Protecting Privacy Against Location-based Personal Identification*

Claudio Bettini[1], X. Sean Wang[2], and Sushil Jajodia[3]

[1] DICo, University of Milan, Italy. bettini@dico.unimi.it
[2] Dept of CS, University of Vermont, Vermont. xywang@cs.uvm.edu
[3] CSIS, George Mason University, Virginia. jajodia@gmu.edu

Abstract. This paper presents a preliminary investigation on the privacy issues involved in the use of location-based services. It is argued that even if the user identity is not explicitly released to the service provider, the geo-localized history of user-requests can act as a quasi-identifier and may be used to access sensitive information about specific individuals. The paper formally defines a framework to evaluate the risk in revealing a user identity via location information and presents preliminary ideas about algorithms to prevent this to happen.

1 Introduction

There are currently over 1.5 billion mobile phone users worldwide and the numbers are still growing very fast. Location technologies can be currently used by wireless carrier operators to provide a good estimate of the user location. These techniques are being refined in order to meet the requirements imposed by federal institutions both in US and Europe for location-enhanced emergency services. Significantly more precise positioning is obtained by GPS technology which is already integrated in some mobile phones and it is likely to become a common feature of mass product phones. Indoor positioning is also available based on a wide range of enabling technologies including ultrasounds, Wi-Fi, and Bluetooth. Considering that mobile phones are rapidly evolving into multipurpose devices that can access a wide range of services, there is a general concern about how positioning information is stored, managed and released to possibly untrusted service providers.

This paper considers the privacy issues involved in accessing location-based services, i.e., services that, based on the user current position, can provide location-aware information. Typical examples are map and navigation services, services that provide information on close-by public resources (e.g., gas stations, pharmacies, ATM machines, ...), services that provide localized news (e.g., weather forecasts, road constructions, etc.), as well as more personalized services like proximity marketing or friend-finder.

* This work is partially supported by NSF under grants IIS-0430402 and IIS-0242237. The work of Bettini is also partially supported by the Italian MIUR (FIRB ”Web-Minds” project N. RBNE01WEJT_005)

In principle, most location-based services do not require the personal identification of the user. However, better service may be provided if personalization is allowed. In order to obtain personalized service without revealing personal information, we may use a trusted middleware infrastructure to make sure that only pseudonyms are sent to the service providers, hence making the service requests anonymous for them. Such pseudonyms can also be helpful when the accounting of service usage is performed.

The problem, however, is that positioning information, in the form of a specific location or of a movement trace, can actually lead to personal identification, hence revealing the association between a pseudonym and a real person. For example, a service request containing as location information the exact coordinates of a private house provides sufficient information to personally identify the house's owner since the mapping of such coordinates to home addresses is generally available and a simple look up in a phone book (or similar sources) can reveal the people who live there. If several requests are made from the same location with the same pseudonym, it is very likely that the user associated with that pseudonym is a member of the household.

An obvious solution might be to make all requests very coarse in terms of spatial and temporal resolution. However, for some services to be useful, sufficiently fine resolution must be used. With fine resolution, however, location data may become a quasi-identifier. A quasi-identifier (see Section 4 for more details) is similar to a social security number in that with some external information source, a specific person can be identified. Any service request containing data that becomes sensitive once associated with a user's identity is a potential threat to the user's privacy. Hence, the challenge is to obtain useful service without revealing personal privacy.

By sensitive data we mean information of general concern, like medical information or financial data that could be transmitted as part of a service request; but it may also be the spatio-temporal information regarding the user, as possibly collected by a location-based service provider. Examples include a) information on the specific location of individuals at specific times, b) movement patterns of individuals (specific routes at specific times and their frequency), c) personal points of interest (frequent visits to specific shops, clubs, or institutions).

The problem we are addressing can be stated as follows:

> *We must ensure that no sensitive data is released to a service provider when the data can be personally identified through a location-based quasi-identifier.*[4]

The main contributions of this paper are the following:

- By defining location-based quasi-identifiers and by introducing the notion of Historical k-anonymity, we provide a formal framework to evaluate the risk of revealing personal sensitive information based on location data.
- We propose a technique to preserve a specified level of anonymity, and identify several promising research directions on this topic.

