Location Diversity: Enhanced Privacy Protection in Location Based Services

Mingqiang Xue, Panos Kalnis, and Hung Keng Pung

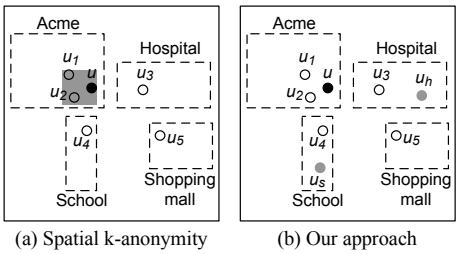
Department of Computer Science,
National University of Singapore
{xuemingq,kalnis,punghk}@comp.nus.edu.sg

Abstract. Location-based Services are emerging as popular applications in pervasive computing. Spatial k-anonymity is used in Location-based Services to protect privacy, by hiding the association of a specific query with a specific user. Unfortunately, this approach fails in many practical cases such as: (i) personalized services, where the user identity is required, or (ii) applications involving groups of users (e.g., employees of the same company); in this case, associating a query to any member of the group, violates privacy.
In this paper, we introduce the concept of *Location Diversity*, which solves the above-mentioned problems. Location Diversity improves Spatial k-anonymity by ensuring that each query can be associated with at least ℓ different semantic locations (e.g., school, shop, hospital, etc). We present an attack model that maps each observed query to a linear equation involving semantic locations, and we show that a necessary condition to preserve privacy is the existence of infinite solutions in the resulting system of linear equations. Based on this observation, we develop algorithms that generate groups of semantic locations, which preserve privacy and minimize the expected query processing and communication cost. The experimental evaluation demonstrates that our approach reduces significantly the privacy threats, while incurring minimal overhead.

1 Introduction

Nowadays, mobile devices are able to access a variety of Location-based Services (*LBS*) in the pervasive environment. As an example of LBS, a car driver may send query about the route to the nearest gas station when she is driving on the expressway.

A characteristic of such services is that they require the user location to answer queries. This raises serious privacy concerns, since the location can reveal to an attacker (which may be the LBS itself) sensitive information about the user, such as her interests, alternative life style, health conditions, etc. The problem is not solved by simply replacing the user ID with a pseudonym, since the location information can be joined with other public data (e.g., white pages service) to re-identify the user. Therefore, most of the existing work combines pseudonyms with Spatial k-anonymity [5, 10, 16]. Specifically, between the users and the LBS



(a) Spatial k-anonymity (b) Our approach

Fig. 1. k-anonymity vs. Location Diversity

exists a trusted Anonymizer service that keeps the current locations of all users. When user u wants to send a query, she contacts the Anonymizer, which removes the ID of u and constructs an Anonymizing Spatial Region (ASR) that includes u and $k-1$ other users near u; the ASR is sent to the LBS. The intuition is that by observing the ASR, the probability of identifying u as the querying user is at most $1/k$. The LBS answers the query for the entire ASR and sends the candidate results to the Anonymizer, which filters the false positives and returns the actual answer to u.

Nevertheless, there are cases where k-anonymity is inadequate. The problem exists because each user location is implicitly associated with semantic information. For example:

Example 1 (Group privacy violation). Acme is an insurance company with a large client list; the list is a valuable business asset and must be kept secret. Acme's employees visit frequently their clients. To plan their trip, they use an LBS (e.g., Google maps) which suggests the fastest routes; due to varying traffic conditions the routes may change over time. Obviously, if the LBS is malicious, it can reconstruct with high probability the entire client list, by observing frequent queries that originate at Acme. To avoid this, queries are issued through an Anonymizer, which implements Spatial k-anonymity. Figure 1.a shows a map of the current users; u is the querying user. The Anonymizer attempts to generate a small ASR[1]*. Assuming $k = 3$ the ASR is the gray rectangle that contains $\{u, u_1, u_2\}$. Unfortunately, the ASR contains only Acme's employees; therefore, the malicious LBS is sure that the query originated at Acme.*

The previous example demonstrates that although k-anonymity is satisfied, the group privacy is violated. This happens because all users inside Acme's headquarters are (with high probability) Acme's employees, therefore they belong semantically to the same group. Motivated by this, we propose the concept of *Location Diversity*, which guarantees that each query can be associated with at least ℓ semantically different locations. This is shown in Figure 1.b, where the Anonymizer sends to the LBS a packet of $\ell = 3$ queries originating at u, u_h and u_s. The semantic locations of these users are Acme, Hospital and School,

[1] Small ASRs are desirable for efficient query processing [10].

respectively. Therefore, an attacker cannot identify with probability larger than $1/\ell$ that the query comes from Acme.

A second case involves LBS applications where the user's true identity is required. Such applications (e.g., location-based social networking[2]) provide the user with personalized services according to her profile. Spatial k-anonymity is not applicable in this case, because it requires pseudonyms. On the other hand, our approach can effectively protect user's location privacy, as shown in the next example.

Example 2 (Personalized services). Assume that u_3 (Figure 1.b) visits frequently a hospital because of her chronic disease. u_3 subscribes to an LBS, which provides a social networking service that allows u_3 to see her friends who are currently near her. However, u_3 believes that hospital is a sensitive location and she does not want the LBS (attacker) to learn her semantic location when she is using the service in the hospital. Obviously, Spatial k-anonymity cannot be used because the true identity of u_3 is needed in order to identify her friends. On the other hand, Location Diversity can be successfully employed. Assuming $\ell = 3$, the Anonymizer sends to the LBS a packet of 3 queries, containing u_3 plus two possible locations from two semantically different sites (e.g., $\{u_3, u_5, u_s\}$). From the query, the LBS thinks that possibly u_3 is in shopping mall, or school, or hospital. Therefore, the LBS can only assume that u_3 is in a hospital with probability at most $1/3$.

