan Liao, Efficient Location Privacy Algorithm for Internet of Things (IoT) Services and Applications, *Journal of Network and Computer Applications,* http://dx.doi.org/10.1016/j.jnca.2016.10.011

privacy. The *k*-anonymity model is an important technique to protect user's location privacy in LBS, it can ensure that a user is identified with a probability of (only) 1/*k*. To achieve *k*-anonymity in LBS, a user first submits a query to a centralized location anonymizer. Then, the location anonymizer enlarges the queried location into a bigger Cloaking Region (CR) for covering many other users (at least *k*-1) geographically distributed. Finally, the location anonymizer sends the query to the LBS server. However, since this technique relies heavily on the location anonymizer, there will be a single point of failure. Moreover, since location anonymizer must process all users' queries, the location anonymizer may become a performance bottleneck.

To address this problem, the "dummy location" has been proposed and used to protect user's location privacy, which does not need any third party service. Dummy location is part of our emerging IoT service. Existing approaches [23-25] try to effectively generate dummy locations which cannot be distinguished by the LBS server. However, these approaches do not consider the side information [26], i.e., users' query probability related to location and time, or information related to the semantics of the query such as the gender and social status of the user. If the side information is obtained by an adversary, incredible chosen dummy locations such as lakes, swamps etc. may be easily filtered out by the adversary. Therefore, these algorithms for dummy locations generation cannot effectively achieve *k*-anonymity. The authors in [27] proposed a dummy location selection (DLS) algorithm for location privacy preservation, which considers the side information that may be exploited by attackers. However, the computational cost (i.e., time complexity) of the DLS algorithm is very high. As a result, how to select dummy locations is still a challenge, particularly for a data-driven IoT service whereby more complexity can be involved with volume, velocity, variety, veracity and validity. Locations based service is one of the major application for a data-driven IoT service and needs to pretest highly sensitive data. LBS based cloud applications needs to collect, process, and analyze geo-position data or send the required geo-locations instantly for millions of users in real-time. LBS is useful in many cases to find a convenient local place in an unfamiliar territory for socialization. However, LBS based applications also come with risk of revealing personal information and data for tracking. Despite personal identification may be hidden in the LBS services, the geo-localized history of user requests can act as a quasi-identifier, which can reveal about individuals' details and their locations. Hence, we need efficient strategies to hide this quasi-identification using dummy LBS data.

In this paper, we first analyze the well-known DLS algorithm, which provides a location privacy preservation for a data-driven IoT service of users' queries in LBS. Then, we discuss an attack algorithm for DLS (ADLS) with a goal to identify the user's real location out from the data-driven IoT service of chosen dummy locations in LBS. We also design a dummy location based privacy (DLP) algorithm for location privacy preservation in LBS. Different from existing algorithms, the DLP makes a tradeoff between computational cost (i.e., time complexity) and the privacy requirements of users. The main contributions of this research are as follows:

- We analyze the current DLS algorithm, and attack algorithm for DLS (ADLS), for the data-driven IoT service of chosen dummy locations.
- We propose an entropy-based DLP algorithm, by selecting dummy locations in a greedy manner for a tradeoff between computational cost (i.e., time complexity) and the privacy requirements for the data-driven IoT service in LBS.
- We analyze the performance on privacy preservation of our proposed DLP algorithm against the *colluding attack* and *inference attack*; and use the attack algorithm to test robustness of our data-driven IoT service.
- We demonstrate that the ADLS algorithm has a high probability of query recognition for the DLS algorithm through simulations. When compared with the DLS algorithm, the results show that the DLP algorithm can efficiently reduce the computational cost (i.e., time complexity) while providing the same privacy level as the DLS algorithm. Moreover, the DLP algorithm has a lower probability of query recognition (i.e., lower probability of losing users' privacy) compared to the DLS algorithm.

The remainder of this paper is organized as follows. Section 2 reviews the related work. Section 3 introduces the preliminaries and the system model. Section 4 gives the detailed analysis on the DLS algorithm. Section 5 presents the ADLS algorithm for identifying the user real location and evaluate its performance. Section 6 presents the detailed descriptions on our DLP algorithm and simulation results. Section 7 gives the discussions and explains how our contributions are relevant to the data-driven IoT service. Section 8 concludes this paper.

2. RELATED WORK

In this section, we describe recent researches related to privacy protection methods in location based services of IoT.

2.1 Privacy-preserving for IoT

Several recent researches have been conducted for the privacy-preserving for the IoT based services [28-36]. In order to handle the massive amount of data, the most convincing solution is the federation of the IoT and cloud computing. Henze, et al. presented an user-driven privacy enforcement appraoch for cloud-based services in the IoT, which focuses on privacy preserving for individual end-users [28]. The authors in [29] proposed PAgIoT, a Privacy preserving Aggregation protocol

suitable for IoT settings and enables multi-attribute aggregation for groups of entities while allowing for privacy-preserving value correlation. A lightweight privacy-preserving trust model had been designed for minimizing privacy loss in the presence of untrusted service providers, so that providers can be prevented from disclosing information to third parties for secondary uses [30]. A conditional privacy-preserving authentication with access linkability (CPAL) for roaming service, to provide universal secure roaming service and multilevel privacy preservation [31]. The authors in [32] estimated the cost of breaking public key crypto systems when the adversary is limited by the available resources and time and presentd the trade-off between the processing load for an IoT node versus the desired time span of privacy protection. Jin, et al., presented a framework for the realization of smart cities through the Internet of Things (IoT), which encompasses the complete urban information system and forms a transformational part of the existing cyber-physical system [33]. The authors in [34] proposed a privacy-by-design (PbD) framework that can guide software engineers to systematically assess the privacy capabilities of IoT applications and middleware platforms, thus the proposed PbD framework can also be used to design new IoT platforms.

2.2 Location Anonymization Approach for LBS

Location anonymization approach is one of most important techniques to protect location privacy, which attempts to make user's location indistinguishable from a certain number of other users. Commonly used techniques include spatial-temporal cloaking and location obfuscation. *k*-anonymity is an important technique for location anonymization, which relies on a centralized location anonymizer to enlarge a user's queried location into a bigger Cloaking Region (CR) for covering many other users. A personalized *k*-anonymity model is proposed in [37]. The model enables a user to have different privacy requirements in different contexts, and different users can require different levels of privacy in the same context. In the proposed model in [37], the trusted anonymization server runs an efficient message perturbation engine, which performs location anonymization considering the trade-off between location privacy and quality of service (QoS). A cloaking algorithm based on *k*-anonymity and *l*-diversity has been proposed in [38]. When constructing a cloaking region, it ensures that a cloaking region has at least *k* vehicles (*k*-anonymity) and *l* road segments (*l*-diversity), which can effectively protect user's location privacy. The authors in [39] studied the problem that how to protect the location privacy under various privacy threats, and proposed a location privacy framework uses *k*-anonymization and pseudo-anonymization methods to provide efficient location privacy preservation. A weighted adjacency graph based *k*-anonymous cloaking technique is proposed in [40], which can support *k*-nearest neighbor queries without revealing private information of the query initiator. The algorithm in [40] not only can ensure user privacy protection, but also reduce bandwidth usages. The concept of mix zones is first proposed in [41]. A mix zone is referred to a spatial region in which none of users has registered any application callback. The authors in [42] allowed users to exchange their pseudonyms when they meet in a mix zone, which ensures a user avoid using a long-term pseudonym. Thus, the relationship between user pseudonyms and locations can be broken though exchanging pseudonyms.

