TRANSACTIONS ON DATA PRIVACY 3 (2010) 123–148

The PROBE Framework for the Personalized Cloaking of Private Locations

Maria Luisa Damiani*, **Elisa Bertino****, **Claudio Silvestri*****

*Universita degli Studi di Milano, **Purdue University,

***Istituto di Scienza e Tecnologie dell'Informazione - Consiglio Nazionale delle Ricerche

E-mail: damiani@dico.unimi.it,bertino@cs.purdue.edu,claudio@silv.eu

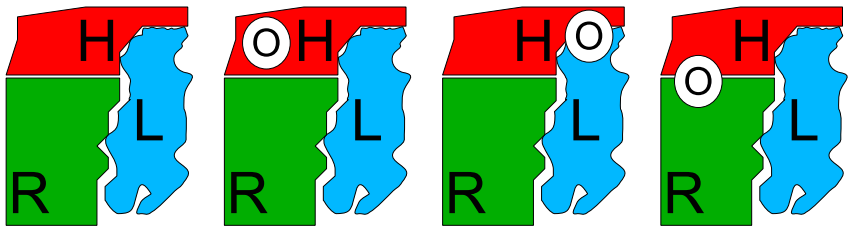
Abstract. The widespread adoption of location-based services (LBS) raises increasing concerns for the protection of personal location information. A common strategy, referred to as obfuscation (or cloaking), to protect location privacy is based on forwarding the LBS provider a coarse user location instead of the actual user location. Conventional approaches, based on such technique, are however based only on geometric methods and therefore are unable to assure privacy when the adversary is aware of the geographical context, in particular of the *semantic locations* and the statistical distribution of positions in the given space. This paper provides a comprehensive solution to this problem. We present a novel privacy model and an architectural framework for the personalized cloaking of semantic locations. In our model, a cloaked location is an uncertainty region which satisfies the privacy constraints specified by the user in the privacy profile(*obfuscated location*). We propose a strategy for generating obfuscated locations and evaluate different algorithms which implement efficiently such a strategy. The paper includes several experimental results assessing performance, storage requirements and accuracy for the approach. The paper also discusses the system architecture and shows that the approach can be deployed also for clients running on small devices.

Keywords. Location privacy, Geo-social networks, Spatial Databases and GIS

1 Introduction

The ever increasing collection of personal location data, pushed by the widespread use of location-sensing technologies, like satellite positioning systems, RFID and sensors, and the development of novel location-based services (LBS), motivates the great concern for the protection of location privacy. The communication of a user's position to a LBS provider upon a service request may result in the unauthorized dissemination of personal location data. Such data, combined with other available information, may in turn lead to the inference of sensitive information about individuals. Various approaches have been thus proposed to assure location privacy. Most of those approaches are based on obfuscation techniques that aim at disguising the actual position of the user by forwarding to the LBS

*This article is an extended version of a paper presented at the 2nd SIGSPATIAL ACM GIS 2009 International Workshop on Security and Privacy in GIS and LBS (SPRINGL 2009), Seattle WA, USA, Nov. 3, 2009.



(a) (b) (c) (d)

Figure 1: Geographical context

provider fake or less accurate (*generalized*) location information. Approaches based on k-anonymization refine obfuscation techniques by making sure that the generalized location of each user is indistinguishable from the generalized locations of other $k-1$ users.

A common problem to all the above approaches is that they do not take into account background knowledge that the adversary may have about the reference spatial context. We claim that by exploiting such knowledge, the adversary may be able to obtain more precise bounds about the actual user location, thus defeating the obfuscation mechanism. Another major drawback of such approaches is that they do not support location privacy preferences, that is, the specification of which locations are sensitive for which users. Not all locations may have the same sensitivity for all the users and therefore a suitable obfuscation mechanism should be able to generate obfuscation locations that are tailored to the privacy preference of each user. We believe that, as we move toward more personalized LBS, privacy should be one of key personalization dimensions. Before moving to introduce the key contribution of the paper, we introduce a running example to illustrate the location inferences that are possible when background geographical knowledge is available to the adversary.

Example 1. Assume that a user of a LBS is located within a hospital. Consider the geographical context in Figure 1.(a): the hospital H is close to a lake L and to a residential district R; all these places, i.e., the lake, the district and the hospital, cover a polygonal region. Suppose that no boats are allowed on the lake and that the adversary has this knowledge. Assume also that the actual position of the user is cloaked by region O containing the user's position.

From the observation of the spatial relationships existing between the obfuscated location and the spatial entities, like spatial containment, overlaps and disjointness, the adversary can easily infer whether the user is located in a sensitive place. In particular consider the following three cases:

i) The obfuscated location is spatially contained in the extent of the hospital (Figure 1.(b)). In this case, the adversary may easily infer that the user is located in a sensitive place, that is, the hospital, although the actual position is blurred to a coarser region.

ii) The region corresponding to the user's obfuscated location includes the extent of both the hospital and the lake (Figure 1.(c)). Since the user cannot be physically located inside the lake, because no boats are allowed on the lake, the only realistic position is within the hospital and thus the obfuscated location is still sensitive. Notice that in this case information about the user's obfuscated location is combined with publicly available information, i.e., that no boat is allowed on the lake, in order to infer more precise information about the actual location of the user.

iii) The region corresponding to the user's obfuscated location overlaps part of the hospital and part of the residential district (Figure 1.(d)). Since the hospital is the only sensitive place, the obfuscated location is "sensitive to some extent".

Suppose now that the user is a physician. In such case the fact of being located in the hospital is likely not to be sensitive for this user.

The example emphasizes the fact that a location, besides a geometric shape, has a semantics (e.g., hospital). Moreover locations are not all equally probable, namely some locations are more (or less) probable than others and this information is publicly known. Moreover, the example clearly shows that privacy breaches occur because existing obfuscation techniques are unable to prevent the inference of *semantic locations* which, depending on the perceptions of users, may represent sensitive information. The protection of sensitive location information thus requires techniques able to take into account the geographical context in which users are located, the statistical distribution of positions, as well as the users' privacy preferences. To our knowledge, a comprehensive approach to this problem has not been investigated yet. In this paper, built on [9], we take a major step in that direction and present a novel privacy model and methods for the personalized obfuscation of semantic locations.

1.1 The PROBE framework

We formalize the notions of sensitivity of a region and user's privacy profile and provide solutions for the efficient generation of cloaked regions satisfying the user's preferences (*obfuscated locations*). Moreover we take into account scenarios in which the distribution of positions is not uniform.

The choice of the sensitivity metric is crucial, and, indeed, different metrics can be devised. We define the sensitivity of an arbitrary region r with respect to a certain category of semantic locations (e.g., hospitals, religious buildings) as the probability that a user in r is inside any place of that category. The user specifies which categories of semantic locations are sensitive and the privacy preferences in the *privacy profile*. A privacy preference is a constraint on the maximum sensitivity of the cloaked region, tolerated by the user.

We propose a sound privacy-preserving strategy for the generation of obfuscated locations which develops in two stages: first an obfuscation algorithm generates off-line a set of obfuscated locations. Each of those regions includes both sensitive and non-sensitive semantic locations and satisfies the user's privacy constraints. Then, at run-time, the user's position is checked against the set of obfuscated locations. If the user is located inside any location r, r is disclosed in place of the exact position. Note that the obfuscation algorithm does not take into account the user's position. This way, an attacker cannot exploit the knowledge of the algorithm to infer more precise bounds over the user's position inside the larger region. The approach is thus robust against reverse engineering attacks.

We present different obfuscation algorithms and evaluate these algorithms on both synthetic and real data with respect to spatial accuracy, efficiency and storage requirements. The above elements are combined in the framework, referred to as PROBE (Privacy-preserving Obfuscation Environment). The PROBE framework can be flexibly deployed on either a two-tier architecture or, in alternative, on a three-tier architecture whenever the client devices have limited capabilities [13]. A prototype of the core component, the Obfuscation Engine, has been developed and implements functions for the specification of privacy profiles, and the generation and visualization of obfuscated locations.

1.2 Contributions and structure of the paper

In summary, the main contribution of the paper are:

- We define a privacy model for the personalized protection of sensitive semantic locations based on a probabilistic model of space.
- We specify a two-phases computational model for the generation of obfuscated locations which can be deployed on different architectures and use different obfuscation algorithms.
- We evaluate three obfuscations algorithms,$Sens_{Reg}$,$Sens_{Div}$ and $Sens_{Hil}$, all implementing a common heuristics, which differ for the granularity of the sensitive areas they obfuscate. Experimental results show that on small scale scenarios, e.g., limited number of sensitive objects, the three methods present similar characteristics. On larger scale scenarios, with stringent privacy constraints and large numbers of sensitive locations, $Sens_{Hil}$ provides more accurate regions and it is considerably more efficient. Moreover the size of obfuscated maps is very small and thus suitable for storage on small devices.

The rest of the paper is organized as follows: in Section 2, we investigate related work. We present the privacy model in Section 3 and the architectural framework in Section 4. In Section 5, we introduce the three obfuscation techniques. We evaluate experimentally the proposed techniques on synthetic and real data in Section 6. We discuss privacy threats in the privacy analysis in Section 7 and we conclude with some final remarks in Section 8.