[4] See Section 4 for a formal definition of location-based quasi-identifiers.

The rest of the paper is organized as follows. In Section 2 we discuss related work. In Section 3 we present the service delivery model we are taking as a reference in our work. In Section 4 we define location quasi-identifiers, while in Section 5 the central notion of historical k-anonymity is presented. In Section 6 we present preliminary ideas about algorithms for preserving historical k-anonymity, and we mention several issues that deserve further investigation. Section 7 concludes the paper.

2 Related Work

The problem we are addressing has many analogies with the problem of guaranteeing anonymity of personal data extracted from a relational database (see e.g., [14]). Typical solutions involve either the de-identification of data, essentially avoiding the presence of quasi-identifiers, the obfuscation of sensitive data, or the separation of quasi-identifiers from sensitive data. The first two solutions are usually based on the generalization or suppression of attribute values. Despite we will show that there are specific issues that distinguish the location-based problem from the analogous one in the relational database scenario, similar techniques can be applied. Indeed, the dynamic change of spatio-temporal resolution that we illustrate in Section 6 is an obfuscation technique based on generalization.

Considering the delivery of positioning data, the IETF Geopriv working group [7] has focused on the design of protocols and APIs that enable devices to communicate their location in a confidential and integrity-preserving manner to a location server. Then, the location server is assumed to deliver data to other services accordingly to the user's privacy policies, possibly including the use of pseudonyms instead of the real user identity. This work can be considered complementary to ours.

The idea of adapting spatio-temporal resolution to provide a form of location k-anonymity can be found in [11]. This work is extended in [9] to support the use of a different value of k for different requests. However, the notion of k-anonymity used in [9] is slightly different: the authors consider a message sent to a service provider to be k-anonymous, only if there are other k-1 users in the same spatio-temporal context that actually send a message. This is a debatable interpretation of the k-anonymity concept, and differs from the one used in our paper as well as in [11]. We only require the presence in the same spatio-temporal context of k-1 *potential* senders, which is a much weaker requirement.

Independently from the above issue, [11] and [9] address a special case of the problem considered in this paper, characterized by assuming that each location is a quasi-identifier, and that the simple fact of issuing a request is sensitive information. We believe that this assumption on the quasi-identifier is actually a very strong one, similar to assuming that an external source (e.g. a camera) would be available at each location allowing the identification of all users that were at that location in any given time interval. The solution proposed in [11] ensures that the request may have been issued by anyone of k users present at a certain location in the time interval specified in the request. With respect to this work, our framework addresses the issue of defining what a location-based quasi-identifier is, enabling a wide range of assumptions about how easy it would be to re-identify a subject in a specific context. Moreover, we extend to traces the notion of k-anonymity.

Location privacy issues have also been addressed in [2, 1]. In particular, the authors propose and deeply investigate the notion of a *mix-zone*. A mix-zone is analogous to a mix node in communication systems [6], and can be intuitively described as a spatial area such that, if an individual crosses it, then it won't be possible to link his future positions (outside the area) with known positions (before entering the area). Here, "link" means the association of different requests to the same user. While it is not the focus of this paper to analyze mix-zones, we consider it a very useful notion, and we use it in Section 6 as part of algorithmic solutions for the preservation of historical k-anonymity.

3 The anonymous location-based service model

Our investigation assumes a specific service provisioning model described by the following scenario (see Figure 1). This model is assumed as well in [11, 9], and it closely reflects the reality of current systems. Although a trusted server presents risks in terms of a single-point trust, since mobil devices are usually limited in their capabilities, the assumed model as we adopt here is a reasonable assumption.

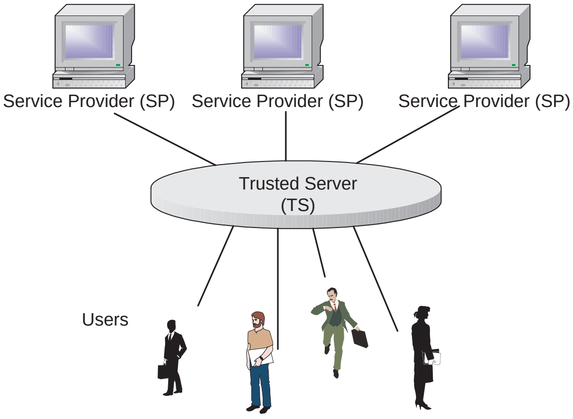

Fig. 1. Service provisioning model.