2.3 Policy or Cryptography Primitive based Approach

Policy and cryptography primitive based approaches [43-45] protect user privacy by using encryption techniques. The authors in [46] propose a privacy preserving framework (PLAM) for local-area mobile social networks. The PLAM framework not only employs a privacy-preserving request aggregation protocol with *k*-anonymity and *l*-diversity properties to keep user's preference privacy without adopting a trusted anonymizer server when querying location-based service, but also integrates unlinkable pseudo-ID technique to achieve users' identity privacy and location privacy. The PLAM framework can not only satisfy the desirable privacy requirements but also resist outside attacks on source authentication, data integrity and availability. For preserving user's privacy, the authors in [47] proposed a dynamic pseudo-ID scheme, where different pseudo-IDs are adopted in different queries in order to unlink the correlation between user's real identity and trajectory. In [48], the authors propose a fine-grained privacy preserving LBS framework (FINE) for mobile devices. The FINE framework not only employs a ciphertext-policy anonymous attribute based encryption technique to achieve fine-grained access control, location privacy, confidentiality of the LBS data and its access rule, and accurate LBS query result without involving any trusted third party, but also integrates the transformation key and proxy re-encryption to migrate most of computation intensive tasks from LBS provider and users to cloud server. In [49], the authors study the *k* nearest neighbor (kNN) queries where mobile users query the LBS provider about *k* nearest points of interests (POIs) on the basis of their current location, and then propose a solution built on the Paillier public-key cryptosystem for preserving the location privacy and data privacy in kNN queries of mobile users. The authors in [50] design a private block retrieval protocol, and propose a secure and efficient location based service system. In the proposed system, users can retrieve information of interest associated with the current location without leaking their location information to the service provider.

2.4 Dummy Location Selection for IoT

Dummy location approach focuses on selecting dummy locations for users in order to protect users' location privacy. In [25], the authors first study the behaviors of self-interested users in the LBS system from a game-theoretic perspective. The

work then formulates two Bayesian game models in both static and timing-aware contexts, and analyzes the existence and properties of the Bayesian Nash Equilibrium for the two models. A Dummy-Location Selection (DLS) algorithm is proposed in [27] to achieve *k*-anonymity for users using LBS. The DLS algorithm selects dummy locations considering that the side information may be exploited by adversaries, which is based on the entropy metric [51]. To make sure that the selected dummy locations are spreaded as far as possible, the authors in [27] also propose an enhanced-DLS algorithm, which can enlarge the cloaking region while keeping similar privacy level as the DLS algorithm. The authors in [52] propose two dummy generation methods: circle-based and grid-based, which take into account privacy area requirements. In [53], the authors proposed two dummy based solutions to achieve *k*-anonymity for privacy-area aware users in LBS with considering that side information may be exploited by adversaries.

However, most of these existing approaches have not considered the side information that may be exploited by attackers when selecting dummy locations in IoT. Even if some approaches have taken into acount the side information, but the computational costs (i.e., time complexities) of them are very high. Therefore, how to efficiently select dummy locations in IoT still remains a challenge, and our proposal will be presented between Section 3 and 6.

3. PRELIMINARIES

In this section, we describe the main basic concepts and the system model.

3.1 Side Information

As mentioned in previous section, the side information [26] may be query probability of users related to location and time, or information related to the semantics of the query such as the gender and social status of the user. In this paper, the side information is considered to be the query probability of users related to location, called query probability. A particular user's query probability at a certain location can be denoted by the ratio of the number of current location queries to the number of total queries of all locations, as shown in Equation (1).

$$q_i = \frac{number\ of\ queries\ in\ location\ i}{number\ of\ queries\ in\ all\ locations} \tag{1}$$

Generally, users can get two kinds of side information from a system: *partial information* and *global information*. Partial information denotes the information collected by other users, for example, a particular user may know the query probabilities related to some locations. Since the LBS server can receive the LBS queries of all users, the LBS server can obtain the global information (i.e., the query probabilities related to all locations). For a particular user, it's necessary to design an optimal strategy to select dummy locations for protecting his/her location privacy under the condition of knowing the global information. In this paper, the LBS server is responsible for disseminating and updating the global side information so that users can get this information from a well-known place (e.g., local database of LBS application).

3.2 Entropy-based Privacy Metric

In this work, the degree of privacy is measured by the entropy. It can be seen as the uncertainty in identifying a user's real location out from the chosen dummy locations [51]. When calculating the entropy, each dummy location should have a probability, which can be the history query probability of users related to location. We use p_i to denote the historic query probability of users related to location *i*. According to the set of dummy locations and the historic query probabilities, we can define the entropy *H* of a user as in Equation (2).

$$H = -\sum_{i=1}^{k} q_i \log_2 q_i \tag{2}$$

where $q_i = p_i \Big/ \sum_{i=1}^{k} p_i$, is the normalized query probability of location *i*; and the sum of all p_i is equal to 1.

Since the greater the entropy the higher the uncertainty in identifying the user's real location from the dummy locations set, our goal is to obtain enough entropy. In particular, when all of the *k* dummy locations have the same historical query probability, we can achieve the maximum entropy $H_{max} = \log_2 k$.

3.3 Service based System Model for IoT

More and more mobile technologies support smart location based services including smart phones, manufacturing industries, smart home technologies, and smart cities. LBS is the key for achieving our future aim of smart living. The system architecture model shown in Figure 1 illustrates our approach towards service-oriented design and implementation for the proposed algorithm.

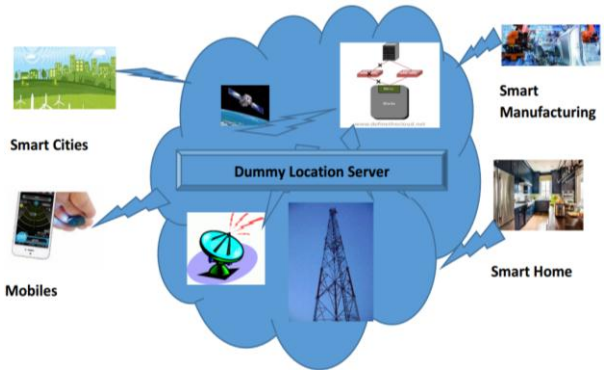


Fig.1: Service based System Model for IoT

We design our model for LBS based on the system architecture in [24]. The system mainly consists of two parties: the *LBS server* and *LBS users* with mobile devices.

1) LBS server: The LBS server can be a service provider, which not only stores all kinds of service databases, but also can update the service data and provide users with various services. In our system, the LBS server is responsible to receive service queries from users, search for requested service data in the database, and reply with the search results back to the users. In addition, the LBS server is able to obtain the global information based on queries of all users at all locations, which can be the historical query probabilities of users related to all locations. Moreover, the LBS server is responsible for disseminating and updating the global side information so that users can get this information from a well-known place (e.g., local database of LBS application).

2) LBS users: The system typically consists of users who are equipped with mobile devices (e.g., smart phones or tablets), with built-in GPS modules that can be used to obtain user's location data. Due to the rapid development of mobile devices and social networks, a variety of LBS applications can be accessible for users. If users want to get services from LBS servers, they need to send queries to LBS server, which include their identity, location information, interests, and the query range (e.g., 1000m). In order to protect user's location privacy, user's location information not only includes user's real location, but also includes many other dummy locations.

