k-DLCA: An Efficient Approach for Location Privacy Preservation in Location-Based Services

Dan Liao[1,2], Xunhui Huang[1], Vishal Anand[3], Gang Sun[1], Hongfang Yu[1]

[1]Key Lab of Optical Fiber Sensing and Communications (Ministry of Education), University of Electronic Science and Technology of China, Chengdu, China

[2]Institute of Electronic and Information Engineering in Dongguan, UESTC, China

[3] Department of Computer Science, The College at Brockport, State University of New York, USA

Abstract**—**Location-Based Service (LBS) is one of the fundamental and central functionalities of mobile social networks. Since users usually have to report their locations to the LBS providers while using services, the protection of user's location privacy poses a critical challenge. Although many existing approaches can preserve user's location privacy effectively, most of them must include and use the user's real location. In this paper, we first propose an efficient *k*-anonymity based Dummy Location and divided Circular Area (*k*-DLCA) approach to protect the user's location privacy. Different from existing studies, the *k*-DLCA algorithm adopts a greedy strategy to select dummy locations and considers the semantic location information of the location. Moreover, the user's real location may not be contained in the chosen dummy locations. We then show that *k*-DLCA algorithm can resist the attacks from adversaries, and has a low probability of exposing the user's real location. We conduct extensive simulations to evaluate the efficiency of the proposed scheme. The simulation results demonstrate that our proposed scheme is promising.

Key words**:** Privacy-preserving; Location privacy; Location based service; *k*-anonymity

I. Introduction

Due to the rapid development of mobile communication technology and mobile devices, Location-Based Services (LBS) has become increasingly prevalent in mobile social networks. When a user needs LBS, the user sends a query or request to their LBS provider that may include the identity, location, points of interest (POI), and region of interest (ROI) of the user. However, an untrusted LBS provider that has all user information may use this information to track users or reveal their personal data to a third party for commercial or other interests. Therefore, how to preserve user's location privacy while using the LBS is an important issue that must be addressed.

Several approaches [1-4] for protecting user's location privacy have been proposed. Most of them are based on the *k*-anonymity cloaking technique, which was first proposed in [5]. The *k*-anonymity cloaking technique employs a trusted third-party to select other *k*-1 LBS queries from the user's neighbors and sends all of them in addition to the user's own query to the LBS provider, instead of only sending the user's request. However, the heavy reliance on a third party leads to a single point failure and also creates a serious performance bottleneck.

To preserve user's location privacy without any third party, the dummy location technique was proposed. Existing approaches [6-8] can effectively generate dummy locations that cannot be distinguished by LBS providers. However, none of these approaches take the semantic location information (e.g., the probability of user sending LBS query on certain location type) into account. If some of the chosen dummy locations are locations in which users send queries with small probabilities, these locations can easily be filtered out by attackers. Although the use of dummy locations [9-14] can protect user's location privacy, the user's real location must be one of the dummy locations. To prevent the loss of user's real location, the authors in [15] proposed a geometric approach, called *n* concealing disks (*n*-CD), to protect user's location privacy. This approach first divides the ROI of each user into *n* equal sectors, and then randomly generates *n* locations in user's ROI. Finally, it generates *n* concealing disks to fully cover the user's real ROI, and sends the *n* locations and radii of *n* concealing disks to the LBS server. By doing this, the query results obtained from the LBS provider must include the user's real POI. Although such a query does not contain the user's real location, attackers still are able to infer that the user's real location must be within a certain overlapping region.

In this paper, we propose an efficient *k*-anonymity based Dummy Location and divided Circular Area (*k*-DLCA) approach for preserving user's location privacy. Different from existing algorithms, in order to effectively protect user's location privacy, the *k*-DLCA algorithm successively selects dummy locations based on entropy in a greedy manner considering the semantic location information that may be exploited by attackers. Also, the *k*-DLCA algorithm may not require a user's real location to be included in the chosen dummy locations, and thus can lower the attackers' average probability of successfully conjecturing the user's real location. Moreover, the *k*-DLCA algorithm can conceal the user's real location in larger anonymity zone.