Location Diversity is similar to ℓ-diversity [15] which is used to anonymize relational data. However, similar idea has never been used in Location-based services. In summary, our contributions are:

- We define the concept of semantic locations and identify cases where Spatial k-anonymity fails to protect privacy. We propose Location Diversity, which protects against such threats by grouping together sets of ℓ semantically different locations.
- We model the attacker's knowledge as a system of linear equations that represent the observed queries. We show that a necessary condition to preserve privacy, is to have infinite solutions for all variables of the equation system.
- Based on our attack model, we develop algorithms that implement Location Diversity while minimizing the query processing cost. Our experimental results show that Location Diversity provides superior privacy protection compared to Spatial k-anonymity without significant computational and communication overhead.

The rest of this paper is organized as follows: Section 2 overviews the related work. Section 3 formulates Location Diversity, whereas Section 4 discusses the attack model. Section 5 presents our anonymization algorithms. The experimental results are presented in Section 6. Finally Section 7 concludes the paper with directions for future work.

[2] For example, MyLoki: http://my.loki.com/

2 Related Work

k-anonymity [17] was initially studied in the context of relational databases, where data must be published without revealing the identity of individual records. Even if the identifying attributes (e.g., *name*) are removed, an attacker may identify specific persons by using combinations of other attributes (e.g., $\langle zip, sex, DoB \rangle$), called quasi-identifiers (QI). A table is anonymized if each record is indistinguishable from at least $k-1$ other records with respect to the QI. A common form of anonymization is generalization, which involves replacing specific QI values with more general ones (e.g., $city \rightarrow state$). Several generalization algorithms have been proposed [13][18] [8].

In some cases k-anonymity is not sufficient. Consider an anonymized database where all tuples in a group have the same value for a sensitive attribute (e.g., *disease*). By associating a person with that group, an attacker will know the disease of that person, even though k-anonymity prevents the association of a specific tuple with the person. ℓ-diversity [15] solves this problem by requiring that each anonymized group contains at least ℓ "well-represented" sensitive attribute values.

Spatial k-anonymity is an adaptation of the relational methods in the context of Location-based Services (LBS). The aim in this case is to hide the association between a specific user and a query submitted to the LBS. Most approaches follow the architecture of Figure 2, which requires a *trusted* Anonymizer between the user and the LBS. The Anonymizer maintains the current location of all users and is responsible for (i) hiding the ID of the querying user u (i.e., by using a pseudonym), and (ii) replacing the original query with an anonymizing spatial region (ASR) that contains u and at least $k-1$ other users near u; this process is called *spatial cloaking*.

Several cloaking methods have been proposed. Clique Cloak [5] combines spatial with temporal cloaking. Each query q specifies a temporal interval δt that the corresponding user u is willing to wait. If within δt, the Anonymizer finds $k-1$ more clients in the vicinity of u that also issue queries, all these queries are combined in a single ASR; otherwise, q is rejected. In Interval Cloak [9], on the other hand, the Anonymizer maintains the current locations of all users in a Quad-tree. Once it receives a query from u, it traverses the tree (top-down) until it finds the quadrant that contains u and fewer than $k-1$ users. Then, it selects the parent of that quadrant as the ASR. Casper [16] is also based on Quad-trees. The Anonymizer builds a hash table on the user ID pointing to the lowest-level quadrant where the user lies. Thus, each user is retrieved directly, without having to access the tree top-down. Casper generates smaller ASRs compared to Interval Cloak, by considering the quadrants at the same level of the tree before ascending to the parent node. Due to the smaller ASRs, the computational and communication cost during query processing are lower.

Interval Cloak and Casper do not guarantee privacy when the distribution of user locations is skewed, since the outlier users are easily exposed [10]. To solve this problem, Kalnis et al. [10] formulated the reciprocity property, which states that, if a user u_j belongs to the ASR of u_i, then u_i belongs to the ASR of u_j.

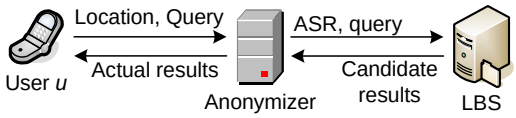


Fig. 2. Architecture for Spatial k-anonymity

They also developed the Hilbert Cloak algorithm, which uses the Hilbert space filling curve [1] to map the 2-D space into 1-D values. The algorithm partitions the 1-D list into groups of k users (the last group may have up to $2k-1$ users). For a querying user u the algorithm finds the group where u belongs, and returns the minimum bounding rectangle of that group as the ASR. The same ASR is returned for any user in a given group; therefore Hilbert Cloak is reciprocal. A similar method has also been implemented on a Peer-to-Peer system [7]. Chow and Mokbel [4] extend reciprocity to continuous queries involving moving users.

The above-mentioned methods assume that the Anonymizer maintains the current locations of numerous users. However, in practice this is problematic because: (*i*) Privacy can be compromised if some users are not trustworthy (e.g., the attacker deploys decoys at known locations). (*ii*) The cost of location updates is high and (*iii*) it is unlikely that a large number of mobile users will accept such cost in constant basis, for the sporadic benefit of issuing an anonymous query. For these reasons, recent methods do not require an Anonymizer to maintain the user locations. In Probabilistic Cloaking [2] the ASR is a region around the query point. Given an ASR, the LBS returns the probability that each candidate result satisfies the query based on its location with respect to the ASR. Space Twist [18] is another approach where the user sends a fake query, called anchor, to the LBS. The LBS progressively returns the nearest-neighbors of the anchor to the user, who computes her own nearest neighbor from those results. Conceptually, the ASR in this case corresponds to an irregular shape that models the information leak during query processing. Similar to these methods, we also avoid storing the current user locations at the Anonymizer and prefer to send to the LBS fake user locations when necessary.