4. ANALYSIS OF THE DLS ALGORITHM

4.1 Review the DLS Algorithm

The main purpose of Dummy-Location Selection (DLS) algorithm [27] is to generate a set of realistic dummy locations to protect user's location privacy. Given the degree of anonymity k, the DLS algorithm needs to select other k-1 dummy locations based on the side information. The following shows the 5 steps how the DLS algorithm addresses this problem:

(*i*) In the first step, a particular user needs to determine the degree of anonymity k.

(*ii*)Then, the algorithm reads all of the obtained query probabilities and then sorts the query probabilities of all locations in ascending order.

(*iii*) In the sorted list, the algorithm needs to choose $2k$ candidate locations, whose history query probabilities are similar to the user's real location. In the $2k$ candidate locations, it randomly selects k –1 locations. Then, it derives m sets, each set contains k locations. For each set, one location is user's real location and the other k−1 locations are randomly chosen from the $2k$ candidates. The entropy for the $j^{th}(j \in [1,m])$ set can be calculated according to Equation (2) as shown in Section 3.

(*iv*) Finally, the algorithm has to determine an optimal location set with the biggest entropy to effectively achieve k-anonymity for the user.

4.2 Preparations for Performance Analysis

Table 1: Summary of key notations

Notation	Meaning
N	Number of all locations.
k	The privacy level requirement of user.
$P[N]$	The historical query probabilities in all locations.
m	Number of randomly selecting k–1 locations from $2k$ locations, i.e., $m = C_{2k}^{k-1}$.
P_i	The historical query probability at location i.
L_{real}	The real location of user.
$P_i[2k]$	The chosen $2k$ candidates at location i, where k candidates are left before L_{real} and the other k candidates are right after L_{real} in the sorted list.
$C_i[k]$	The chosen optimal location set at location i.
k'	The number of locations which have the same historical query probability as L_{real} in P_i.

Let the historical query probabilities of all locations $P = [p_1, p_2,\ldots,p_N]$, the chosen $2k$ candidate locations at location i as $P_i = \{p_{i,1}, p_{i,2},\ldots, p_{i,2k}\}$and the chosen $2k$ candidate locations at location j as $P_j = \{p_{j,1}, p_{j,2},\ldots, p_{j,2k}\}$. Then, let $P_{ii} = P_i \cup \{p_i\}$, and $P_{ij} = P_{ii} \cap P_{jj}$. Let M denote the size of set P_{ij}. We define P_{ij} as follows.

$$P_{ij} = \begin{cases} \{p_{(1)}, p_{(2)},..., p_{(M)}\}, M > 0 \\ \varnothing\ , M = 0 \end{cases} \tag{3}$$

Theorem 1: Under the condition of $m = C_{2k}^{k-1}$, for $\forall\ i, j \in [1, N]$, $C_i \neq C_j$ $(i \neq j)$, set P must satisfy the following conditions:

(*i*) $\forall\ i \neq j$, $p_i \neq p_j$, i.e., each location has a unique historical query probability.

(*ii*) $0 \leq M \leq 2k$, $\forall\ i, j$ $(i \neq j)$, $P_{ij} \cap C_i \neq C_i$ or $P_{ij} \cap C_j \neq C_j$; that is to say when $0 \leq M \leq 2k$, $\forall\ i, j$ $(i \neq j)$, the chosen optimal location set at location i or location j is not included in the intersection of the chosen $2k$ candidate locations at location i and the chosen $2k$ candidate locations at location j.

Proof:

Adequacy:

(1) We first prove that set P must satisfy condition (*i*).

We assume that set P does not satisfy condition (*i*), and then $\exists\ i, j \in [1, N]$, $p_i = p_j$ $(i \neq j)$. Thus, P_i and P_j will be the same according to the step (*ii*) in DLS algorithm.

When $k' \geq k + 1$, although C_i may not be the same as C_j according to the step (*iii*) and (*iv*) in DLS algorithm, it is possible that $C_i = C_j$. However, according to our assumption that C_i cannot be the same as C_j. Thus, set P must satisfy condition (*i*).

When $k' \leq k$, C_i must be the same as C_j according to steps (*iii*) and (*iv*) in DLS algorithm. However, according to our assumption, C_i cannot be the same as C_j. Thus, set P must satisfy condition (*i*).

(2) We then prove that set P must satisfy condition (*ii*) after satisfying the condition (*i*).

We assume that set P satisfies condition (*i*), but does not satisfy condition (*ii*). Thus, $\exists\ i, j \in [1, N]$, $P_{ij} \cap C_i = C_i$ and $P_{ij} \cap C_j = C_j$. Since C_i and C_j both are the optimal location set in set P_{ij}, i.e., $C_i = C_j$. However, according to our assumption, set C_i cannot be the same as set C_j. Thus, set P must satisfy condition (*i*) and condition (*ii*).

Necessity:

According to condition (*i*), we can get that for $\forall\ i \neq j$, $P_i \neq P_j$. Then, we discuss the condition (*ii*) as follows.

(1) $0 \leq M \leq 2k$, $\forall\ i \neq j$, $P_{ij} \cap C_i \neq C_i$ and $P_{ij} \cap C_j \neq C_j$. For this situation, set C_i must include the location from set P_i-P_{ij}, which does not belong to set C_j. Moreover, set C_j must also include the location from P_j-P_{ij}, which does not belong to set C_i. Thus, for $\forall\ i \neq j$, $C_i \neq C_j$.

(2) $0 \leq M \leq 2k$, $\forall\ i \neq j$, $P_{ij} \cap C_i \neq C_i$ and $P_{ij} \cap C_j = C_j$. For this situation, set C_i must include the location from set P_i-P_{ij}, which does not belong to set C_j. Therefore, for $\forall\ i \neq j$, $C_i \neq C_j$.

(3) $0 \le M \le 2k$, $\forall i \ne j$, $P_{ij} \cap C_i = C_i$ and $P_{ij} \cap C_j \ne C_j$. For this situation, set C_j must include the location from set P_j-P_{ij}, which does not belong to set C_i. Thus, for $\forall i \ne j$, $C_i \ne C_j$.

Therefore, we can conclude that for $\forall i \ne j$, $C_i \ne C_j$ when set P satisfies conditions (*i*) and (*ii*).

4.3 Performance Analysis for DLS Algorithm

Based on step (*iii*) in the DLS algorithm, we can see that the greater of value of m the higher the computational cost of the DLS algorithm is. We also can see that different values of m may result in different optimal location sets in DLS algorithm, and the DLS algorithm can obtain the optimal location set when $m = C_{2k}^{k-1}$. We analyze the performance of the DLS algorithm when $m = C_{2k}^{k-1}$ as follows.

(1) $\exists i, j \in [1, N]$, $p_i = p_j (i \ne j)$ in set P. We assume that a particular user is at location i, and the number of locations whose query probabilities are the same as that of the user's real location in the chosen candidate locations is denoted by k'. Since $p_i = p_j$, set P_i the user selects at location i is the same as set P_j the user selects at location j in DLS algorithm. We then discuss the performance of the DLS algorithm in the following situations. When $1 \le k' \le k$-1, set C_i is the same as set C_j in DLS algorithm under the condition $m = C_{2k}^{k-1}$. In this situation, although the LBS server can infer the probability for a user to submit a LBS query, the server cannot know the user's real location. This is because there are other locations whose query probabilities are the same as that of the user's real location. Moreover, the larger k' is, the better the performance of the DLS algorithm is. When $k' \ge k$, C_i may be different from C_j. The reason is that randomly selecting k-1 locations from the k' locations whose query probabilities are the same as p_i may be the optimal location set. In this situation, since each location has the same query probability, the DLS algorithm achieves the best performance.