2 Related work

Location privacy has become a prominent research topic in a variety of areas spanning from spatio-temporal data management to human-computer interaction and mobile computing [21, 18, 3]. A broad range of techniques exist which target location privacy in various application areas, including, LBS, sensor networks and spatio-temporal data publishing. We focus on privacy in LBS and relate our approach to the three major privacy paradigms developed in this area, i.e., *identity privacy*, *location privacy*, and *privacy policies.*

Identity privacy. The goal of the identity privacy techniques is to prevent the re-identification of seemingly anonymous users [2, 11, 15, 6, 19, 22, 20, 23, 35]. Individuals can be identified with great precision, based for example on the locations that they usually frequent such as homes, work places and historical sites, even though their identities are hidden [2]. A major privacy paradigm for the protection of identity is location k-anonymity [11, 15, 6, 19]. k-anonymity is achieved by generalizing the user's location so as to contain k individuals. Methods are developed to k-anonymize both single positions and users' trajectories [23, 33, 25]. On the other hand, location k-anonymity has been shown to have conceptual and architectural limitations. For example, it does not provide any protection to semantic locations, e.g., if a k-anonymous region falls inside a hospital, then one can infer that the k users in that location may have health problems. Moreover, privacy preferences can be only expressed through a number, the value of k, that is unclear how to set. In addition, k-anonymity commonly requires a costly infrastructure, including a trusted intermediary, the *anonymizer*, which computes the cloaked regions based on the knowledge of the position of the whole set of users.

Several techniques have been proposed to address such drawbacks. For example, Xue et al. [36] apply the notion of *l-diversity* so as to generate k-anonymous regions which contain at least l semantically different locations. However, users cannot specify which semantic locations are sensitive and the degree of sensitivity. Cryptographic techniques address architectural limitations. Ghinita et al. [14] have proposed a PIR-based approach to compute nearest-neighbor (NN) queries without disclosing any information to third parties about location and the requested points of interest. This solution however incurs high communication costs. Xu et al. [35] have proposed an approach in which the user can specify, in place of the value of k, a *public region* and request that her disclosed location must be at least as popular as that space.

Our approach is different, in that the goal is not to protect the user's identity, but rather the location. Nevertheless the location anonymization techniques are also of concern for the obfuscation of semantic locations. For example, one of the obfuscation methods that we present in this paper, the $Sens_{Hilb}$ algorithm, applies to the cells of a linear and gridded space organized as Hilbert filling curve. The Hilbert filling curve is also at the basis of the location anonymization technique proposed in [19].

Location privacy. The privacy objective is to prevent the disclosure of the exact position of non-anonymous users [4]. A major class of techniques attempt to protect the user's position by forwarding a query, typically a k-nearest neighbors (k-NN) query, along with a fake location [1, 10, 37]. The query processor then computes a set of candidate answers that are returned to the client either in a unique block [1] or using a negotiation protocol [10] or according to an incremental strategy [37]. The client then selects among the candidates the exact solution, based on the knowledge of the real position of the user. None of these techniques is able to protect sensitive locations.

The privacy model which is closer to ours is by Cheng et al. [5]. The cloaked location is defined as an uncertainty region. The degree of privacy can be measured in two ways: (i) as the size of uncertainty region; and (ii) as the size of the coverage of sensitive area, where the coverage is the size of the portion of sensitive locations enclosed in the cloaked region, under the assumption that positions have the same probability. We adopt a similar approach, yet our privacy model differs in two main aspects; we focus on the generation of cloaked locations having certain degrees of privacy and not on the evaluation of imprecise queries; we assume that positions are not equally probable.

A different approach emphasizing the notion of sensitive place in user tracking applications has been proposed by Gruteser et al. [16]. Following the *default* privacy policy, well-frequented areas (e.g., streets) are considered non-sensitive while less-frequented areas (e.g., buildings) are sensitive. The idea is to partition the reference space in regions each containing at least k sensitive areas. For example a building is hidden in a region which contains other $k-1$ buildings. Location updates are then delayed until the user leaves the region, thus an observer cannot infer which building the user entered. This approach does not specify how to support personalized privacy.

Privacy policies. Privacy personalization is the key contribution of the *policy-based* paradigm. Location privacy policies [24, 38, 17, 30, 28, 12] regulate the access and usage of location data through access control rules. The IETF GeoPriv standard [12] provides a reference framework for the specification and enforcement of location privacy policies. For example the user can specify in a policy which subjects are authorized to access the user's locations, at what time, and at which granularity. Location granularities are commonly system-defined,

e.g., zip code area and city. Major research issues concern the expressivity and usability of privacy policy languages [31, 32].

A different approach to privacy personalization is by Xiao et al. [34] and applies to sensitive attributes in non-spatial, k-anonymous datasets. The idea is to structure the values of a sensitive attribute in a hierarchy. The user can then specify at which level of the taxonomy the attribute value can be disclosed. This approach, however, has only been applied to categorical data.

To summarize, there are various proposals which aim to protect location either through spatial cloaking, policy-based approaches or using strong but expensive solutions like cryptographic techniques. None of them, however, is able to provide at the same time a personalized and cost effective protection of semantic locations, which instead is the contribution of PROBE.

3 Privacy model and requirements

In this section we introduce the basic concepts of the privacy model and then formulate the location obfuscation problem.

3.1 The privacy model

3.1.1 Preliminaries

We consider a reference space, denoted as Ω, consisting of a bounded area in the two-dimensional space. The spatial objects which populate the space have a spatial type compliant with geo-spatial standards [26]. The user's position is a point. Moreover spatial types are closed under (appropriately defined) geometric union $\cup^s$, intersection $\cap^s$, difference ($\setminus^s$).

We model semantic locations in terms of *spatial features* (simply *features* hereinafter). Features have a type, e.g., hospital, and a spatial extent which is a region. Features are assumed to be spatially disjoint. Thus if two places are one contained in the other, the corresponding features must be defined so that they do not overlap. For example, if a restaurant is within a park, the extent of the park feature should have a hole corresponding to the extent and location of the restaurant feature. We denote with $Cov(ft)$ the spatial union of the extents of all features of type ft, and with FT and F the set of feature types and features respectively.

3.1.2 Assumptions

We assume that the probability density function pdf of the position in space is publicly known. $P(r) = \int_r pdf$ is the probability that a position falls in region r. It holds that: $P(\Omega) = 1$ and $P(\emptyset) = 0$. In general, positions are not uniformly distributed, namely some places are more likely to be frequented than others. If $P(r) = 0$ then region r is *unreachable*; otherwise r is *reachable*. We assume that the distribution of positions is fixed.

We also assume that users are not anonymous and that the users' movements are not predicable. Thus an observer cannot infer, based on a user's profile or history, where the user could be located. In addition, we assume that the user's position is disclosed sporadically, thereby an observer cannot use information about the user's movement, e.g., the speed, to infer more precise positions.

3.1.3 Sensitivity metric

Feature types can be classified as *sensitive* and *non-sensitive* based on personal preferences. This classification easily extends to regions. Consider a region r and denote with $FT_S \subseteq FT$ the subset of sensitive feature types. Formally region r is sensitive if it overlaps $Cov(ft)$, for some sensitive feature type ft, that is,

$$\bigcup_{ft \in FT_S}^{s} Cov(ft) \cap^s r \neq \emptyset.$$

We quantify "how much private" an arbitrary region is by introducing a *sensitivity* metric. The sensitivity of a region depends on both the places which are enclosed in the regions and the function *pdf*. The sensitivity of a region r with respect to a feature type ft, denoted $P_{sens}(ft, r)$, is defined by the probability that a user, known to be in r, is actually within the extent of any sensitive feature of type ft overlapping with r:

$$P_{sens}(ft, r) = \begin{cases} P(Cov(ft)|r) & \text{if} P(r) \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

The sensitivity of an unreachable region is set to 0 for any feature type because a user cannot be located in such a region. Furthermore, if the region is entirely covered by sensitive features of the same type ft, then $P_{sens}(ft, r) = 1$.

Example 2. Consider the region r in Figure 2. Such a region overlaps two features H_1 and H_2 of type $Hospital$, which is sensitive feature type. H_1 is partially contained in r and H_2 is entirely inside the region. Moreover the region includes lake L. Assume the following distribution of user's positions: L is unreachable; the user's positions in $r \setminus^s L$ are equally probable. The *pdf* is thus defined as $\frac{1}{Area(r \setminus^s L)}$, where *Area* computes the surface of a region. The sensitivity of r w.r.t. $Hospital$ can be simply computed:

$$P_{sens}(Hospital, r) = \frac{Area(H_1 \cap^s r) + Area(H_2)}{Area(r \setminus^s L)}$$

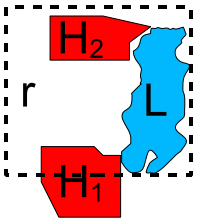


Figure 2: Example of a sensitive region that includes an unreachable region

3.1.4 The privacy profile

The user's privacy requirements are summarized in the *privacy profile*. Formally, the privacy profile is the pair (FT_S, T) where FT_S are the user-defined sensitive feature types, and T the set of *privacy preferences*. A privacy preference is a constraint over the maximum sensitivity that the user tolerates in any location wrt a certain sensitive feature type. A

preference takes the form: (ft, τ) where $\tau \in (0, 1)$ is the *threshold sensitivity value* of $ft \in FT_s$. A region r *satisfies* the preference (ft, τ) if the following inequality holds:

$$P_{sens}(ft, r) \leq \tau$$

We exclude the preference $(ft, 1)$ because it would mean that ft is not sensitive, against the initial assumption. We also rule out the preference $(ft, 0)$ because it can be only satisfied if ft has no instances which is not an interesting case.