None of the previous methods considers the semantic locations that are included in the ASR, so they cannot be used in our problem. A possible solution is to encrypt the location of the user, before sending the query to the LBS. This approach is followed by Khoshgozaran and Shahabi [11], who employ the Hilbert transformation together with encryption to conceal both the spatial data and the queries from the LBS; their system supports approximate nearest-neighbor queries. A serious drawback is that all users must be trustworthy. Ghinita et al. [6] provide much stronger privacy by employing Private Information Retrieval [12]. The user does not need to trust anybody (e.g., Anonymizer or other users) and the LBS does not gain any information about the query. Unfortunately, the system supports only 1-Nearest-Neighbor queries; moreover the computational and communication cost are very high.

3 Location Diversity

In Section 1 we mentioned that the location of a user carries contextual information. For example, a user who is currently at a shopping mall is most probably in the process of shopping; a user at a girl's school is a young female student with high probability. Below is the definition of semantic location, which will be used throughout the rest of the paper:

Definition 1 (Semantic Location). *A semantic location is a region on the map that typically gathers user population that has similar contextual information, such as age, gender, activities, etc. Examples of semantic locations are schools, hospitals, a golf course, the office of a specific company, etc.*

Let SL be the set of all semantic locations (e.g., $SL = \{school, hospital, Acme\}$). Each semantic location may have many instances. We denote by $T(L_i)$, $L_i \in SL$ the set of all instances of semantic locations that belong to type L_i. For example, if $L_i \equiv$'*school*' and there are three schools on the map, then $T($'*school*'$) = \{school_1, school_2, school_3\}$. Obviously, $T(L_i) \neq \emptyset$.

One question is when should we classify two semantic locations into the same category. For example, German and English schools could both be classified under 'school', but they could also form two different categories. Such classification is application dependent and is outside the scope of this paper. We assume that the categories of semantic locations are given as a-priori knowledge to both the attacker and the Anonymizer.

Let SQ be the set of all spatial queries submitted to the LBS and let $Q_i \in SQ$. Each Q_i is associated with the semantic location of the querying user. For example, if a user asks Q_i from Acme's headquarters, then the query is associated with 'Acme'. The attacker observes all queries; therefore, he can construct estimations about the distribution of each query for each semantic location. Formally, $\forall L_i \in SL$ the attacker estimates the distribution $D_{L_i}(q)$, where q is a random variable taking values from SQ. Through these estimations the attacker may breach privacy; this is demonstrated in the following example:

Example 3. Let $SL = \{L_1, L_2, L_3\}$, $SQ = \{Q_1, Q_2, Q_3\}$ and assume that the attacker observes 100 queries whose distribution is shown in Table 1. The attacker can generate association rules in the form $L_j \Rightarrow Q_i$. Assume $L_1 \equiv$ 'Acme' and $Q_1 \equiv$ 'Find the fastest route to 107 River Ave'. Since this query is sent from 'Acme' with high probability (i.e., 89%) the attacker can assume that the '107 River Ave' address corresponds to an Acme's client; the name of the client can be found though a white pages service.

Observe that the same query Q_i can be asked from several semantic locations; we denote the set of these locations by QL_i. In the previous example, $QL_1 = \{L_1, L_3\}$ corresponds to query Q_1. Formally, a location L_j belongs to QL_i if the probability that the query Q_i was asked from L_j is greater than zero. Formally:

Definition 2 (Weak Location Diversity). *A query Q_i exhibits Weak Location Diversity if $|QL_i| \geq \ell$ (i.e., Q_i is associated with at least ℓ semantic locations). ℓ is a system-wide privacy parameter.*

Table 1. Queries collected by the attacker

Location	Query	Count	Rule	Conf.	Sup.
L_1	Q_1	40	$L_1 \Rightarrow Q_1$	89%	40%
L_1	Q_2	10	$L_1 \Rightarrow Q_2$	50%	10%
L_2	Q_2	10	$L_2 \Rightarrow Q_2$	50%	10%
L_2	Q_3	10	$L_2 \Rightarrow Q_3$	29%	10%
L_3	Q_1	5	$L_3 \Rightarrow Q_1$	11%	5%
L_3	Q_3	25	$L_3 \Rightarrow Q_3$	71%	25%

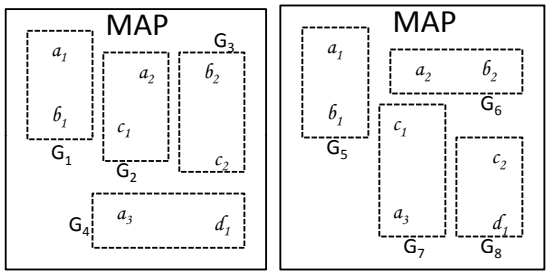
Refer to the example of Figure 1.b: To implement Weak Location Diversity, the querying user u sends her query Q_u to the Anonymizer. As mentioned above, the Anonymizer has a-priori knowledge of the set $\bigcup_{\forall L_i \in SL} T(L_i)$ (i.e., all instances of semantic locations). The Anonymizer selects a set QL_u such that QL_u contains the location of u as well as $\ell - 1$ other semantic locations; in the example, $QL_u = \{Acme, Hospital, School\}$. Each semantic location represents a large spatial region. From each region, the Anonymizer selects a user who represents a query point. Obviously, u is selected from the 'Acme' region. For the other two regions, the Anonymizer selects two random points which correspond to fake users[3] u_h and u_s. The fake users are selected because the Anonymizer does not maintain the exact user locations (see Section 2). The three query points (i.e., u, u_h, u_s) are packed together and sent to the LBS, which executes three separate queries and returns all results to the Anonymizer. Next, the Anonymizer filters out the false positives and returns the actual result to u.