(2) $\forall i, j \in [1, N]$, $p_i \ne p_j (i \ne j)$ in set P. We assume that a particular user is at location i. Since $p_i \ne p_j$, set P_i the user selects at location i must be different from the set P_j user selects at location j. However, when $M \ge k$-1, C_i may be the same as C_j, that is to say the chosen optimal location set at location i is likely to be the same as the chosen optimal location set at location j. In this situation, although the LBS server may try to infer which location is most likely to select this location set, the server may make a incorrect decision. The reason is that the optimal location sets chosen by the user in other locations are the same as that of the user's real location. Moreover, the larger the number of locations whose chosen optimal location sets are the same as the that of user's real location is, the better the performance of the DLS algorithm is. However, once there is no location whose chosen optimal location set is the same as other locations in set P, the DLS algorithm would have bad performance.

5. ADLS ALGORITHM

In this section, we first introduce an attack model and related theories, then give detailed descriptions of ADLS algorithm and the performance evaluations.

5.1 Attack Model

In order to protect location privacy, the dummy location generation algorithm is used for generating some dummy locations. Thus, the users' location information not only includes users' real location, but also includes other chosen dummy locations [52]. The goal of the adversary is to obtain the user's real location from the user's location information. Since adversaries can compromise the LBS server and obtain all the information that the LBS server knows and holds. Thus, in this work, we assume that the LBS server is the adversary. Note that, LBS server is able to obtain global side information and monitor the current queries being sent from users. Furthermore, the LBS server can obtain the historic data of a particular user as well as the current situation and information. Additionally, the mechanisms used for location privacy protection in the system are also known by the LBS server.

5.2 Related Theories

Let set $P = [p_1, p_2, \ldots, p_n]$, where $0 < p_i < 1 (1 \le i \le n)$. We define function $H(P, p_{n+1})$ in Equation (4).

$$H(P, p_{n+1}) = -\sum_{p_i \in P} \frac{p_i}{\sum_{p_i \in P} p_i + p_{n+1}} \ln \frac{p_i}{\sum_{p_i \in P} p_i + p_{n+1}} - \frac{p_{n+1}}{\sum_{p_i \in P} p_i + p_{n+1}} \ln \frac{p_{n+1}}{\sum_{p_i \in P} p_i + p_{n+1}} \tag{4}$$

$$D(P, p_{n+1}) = \frac{dH(P, p_{n+1})}{dp_{n+1}} = \frac{\sum_{p_i \in P} p_i \ln p_i - \sum_{p_i \in P} p_i \ln p_{n+1}}{(\sum_{p_i \in P} p_i + p_{n+1})^2} \quad (5)$$

In Equation (4), function $H(P, p_{n+1})$ varies with p_{n+1}, where $0< p_{n+1} <1$. In order to get the maximum value of $H(P, p_{n+1})$, we first calculate the derivative of function $H(P, p_{n+1})$, denoted by function $D(P, p_{n+1})$ as shown in Equation (5). Then, let function $D(P, p_{n+1})$ be zero to get the value of p_{n+1} as shown in Equation (6). Finally, we can get the extreme points of function $H(P, p_{n+1})$. From Equation (6), we can know that function $H(P, p_{n+1})$ has a unique extreme point.

$$\begin{aligned} &\frac{\sum_{p_i \in P} p_i \ln p_i - \sum_{p_i \in P} p_i \ln p_{n+1}}{(\sum_{p_i \in P} p_i + p_{n+1})^2} = 0 \Rightarrow \\ &\sum_{p_i \in P} p_i \ln p_i - \sum_{p_i \in P} p_i \ln p_{n+1} = 0 \Rightarrow \\ &p_{n+1} = exp(\frac{\sum_{p_i \in P} p_i \ln p_i}{\sum_{p_i \in P} p_i}) \end{aligned} \quad (6)$$

Let $\bar{p}_{n+1} = exp\left(\sum_{i=1}^{n} p_i \ln p_i \Big/ \sum_{i=1}^{n} p_i\right)$.

When $p_{n+1} < \bar{p}_{n+1}$, the value of function $D(P, p_{n+1})$ is greater than zero, and the value of function $H(P, p_{n+1})$ increases with the growth of p_{n+1}. When $p_{n+1} > \bar{p}_{n+1}$ the value of function $D(P, p_{n+1})$is less than zero, the value of function $H(P, p_{n+1})$ decreases with the growth of p_{n+1}. Thus, we have that the maximum point of function $H(P, p_{n+1})$ is $p_{n+1} = \bar{p}_{n+1}$. We can obtain the range of $\bar{p}_{n+1}$ by Equation (7). From Equation (7), we can see that the value of p_{n+1} is not greater than the maximum of set P, and also not less than the minimum of set P. In our ADLS algorithm, we can use this property to select dummy locations.

$$\begin{aligned} &\min(P) \le p_i \le \max(P) \\ &\Rightarrow \ln(\min(P)) \le \ln p_i \le \ln(\max(P)) \\ &\Rightarrow \ln(\min(P)) \sum_{p_i \in P} p_i \le \sum_{p_i \in P} p_i \ln p_i \le \ln(\max(P)) \sum_{p_i \in P} p_i \\ &\Rightarrow \min(P) \le p_{n+1} = \exp(\frac{\sum_{p_i \in P} p_i \ln p_i}{\sum_{p_i \in P} p_i}) \le \max(P) \end{aligned} \quad (7)$$

5.3 The ADLS Algorithm

The main goal of the ADLS algorithm is to identify the user's real location out from the dummy locations obtained by the DLS algorithm. When obtaining a user's LBS query, an adversary can adopt two methods to infer the user's real location based on user's location information. One method is to randomly choose one location from user's location information as the user's real location. By this method, the probability of successfully identifying the user's real location is 1/*k*, and the probability remains stable. The other method is to analyze the dummy location generation algorithm, and then design an attack algorithm. By this method, the adversary can enhance the probability of successfully identifying the user's real location by designing a good attack algorithm. In this paper, we adopt the latter method to infer user's real location. Based on the analysis of the DLS algorithm in section 4, we know that once the history query probabilities of two locations are different in DLS algorithm, their chosen optimal dummy location sets must be different. In this paper, we use this property to infer the user's real location out from the user's location information.

The ADLS algorithm first gets the anonymity degree *k* according to the user's location information. Then, for the $i^{th}(1 \le i \le k)$ location in user's location information, the ADLS algorithm selects other *k*-1 dummy locations based on entropy in a greedy manner, and then obtains the dummy location set C_i. After obtaining the *k* dummy location sets, the ADLS algorithm sorts the probabilities of set C_i $(1 \le i \le k)$ and the user's dummy location set in ascending order. Then, for each dummy locations set C_i

($1 \le i \le k$), the ADLS algorithm calculates the variance between the set C_i and the user's dummy location set, and determines the user's real location based on the variance. For example, if the variance between the set C_i and the user's dummy location is the smallest, the ADLS algorithm infers that the user's real location is location i. The following shows how the ADLS algorithm works.

(*i*) In the first step, the LBS server needs to get the anonymity degree of a user based on the user's location information. Let k denotes a user's anonymity degree, set R denotes a user's location information.

(*ii*) LBS server reads all the query probabilities and then sorts query probabilities of all locations in ascending order.