Example 3. The privacy profile of a user concerned with the disclosure of positions in religious buildings and in health organizations can be defined as follows:

$$FT_S = \{Hospital, ReligiousBuilding\}$$

$$T = \{(Hospital, 0.4), (ReligiousBuilding, 0.1)\}.$$

In this example, the threshold value is lower for the feature type *ReligiousBuilding* than for the feature type *Hospital* to mean that the privacy demand is stronger in the former case.

3.2 Privacy requirements

3.2.1 Obfuscated location

Consider a region r. We say that r is an *obfuscated location* for the privacy profile (FT_S, T) if r is sensitive and every preference in the set T is satisfied, i.e.:

$$\forall\ (ft, \tau) \in T, P_{sens}(ft, r) \leq \tau$$

In essence, an obfuscated location is a coarse region including a sensitive portion of space which satisfies the user's privacy constraints.

Obfuscated locations may have an arbitrary complex shape and large size and even cover the reference space. Thus, in order not to compromise the quality of service (QoS), the obfuscated locations must have a small size. We define the spatial accuracy of a set $R = \{r_1, ..r_n\}$ of obfuscated locations as the average size of the regions:

$$\frac{1}{N} \sum_{i \in [1,n]} Area(r_i)$$

3.2.2 Privacy-preserving location

Consider a generic user's location p, either exact or coarse. We say that p is *privacy-preserving* if one of the following conditions is true:

(a) p is an exact position which does not fall in any sensitive location

(b) p is an obfuscated location.

The goal is to provide a framework which assures the disclosure of privacy-preserving positions and which provides efficient and scalable methods for the generation of obfuscated locations while limiting the loss of spatial accuracy.

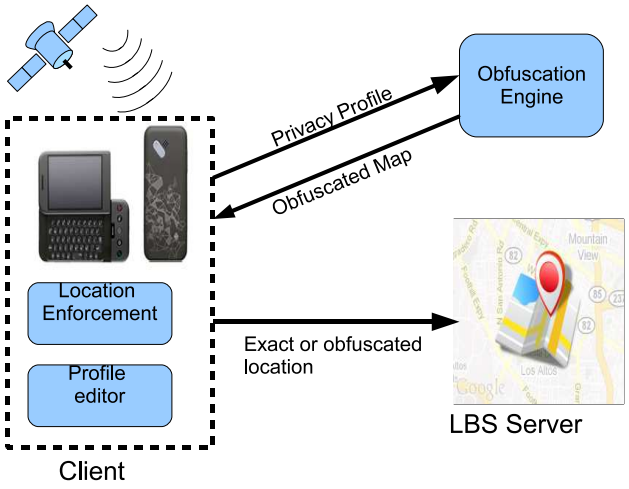


Figure 3: Reference architecture

4 Architecture

Figure 3 shows the PROBE architecture. We consider a conventional client-server architecture in which clients are location-aware mobile devices. The position of a client is supposed accurate and can be provided by the mobile device itself or by some trusted external location service. The client forwards to the server a position, which can be either exact or obfuscated. Such a position is then used, for example, to answer location-based queries or be disclosed to other individuals, e.g., the members of a geo-social network.

The core component of PROBE is the *Obfuscation Engine*. The Obfuscation Engine generates the set of obfuscated regions based on the input privacy profile. Note that the sensitive locations are obfuscated *off-line*, i.e., before any service is requested. This choice has two main advantages: first, the solution is robust against reverse engineering attacks because the generation of the obfuscated region is decoupled from the actual position of the user. Second, the solution is efficient because the obfuscation process must be re-applied only when the user changes his privacy profile (or the background knowledge is updated). Note also that the obfuscated locations are generated once so as to cover the whole sensitive area. The resulting set of regions is called *obfuscated map*.

The obfuscation process, i.e., the process which cloaks the user's position, develops in three phases.

(a) First, the user specifies the privacy profile through the *Profile Editor* possibly running in the client. For example, the user can select the sensitive features types from a predefined list even provided by a third party and then specify the privacy preferences.

(b) In the subsequent phase, the user invokes the *Obfuscation Engine* to generate the obfuscated map based on the input privacy profile. The Obfuscation Engine can run on a third party, such as a Web application or a laptop, or even on the mobile device if such a device has sufficient computational resources [13]. The obfuscated map is then stored on the mobile device.

(c) Finally at run-time, upon a service request, the *Location Enforcement* application run-

ning on the client checks the user's position against the local obfuscated map. If the position falls inside an obfuscated location, the actual position is replaced by the coarser position otherwise the exact location is transmitted to the Server. Note that location enforcement is performed on the client to provide privacy guarantees and efficiency.

5 The generation of obfuscated locations

We now introduce the heuristic strategy for the generation of the obfuscated map, and then the methods which provide different implementations of such a strategy.

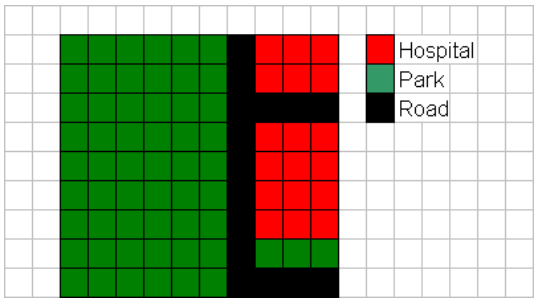


Figure 4: Grid based representation of features

5.1 Model representation

We first translate the privacy model onto a discrete space. The space consists of a grid G of n regular cells (Figure 4). The cells are assumed to be small enough so as to cover a portion of a unique feature, e.g., a cell can be part of a hospital or a road, but not of both. A feature consists of one or more cells. Moreover, cells are sensitive if they are part of sensitive features. In that case, the sensitivity of a cell c is $P_{sens}(ft, c) = 1$ where ft is the type of the feature the cell is part of. In addition, cells are assigned a probability value, i.e., $P(c) = \int_c pdf$. A high probability value means that the cell is expected to be much frequented. The set of probability values $\{P(c_i)\}_{i\in[1,n]}$ is called *Probability Grid*.

5.2 The obfuscation strategy

We propose a greedy strategy. The idea is to progressively expand every sensitive region by aggregating neighbor cells until the privacy constraints are satisfied or the region degenerates in the whole space. Moreover we impose the condition that the resulting obfuscated locations must be spatially disjoint. This requirement is motivated by the fact that overlapping regions may compromise the privacy guarantees of obfuscation. For example, consider the two obfuscated locations A and B in Figure 5.(a), each containing a sensitive area (e.g., a hospital). Assume that the obfuscation algorithm is publicly available. An observer who knows the obfuscated map is thus aware that whenever the user is inside $A \bigcap B$[1], then the returned obfuscated location is A. Now assume that the client discloses

[1]The choice of the region cannot be random, because an adversary can easily make inferences on the position repeatedly generating the obfuscated location.

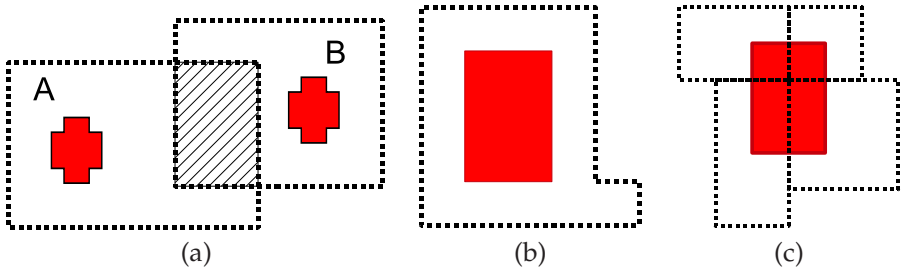


Figure 5: (a) Overlapping obfuscated regions; (b) expansion of the whole sensitive location; (c) expansion per parts

B to the Server. Since the user cannot be located in the intersection region, otherwise the region A would have been returned, the observer can easily bound location B and thus defeat the obfuscation mechanism.

5.2.1 Granularity of aggregation

The greedy strategy can be implemented using different aggregation methods which basically differ in two main aspects: (a) the granularity of the sensitive area to enlarge, i.e., the whole location or part of it; and (b) the granularity of the expansion, i.e., by single cells or group of cells.

Consider first the case in which the sensitive location is taken as a whole and progressively enlarged. It is trivial to observe that the broader the region is, the lower the spatial accuracy of the obfuscated location is (Figure 5.b). Ideally by subdividing the sensitive regions in parts and expanding each part separately, one could obtain obfuscated locations which are more accurate than the regions that would be obtained by obfuscating the entire region (Figure 5.c). On the other hand, the smaller these parts are and the more likely the regions overlap when they are enlarged. Thus, if those regions are merged, the resulting obfuscated region can be excessively broad. Moreover, a flexible and fine-grained obfuscation can lead to unacceptable computational costs [7]. In summary, it is not clear which aggregation method is the most appropriate in which circumstance. To address with this problem, we have experimented three algorithms:

(a) $Sens_{Reg}$ (region-based method). This technique applies to sensitive locations of rectangular shape. It expands the whole sensitive location by aggregating rows or columns.

(b) $Sens_{Div}$ (division-based method) simply applies the previous method to portions of the sensitive location obtained by dividing the region in smaller rectangles.

(c) $Sens_{Hil}$ (Hilbert-based method). It can be used with locations of arbitrary shape. The regions which are enlarged are the single cells. Cells are aggregated one at a time.