Weak Location Diversity guarantees privacy only if each semantic location has a single instance (i.e., $\forall L_i \in SL \Rightarrow |T(L_i)| = 1$). In Section 4 we will show that, if the previous condition is not satisfied, the Cross Group Inference Attack can compromise privacy. Motivated by this, we propose *Strong Location Diversity* which is not vulnerable to the Cross Group Inference Attack.

Definition 3 (Strong Location Diversity). *An anonymization method exhibits Strong Location Diversity if, for every query $Q_i \in SQ$ and every semantic location $L_j \in SL$, the probability of associating Q_i with L_j is at most $1/\ell$.*

If a method satisfies the Strong Location Diversity property, then every query anonymized by that method satisfies Weak Location Diversity; the opposite is not true. As a consequence, it is easy to find a solution for the Weak version; the only requirement is that there exist at least ℓ different semantic locations (i.e., $|QL_i| \geq \ell$). In contrast, depending on the number of instances per semantic location (i.e., $|T(L_i)|$), there are cases where no solution exists for Strong Location Diversity. In Section 5 we develop algorithms that generate solutions in between the two extremes.

[3] In general, this does not raise any privacy issue, unless the attacker knows the actual locations of all users; however, we believe that such an assumption is unrealistic.



(a) Vulnerable grouping (b) Robust grouping

Fig. 3. Cross group inference attack

4 Cross Group Inference Attack

In this section, we show an attack (*CGIA*) which demonstrates that the Weak Location Diversity may not always guarantee privacy.

Remember that in our definition of semantic location, there can be multiple instances of the same semantic location. For example, several branches of the insurance company Acme (as in example 1) located at different places in the city are all instances of the Acme. Since these branches belong to the same company, it is possible that they share the same client list. Compare to other instances of other semantic locations, these branches tend to send similar queries because of the same client list. In the worst case, they produce very similar query patterns in long term. Remember that the client list is the secret business asset to Acme, and the query patterns at any branches would help the attacker to reconstruct the client list. In the following, we present a worst case analysis on the information gain of the attacker under the worst case assumption that instances of the same semantic location produce very similar query patterns.

Consider the example of Figure 3.a, where $SL = \{A, B, C, D\}$ is the set of semantic locations. $T(\text{`}A\text{'}) = \{a_1, a_2, a_3\}$, $T(\text{`}B\text{'}) = \{b_1, b_2\}$, $T(\text{`}C\text{'}) = \{c_1, c_2\}$ and $T(\text{`}D\text{'}) = \{d_1\}$ are the instances of the corresponding semantic locations. The figure shows the groups that are used for anonymization. For example group $G_1 = \{a_1, b_1\}$ indicates that, if the querying user is in the region of a_1 or b_1, then the anonymizer will construct a query packet that includes fake users from a_1 and b_1. Observe that all groups contain two distinct semantic locations, therefore they satisfy Weak Location Diversity for $\ell = 2$. The intuition of CGIA is that, although the attacker cannot infer any information from a single group, he is able to reconstruct the query patterns of individual semantic locations by examining the query history.

Let $Q \in SQ$ be a particular query of the attacker's interest. By the definition of semantic locations, we expect that the query distribution in all instances of a semantic location is similar. Under the worst case assumption, the number of Q asked from the instances of the same semantic location are the same. For example, a_1, a_2, a_3 each asks the same number of Q in the worst case. Specifically,

let A^Q be the number of times that Q was asked from an instance of semantic location A (i.e. a_1 or a_2 or a_3). Similarly, we define B^Q, C^Q and D^Q for the rest of the semantic locations. Moreover, let G_i^Q be the total number of times that Q was asked from all semantic locations of group G_i. Then, from Figure 3.a we form the following system of linear equations:

$$\begin{cases} A^Q + B^Q = G_1^Q = z_1 \\ A^Q + C^Q = G_2^Q = z_2 \\ B^Q + C^Q = G_3^Q = z_3 \\ A^Q + D^Q = G_4^Q = z_4 \end{cases} \tag{1}$$

The system has a unique solution:

$$\begin{cases} A^Q = \frac{z_1+z_2-z_3}{2} \\ B^Q = \frac{z_1-z_2+z_3}{2} \\ C^Q = \frac{z_2-z_1+z_3}{2} \\ D^Q = \frac{2z_4-z_1-z_2+z_3}{2} \end{cases} \tag{2}$$

Since the attacked knows $z_{1...4}$ (by observing the query history), he can calculate A^Q, B^Q,C^Q, and D^Q. Thus, he can identify the probability density function of query Q for all semantic locations. By repeating the process for other queries in SQ, the attacker can construct a table similar to the one of Example 3 (see Section 3). Therefore, the grouping of Figure 3.a is vulnerable under CGIA, although Weak Location Diversity is satisfied.

A different grouping is shown in Figure 3.b. Using the same methodology, we obtain the following system:

$$\begin{cases} A^Q + B^Q = G_5^Q = z'_1 \\ A^Q + B^Q = G_6^Q = z'_1 \\ A^Q + C^Q = G_7^Q = z'_2 \\ C^Q + D^Q = G_8^Q = z'_3 \end{cases} \tag{3}$$

The solution is:

$$\begin{cases} A^Q = x, x \in \mathbb{N} \\ B^Q = z'_1 - x \\ C^Q = z'_2 - x \\ D^Q = z'_3 - z'_2 + x \end{cases} \tag{4}$$

where x is a free variable. Since A^Q, B^Q, C^Q and D^Q all depend on x, the attacker cannot calculate the number of times that Q is associated with a specific location. Therefore, the grouping of Figure 3.b is robust against CGIA.