(*iii*) For each location in set R, the LBS server needs to selects $2k$-2 candidate locations (denoted as set D_j), in which k-1 locations are left before the user's real location and the other k-1 locations are right after the user's real location in the sorted list. Then, the LBS server puts the user's real location in C_j ($j \in [1, k]$).

(*iv*) Find the maximum and minimum from set C_j. Let $p_{\max}$ denote the maximum and $p_{\min}$ denote the minimum. Then, it finds two locations in set D_j, which is the maximum of the probability set being less than $p_{\min}$, denoted by $p_{\text{min-max}}$, and the other is the minimum of the probability set being greater than $p_{\max}$, denoted by $p_{\text{max-min}}$. Finally, it compares the entropy $H(C_j, p_{\text{max-min}})$ and $H(C_j, p_{\text{min-max}})$, and puts the location in set C_j, which achieves a larger entropy.

(*v*) Repeat step (*iv*) until the size of set C_j is k.

(*vi*) Finally, LBS server needs to determine which one is the user's real location. Specifically, for a particular chosen set C_j , it computes the variance according to Formula (8).

$$S_j = \sum_{i=1}^{k} (r_i - c_i)^2 \tag{8}$$

where $r_i \in R$, $c_i \in C_j$. The ADLS algorithm then uses the locations with the least variance as the user's real location:

$$S = arg \ \min S_j \tag{9}$$

***Algorithm* 1: Attack algorithm for DLS (ADLS)**

Input: Historical query probabilities of all locations denoted as P; a user's location information R.
Output: The optimal location.

1: Sort the elements in P and R in ascending order;
2: $k \leftarrow$ user's anonymity degree
3: for (i=1; $i \le k$; i++) **do**
4: Set $C_i \leftarrow$read one location L from set R which isn't read before;
5: Choose k-1 locations left before and k-1 locations right after location L in the sorted list as candidate location set D_i ;
6: **for** (j=1; $j \le k$; j++) **do**
7: $p_{\max} \leftarrow \max(C_i)$;
8: $p_{\min} \leftarrow \min(C_i)$;
9: Find one location from set D_i, which is the maximum of the probability set being less than $p_{\min}$ in set D_i, denoted as $p_{\text{min-max}}$;
10: Find one location from set D_i, which is the minimum of the probability set being greater than $p_{\max}$ in set D_i, denoted as $p_{\text{max-min}}$;
11: **if** $H(C_i , p_{\text{max-min}}) > H(C_i , p_{\text{min-max}})$ **then**
12: $C_i \leftarrow C_i \cup \{p_{\text{max-min}}\}$, $D_i \leftarrow D_i \backslash \{p_{\text{max-min}}\}$;
13: **else**
14: $C_i \leftarrow C_i \cup \{p_{\text{min-max}}\}$, $D_i \leftarrow D_i \backslash \{p_{\text{min-max}}\}$;
15: **end**
16: **end for**
17: Sort the elements in C_i in ascending order;
18: $S_i \leftarrow \sum_{i=1}^{k} (r_i - c_i)^2$
19: end for
20: return *arg min* S_i

5.4 Performance Evaluation

In this subsection, we evaluate the effectiveness of our proposed ADLS algorithm through simulation experiments.

5.4.1 Simulation Environment

In this set of simulations, the service area of LBS provider is divided into $n \times n$ cells with equal size. We assume that each cell has already had a historical query probability based on the users' previous queries. For measuring the *probability of query recognition*, which denotes the probability for the proposed ADLS algorithm to successfully identify a user's real location from the chosen dummy locations, we use the DLS algorithm to generate dummy locations and submit 1000 queries in the simulations.

In our simulations, k is related to k-anonymity and denotes the anonymity degree. Given the value of k, m denotes the number of cases that randomly choose k-1 cells from 2k cells, whose maximum value is C_{2k}^{k-1}. For evaluating the ADLS algorithm, the following four scenarios are considered in our simulations:

- *Scenario*-1: The value of m varies from 100 to 1000.
- *Scenario*-1.1: The value of k varies from 5 to 7.
- *Scenario*-1.2: The value of k varies from 10 to 14.
- *Scenario*-2: The value of k varies from 5 to 15, and the values of m are set to be 1×10^4, 5×10^4 and1×10^5, respectively.

5.4.2 Simulation Results

For evaluating the effectiveness of the proposed ADLS algorithm, we have conducted extensive simulations. We have evaluated the performance of the ADLS algorithm in terms of probability of query recognition under different scenarios with different values of k and m. Based on the analysis of the DLS Algorithm in Section 4, we can see that if a user's chosen optimal location set at location i is different from that of location j (i and j denote two different locations), the *ADLS Algorithm* with high probability to infer the user's real location from the dummy locations generated by DLS Algorithm.

Simulation Results Under Scenario-1.1: We explore the relationship between m and the probability of query recognition**.** From Figure 2, we can see that the probability of query recognition generally increase with the growth of m. The reason is that larger m leads to the chosen dummy location in DLS algorithm to be closer to the optimal dummy location set, which enables the ADLS algorithm to identify the user's real location with high probability. Figure 2 also shows that greater k leads to lower probability of query recognition while lower k results in higher probability of query recognition and this can be explained as follows. First, the maximum of m is C_{2k}^{k-1}, and C_{2k}^{k-1} exponentially increases with the growth of k. Second, for a given value of m, smaller anonymity degree k results in that the value of m is more close to the maximum one. Therefore, the user's chosen dummy locations are more likely to be close to the optimal dummy locations.

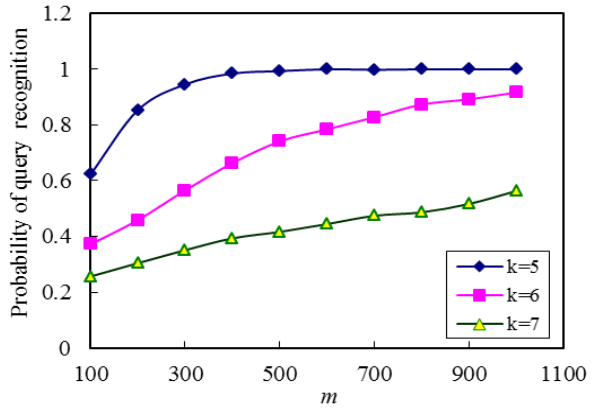


Fig.2: The probability of query recognition achieved with different anonymity degrees k under Scenario-1.1.

Simulation Results Under Scenario-1.2: In this set of simulations, we explore the relationship between m and the probability of query recognition when m and k become greater**.** Comparing with the results of Scenario-1.1, we observe that although the value of m and the value of k become greater, the probability of query recognition does not be improved. The reason is that when the value of k becomes greater, the higher probability of query recognition can be obtained only with greater value of m. Moreover, a small difference in the anonymity degree k will lead to a great difference in the value of m when achieving the same probability of query recognition in the ADLS algorithm.

Simulation Results Under Scenario-2: Figure 4 shows the relationship between k and the probability of query recognition. Generally, for a given value of m, the probability of query recognition will be influenced by the value of k. The results show that the greater the value of k is, the lower the probability of query recognition is. Furthermore, greater m leads to higher probability of query recognition while *lower* k results in lower probability of query recognition when $k\geq8$. Moreover, different values of m have almost the same probability of query recognition when $k\leq7$. The reason is that the smaller k makes the value of m to be close to the maximum value. Therefore, the user can select the optimal location set with higher probability.