The remainder of this paper is organized as follows. Section II gives the problem descriptions. Section III describes the proposed *k*-DLCA algorithm. Section IV conducts the analysis for *k*-DLCA algorithm. Section V provides the simulation results, and Section VI concludes the paper.

II. Problem Description

A. System Model

We consider a LBS system consisting of many mobile users. An original LBS query of a user can be denoted as $q = (u_{id}, \{(x, y), R, P\})$, where u_{id} and (x, y) are the identity and location coordinate of the user, respectively; R is the radius of user's ROI, and P is the user's POI, e.g., a restaurant, a gas station, etc. To preserve user's location privacy, before submitting a request to the LBS provider, a user first employs a dummy location generation algorithm to generate k dummy locations, denoted by $(x_1, y_1), \ldots, (x_k, y_k)$, where the value of k

will be determined by the user's privacy requirement. If a user wants to protect their POI in certain sensitive locations, e.g., a hospital, the algorithm can generate some other dummy POIs as well. The original query q is then transformed into a new query $q^* = (u_{id}, \{(x_1,y_1),\ldots,(x_k,y_k)\}, \overline{R}, \{P_1,\ldots P_h\})$, where $\overline{R}$ may be different from R, and the set $\{P_1,\ldots,P_h\}$ is the set of dummy POIs including P. Finally, the query q^* is sent to the LBS server. Upon receiving the query q^*, the LBS provider will search for all POIs in the ROI for all the dummy locations and return them to the user.

B. Basic Concepts

In this paper, the semantic location information is referred to the probability of user sending LBS query on certain location type. User assigns a semantic parameter for each location type. The greater the semantic parameter, the higher the probability of user sending LBS query on this location type. In this paper, since the LBS provider knows the locations in its service area, the server is responsible for classifying all locations based on location property into different types, such as roads, shops, residences etc.

In this work, we adopt entropy [13] to measure the degree of privacy, which can be seen as an uncertainty in identifying user's real location out of chosen dummy locations. Thus, we need to have a current query probability for each dummy location so that we can calculate the entropy of the user. We use $\overline{q_i}(1 \le i \le k)$ to denote the current query probability related to location i, which can be obtained by the semantic location information and the historical query probability of location i. Similar to [13], the historical query probability related to location i can be defined in Equation (1).

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

The current query probability $\overline{q_i}$ can be calculated as follows. If the historical query probability related to location i is q_i and the semantic parameter of location i is b_i, then the current query probability of the user at location i is the product of q_i and b_i, i.e., $\overline{q_i} = q_i \cdot b_i$. In the dummy locations set, the normalized probability of each dummy location is denoted by p_i. The entropy H of each user thus is defined as:

$$H = -\sum_{i=1}^{k} p_i \log_2 p_i \quad , \tag{2}$$

where $p_i = \overline{q_i} \big/ \sum_{i=1}^{k} \overline{q_i}$.

When all the k dummy locations have the same probability, the maximum entropy can be achieved, i.e., $H_{max} = \log_2 k$.

III. Algorithm Design

In this section, we first introduce an attack model for the k-DLCA algorithm, and then give the detailed description of our k-DLCA algorithm.

A. Attack Model

The dummy location generation algorithm can be used to efficiently generate dummy locations for protecting the user's location privacy in LBS. The LBS server is able to obtain the historical query probabilities related to all locations and monitor the current queries being sent from users. Moreover, the LBS server also can obtain the historical information of a particular user as well as the current situation and information of the user. Additionally, the LBS server knows how the location privacy protection mechanism works, and the server is responsible for disseminating and updating the historical query probabilities so that users can get this information from a well-known place (e.g., local database of LBS application). The goal of the adversary is to identify the user's real location out of the user's location information. Since the adversaries may compromise the LBS server and obtain all the information that the server knows and holds, we assume that the LBS server is the adversary in this work.