Before proceeding to illustrate the characteristics of these methods, we discuss some general properties of the greedy strategy.

5.2.2 Properties of the greedy strategy

We introduce some preliminary definitions. Let $\mathcal{C} = \{c_1, c_2, ...c_n\}$ be a partition (not necessarily the initial one) of grid G. Two cells [2] $c_1, c_2 \in \mathcal{C}$ are adjacent if they have a common border. Given two adjacent cells c_1, c_2, the operation which merges the two cells generates a new partition $\mathcal{C}'$ in which cells c_1 and c_2 are replaced by cell $c = c_1 \bigcup^S c_2$. We say that partition $\mathcal{C}'$ *is derived* from partition $\mathcal{C}$, written as $\mathcal{C}' \succcurlyeq \mathcal{C}$. Consider the set $\mathcal{P}_{\mathcal{C}_{in}}$ of partitions derived directly or indirectly from the initial partition $\mathcal{C}_{in}$ through subsequent merge operations. The poset $H = (\mathcal{P}_{\mathcal{C}_{in}}, \succcurlyeq)$ is a bounded lattice in which the least element is the initial partition while the greatest element is the partition consisting of a unique element, that is, the whole space (called *maximal partition*).

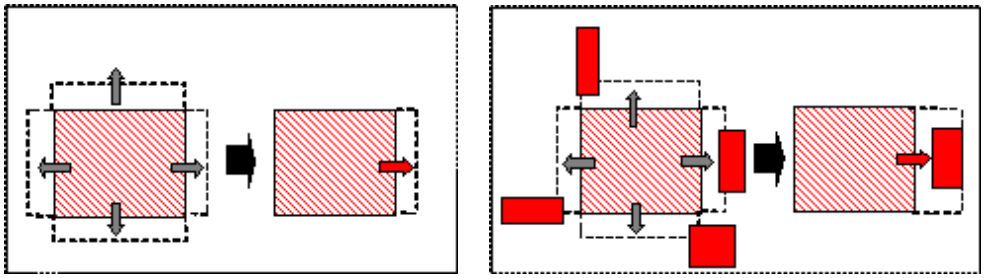
It can be shown [8] that when two cells are merged, the sensitivity of the resulting cell is lower than the maximum between the sensitivity values of the two starting cells c_1 and c_2, that is:

$$P_{sens}(ft, c) \leq \max\{P_{sens}(ft, c_1), P_{sens}(ft, c_2)\}$$

Moreover, the maximum sensitivity value in a partition $\mathcal{C}_A$ is equal or greater than the maximum sensitivity of a derived partition, that is:

$$\mathcal{C}_A \succcurlyeq \mathcal{C}_B \Longrightarrow \max_{r \in \mathcal{C}_A} P_{sens}(ft, r) \leq \max_{r \in \mathcal{C}_B} P_{sens}(ft, r)$$

In essence, the coarser is the partition, the lower is the maximum sensitivity value of the partition. From this consideration, it is trivial to show that a necessary condition for an obfuscated map to exist is that the maximal partition satisfies the privacy preferences. In what follows we assume that this condition is true and thus at least one obfuscated map exists. In the worst case the obfuscation process degenerates returning a unique obfuscated location for the whole space.



(a) (b)

Figure 6: Expansion of the region with no overlap (a); with overlap (b)

5.3 The $Sens_{Reg}$ algorithm

The $Sens_{Reg}$ algorithm implements the most straightforward approach which expands each sensitive location as a whole. The sensitive regions are represented by rectangles. The idea is to extend each sensitive feature in the direction, i.e., West, North, South, East, which locally results the most convenient.

[2] In this case the cell is an element of the partition.

Figure 6 illustrates the main steps of the algorithm. At each step the sensitive area (i.e. the rectangle) is progressively enlarged of one row or column. If the region can grow without overlapping other rectangles (sensitive features or obfuscated locations), then the region is enlarged in the direction which minimizes the sensitivity of the region (Figure 6.(a)) If the expanding region necessarily overlaps other regions, the region is enlarged so as to include the intersecting rectangles and achieve the minimal expansion among those possible. The enlargement is achieved by computing the MBB enclosing the sensitive area and the overlapping regions (Figure 6.(b)). If the resulting region intersects other regions, then the MBB is extended to iteratively include also those regions. Figure 7 summarizes the expansion process. The algorithm scans the list of sensitive regions and applies the obfuscation process to each of those regions. The set of obfuscated locations map is initially empty. Each sensitive region is then extended in one of the directions, either overlapping or not other rectangles (loop at line 4). If the resulting region obf is not degenerated, i.e., it does not coincide with the whole space, then obf is added to map.

$Sens_{Reg}$
Input: set of sensitive rectangles R, privacy preferences pp= $\{(ft_i, \tau_i)\}_{i\in[1,n]}$
Output: Obfuscated locations map
1. map=EmptySet
2. dir=EmptySet
2. **for all** $r \in R$
3. $obf \leftarrow r$
4. **while** $(P_{Sens}(obf, ft_i) > \tau_i)$ for some $i \in [1, n]$
5. $dir \leftarrow$ the directions in {N,S,W,E} in which obf can expand without overlapping other regions
6. **if** $dir \neq null$ **then**
7. $obf \leftarrow$ the enlarged region with the best sensitivity
8. **else** $obf \leftarrow$ the MBR with the minimum extent enclosing r and the overlapping regions
9. **if** (obf is not degenerated) **then**
10. Remove the sensitive rectangles in obf from R
11. Add obf to map
12. **return** Map

Figure 7: $Sens_{Reg}$ Pseudocode

5.4 The $Sens_{Div}$ algorithm

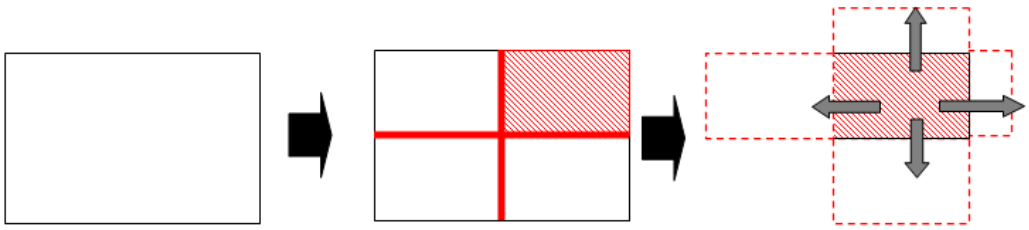
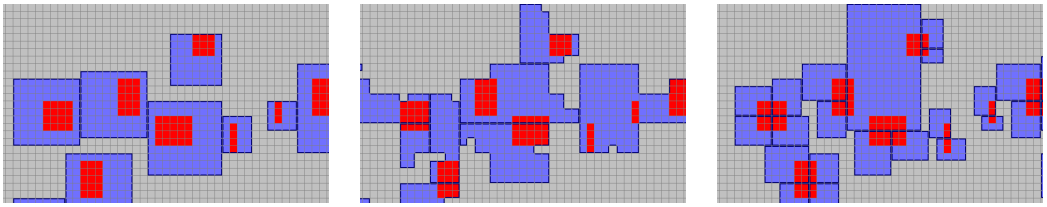


Figure 8: Example

$Sens_{Div}$
Input: set of sensitive rectangles R, privacy preferences pp= $\{(ft_i, \tau_i)\}_{i\in[1,n]}$
Output: Obfuscated locations map
1. map=EmptySet
2. R'=EmptySet
3. **for all** $r \in R$
4. subdivide r in 4 quadrants and add those quadrants to R'
5. $map \leftarrow Sens_{Reg}(R', pp)$
6. **return** map

Figure 9: $Sens_{Div}$ Pseudocode

$Sens_{Div}$ obfuscates a sensitive region per parts. The idea is to subdivide in four quadrants each rectangular sensitive feature and then obfuscates each quadrant separately using the $Sens_{Reg}$ method. The advantage is that the area to obfuscate is smaller. Consider the sensitive rectangle in Figure 8. The rectangle is divided in four quadrants. Each quadrant is then obfuscated as if it were a distinct object. Thus it can be enlarged in any direction. Since the techniques is based on $Sens_{Reg}$, the quadrant will be first enlarged in the direction in which the region does not overlap other rectangles. If the expanded region overlaps other rectangles then the MBB is computed. The algorithm is reported in Figure 9.



(a) (b) (c)

Figure 10: Sensitive regions (red) and obfuscated regions (blue boundary) generated by: $Sens_{Reg}$(a), $Sens_{Hil}$(b), $Sens_{Div}$(c)

5.5 The $Sens_{Hil}$ algorithm

This algorithm obfuscates sensitive cells by aggregating one cell at a time. The idea is to map the grid onto a Hilbert space-filling curve. The Hilbert space-filling curve is a one-dimensional curve which visits once every cell of the grid.