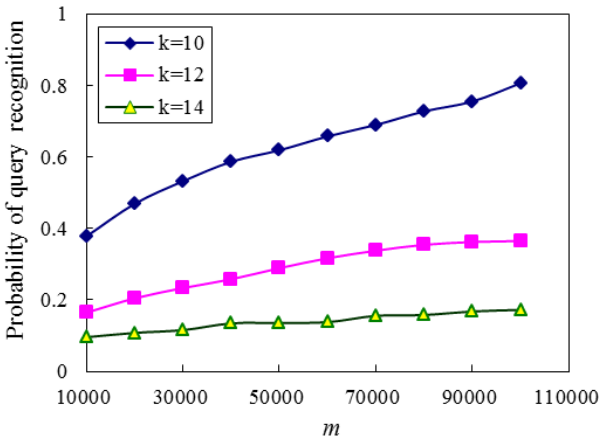


Fig.3: The probability of query recognition achieved in different anonymity degree *k* under Scenario-1.2.

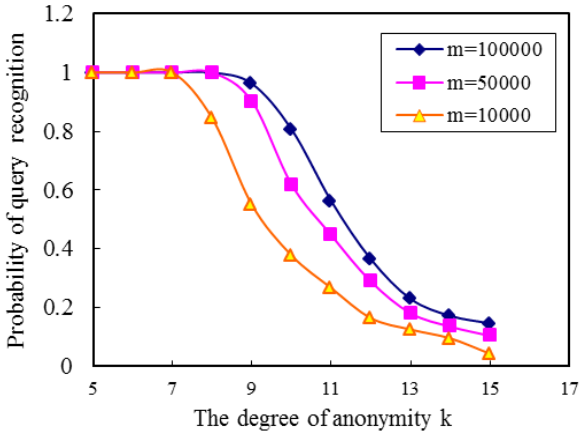


Fig.4: The probabilities of query recognition under Scenario-2.

6. DLP ALGORITHM DESIGN AND ANALYSIS

In this section, we give the detailed descriptions for the DLP algorithm, and present the performance evaluations.

6.1 DLP Algorithm Description

The basic idea of Dummy Location Privacy-preserving (DLP) algorithm is to select the optimal dummy locations considering that the adversary may exploit some side information, and make different choice for different privacy requirements of different users. We adopt a greedy approach to search a large database to find an optimal set of dummy locations. For achieving *k*-anonymity, we successively select *k*-1 other locations from all locations in the location map, which must make sure that the current entropy is the biggest. For example, if the DLP algorithm has already chosen *i* locations (where $i < k$), when choosing the $(i+1)^{th}$ location, it must ensure that H_{i+1} is the largest for all residual locations. H_{i+1} is defined in Equation (10).

$$H_{i+1} = -\sum_{j=1}^{i+1} \frac{p_j}{\sum_{l=1}^{i+1} p_l} \log_2 \frac{p_j}{\sum_{l=1}^{i+1} p_l} \quad , \tag{10}$$

where p_j denotes the users' historical query probability at location *j*. The following shows how the proposed DLP algorithm works.

(*i*) First, a user needs to set a proper anonymity degree *k*, which is closely related to the user's requirement on location privacy. Although a bigger *k* leads to higher anonymity degree, it also causes a higher overhead due to the cost for selecting dummy locations.

(*ii*) At the beginning, the DLP algorithm needs to read all the obtained query probabilities from the LBS server and then sort the query probabilities in ascending order. Let *p* denote the query probability of the user's real location. For the sorted

list, the DLP algorithm calculates the number of locations which have the same query probability as p, which is denoted by $\bar{k}$. If $\bar{k}$ is large enough, it puts half of them before and the other half of them after the real location.

(*iii*) If $\bar{k} \geq k$, DLP algorithm selects k-1 locations which have the same query probability as p from the sorted list. Then, it outputs the chosen k-1 dummy location and the user's real location.

(*iv*) If $k/4 \leq \bar{k} \leq k$, the algorithm selects $\bar{k}$-1 locations which have the same query probability as p from the sorted list. We use set C to denote the $\bar{k}$-1 dummy locations and the user real location. In the sorted list, the algorithm selects k-$\bar{k}$ locations left before and other k-$\bar{k}$ locations right after the real location as 2(k-$\bar{k}$)) candidate locations, whose query probabilities are different from p. Let set S denotes the 2(k-$\bar{k}$) candidates. The reason for choosing 2(k-$\bar{k}$) candidates for dummy locations is to make sure to get large enough entropy. Otherwise, it goes to Step (*vii*).

(*v*) To achieve k-anonymity, it needs to successively select residual k-$\bar{k}$ locations from set S. For the i^{th} ($\bar{k} < i \leq k$) dummy location, it must ensure that the H_i is maximum for all residual locations in set S.

(*vi*) When the size of C is k, DLP outputs the set C.

(*vii*) If $\bar{k} < k/4$, the DLP chooses 2k-ε locations left before and other 2k-ω locations right after the real location as 4k-ω-ε candidates from the sorted list. We use set $\bar{S}$ to denote the 4k-ω-ε candidates. Both ω and ε are set by users based on their privacy requirements. Generally, ω is smaller than ε. Let set $\bar{C}$ denote a user's real location. It randomly selects one location as a dummy location from set S, and put this location into set $\bar{C}$.

(*viii*) For achieving k-anonymity, the successively selects residual k-2 locations from set $\bar{S}$. For the i^{th} ($2 < i \leq k$) dummy location, it must ensure that H_i is the largest for all residual locations in set $\bar{S}$.

(*ix*)When the size of $\bar{C}$ is k, DLP outputs the set $\bar{C}$.

***Algorithm* 2: Dummy Location Privacy-preserving (DLP)**

Input: The set of historical query probabilities P; users' real location.
Output: The optimal set of dummy locations, C.

1: Sort P in ascending order;
2: $H \leftarrow$ select the locations which have the same query probability as users' real location from sorted P;
3: if (*size* (H) $\geq$ k) **then**
4: $C \leftarrow$ randomly select k locations including the user real location from H;
5: else if ($k/4 <$ *size*(H) $< k$) **then**
6: $\bar{k} \leftarrow$ *size*(H), $C \leftarrow H$;
7: $S \leftarrow$ choose 2(k-$\bar{k}$) candidate locations whose query probabilities are similar to the user's real location;
8: **for** (j = 1; $j \leq k$-$\bar{k}$; j ++) **do**
9: Choose one location l from set S, such that $H(C, q)$ is the maximum in set S;
10: $C \leftarrow C \cup \{l\}$, $S \leftarrow S \setminus \{l\}$;
11: **end for**
12: else
13: $S \leftarrow$ choose 4k-ω-ε candidate locations whose query probabilities are similar to the user's real location;
14: Randomly choose location i from S;
15: $C \leftarrow H \cup \{i\}$;
16: **for** (j = 1; $j \leq k$-2; j + +) **do**
17: Choose one location h from S, which makes sure that $H(C, q)$ is the maximum in set S;
18: $C \leftarrow C \cup \{h\}$, $S \leftarrow S \setminus \{h\}$;
19: **end for**
20: end if
21: return the optimal set of dummy locations, C .

6.2 Security Analysis

This subsection shows that how to resist the *colluding attacks* and *inference attacks* to protect user's location privacy through the proposed DLP algorithm.

1) *Resistance to the Colluding Attack:* To obtain user's location privacy, passive attackers may collude with other users or with the LBS provider for various purposes.