In general, consider a map with N groups $G_1, G_2, \ldots G_N$ and let $L_i \in SL$, where SL is the set of semantic locations. The attacker can construct a system of N linear equations for every query $Q \in SQ$. The form of the j-th equation that corresponds to a query Q is:

$$\begin{aligned} &\sum_{i=1}^{|SL|} h \cdot L_i^Q = G_j^Q, 1 \leq j \leq N \\ &h = \begin{cases} 1 \text{ if } L_i \in G_j \\ 0 \text{ otherwise} \end{cases} \end{aligned} \tag{5}$$

In a consistent system of linear equations, if the rank of the system is the same as the number of variables, then there is a unique solution. In this case, the grouping is vulnerable to CGIA. On the other hand, if the rank is smaller than the number of variables, then the system has infinite number of solutions. However, there are cases where the attacker can still compromise privacy, because some of the variables may have unique solutions. Consider the following example:

$$\left(\begin{array}{cccc|c} 1 & 0 & 0 & 1 & z_1 \\ 0 & 1 & 0 & 0 & z_2 \\ 0 & 0 & 1 & 1 & z_3 \end{array}\right) \tag{6}$$

The above is the reduced row-echelon form of a linear system after Gauss-Jordan elimination. Although the system has an infinite number of solutions, the second row has only one none-zero entry; therefore the second variable has a unique solution, rendering the grouping vulnerable to CGIA. Based on this observation, be propose the following definition:

Definition 4 (Robust system). *A system of linear equations is robust if every equation in the reduced row-echelon form has two or more non-zero entries.*

If a grouping generates a robust system, then it is guaranteed that the attacker cannot calculate the exact number of queries for any semantic location. Nevertheless, the following example shows that, even with a robust system, the attacker can gain *probabilistic* information about the query distribution:

5 Implementing Location Diversity

5.1 Assumptions about the Attacker

We assume that in the worst case the attacker is the LBS; obviously, in this case the attacker has complete knowledge about the map. The attacker will also keep the history of all user queries and observe the incoming queries. We assume that the attacker is semi-honest, which means that he is only curious in gaining privacy information from the history of user queries, but not cheat the users in query answering. The attacker has the following goals: 1) Infer the semantic location where a query is generated from. 2) Learn the query pattern of a semantic location, i.e., $D_{L_i}(q)$ for a semantic location under category L_i.

5.2 System Architecture

The system utilizes the same architecture as in spatial k-anonymity (see Figure 2). Instead of generating ASRs the Anonymizer injects Location Diversity into the queries. The Anonymizer will send ℓ or more point queries to the LBS server instead of a single query with an ASR region. The Anonymizer needs to keep the semantic information of the map. Before it can provide Anonymization service to the users, it runs the grouping algorithm so that groups of semantic locations can be formed.

5.3 Grouping Algorithms

The grouping algorithms form groups among semantic locations so that each group contains at least ℓ semantic locations from different categories. As we have shown in Section 4, certain groups are vulnerable to CGIA. A good grouping algorithm should take the attacker's capability into consideration, and form groups that result to robust system of linear equations. Our algorithm is doing so by finding maximum number of common groups in the map. Common groups refer to the groups that have members from common categories. Since each common group is a repetition of another, it gives no extra information to the attacker. In terms of forming system of linear equations, this methodology finds common equations that can be canceled during the reduction (e.g., Gauss-Jordan elimination), so that the rank of the system can be minimized.

Algorithm 1 depicts how the grouping algorithm works. We use $T(L_i)$ to represent the vector structure that enumerates all the instances under category L_i. We store all $T(L_i)$ in an array $CatList$. In each loop, we first sort $CatList$ in descending order according to the number of instances in each $T(L_i)$. Then we take one instance from each of the ℓ first $T(L_i)$ in the array and form a group. Once the group is formed, we remove the corresponding semantic location instances from $T(L_i)$. We label this group as the leading group. We continue to check if the instances from the first ℓ $T(L_i)$ could form a common group with the leading group. If this is true, we form a new group. We continue adding groups until no common group with the leading group can be found; then we go back to the beginning of the loop. The loop stops when all $T(L_i)$ are empty. It is possible that the $CatList$ has less than ℓ non-empty $T(L_i)$ left, meaning that we cannot form a group with semantic locations from ℓ categories. If it this case, we form a group with as many different semantic locations as possible, and we combine it with another group that already has ℓ distinct semantic locations. In this way all groups satisfy the Location Diversity property. Figure 4 is the visualization of $CatList$ at the beginning. a_i to f_i are instances of categories L_1 to L_6 respectively. The $CatList$ is in sorted order. With ℓ= 3, $\{a_1, b_1, c_1\}$ in the box of dotted line forms the leading group. Next, four more groups are generated: i.e., $\{a_2, b_2, c_2\}$, $\{a_3, b_3, c_3\}$, $\{a_4, b_4, c_4\}$ and $\{a_5, b_5, c_5\}$; these groups contain instances from the same semantic locations as the leading group. After these four groups have been found, the algorithm sorts $CatList$ again and enters another loop.