B. Algorithm Description

The main idea of the k-DLCA algorithm is to generate a set of dummy locations by considering the semantic location information and the historical query probability that may be exploited by adversaries, so that all dummy locations will have the similar current query probability. Given the degree of anonymity k, the k-DLCA algorithm successively selects k dummy locations in a greedy manner, such that the current entropy is maximal. For example, if the k-DLCA algorithm has already chosen the first i locations, where $i < k$, when the k-DLCA algorithm chooses the $(i+1)^{th}$ location, it must ensure that the value of H_{i+1} is the largest among all the remaining locations. H_{i+1} is defined in Equation (3).

$$H_{i+1} = -\sum_{j=1}^{i+1} \frac{p_j}{\sum_{l=1}^{i+1} p_l} \log_2 \frac{p_j}{\sum_{l=1}^{i+1} p_l} \quad , \tag{3}$$

where p_j denotes the user's current query probability at location j. Since the chosen dummy locations may do not include the user's real location in this work, the k-DLCA algorithm adopts different strategies to select dummy locations. If the chosen dummy locations include the user's real location, the k-DLCA algorithm directly selects dummy locations based on entropy. If t the chosen dummy locations do not include the user's real location, the k-DLCA algorithm divides the user's ROI into n equal sectors, where $n \le k$. The main idea of dividing user's ROI is that the k-DLCA algorithm adopts other ROIs to fully cover user's ROI so that although the chosen dummy locations do not include the user's real location, the user still can obtain what he/she is interested in from the search results. In this paper, the values of k and n are set according to the user's privacy requirement. If $n < k$, the k-DLCA algorithm needs to select other k-n dummy locations based on entropy. Finally, if the user wants to protect his/her own real POI, the k-DLCA algorithm has to generate other suitable dummy POIs for the user according to the chosen dummy locations. The detailed description of the k-DLCA algorithm is as follows.

```
Algorithm 1: k-DLCA Algorithm
Input: $q = (u_{id}, \{(x,y), R, P\})$, historical query probability set $P*$,
        the semantic parameters of each location type;
Output: New query $q*$.
1: Calculate the current query probability set $Q$ based on
   semantic parameters and $P*$;
```

```
2: Classify the elements in set $Q$ based on location types;
3: if ($strategy$ = =1) then
4:     Call Procedure 1;
5: else
6:     Call Procedure 1;
7: end if
8: if a user needs to protect query privacy then
9:     Select $\lfloor l/2 \rfloor$-1 dummy POIs based on the $k$ dummy locations;
10:    return $q^* = (u_{id}, \{(x_1, y_1), ..., (x_k, y_k)\}, R, \{P_1, ..., P_{\lfloor l/2 \rfloor}\})$
11: end if
12: return $q^* = (u_{id}, \{(x_1, y_1), ..., (x_k, y_k)\}, R, P)$
```

***Procedure* 1: Generation of Dummy Locations including user's real location**

```
Input: User's location($x$, $y$), the semantic parameters of each location type, current query probability set of each location type;
Output: $k$ dummy locations $\{(x_1, y_1), ..., (x_k, y_k)\}$.
1: Select other $l$-1 location types, based on the smallest difference on the semantic parameter with the user's real location;
2: for each of the chosen location types
3:     Choose $k$-$l$ candidate locations based on the smallest difference on the current query probability with the user's real location in a greedy manner;
4: end for
5: Select the other $k$-1 locations, based on entropy in the chosen candidate locations and make sure that the number of the location types is no less than $\lfloor l/2 \rfloor$;
6: return $\{(x_1, y_1), ..., (x_k, y_k)\}$.
```