***Definition* 1**: A scheme can resist the colluding attack if the probability of successfully identifying a user's real location from the user's location information does not increase with the growth of the size of the colluding group.

***Theorem* 1**: The DLP algorithm can resist the colluding attack.

Proof: A colluding attack happens among a set of users who want to identify a user's real location out from the submitted *k* locations. In our scheme, each user protects her/his location privacy by selecting other dummy locations. When an attacker first compromises a user U_A, he/she will obtain the user's location information including *k* locations. Since the *k* locations have similar historical query probabilities, the attacker has no clue about the user's real location and only randomly guesses the user's real location out from the intercepted *k* locations. Thus, the probability of successfully identifying the user's real location is 1/*k*. Then, the attacker intercepts the LBS query of user U_B, and obtains the user's location information. However, the probability of successfully identifying a user's real location remains stable in our scheme. The reason is that there are no correlations between the selected dummy locations of users U_A and U_B. Therefore, the attacker can only identify each user's real location randomly from the intercepted *k* dummy locations. Similarly, when a colluding group has more members involve, the attacker can only randomly guess each user's real location from the intercepted *k* dummy locations. This implies that the probability of successfully identifying the user's real location out from the chosen dummy locations remains stable (i.e., 1/*k*) in our scheme.

In an extreme case that the passive adversary compromise the LBS server and get all information the LBS server has, he/she can turn to be an *active* adversary. For an active adversary, he/she can perform the inference attack.

2)Resistance to the Inference Attack: In this part of analysis, we assume that the LBS provider is an active attacker. The LBS provider knows a user's historical query probabilities of all locations, the historical queries and the current queries of users.

***Definition* 2**: A scheme can resist the inference attack if attackers cannot successfully identify the user's real location from user's location information.

***Theorem* 2**: DLP scheme can resist the inference attack.

Proof: In the DLP scheme, since the chosen *k* locations have similar historical query probabilities, although the LBS provider knows the historical query probabilities of all locations, he/she cannot determine which one is the user's real location in the *k* locations. Even then he/she tries to reverse the algorithm, but he/she will also be failed. The reasons are explained in the follows. Let us recall the step (3) to step (11) of the DLP scheme mentioned in Section 6.1. In these steps, since the DLP scheme can guarantee that there are enough locations whose historical query probabilities are as same as that of the user's real location in the chosen dummy locations, thus the LBS server still cannot obtain the user's real location by reversing the algorithm. Furthermore, let us recall the step (13) to step (18) of the DLP scheme. In these steps, since step (13) and step (14) of DLP can ensure the uncertainty of the selection, the LBS server also cannot obtain the real location by running our algorithm several times.

6.3 Performance Evaluation

For evaluating the performance of DLP algorithm, we have conducted extensive simulations in this subsection.

6.3.1 Simulation Environment

Similar to Section 5.4, we divide the location map into $n \times n$ cells with equal size. Each cell has a query probability based on the query history. We conduct simulations on the following three scenarios to evaluate the performance of the DLP algorithm.

- ***Scenario A:*** Let user be located in a cell such that there are many (more than *k*) cells that have the same historical query probability as the user's current location. In this scenario, the chosen dummy locations have the same query probability as that of the user's real location.
- ***Scenario B:*** Let user be located in a cell such that the number of cells that have the same historical query probability as that of the user's current location is slightly less than *k* but greater than one quarter of *k*. In this scenario, it can guarantee that there are enough locations have the same query probability as that of the user's real location in the chosen dummy locations.
- ***Scenario C:*** Let user be located in a cell such that there are a few (i.e., less than one quarter of *k*) cells have same historical query probability as that of the user's current location. In this scenario, there are few locations that have the same query probability as that of the user's real location in the chosen dummy locations.

6.3.2 Simulation Results

For evaluating the effectiveness of our proposed DLP algorithm, we have conducted extensive simulations.We have compared the performance of two algorithms in terms of the running time and the privacy level under various anonymity degree requirements of users. We also compare the probability of query recognition under Scenario C.

Figure 5, Figure 6 and Figure 7 illustrate the results for DLS algorithm and DLP algorithm, respectively. The results show the running time and the privacy level in terms of entropy under different scenarios. In Figure 5, the DLP algorithm and the DLS algorithm have the same entropy, but there are large differences in the running times. Moreover, the running time of DLS algorithm rapidly increases with the growth of the value of *k* (i.e., anonymity degree), but the running time of DLP algorithm varies little. The reason is that the DLS algorithm adopts enumeration method to select *k* dummy locations which

make the entropy is largest while the DLP algorithm adopts greedy method to successively select k dummy locations. The computational complexity of the DLS algorithm increases with the growth of the value of k, but the computational complexity of the DLP algorithm almost remains stable. From Figure 6 and Figure 7, we can see that Scenario B and Scenario C have the similar trend on results as Scenario A. We also note that the largest entropy appears in Scenario A, whereas the smallest entropy appears in Scenario C for both the DLS and DLP algorithms. This is because that there are more than k locations whose historical query probabilities are the same as that of the user's real location in Scenario A, but there are only enough or few locations whose historical query probabilities are the same as that of the user's real location in Scenario B or C. Moreover, we can obtain the maximum entropy $Hmax = \log_2 k$ under Scenario A. Thus, the DLS and DLP algorithms can achieve larger entropy in Scenario A than that in Scenario B or C.