Algorithm 1 Grouping Algorithm

```
1: sortCatList(catList)
2: length ← 0, width ← 0
3: for i ← L − 1 to 0 do
4:    if catList[i].size = 0 then
5:       continue
6:    end if
7:    length ← i + 1, width ← catList[i].size
8:    break
9: end for
10: if width = 0 and length = 0 then
11:    return
12: else
13:    for i ← 0 to width − 1 do
14:       new eqn
15:       for j ← 0 to length − 1 do
16:          eqn.add(catList[j].remove(0))
17:       end for
18:       eqnList.add(eqn)
19:    end for
20:    go to line 1
21: end if
```

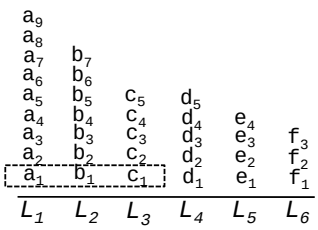


Fig. 4. Grouping algorithm

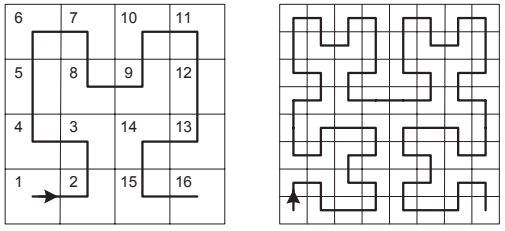


Fig. 5. $2^2 \times 2^2$ and $2^3 \times 2^3$ Hilbert curves

5.4 Minimization of Group Area

We prefer to form groups whose members are closer to each other, as geographically closer semantic locations tend to return common answers. Finding the optimal grouping is difficult, as it is a typical combinatorial problem. In the following, we introduce two heuristics to the grouping algorithm that aim to minimize the distance between group members by swapping group memberships of semantic locations that are in the same category while preserving Location Diversity. We use the area of the Minimum Bounding Region (MBR) that contains all members of a group as the metric of the relative distances between members, i.e., smaller MBR implies better grouping.

5.5 Greedy Optimization

Suppose $CatList$ is in the sorted order, and $CatList[j]$ refers to the j-th ($j \geq 0$) element of $CatList$. The greedy optimization can be applied when the grouping algorithm is finding a common group with the leading group. In greedy optimization, the first member of the new group is chosen to be the first instance in $CatList[0]$. Subsequently, the $(j+1)$-th member of the new group is chosen from $CatList[j]$ which forms smallest area of MBR with the 1-st to the j-th members in the group. For example, in Figure 4, a_1 is selected as the first member of the new group, whereas b_1 is selected only if it forms smallest MBR with a_1 among all the b_i. Similarly, c_1 is selected only if it forms smallest MBR with a_1 and b_1 among all the c_i. The pseudo code of the greedy optimization is given in Algorithm 2. The time complexity of the grouping algorithm with Greedy optimization is $O(N^2)$, where N is the number of instances of semantic locations.

Algorithm 2 Greedy Optimization

```
1: function greedySelect(catList, width, length)
2: for i ← 0 to width − 1 do
3:    new eqn
4:    for j ← 1 to length − 1 do
5:       minArea ← +∞, minIndx ← −1
6:       for k ← 0 to catList.size − 1 do
7:          area = computeArea(catList, i, j, k)
8:          if area < minArea then
9:             minArea ← area, minIndx ← k
10:         end if
11:      end for
12:      eqn.add(catList[j].get(minK))
13:      catList[j].remove(minK)
14:   end for
15:   eqnList.add(eqn)
16: end for
```

5.6 Hilbert Optimization

Another algorithm with better time complexity is Hilbert optimization. The Hilbert optimization is based on 2-dimensional Hilbert curve (see Figure 5). One of the properties of Hilbert curve is that, if two points are close to each other in multi-dimensional space, they also tend to be close in the one dimensional space generated by the Hilbert curve. With Hilbert curve, we can divide a square map into $2^n \times 2^n$ number of squares, where n is called the level of Hilbert curve. For example, in Figure 5(left) , a square map with the level 4 Hilbert curve is divided into $2^2 \times 2^2$ squares. Following the Hilbert walk, each of the square is assigned an ID with increasing number. By making using of Hilbert curve, we can sort all the semantic locations under the same category in increasing order of their ID before running the grouping algorithm. For example, in Figure 4 each column is sorted according to Hilbert order. Large Hilbert level will result in smaller squares in the map. Imagine when the Hilbert level is infinitely large each of the square becomes a point in the map, the accuracy will be improved. The time complexity of grouping algorithm with Hilbert optimization is $O(N)$.

6 Experimental Evaluation

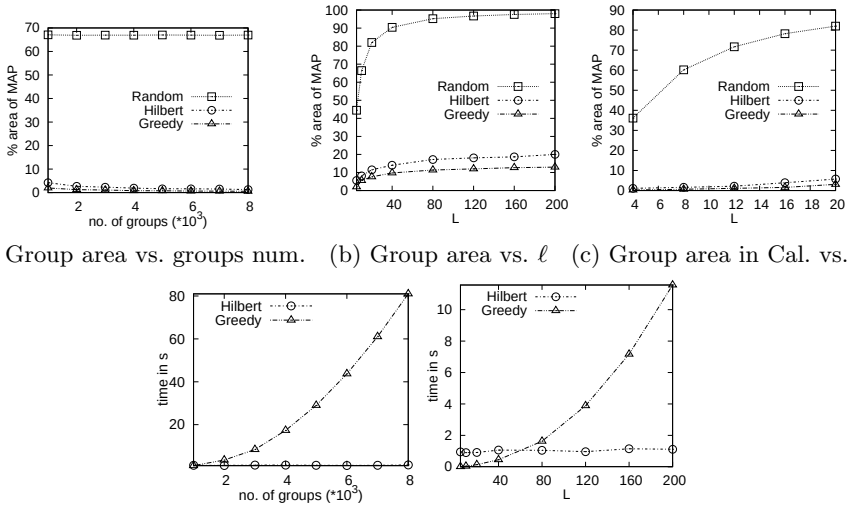
In this section we evaluate the performance of our algorithms. All experiments were conducted on a Intel-Duo 2.33GHz CPU with 3.25GB RAM and Windows XP. Each experiment was run 10 times; the figures show the average values of the results. We used a variety of synthetic datasets; their parameters are explained in the corresponding experiments. We also used a real dataset consisting of points of interest (POIs) in California[4]. The dataset is classified in 63 categories (i.e., set of semantic locations SL) and contains $104,770$ landmarks (i.e., instances of semantic locations). Examples of such semantic locations are airport, church, dam, park, and etc. Each semantic location is defined by a pair of longitude and latitude values. Another useful resource of classifications of locations can be found in NAICS[5] website.