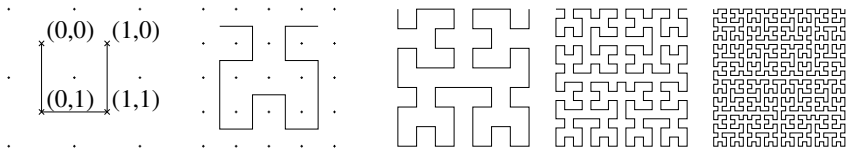


Figure 11: Hilbert curves for different grid sizes

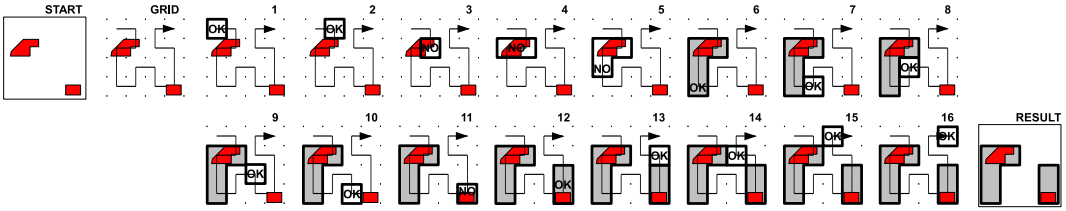


Figure 12: $Sens_{Hil}$ An example. Subfigures numbered from 1 to 16 illustrate the intermediate steps between START and RESULT.

Figure 11 illustrates the curves representing grids with 2×2, 4×4, 8×8, 16×16, and 32×32 cells, respectively. In this linear space, cells are identified by an integer value corresponding to the order in which cells are visited.For example in the first grid in Figure 11, the interval $[2,4]$ denotes the set of cells $\{(0,1),(1,1),(1,0)\}$, corresponding to the 2^{nd}, 3^{rd}, and 4^{th} cells touched by the curve.

The $Sens_{Hil}$ algorithm obfuscates a sensitive cell by progressively aggregating adjacent cells in the linear space. Since this kind of filling curve has the property that the cells that are adjacent in the linear space are also adjacent in the 2D space [29], the result of the aggregation process is a region. The method works as follows: at each step of the obfuscation process, the sensitivity of the growing region is evaluated. If the region is over-sensitive (i.e., the sensitivity exceeds the sensitivity threshold specified in privacy preferences) then the next cell in the ordering is included in the region, independently of whether the cell is sensitive or not. The region thus grows along a predefined direction determined by the total ordering (from the start to the end of the curve).

Figure 12 illustrates the step-wise execution of the algorithm applied to a grid containing two sensitive features (the red polygons). The initial grid is labelled as 'START' in the figure (on the left) while the obfuscated map is labelled 'RESULT' (on the right). The two sensitive features are of the same type. Further, we set the privacy threshold to 25%. The reference space is described by a 4×4 grid, consisting of 16 cells. Subfigures 12.1–16 show the steps of the algorithm. Cells are transversed following the Hilbert ordering. At each step, the region being examined is labelled either 'OK' or 'NO', depending on whether the privacy preference is satisfied or not. Steps 1 and 2 simply skip the first two cells in the ordering because are not sensitive. The 3^{rd} cell, instead, is covered by sensitive features. Thus, in the subsequent steps (subfig. 3-6) the cell is progressively aggregated with neighbor cells until an obfuscated region is found (subfig. 6). The algorithm proceeds until the last cell has been processed.

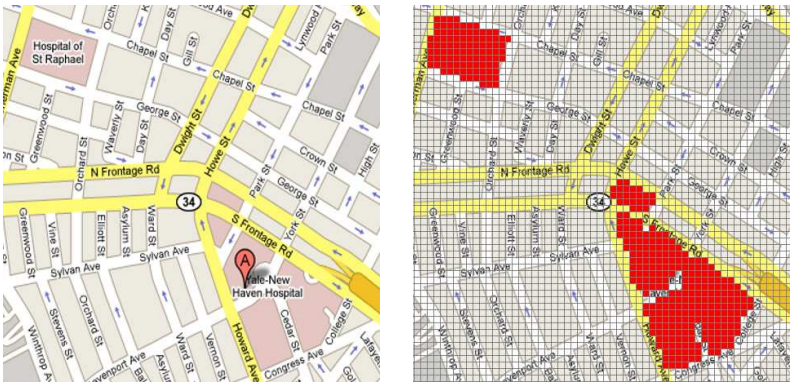
The obfuscation process is summarized in Figure 13. The algorithm starts by scanning the sequence of cells. As an over-sensitive cell is found, the algorithm tries to generate an obfuscated region (represented by an interval) starting from the current cell (line 6). If such interval is found, the interval is added to the set of obfuscated locations and the scan proceeds until every cell has been examined. The cells which are not over-sensitive and which are not included in any obfuscated region are equally added to the set of obfuscated locations. Upon completion of the scan, if the last sensitive cell cannot be obfuscated, then the region is expanded backward. In the worst case the obfuscation degenerates and the entire space is returned.

$Sens_{Hil}$
Input: Hilbert filling curve C, privacy preferences pp= $\{(ft_i, \tau_i)\}_{i\in[1,n]}$
Output: set of obfuscated locations map
1. $map \leftarrow EmptySet$
2. $idx1 \leftarrow FirstCell$
3. $idx2 \leftarrow FirstCell$
4. while $idx2 \leq EndCurve$ do
5. obf $\leftarrow$ the region defined by the interval $[idx1, idx2]$
6. while $(P_{Sens}(obf, ft_i) \geq \tau_i)$ for some $i \in [1, n]$ and $idx2 \leq EndCurve$ do
7. idx2 $\leftarrow$ idx2 +1 *the index of the adjacent cell in the linear space*
8. $obf \leftarrow [idx1, idx2]$ *enlarge the region*
9. if $idx2 \leq EndCurve$ then
10. add obf to map
11. $idx2 \leftarrow idx2 + 1$
12. $idx1 \leftarrow idx2$
13. Obfuscate the last region $[idx1, idx2]$ if it is not obfuscated
14. **return** map

Figure 13: $Sens_{Hil}$ Pseudocode

5.6 Examples

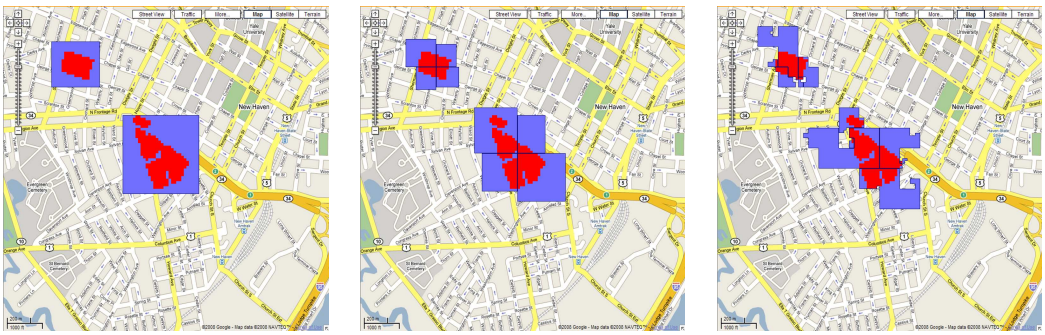
Figures 10 highlights the different shape of the obfuscated locations generated by the three methods. We use synthetic data generated by the spatially-aware generalization (SAG) tool (see later on in the paper): the red rectangles represent the sensitive locations. The obfuscated locations are the blue-filled polygons enclosing the red rectangles or part of them. Depending on the method, a sensitive region can be covered by multiple regions. It can be seen that the regions generated by $Sens_{Hil}$ have an irregular shape. That is due to the characteristics of the Hilbert filling curve space.



(a) (b)

Figure 14: The two hospitals in New Haven, Connecticut

Another example reports the cloaked regions resulting from the obfuscation of two existing hospitals located in New Haven (US). The two hospitals are modelled as two features



(a) (b) (c)

Figure 15: Obfuscation of hospitals by: $Sens_{Reg}$ (a), $Sens_{Div}$(b), $Sens_{Hil}$(c)

of type Hospital. Figure 14.a. zooms on the two sensitive locations, located respectively on the North and on the South. Figure 14.b shows the grid-based representation of the spatial extent. The grid has size 128×128 cells and the resolution of cells is about 20 metres. The privacy profile (FT_S, T) used for the obfuscation is: $FT_S = \{Hospital\}$ and $T = (\{(Hospital, 0.3)\}$. The resulting obfuscated locations are displayed in Figure 15. The $Sens_{Reg}$ and $Sens_{Div}$ are applied to the Minimum Bounding Boxes of the hospitals.

6 Experiments

We have developed a Java prototype of the three obfuscation algorithms. In this section we evaluate the spatial accuracy and the efficiency of these three methods. To run the experiments over synthetic data, we have implemented a platform called SAG (Semantic-aware Generalization). SAG provides a set of functionalities. One of these is to populate the reachable portion of the grid space with rectangles randomly generated, each representing a feature of user-defined type.

The length of the rectangle sides can be set to a fixed value or be randomly generated using a binomial distribution. In most experiments the size of the rectangle side is set to 4 cells. Grids can have different size. A common size in the experiments is 512 x 512 cells: at a resolution of 10 metres, the grid covers the area of a medium-size city. Moreover for each experiment we consider the two cases in which positions are uniformly and non-uniformly distributed. Privacy profiles are simply specified through a graphical interface by flagging the sensitive features types in the list of feature types and then specifying for each sensitive feature the threshold value. The experimental testbed consists of a laptop PC equipped with an Intel Core Duo P8400 2.2GHz CPU, 4GB of RAM and Windows Vista.