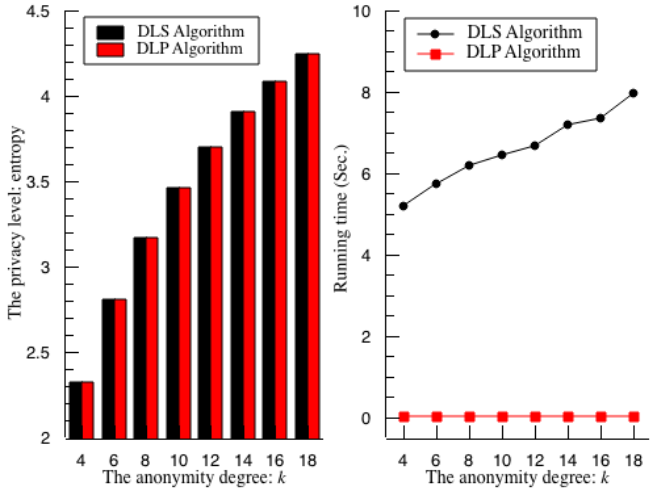


Fig.5: Entropy and running times under Scenario A

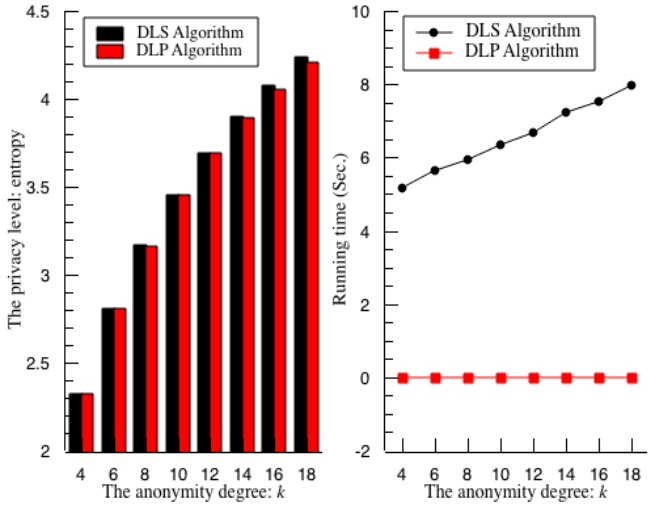


Fig.6: Entropy and running times under Scenario B

[17] F. Tang, J. Li, I. You, et al. Long-term location privacy protection for location-based services in mobile cloud computing. Soft Computing, 20(5): 1735-1747, 2016.
[18] B. Niu, S. Gao, F. Li, et al. Protection of location privacy in continuous LBSs against adversaries with background information. IEEE International Conference on Computing, Networking and Communications (ICNC), 1-6, 2016.
[19] B. Niu, X. Zhu, H. Chi, H. Li. 3PLUS: Privacy-preserving pseudo-location updating system in location-based services. IEEE Wireless Communications and Networking Conference (WCNC), 4564- 4569, 2013.
[20] D. Liao, X. Huang, V. Anand, et al. k-DLCA: An efficient approach for location privacy preservation in location-based services. IEEE International Conference on Communications (ICC), 1-6, 2016.
[21] T. Peng, Q. Liu, D. Meng, et al. Collaborative trajectory privacy preserving scheme in location-based services. Information Sciences, In Press, 2016.
[22] W. Ni, M. Gu, X. Chen. Location privacy-preserving *k* nearest neighbor query under user's preference. Knowledge Based Systems, 103: 19-27, 2016.
[23] K. Vu, R. Zheng and J. Gao. Efficient algorithms for k-anonymous location privacy in participatory sensing. IEEE INFOCOM, 2399-240, 2012.
[24] X. Zhu, H. Chi, B. Niu, W. Zhang. Mobi Cache: When k-anonymity meets cache. IEEE Globecom, 820-825, 2013.
[25] X. Liu, K. Liu , L. Guo. A game-theoretic approach for achieving k-anonymity in Location Based- Services. IEEE INFOCOM, 2985- 2993, 2013.
[26] Y. Ma, K. Yau, N. Yip, S. Rao. Privacy vulnerability of published anonymous mobility traces. IEEE/ACM Transactions on Networking, 720-733, 2013.
[27] B. Niu, Q. Li, X. Zhu, G. Cao. Achieving k-anonymity in privacy -aware location-based services. IEEE INFOCOM, 754 -762, 2014.
[28] M. Henze, L. Hermerschmidt, D. Kerpen, et al. A comprehensive approach to privacy in the cloud-based Internet of Things. Future Generation Computer Systems, 56: 701-718, 2016.
[29] L. González-Manzano, J. de Fuentes, S. Pastrana, et al. PAgIoT-Privacy-preserving Aggregation protocol for Internet of Things. Journal of Network and Computer Applications, 71: 59-71, 2016.
[30] P. Appavoo, M. Chan, A. Bhojan, et al. Efficient and privacy-preserving access to sensor data for Internet of Things (IoT) based services. IEEE International Conference on Communication Systems and Networks, 1-8, 2016.
[31] C. Lai, H. Li, X. Liang, et al. CPAL: A conditional privacy-preserving authentication with access linkability for roaming service. IEEE Internet of Things Journal, 1(1): 46-57, 2014.
[32] S. Premnath, Z. Haas. Security and privacy in the internet-of-things under time-and-budget-limited adversary mode. IEEE Wireless Communications Letters, 4(3): 277-280, 2015.
[33] J. Jin, J. Gubbi, S. Marusic, et al. An information framework for creating a smart city through internet of things. IEEE Internet of Things Journal, 1(2): 112-121, 2014.
[34] C. Perera, C. McCormick, A. Bandara, et al. Privacy-by-Design Framework for Assessing Internet of Things Applications and Platforms. arXiv preprint arXiv:1609.04060, 2016.
[35] L. Malina, J. Hajny, R. Fujdiak, et al. On perspective of security and privacy-preserving solutions in the internet of things. Computer Networks, 102: 83-95, 2016.
[36] A. Sadeghi, C. Wachsmann, M. Waidner. Security and privacy challenges in industrial internet of things. ACM the 52nd Annual Design Automation Conference, 54, 2015.
[37] B. Gedik, L. Liu. Protecting location privacy with personalized k-Anonymity: architecture and algorithms. IEEE Transactions on Mobile Computing (TMC), 7(1):1-18, 2008.
[38] B. Ying , D. Makrakis. Protecting Location Privacy with Clustering Anonymization in vehicular networks. IEEE INFOCOM Workshops, 305-310, 2014.
[39] A. Aryan and S. Singh. Protecting location privacy in Augmented Reality using k-anonymization and pseudo-id. International Conference on Computer and Communication Technology, 119-124, 2010.
[40] A. Hossain, S. Jang, J. Chang. Privacy-Aware Cloaking Technique in Location-Based Services. IEEE the First International Conference on Mobile Services, 9-16, 2012.
[41] A. Beresford and F. Stajano. Location privacy in pervasive computing. IEEE Pervasive Computing, 2(1): 46-55, 2003.
[42] X. Liu, H. Zhao, M. Pan, H. Yue, X. Li, and Y. Fang. Traffic aware multiple mix zone placement for protecting location privacy. IEEE INFOCOM, 972-980, 2012.
[43] T. Jung, X. Li, Z. Wan and M. Wan. Privacy preserving cloud data access with multi-authorities. IEEE INFOCOM, 2625-2633, 2013.
[44] J. Bethencourt, A. Sahai and B. Waters. Ciphertext-policy attribute based encryption. IEEE Symposium on Security and Privacy, 321-334, 2007.
[45] X. Li, T. Jung. Search me if you can: Privacy-preserving location query service. IEEE INFOCOM, 2760-2768, 2013.
[46] R. Lu , X. Lin , Z. Shi. PLAM: A privacy preserving framework for local-area mobile social networks. IEEE INFOCOM, 763-771, 2014.
[47] X. Zhu, H. Chi and S. Jiang. Using dynamic pseudo-IDs to protect privacy in location-based services. IEEE International Conference on Communications (ICC), 2307-2312, 2014.
[48] J. Shao , R. Lu and X. Lin. FINE: A fine-grained privacy-preserving location-based service framework for mobile devices. IEEE INFOCOM, 244-252, 2014.
[49] X. Yi , R. Paulet and E. Bertino. Practical k nearest neighbor queries with location privacy. IEEE 30th International Conference on Data Engineering (ICDE), 640-651, 2014.
[50] X. Zhao , H. Gao, L. Li , H. Liu and G. Xue. An efficient privacy preserving location based service system. IEEE GLOBECOM, 576-581, 2014.
[51] A. Serjantov and G. Danezis. Towards an information theoretic metric for anonymity. International Conference on Privacy enhancing technologies, 41-53, 2003.
[52] H. Lu, C. Jensen and M. Yiu. Pad: privacy-area aware, dummy based location privacy in mobile services. ACM MobiDE, 16-23, 2008.

[53] B. Niu, Z. Zhang, X. Li , H. Li . Privacy-area aware dummy generation algorithms for location-based services. IEEE ICC, 957-962, 2014.
[54] V. Chang, Y. Kuo, M. Ramachandran. Cloud computing adoption framework: a security framework for business clouds. Future Generation Computer Systems, 57: 24-41, 2016.
[55] V. Chang, M. Ramachandran. Towards achieving Data Security with the Cloud Computing Adoption Framework. IEEE Transactions on Services Computing, 9(1): 138-151, 2016.