6.1 Performance of Grouping Algorithms

In the first experiment we measured the average area of the anonymization groups. Recall that we prefer groups with small area, since they incur less query processing time and the number of candidate results (which is proportional to the communication cost) is smaller. As a base-line competitor, we implement an algorithm (call *Random*) that selects randomly the semantic locations of each group.

First we used synthetic data, where we generated randomly the semantic locations in the map. In Figure 6.a we fixed $\ell = 10$ and vary the number of groups of sensitive locations between $1,000$ and $8,000$. Both Hilbert and Greedy create

[4] Available at: http://www.cs.fsu.edu/ lifeifei/SpatialDataset.htm
[5] Available at http://www.census.gov/eos/www/naics/



(a) Group area vs. groups num. (b) Group area vs. ℓ (c) Group area in Cal. vs. ℓ

(d) Time vs. groups num. (e) Time vs. ℓ

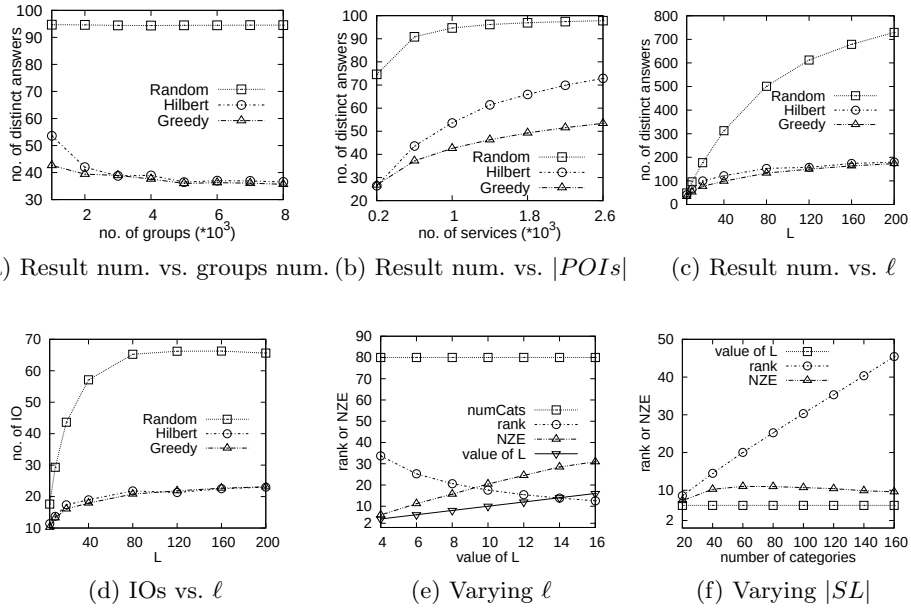
Fig. 6. Group area comparison and California POIs

groups with much smaller areas than Random. In Figure 6.b we set $|SL| = 1,000$ and vary ℓ; the trends are similar to the previous graph. We also performed the same experiments with the California dataset. The results are shown in Figure 6.c; the trends are similar to those of the synthetic data.

Using the random data, we measured the time to generate the groups. Figures 6.d and 6.e show that the running time for Hilbert tends to be almost unaffected by number of groups and ℓ. Greedy, on the other hand, grows roughly quadratically with number of groups and ℓ. Therefore, there is a tradeoff between the two algorithms: Greedy generates groups with slightly smaller area, but it is considerably slower than Hilbert.

6.2 Query Processing Cost at the LBS

In this set of experiments we evaluated the impact of the grouping algorithms on query processing. Again, we used the synthetic maps, where we generated 2,600 POIs; the POIs are indexed by an R*-tree. The LBS implements nearest-neighbor queries; for each request, it returns the 10 nearest POIs. Figures 7.a and 7.b show the results for varying number of groups and number of POIs, respectively. We observe that the number of answers that correspond to Hilbert and Greedy groupings is significantly smaller than those for Random; the number of answers is proportional to the communication cost. A similar trend is observed in Figure 7.c, where we vary ℓ. Moreover, Figure 7.d shows the I/O cost (i.e., number of disk accesses) for varying ℓ; again the trend among grouping algorithms is the same. This is because the number of distinct answers is typically proportional to the number of I/Os.



(a) Result num. vs. groups num. (b) Result num. vs. $|POIs|$ (c) Result num. vs. ℓ

(d) IOs vs. ℓ (e) Varying ℓ (f) Varying $|SL|$

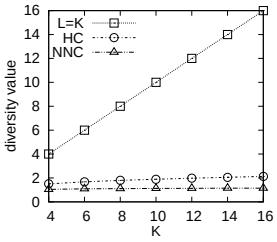
Fig. 7. Query processing cost and privacy

6.3 Evaluation of Privacy Guarantees

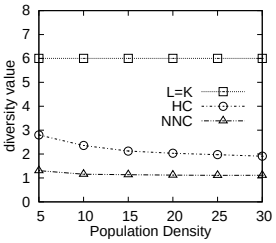
The following experiments evaluate the privacy guarantees of our algorithms. We focus on the rank and the number of non-zero entries (NZE) in the reduced row-echelon form of the linear system, since they characterize the ability of the attacker to find a good estimation of the real associations of queries with semantic locations. We randomly generated $|SL| = 80$ semantic locations; the number of instances for each semantic location was picked randomly and varies between 0 and 400. Using our grouping algorithms, a system of linear equations with 80 variables is generated. Thus, the maximum number of NZEs in a row and the rank, cannot exceed 80. Figure 7.e shows that by using our grouping algorithms, the rank of the linear system becomes significantly less than 80, and only increases slowly with ℓ (low rank corresponds to better privacy). Moreover, the average number of NZEs per equation is always above 2, which is the threshold to the privacy violation. In Figure 7.f we fix $\ell = 6$ and vary $|SL|$. Observe that although the rank increases, it is still much smaller than the number of variables (by definition the number of variables is equal to $|SL|$). Also, the average number of NZEs is always above 2.