Generation of non-uniform distributions. SAG supports the generation of synthetic non-uniform distributions based on an empirical approach. The starting assumption is that the locations that are in proximity of a number of places acting as *attraction points* are the most frequented. Attraction points are, for example, shopping and entertainment centers. The idea is to generate a number of attraction points on a random basis. Each of these points is then attributed an *area of influence*,e.g., a circle of specified radius. A probability value is then assigned to each cell of the region. If the regions of influence overlap then the values of probability in the intersecting areas are summed up. Probabilities are then

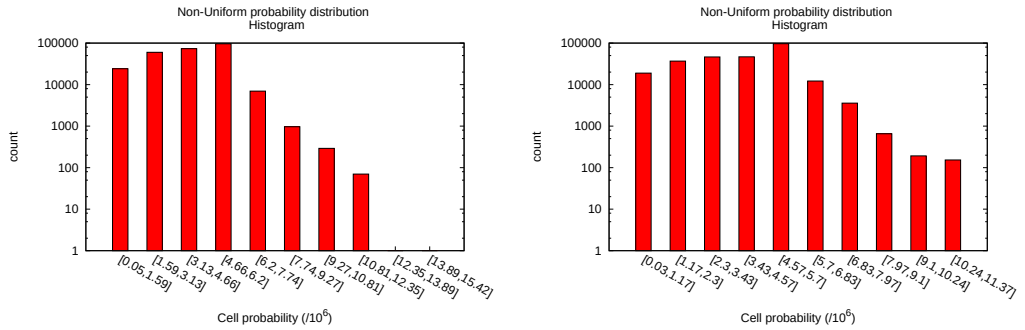


Figure 16: Histograms

normalized to guarantee that the sum of cell values is 1. In the grid 512×512 we consider 1000 attraction points. The probability value assigned to each area is selected randomly inside a pre-defined range of values. We have generated 10 distributions. Figure 16 shows 2 (out of 10) histograms reporting the number of cells for each interval of probability. We run the experiments considering this specific pattern of position distribution. Of course, a myriad of other patterns can be chosen. Yet we believe that this pattern (corresponding to an area in which most of the cells are sufficiently frequented) is realistic enough of be meaningful.

6.1 Spatial accuracy with uniform and non-uniform distribution of positions

6.1.1 Experimental setting

The first series of experiments evaluates the spatial accuracy of the obfuscated locations generated by the three methods (Figure 17). We run the experiments on two scenarios in which the positions are uniformly and non-uniformly distributed respectively. First we measure the accuracy of the regions returned by the obfuscation process. We recall that the accuracy is defined by the average size (i.e., number of cells) of obfuscated locations. We consider a unique feature type, that is sensitive, and features of fixed size (4x4 cells, e.g., a building with side of 40 metres). We consider a number of sensitive objects varying between 100 and 1000 objects. Therefore the fraction of space covered by sensitive objects (*coverage*) ranges between 0.6% and 6%. Moreover we run the experiments using three different values of the sensitivity threshold (τ), i.e., $\tau \in \{0.05, 0.1, 0.2\}$. Consider that a threshold of 5% is very restrictive. For example, in a grid of equiprobable cells, a sensitive rectangle of 160 m^2 is obfuscated by a region of 3200 m^2. Moreover, if there is a significant number of regions, the locations being obfuscated frequently overlap with other regions to form large agglomerates. With a sensitivity threshold set to 5% and coverage 6%, the obfuscated process is expected to degenerate or the accuracy be seriously compromised.

Each experiment is repeated 10 times and each time, a different seed is used for the generation of sensitive regions. Each row reports the results for a different value of τ. The results on the right are obtained with equi-probable cells; the results on the left with a non-uniform distribution.

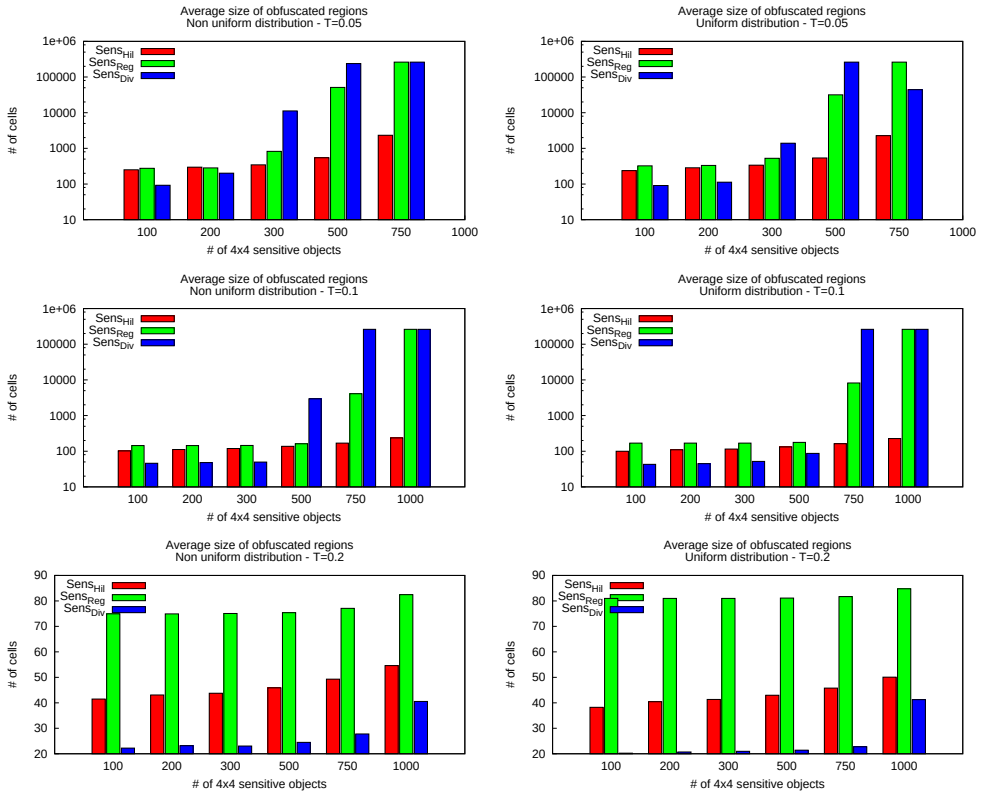


Figure 17: Avg. size of obfuscated locations for different number of sensitive objects

6.1.2 Accuracy

The experimental results show that the most influential parameter is the sensitivity threshold τ, while the fact of using positions of equal probability or not does not seem to significantly affect the results. If the number of sensitive objects is relatively low (< 300 for $\tau = 0.05$) or with weak privacy constraints (e.g., $\tau = 0.2$), the best location accuracy is achieved by the $Sens_{Div}$. However, when the privacy constraints are stronger (e.g., $\tau \leq 0.1$), the accuracy achieved by both $Sens_{Reg}$ and $Sens_{Div}$ rapidly decreases with the increase in the number of objects. The reason is that the regions frequently overlap with other regions and thus are merged in large agglomerates of cells. The $Sens_{Reg}$ algorithm has the worst accuracy in all cases. $Sens_{Hil}$ assures a reasonable accuracy also in the cases of high density of sensitive objects.

6.1.3 Degeneration

The degeneration of the obfuscation process, i.e., the regions become of unacceptable size, is evident in the graphs in Figure 18 which report the number of obfuscated regions for varying numbers of sensitive objects. The bell-shape of the curves (more evident for low values of τ) is due to the fact that the number of obfuscated regions first grows up to a maximum value with the increase in the number of objects. Then, when the objects are too

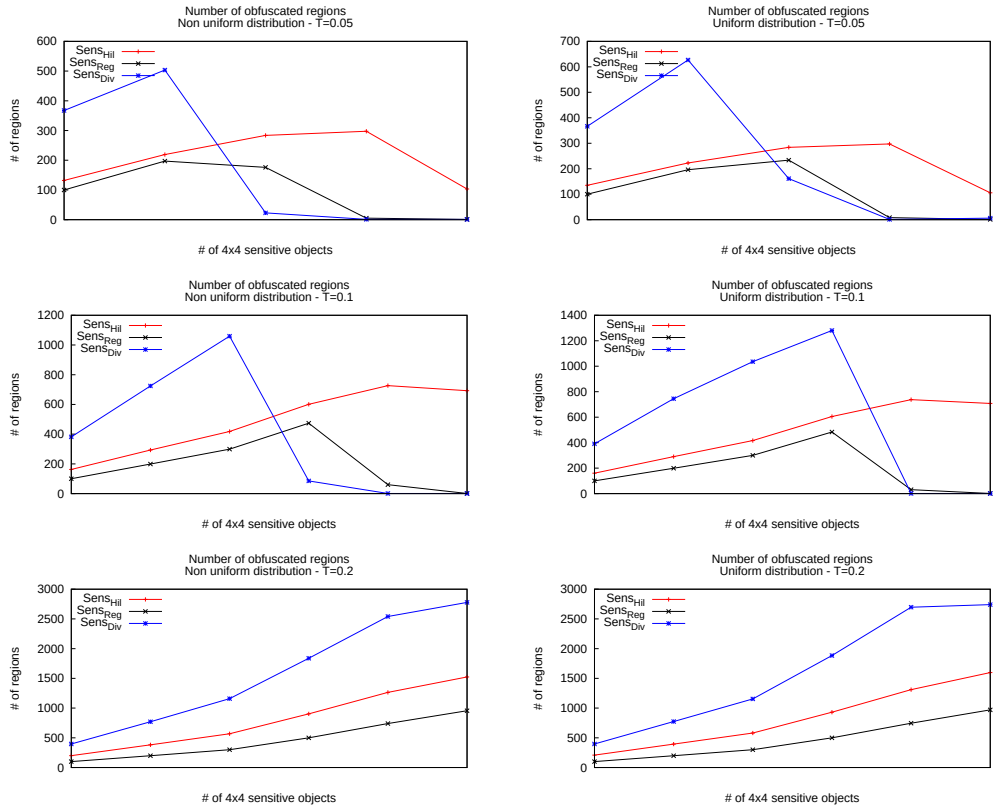


Figure 18: Number of regions for varying number of objects

many, the regions are merged in agglomerates that get larger and larger and the number of obfuscated locations thus decreases. $Sens_{div}$ reaches the maximum earlier than $Sens_{Hil}$, afterwards the curve declines very rapidly.