***Procedure* 2: Generation of Dummy Locations excluding user's real location**

```
Input: User's location($x$, $y$), semantic parameters of location types, current query probability of each location type;
Output: $k$ dummy locations and new radius $\{(x_1, y_1), ..., (x_k, y_k)\}, \bar{R})\}$.
1: Divide the user's ROI into $n$ sectors;
2: Randomly select one location from user's ROI, which belongs to the first sector; select other $n$-1 locations from other $n$-1 sectors based on entropy in a greedy manner;
3: Generate $n$ circular areas to fully cover the ROI of each user's; update $R$ as the largest radius of $n$ circular areas;
4: if ($n < k$) then
5:     $l^* \leftarrow$ the number of location types of the $n$ locations;
6:     if ($l^* < l$) then
7:         Select other $l$-$l^*$ location types whose semantic parameters have the smallest difference with first $l^*$ location types;
8:     end if
9:     for each of chosen $l$ location types
10:        Choose $k$-$n$ candidate locations whose current query probabilities have the least difference with the user's real location;
11:    end for
12:    Select the other $k$-$1$ locations based on entropy in the chosen candidate locations in a greedy manner, and make sure that the number of their location types is no less than $\lfloor l/2 \rfloor$;
13: end if
14: return $\{(x_1, y_1), ..., (x_k, y_k)\}, R)$.
```

IV. Algorithm Analysis

In this section, we first show how our scheme resists the *colluding attack* and *inference attack*. We then analyze how our scheme lowers the probability of identifying a user's real location by attackers.

A. Resistance to 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 conjecturing the user's real location out of user's location information does not increase with the growth of the size of the colluding group.

Theorem 1: The k-DLCA algorithm can resist colluding attack.

Proof: A colluding attack happens among a set of users. Our scheme preserves user's location privacy through selecting other dummy locations. When an attacker first compromises user U_A, the attacker will obtain the user's location information including k locations. Since the k locations have similar current query probability, the attacker has no clue about the user's real location and only can guess the user's real location out of the intercepted k locations. Since the k locations may not include user's real location, the probability of successfully conjecturing the user's real location may be $1/k$ or 0. Then, the attacker intercepts the LBS query of user U_B, and obtains the location information of this user. Similarly the probability of successfully conjecturing the real location of U_B is as same as that for U_A. The reason is that there are no relationships between the dummy location selections of U_A and U_B. Therefore, the attacker can only conjecture each user's real location randomly within the intercepted k dummy locations. Similarly, when a colluding group has more members involved, the attacker also can only guess each user's real location out of the intercepted k dummy locations. This implies that the probability of successfully conjecturing a user's real location does not increase with the growth of the size of the colluding group.

In the extreme where the passive adversary compromises the LBS server and gets all the information that the LBS server has, a passive adversary can become an *active adversary*. An *active adversary* can perform the *inference attack.*

B. Resistance to Inference Attacks

In this part of the analysis, we assume the LBS provider to be an active attacker. The LBS provider has the historical query probabilities of all locations, as well the historical queries and the current queries of users. Moreover, the LBS provider knows how the proposed algorithm works.

Definition 2: A scheme can resist the inference attack if attackers cannot successfully obtain the user's real location according to the user's location information.

Theorem 2: The k-DLCA algorithm can resist inference attack.

Proof: Whether the user's real location is included in the chosen k locations or not, the k-DLCA algorithm can guarantee that the chosen k locations have similar current query probabilities. Thus, when the LBS provider receives the query of a user, the provider cannot know whether the user's location information includes the user's real location.

When the LBS provider knows that a user's location information includes the real location, we analyze how our scheme resists the inference attack as follows. Since the chosen k locations have similar current query probabilities, the provider cannot identify which one of the k locations is the user's real location. Moreover, although the LBS provider knows the historical query probabilities of all locations and the provider tries to reverse the k-DLCA algorithm, it will fail. The reason is that the provider does not know the semantic parameter of each location type set by users. In the k-DLCA algorithm, different semantic parameters can lead to different dummy location selections.