6.4 Worst-case Privacy evaluation

The previous graphs indicate only the average privacy guaranty of our grouping algorithms. Here, we investigate the worst case performance. In Figure 8.c we



(a) Diversity vs. k



(b) Diversity vs. pop. density

# of Single NZE		Number of Categories			
		40	80	120	160
L	4	2	0	0	0
	8	0	0	0	0
	12	0	0	0	0
	16	0	0	0	0

(c) Worst case analysis

Fig. 8. Comparison against k-anonymity and worst case analysis

consider different combinations of ℓ and $|SL|$; for each such combination, we randomly form 10,000 systems of linear equations. The table shows the number of compromised systems (i.e., systems that contain at least one equation with a singe NZE). Only 2 such cases where found; this is an indication of the robustness of our algorithms.

6.5 Comparison against k-anonymity

In the last experiment we compare Location Diversity with Spatial k-anonymity. We implemented two recent k-anonymity algorithms, namely Hilbert Cloak (HC) and Nearest Neighbor Cloak (NNC) [10], and we used the California dataset. Users are randomly generated around each semantic location; each users belongs to the semantic location nearest to her current location. We measure diversity as the number of distinct semantic locations in a group, over the total number of semantic locations in the same group; we take the average of all groups.

Figure 8.a depicts the diversity value for varying k. The line $\ell = k$ corresponds to the ideal case (i.e., ℓ distinct semantic locations per group). Our algorithms are always identical to this line; for clarity, they are not shown in the graph. On the other hand, the diversity value of HC and NNC is much lower than the ideal one. Similarly, in Figure 8.b we fix $k = \ell = 6$ and vary the user population density. We found that the diversity value of HC and NNC decreases. This result is expected because, as the population increases, HC and NNC can find groups with smaller cloaking areas.

7 Conclusions

In this paper, we focused on privacy in location-based services and identified two privacy threats (i.e., group privacy violation and privacy in personalized services), which are not handled by existing Spatial k-anonymity algorithms. We proposed the concept of Location Diversity, which solves these problems. We also showed that Location Diversity is not equivalent to relational ℓ-diversity, since the latter is vulnerable to the cross group inference attack. Finally, we developed algorithms for Location Diversity.

Our work is an initial step towards solving the above problems, and faces some limitations. The most important one is that it assumes a static set of sensitive locations. Therefore, it may report that a fake user is in a library at middle night which is unlikely to be true in reality. Moreover, a solution that supports location diversity without using a trusted anonymizer is also interesting for research. These issues will be targeted in our future work.

References

1. A.R. Butz: Alternative Algorithm for Hilbert's Space-Filling Curve. In: Trans. on Computers
2. R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar: Preserving user location privacy in mobile data management infrastructures. In: Intl. Workshop on Privacy Enhancing Technologies
3. F. Chin and G. Ozsoyoglu: Auditing and Inference Control in Statistical Databases. In: Trans. on Software Engineering
4. C.-Y. Chow and M. F. Mokbel: Enabling Private Continuous Queries for Revealed User Locations. In: Proc. of SSTD
5. B. Gedik and L. Liu: Location Privacy in Mobile Systems: A Personalized Anonymization Model. In: Proc. of ICDCS
6. G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan: Private Queries in Location Based Services: Anonymizers are not Necessary. In: Proc. of SIGMOD
7. G. Ghinita, P. Kalnis, and S. Skiadopoulos: PRIVE: Anonymous Location-based Queries in Distributed Mobile Systems. In: Proc. of WWW
8. G. Ghinita, P. Karras, P. Kalnis, and N. Mamoulis: Fast Data Anonymization with Low Information Loss. In: Proc. of VLDB
9. M. Gruteser and D. Grunwald: Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In: Proc. of USENIX MobiSys
10. P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias: Preventing Location-Based Identity Inference in Anonymous Spatial Queries. In: TKDE
11. A. Khoshgozaran and C. Shahabi: Blind Evaluation of Nearest Neighbor Queries Using Space Transformation to Preserve Location Privacy. In: Proc. of SSTD
12. E. Kushilevitz and R. Ostrovsky. Replication is NOT needed: Single database, computationally-private information retrieval. In: Symp. on Foundations of Computer Science
13. K. LeFevre, D. J. DeWitt, and R. Ramakrishnan: Mondrian Multidimensional k-Anonymity. In: Proc. of ICDE
14. N. Li, T. Li, and S. Venkatasubramanian: t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In: Proc. of ICDE
15. A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam: l-Diversity: Privacy Beyond k-Anonymity. In: Proc. of ICDE
16. M. F. Mokbel, C. Y. Chow, and W. G. Aref: The New Casper: Query Processing for Location Services without Compromising Privacy. In: Proc. of VLDB
17. L. Sweeney: k-Anonymity: A Model for Protecting Privacy. In: Intl. Journal of Uncertainty, Fuzziness and Knowledge-Based Systems
18. J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A. W.-C. Fu: Utility-Based Anonymization Using Local Recoding. In: Proc. of KDD
19. M. L. Yiu, C. S. Jensen, X. Huang, and H. Lu: SpaceTwist: Managing the Trade-Offs Among Location Privacy, Query Performance, and Query Accuracy in Mobile Services. In: Proc. of ICDE