6.1.4 Effect of the object size

We have also carried out another series of experiments to evaluate the effect of the size of sensitive locations over the accuracy. For the sake of space, in Figure 19 we only report the outcomes for the non-uniform distribution scenario and $\tau \leq 0.1$. We assume a fixed number of sensitive rectangle (100 objects) whose side ranges between 1 and 10 cells. The outcomes are in line with the previous results, in that $Sens_{Div}$ provides the best accuracy when objects are relatively small; with large size objects or strong privacy constraints, $Sens_{Hil}$ is better.

6.2 Computational time and storage requirements

In these experiments we evaluate the three methods with respect to the computational costs and the storage size of the obfuscated map.

We compute the computational cost (i.e., obfuscation time) with grids of increasing side from 128 to 1024 cells. Those grids cover approximatively an area ranging from $1km^2$

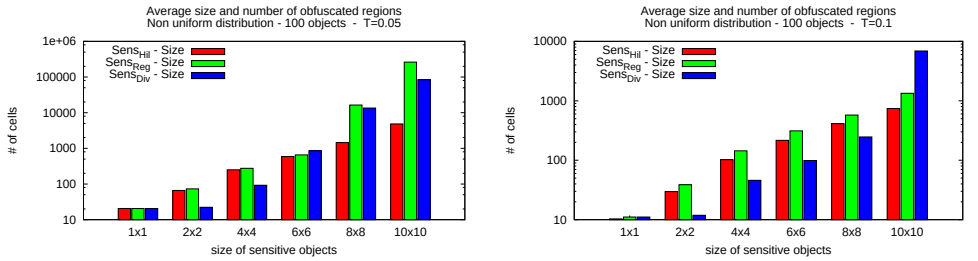


Figure 19: Avg. number of cells for different sizes of sensitive objects

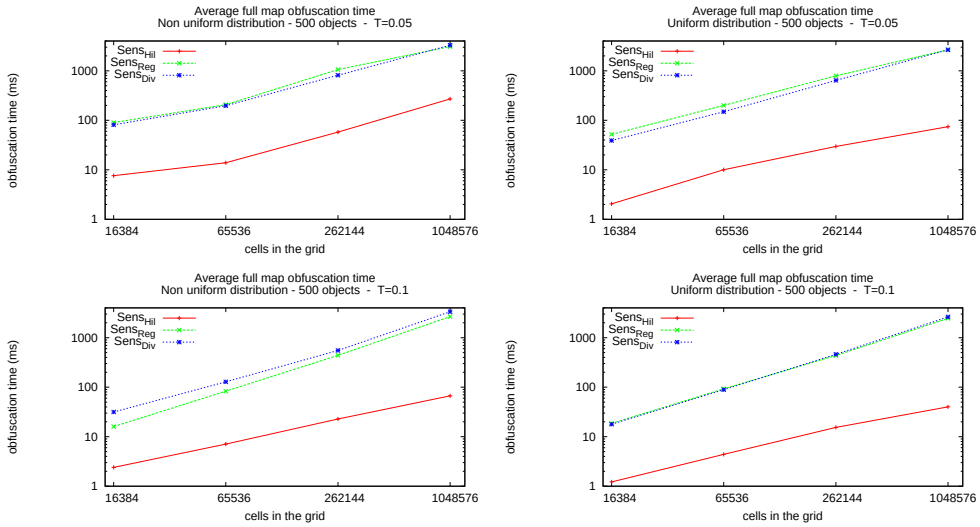


Figure 20: Avg. obfuscation time for different grid sizes

to $100km^2$. We consider a fixed coverage (3% of space occupied by sensitive cells) and values of τv lower or equal than 0.1. The experimental results show that in all situations the $Sens_{Hil}$ algorithm outperforms the other two methods (Figure 20).

The storage size of the obfuscated maps is reported in Table 1 for the non-uniform distribution scenario. The columns report: the average number of obfuscated locations (*Avg # of Regions*), the average size of those regions (*Avg. cells per region*), and the size in bytes of the obfuscated map (*Map size*) for grids of varying size. We consider that the storage size of each obfuscated location created by the $Sens_{Hil}$ algorithm takes 8 bytes [3]; in the other case, each region has a size of 16 bytes. It can be seen that in all cases the storage requirements are limited. Even in the worst case, the size of the map does not exceed 75KB. Therefore the obfuscated locations can be easily stored on low-end mobile devices.

6.3 Experimenting with real data

The last series of experiments uses a more realistic, though approximated, probability distribution. We estimate the cell probabilities by calculating the relative frequency of posi-

[3]The region is represented by two integer values and an integer takes 4 bytes.

Alg.	Grid Side	Cells (/1000)	Avg # of Regions	Map Size (Bytes/1000)
HIL	128	16	31	.2
	256	66	149	1.2
	512	262	601	4.8
	1024	1049	2419	19
	2048	4194	9623	76
REG	128	16	25	0.4
	256	66	119	1.9
	512	262	474	7.6
	1024	1049	1869	30
	2048	4194	1	0.01
DIV	128	16	57	0.9
	256	66	53	0.9
	512	262	85	1.4
	1024	1049	369	5.9
	2048	4194	1	0.01

Table 1: Measures variable grid size. Non-uniform distribution. Coverage = 3%. $T = 0.1$

tions based on an existing data set.

The *Milan data set* is provided by the MODAP project [4] and contains 200K trajectories of 17K vehicles collected during one week in Milan. Each trajectory is defined by a sequence of snapshots taking the form (p, t), where $p = (x, y)$ is the location of the vehicle at time t. We define a grid of size 1024×1024 cells with resolution of about 15 metres. The relative frequency of each cell is computed as the ratio of the number of observations in the cell over the total number of observations. An observation takes place when a vehicle is inside the cell. Each vehicle inside a given cell is counted only once, although the same vehicle visits the cell more than one time. For the computation of the frequency values we exploit the functionalities of the trajectory data warehouse developed by Orlando et al. [27]. In the grid of Figure 21.a, the cells are visualized using different grey levels based on the frequency values (the color is darker for higher frequency values). The corresponding histogram is reported in Figure 21.b.

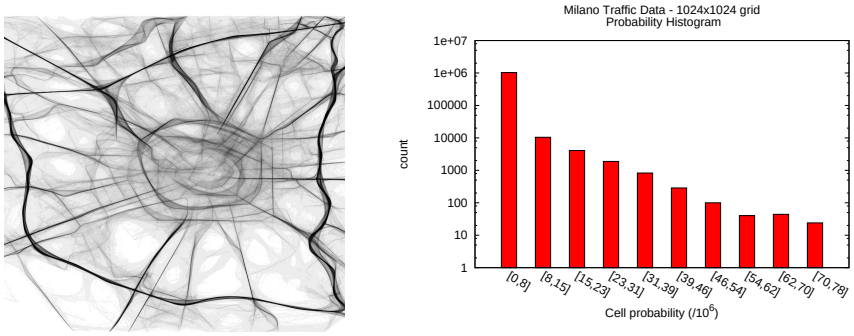
The graphs in Figure 22 report the accuracy for varying numbers of sensitive objects (of size 4×4)). For sake of uniformity with the previous experiments, the coverage varies between 0.2% and 6% (corresponding to a number of objects varying between 100 and 4000). These experimental results confirm that the $Sens_{Div}$ algorithm achieves the best accuracy when the objects are not excessively dense or the privacy requirements are less stringent; $Sens_{Hil}$ is the most scalable.

7 Discussion

Finally we discuss privacy threats and possible countermeasures.

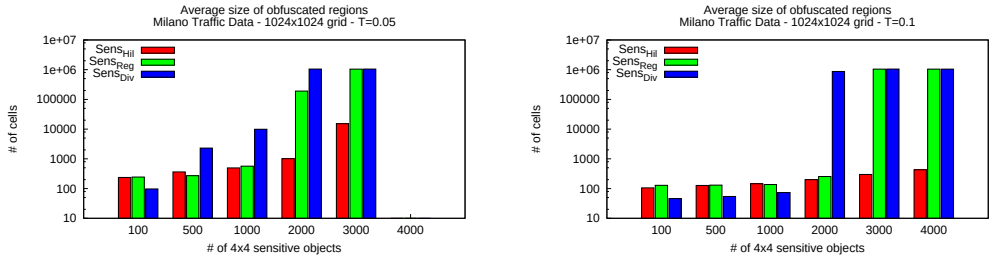
The privacy of the privacy profile. If the user discloses a coarse location in place of the exact position, one can infer that the user has something to hide in that region. Moreover,

[4] http://www.modap.org



(a) (b)

Figure 21: (a) The grid of the frequency values for the Milan data set; (b) histogram of frequency values



(a) (b)

Figure 22: Accuracy of the algorithms applied to the Milan data set

from the observation of which positions are cloaked, on observer can reconstruct the obfuscated map and thus the user's privacy profile. For example, if an obfuscated location includes a hospital and other apparently non-sensitive features, e.g., park and shops, then the observer can infer that the hospital is a sensitive category for the user and thus it is likely that the users wants to obfuscate those places because he has health problems. This subtle privacy threat is motivated by the implicit assumption that the user's profile is itself private.