Next, we analyze how our scheme resists the inference attack when the LBS provider knows that the user's real location is not included in the location information. We note that the LBS server does not know how many sectors the user's ROI is divided into (i.e., the value of n) and the first chosen n locations of our scheme. Therefore, even if the provider tries to reverse the k-DLCA algorithm, the provider cannot obtain the user's real location from the k submitted locations. The reason is that our k-DLCA algorithm can guarantee the uncertainty of the selection results, that is, different value of n and different first n chosen locations lead to different dummy location results in the k-DLCA algorithm.

C. How the k-DLCA algorithm lowers the probability of identifying user's real location.

In this part of analysis, we assume that the LBS provider is an active attacker. Let X be a random variable with probability function $P(X)$ that represents the probability that the provider successfully infers the user's real location from the k locations. Then $P(X=0)$ is the probability of failure while $P(X=1)$ is the probability of success, and $E(X)$ denotes the average probabilities of success.

In the DLS algorithm [12], although the chosen dummy locations include the user's real location, they have similar historical query probabilities. Thus, the provider can only randomly guess the user's real location out of the k locations. Therefore, we get $P(X=0)=(k-1)/k$ and $P(X=1)=1/k$. Therefore, we get the expectation of X, as shown in Equation (4).

$$E(X)_{\text{DLS}} = 1/k \qquad (4)$$

In the k-DLCA algorithm, the LBS provider does not know whether the location information includes the user's real location. Thus, the provider should first determine if the k locations include the user's real location with the probability of 1/2, and then infer the user's real location from the k locations. If the LBS provider determines that the user's submitted k locations do not include the user's real location, the provider needs to infer the value of n and which n locations enclose the user's real location. Without loss generality, we assume that the LBS provider knows the value of n and the number of randomly chosen n location from the k locations which enclose the user's real location is *num.* Moreover*,* we also assume that once the LBS provider finds out the n locations, the provider can obtain the user's real location. Thus, when the LBS provider judges that the user's submitted k locations do not include the user's real location, the probability of obtaining the user's real location is $1/num$. If the LBS provider judges that the k locations include the user's real location, the probability of obtaining the user's real location is $1/k$. Therefore, we can get $P(X=0)=1-(1/k+1/num)/2$ and $P(X=1)=(1/k+1/num)/2$ in the k-DLCA algorithm. Then, we can get the expectation of X, as shown in Equation (5).

$$E(X)_{\text{k-DLCA}} = \frac{1}{2k} + \frac{1}{2num} \quad , \qquad (5)$$

where num is affected by n and k.

Since we assume that once the LBS provider finds out the n locations which enclose the user's real location, the provider can obtain the user's real location, $P(X=0)=0$ and $P(X=1)=1$ in the n-CD algorithm [16]. Then, we can get the expectation of X, as shown in Equation (6).

$$E(X)_{\text{n-CD}} = 1 \qquad (6)$$

From Equation (4)-(6), we can see that $E(X)_{\text{n-CD}}$ is the greatest among the three algorithms. Moreover, for a given k, when n is less than k, num is greater than k. Thus, $1/2k+1/2num$ is less than $1/k$, i.e., $E(X)_{\text{k-DLCA}}$ is less than $E(X)_{\text{DLS}}$.

V. Simulation And Results

To validate the effectiveness of the proposed k-DLCA algorithm, we have conducted extensive simulations. We assume that the LBS service area is divided into 40×40 cells with equal size, in which 20 points of interests (POIs) are randomly and uniformly generated. Each location is classified into one of the corresponding location types by the LBS provider.

A. Privacy Achieved

When the dummy locations include the user's real location, we compare the proposed k-DLCA against the DLS algorithm [12], wherein the latter selects dummy locations by considering the historical query probability related to location.

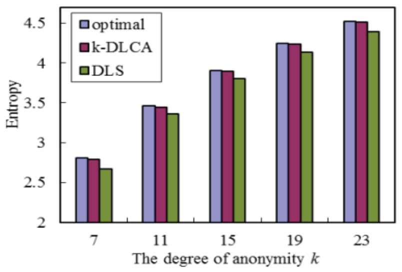

Fig.1: The entropy achieved in different schemes

Figure 1 illustrates the privacy achieved by different schemes in terms of entropy when the dummy locations include each user's real location. The optimal solution can be obtained according to $H_{max} = \log_2 k$ as shown in section II. Generally, the entropy of each scheme increases with the growth of k. Moreover, the optimal scheme has the maximum entropy since all the submitted k locations have the same current query probability. Figure 1 shows that the k-DLCA scheme achieves higher entropy than that of the DLS scheme, because the latter only takes into account the side information that may be exploited by attackers, while the *k*-DLCA scheme considers not only the side information but also the semantic location information of each location type. We can also see that the result of *k*-DLCA is very close to the one delivered by the optimal scheme. Therefore, the *k*-DLCA scheme can efficiently protect user's location privacy.

Table 1 and Table 2 list the entropy and the running time of *k*-DLCA algorithm. From Table 1 we can see that the entropy always increases with the growth of *k*. It is obvious that there is marginal difference in entropy under different *l*, and larger *l* leads larger entropy. This is because the greater the value of *l*, the more candidate dummy locations the *k*-DLCA scheme can generate, which will make the *k*-DLCA scheme select more appropriate dummy locations for users. From Table 2, we can see that the running time of *k*-DLCA algorithm always increases with the growth of the value of *k*. This is because the greater the value of *k*, the more candidate dummy locations the *k*-DLCA algorithm will select, which will make the *k*-DLCA scheme take more time to generate dummy locations. From Table 2, we can also see that the larger *l* results in more running time for *k*-DLCA algorithm. The reason is that the chosen *k* locations must satisfy the condition that the number of different location types is not less than *l*/2. If this condition cannot be met, the algorithm must select other sub-optimal dummy locations to substitute part of them until the chosen *k* suboptimal dummy locations satisfy the condition.

Table 1: Entropy for different *l* (*k*-DLCA)

the privacy level: entropy				
k	*l* = 3	*l* = 5	*l* = 7	*l* = 9
9	3.16519	3.16613	3.16631	3.16818
11	3.45453	3.45537	3.45739	3.45905
13	3.69538	3.69669	3.69826	3.69996
15	3.90154	3.90287	3.90448	3.90629
17	4.08171	4.08377	4.08451	4.08672
19	4.24194	4.24380	4.24340	4.24705

Table 2: Running time for different *l* (*k*-DLCA)

Running time (Seconds)				
K	*l* = 3	*l* = 5	*l* = 7	*l* = 9
9	0.022	0.024	0.028	0.035
11	0.023	0.026	0.030	0.037
13	0.025	0.029	0.034	0.040
15	0.027	0.033	0.037	0.044
17	0.030	0.037	0.041	0.047
19	0.033	0.041	0.045	0.051

B. *Anonymity Zone*

When the dummy locations do not include user's real locations, we compare the proposed *k*-DLCA against the *n*-CD algorithm. In the simulation, we consider the worst case for *k*-DLCA algorithm, i.e., the attackers know that user's real location is not contained in the user's location information and the value of *n*. Moreover, attackers can find out the first *n* chosen locations of a user. Although the attacker may not be able to obtain the user's real location, they know the user's real location must be within a certain region according to the radius of user's ROI, which we refer to as the *anonymity zone* of a user.