To protect from this privacy breach, a simple countermeasure is to force the use of system-defined privacy profiles. For example, some locations are naturally sensitive to the majority of individuals, like hospitals and religious buildings. Obfuscating those places independently from the user preferences would have the advantage of mitigating the curiosity of potential observers. The drawback is the lack of personalization flexibility.

A second countermeasure is to enable the specification of sensitive feature types at different granularities. For example, the user could specify the sensitive feature type *Public services* in place of *Hospitals* where *Public services* groups different types of sensitive and non-sensitive objects. This approach however is not suitable when the user's locations present some degree of homogeneity, e.g., contains the same types of places, because an observer can mine the historical data and find patterns which reveal the user's profile. A third and more complex level of protection, which assures a stronger against the mining of historical data, is to define an adaptive mechanism of protection which dynamically

changes the privacy preferences based on the user's behavior

Context-dependent privacy. The distribution of positions is time-varying. Similarly the places which are sensitive for a user may change in time. For example, pubs can be be sensitive only during the night; a shopping center could be crowded during the day and almost empty at night. For example, suppose that the shopping center is close to a hospital and that both places are included in an obfuscated location. If the obfuscated location is returned in nightly hours, when the shopping center is closed and there a few people outside, one can guess that the user is more likely located in the hospital. A straightforward defense against this privacy breach is to generate different obfuscated maps, each based on a different distribution of positions.

In another situation, an observer can infer a more precise location by using publicly available knowledge about the user, i.e., the job of the individual. Consider the case in which the user is a taxi driver and suppose that his position is cloaked by a region covering many different places, including a park. An observer aware of the user's job can promptly infer that the area inside a park is unreachable for the taxi driver. Like in the previous case, a countermeasure is to obfuscate locations based on the same distribution the attacker is aware of. This solution however does not suffice when multiple attributes of the user are known, each motivating a potentially different distribution of positions.

8 Conclusions

PROBE is a novel framework for the protection of sensitive locations which combines privacy personalization and location privacy. It provides a set of obfuscation methods which have very small storage requirements and that are suited for use on small devices, such as cellular phones. The approach can be integrated with k-anonymity techniques and with policy-based approaches to provide stronger privacy protection especially in novel geo-social applications.

References

[1] M. Atallah and K. Frikken. Privacy-preserving location-dependent query processing. In *ACS/IEEE Intl. Conf. on Pervasive Services (ICPS)*, 2004.

[2] A. Beresford and F. Stajano. Location privacy in pervasive computing. *IEEE Pervasive Computing Magazine*, pages 46–55, 2003.

[3] S. Bettini C. Jajodia S., Samarati P. Wang, editor. *Privacy in Location-based Applications*. Springer, 2009.

[4] M. L. Y. C. S. Jensen, Hua Lu. *In Privacy in location-based applications*, chapter Location Privacy Techniques in Client-Server Architectures. Springer, 2009.

[5] R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar. Preserving user location privacy in mobile data management infrastructures. In *Proc. of the 6th Workshop on Privacy Enhancing Technologies*, 2006.

[6] C. Chow, M. F. Mokbel, and W. G. Aref. Casper*: Query Processing for Location Services without Compromising Privacy. *ACM Transactions on Database Systems*, (34)4:4, 2009.

[7] M. Damiani, E. Bertino, and C. Silvestri. Protecting Location Privacy through Semantics-aware Obfuscation Techniques. In *Proc. of IFIPTM 2008*, pages 231–245. Springer Boston, June 18-20 2008.

[8] M. L. Damiani, E. Bertino, and C. Silvestri. PROBE: an obfuscation system for the protection of sensitive location information in lbs. CERIAS Technical Report, Purdue University, 2008.

[9] M. L. Damiani, E. Bertino, and C. Silvestri. Protecting location privacy against spatial inferences: the probe approach. In *SPRINGL '09: Proceedings of the 2nd SIGSPATIAL ACM GIS 2009 International Workshop on Security and Privacy in GIS and LBS*, New York, NY, USA, 2009. ACM.

[10] M. Duckham and L. Kulik. A formal model of obfuscation and negotiation for location privacy. In *Pervasive Computing*. Springer, 2005.

[11] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In *Proc. of the 25th IEEE ICDCS*, 2005.

[12] GEOPRIV. http://www.ietf.org/html.charters/geopriv-charter.html.

[13] G. Ghinita, M. Damiani, E. Bertino, and C. Silvestri. Interactive Location Cloaking with the PROBE Obfuscator. In *Proc. of the Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, 2009.

[14] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan. Private queries in location based services: anonymizers are not necessary. In *SIGMOD '08*, pages 121–132, New York, NY, USA, 2008. ACM.

[15] M. Gruteser and D. Grunwald. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In *Proc. of the 1st international conference on Mobile systems, Applications and Services*. ACM Press, 2003.

[16] M. Gruteser and X. Liu. Protecting privacy in continuous location tracking applications. *IEEE Security and Privacy*, 2(2):28–31, 2004.

[17] U. Hengartner and P. Steenkiste. Access control to people location information. *ACM Trans. Inf. Syst. Secur.*, 8(4):424–456, 2005.

[18] G. Iachello and J. Hong. End-User Privacy in Human-Computer Interaction. *Foundations and Trends in Human-Computer Interaction* , (1)1:1–137, 2007.

[19] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing location-based identity inference in anonymous spatial queries. *IEEE TKDE*, 2007.

[20] B. Krishnamachari, G. Ghinita, and P. Kalnis. Privacy-Preserving Publication of User Locations in the Proximity of Sensitive Sites. In *Proc. SSDBM*, 2008.

[21] J. Krumm. A survey of computational location privacy. *Personal and Ubiquitous Computing*, (13)6:391–399, 2009.

[22] N. Li, T. Li, and S. Venkatasubramanian. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In *Proc. ICDE*, 2007.

[23] K. Mouratidis and M. L. Yiu. Anonymous query processing in road networks. *IEEE Trans. on Knowl. and Data Eng.*, 22(1):2–15, 2010.

[24] G. Myles, A. Friday, and N. Davies. Preserving privacy in environments with location-based applications. *IEEE Pervasive Computing*, 2(1):56–64, 2003.

[25] M. E. Nergiz, M. Atzori, Y. Saygin, and B. Güç. Towards trajectory anonymization: a generalization-based approach. *Transactions on Data Privacy*, 2(1):47–75, 2009.

[26] Open GIS Consortium. Open GIS simple features specification for SQL, 1999. Revision 1.1.

[27] S. Orlando, R. Orsini, A. Raffaetà, A. Roncato, and C. Silvestri. Trajectory Data Warehouses: Design and Implementation Issues. *Journal of Computing Science and Engineering*, 1(2):240–261, 2007.

[28] N. Poolsappasit and I. Ray. Towards Achieving Personalized Privacy for Location-Based Services. *Transactions on Data Privacy*, 2:1:77 – 99, 2009.

[29] H. Samet. *Foundations of Multidimensional and Metric data Structures*. Morgan Kaufmann, 2006.

[30] E. Snekkenes. Concepts for personal location privacy policies. In *EC '01: Proceedings of the 3rd*

ACM conference on Electronic Commerce, pages 48–57, New York, NY, USA, 2001. ACM Press.

[31] E. Toch, R.Ravichandran, L. Cranor, P. H. Drielsma, J. Hong, P. Kelley, N. Sadeh, and J. Tsai. Analyzing Use of Privacy Policy Attributes in a Location Sharing Application. In *Proc. ACM SOUP*, 2009.

[32] J. Y. Tsai, P. Kelley, P. Drielsma, L. F. Cranor, J. Hong, and N. Sadeh. Who's viewed you?: the impact of feedback in a mobile location-sharing application. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 2003–2012, New York, NY, USA, 2009. ACM.

[33] T. Wang and L. Liu. Privacy-aware mobile services over road networks. In *Proc. of the 35th International Conference on Very Large Data Bases (VLDB'09).*, pages 1042–1053, 2009.

[34] X. Xiao and Y. Tao. Personalized privacy preservation. In *Proc. of the 2006 ACM SIGMOD*, pages 229–240, New York, NY, USA, 2006. ACM.

[35] T. Xu and Y. Cai. Feeling-based location privacy protection for location-based services. In *CCS '09: Proceedings of the 16th ACM conference on Computer and communications security*, pages 348–357, New York, NY, USA, 2009. ACM.

[36] P. H. Xue M., Kalnis P. Location Diversity: Enhanced Privacy Protection in Location Based Services. In *Proc. of the International Symposium on Location and Context Awareness (LoCA)*, 2009.

[37] M. Yiu, C. Jensen, X.Huang, and H. Lu. SpaceTwist: Managing the Trade-Offs Among Location Privacy, Query Performance, and Query Accuracy in Mobile Services. In *Proc. IEEE 24th International Conference on Data Engineering*, 2008.

[38] M. Youssef, V. Atluri, and N. R. Adam. Preserving mobile customer privacy: an access control system for moving objects and customer profiles. In *Proc. MDM*, 2005.