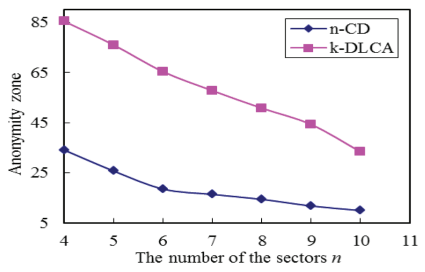

Fig.2: The anonymity zone achieved in different schemes

Figure 2 illustrates the anonymity zone obtained by the *k*-DLCA and *n*-CD scheme when $R = 10$. It can be seen from the figure that the anonymity zone decreases with the growth of the value of *n*. The reason is that the larger the value of *n,* the smaller the radii of *n* generated circular areas, which will lead to a smaller *anonymity zone* where the *n* circular areas intersect with each other. We also can see that the *k*-DLCA algorithm delivers a much bigger anonymity zone than that of the *n*-CD algorithm. Furthermore, the decreasing rate of the anonymity zone by the proposed *k*-DLCA algorithm is much faster than that of the *n*-CD algorithm. The reason is that the *k*-DLCA algorithm selects the maximum radius of generated *n* circular area as the radius of the user's ROI, while the *n*-CD algorithm selects a radius for each chosen locations which are the center of generated *n* circular areas. When compared with the *n*-CD algorithm, we can conclude that the *k*-DLCA algorithm can conceal a user's real location into a much bigger *anonymity zone*.

C. *Probability of Conjecturing the User's Real Location*

In the simulation, $E(X)$ denotes the attacker's average probability of successfully conjecturing the user's real location. $E(X)$ for DLS, *k*-DLCA and *n*-CD algorithm can be calculated by Equation (4)-(6), respectively. Since $E(X)_{n\text{-CD}}$ is the greatest among the three algorithms, we only compare $E(X)_{k\text{-DLCA}}$ with $E(X)_{\text{DLS}}$ in the following. We note that $E(X)$ is not only affected by *k* but also by *n* in the *k*-DLCA algorithm. In this set of simulations, the value of *n* is fixed at 4. From Figure 3, we can see that $E(X)$ generally decreases with the growth of *k*. The reason is that the greater the value of *k*, the lower the attacker's average probability of successfully conjecturing the user's real location. Compared with the DLS algorithm, the achieved $E(X)$ is much smaller in the *k*-DLCA algorithm. The reason is that the *k*-DLCA algorithm allows the chosen dummy locations not to include the user's real location, which increases the difficulty for the adversary to infer the user's real location.

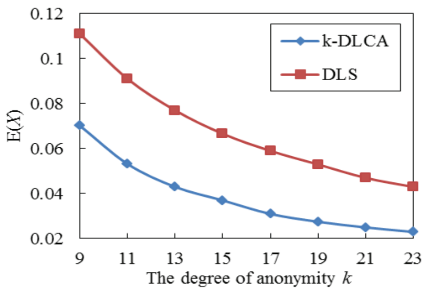

Fig.3: Comparison for $E(X)$

Table 3 shows the effect of *n* on $E(X)$ in the *k*-DLCA algorithm. We can see that, for a given value of *n*, the value of $E(X)$ always decreases with the growth of value of *k*. However, for a given value of k, the value of $E(X)$ may not always increase or decrease with the growth of n. For example, when k=9, $E(X)$ increases with the growth of n; whereas when k =11, $E(X)$ first decreases with the growth of n, and then increases with the growth of n; and when $k \geq 13$ and $k \leq 15$, $E(X)$ decreases with the growth of n; and when $k \geq 17$, $E(X)$ first decreases with the growth of n and then remains stable. The reasons are as follows. First, the number of location sets which randomly select n locations from k locations is C_k^n . Since $C_k^n = C_k^{k-n}$, for a given k, C_k^n increases with the growth of the value of n when n is less than k/2, and C_k^n decreases with the growth of the value of n when n is greater than k/2. Second, since *num* is the number of location sets which randomly select n locations from k locations and enclose the user's real location in the k-DLCA algorithm, for a given value of k, *num* increases with the growth the n when n is less than certain value but decreases with the growth n when n *is* greater than certain value. Thus, $E(X)$ decreases when *num* increases with the growth of n, and $E(X)$ increases when *num* decreases with the growth of n. Moreover, when the value of *num* is far greater than that of k, the value of $E(X)$ is near 1/2k and remains stable.

Table 3: $E(X)$ vs. n

k	E(X)				
	n = 4	n = 5	n = 6	n = 7	n = 8
9	0.0702	0.0720	0.0735	0.0830	0.0970
11	0.0532	0.0490	0.0480	0.0500	0.0514
13	0.0435	0.0430	0.0393	0.0395	0.0392
15	0.0375	0.0340	0.0337	0.0336	0.0335
17	0.0310	0.0298	0.0295	0.0294	0.0294
19	0.0275	0.0265	0.0260	0.0260	0.0260

VI. CONCLUSION

In this paper we studied the problem of preserving user's location privacy in LBS. In order to effectively protect user's location privacy, we first propose an efficient dummy location generation approach, k-DLCA, which greedily selects dummy locations based on entropy by considering the semantic location information that may be exploited by attackers. We then show that the k-DLCA algorithm can resist colluding and inference attacks from adversaries. We conducted simulations to evaluate the efficiency of the proposed algorithm. The simulation results demonstrated that the proposed k-DLCA algorithm can enhance entropy for preserving user's location privacy, and conceal user's real locations in larger anonymity zone, i.e., lower the probability of revealing user's real location.

ACKNOWLEDGEMENT

This research was partially supported by the National Grand Fundamental Research 973 Program of China under Grant (2013CB329103), Natural Science Foundation of China grant (61271171, 61571098), China Postdoctoral Science Foundation (2015M570778), Fundamental Research Funds for the Central Universities (ZYGX2013J002), Guangdong Science and Technology Project (2012B090400031, 2012B090500003, 2012B091000163), and National Development and Reform Commission Project.

REFERENCES

[1] C. Chow, M. Mokbel, and W. Aref. Casper*: Query processing for location services without compromising privacy. ACM Transaction on Database Systems, 34(4):24, 2009.

[2] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing location-based identity inference in anonymous spatial queries. IEEE Transactions on. Knowledge and Data Engineering, 19(12): 1719-1733, 2007.

[3] 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.

[4] B. Ying, D. Makrakis. Protecting Location Privacy with Clustering Anonymization in vehicular networks. IEEE INFOCOM, 2014.

[5] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. ACM Mobisys, 31-42, 2003.

[6] 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, 2012.

[7] 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, 2013.

[8] X. Zhu, H. Chi , B .Niu , W. Zhang. Mobi Cache: When k-anonymity meets cache. IEEE GLOBECOM, 2013.

[9] X. Liu, K. Liu, L. Guo. A game-theoretic approach for achieving k-anonymity in Location Based-Services. IEEE INFOCOM, 2013.

[10] H. Kido, Y. Yanagisawa, and T. Satoh. An anonymous communication technique using dummies for location-based services. International Conference on Pervasive Services, 2005.

[11] H. Lu, C. S. Jensen, and M. L. Yiu. Pad: privacy-area aware, dummy based location privacy in mobile services. ACM MobiDE, 16-23, 2008.

[12] B. Niu, Q. Li, X. Zhu, G. Cao. Achieving k-anonymity in privacy-aware location-based services. IEEE INFOCOM, 2014.

[13] A. Serjantov G. Danezis .Towards an information theoretic metric for anonymity. International Conference on Privacy enhancing technologies, 41-53, 2003.

[14] B. Niu , Z. Zhang , X. Li , H. Li . Privacy-area aware dummy generation algorithms for Location-Based Services. IEEE International Conference on Communications (ICC), 2014.

[15] M. Li, S. Salinas, A. Thapa, and P. Li. n-CD: A Geometric Approach to Preserving Location Privacy in Location-Based Services. IEEE INFOCOM, 2013.