– Users invoke or subscribe to location-based remote services that are going to be provided to their mobile devices. Users can turn on and off a privacy protecting system which has a simplified user interface with qualitative degrees of concern: low, medium, high. The user choice may be applied uniformly to all services or selectively. More expert users can have access to more involved rule-based policy specifications.

- User sensitive information, including user location at specific times and possibly other data needed for the service request, is collected and handled by a Trusted Server (TS). TS has the usual functionalities of a location server (i.e., a moving object database storing precise data for all of its users and the capability to efficiently perform spatio-temporal queries). Qualitative privacy preferences provided by each user are translated by the TS into specific parameters. The TS has also access to the location-based quasi-identifier specifications (see Subsection 6.1).
- Service Providers (SP) receive from TS service requests of the form

  $$(\textbf{msgID, UserPseudonym, } \langle \textbf{Area, TimeInterval} \rangle \textbf{, Data}).$$

  The **msgID** is used to hide the user network address and will be used by the TS to forward the answer to the user's device; **UserPseudonym** is used to hide the user identity while allowing the SP to authenticate the user, to connect multiple requests from the same user, and possibly to charge the user for the service (through a third party which has the mapping to identity and payment instruments). The field ⟨**Area, TimeInterval**⟩ defines a spatio-temporal context in which the request was issued. While the TS knows the exact point and exact time when the user issued a request, both **Area** and **TimeInterval** provide possibly generalized information in the form of an area containing the exact location point, and of a time interval containing the exact instant. Finally, **Data** is a set of attribute-value pairs depending on the specific service and request.
- Service providers fulfill the requests sending the service output to the user's device through the trusted server.

4 Location-based Quasi-Identifiers

In a database table storing personal information about individuals, a set of attributes is called a *quasi-identifier* [8] if their values, in combination, can be linked with external information to reidentify the respondents to whom the information refers. A typical example of a single-attribute quasi-identifier is the Social Security Number, since knowing its value and having access to external sources it is possible to identify a specific individual. In this work we consider in particular the possibility of reidentifying the respondents based on attributes revealing spatio-temporal data, possibly considering histories of attribute values. A relevant issue, not actually addressed in previous work, is how to define and how to represent such location-based quasi identifiers (LBQIDs in the sequel). Since the choice of an LBQID implies certain assumptions on the accessibility of external sources to identify the user, we believe that this is a crucial point in defining what is really a privacy concern for location-based services.

We propose to represent LBQIDs as spatio-temporal patterns, as intuitively illustrated in Example1.

Example 1. A user may consider the trip from the condominium where he lives to the building where he works every morning and the trip back in the afternoon as an LBQID if observed by the same service provider for at least 3 weekdays in the same week, and for at least 2 weeks.

The derivation of a specific pattern or a set of patterns acting as LBQIDs for a specific individual is an independent problem, and is not addressed in this paper. However, it should be clear that the derivation process will have to be based on statistical analysis of the data about users movement history: If a certain pattern turns out to be very common for many users, it is unlikely to be useful for identifying any one of them. The selection of candidate patterns may also possibly be guided by the user. Since in our model it is the TS which stores, or at least has access to, historical trajectory data, it is probably a good candidate to offer tools for LBQID definition.

Definition 1. *A* Location-Based Quasi-Identifier *(LBQID) is a spatio-temporal pattern specified by a sequence of spatio-temporal constraints each one defining an area and a time span, and by a recurrence formula.*

Each location in the sequence is represented analogously to the spatio-temporal context used in each request, i.e., ⟨**Area, U-TimeInterval**⟩ where **Area** identifies a set of points in bidimensional space (possibly by a pair of intervals $[x_1, x_2][y_1, y_2]$), and **U-TimeInterval** is a *unanchored* time interval $[t_1, t_2]$. Differently from the specification of the time interval at request time, here the values t_1 and t_2 represent unanchored instants of time. For example, the **U-TimeInterval** = [7am,9am] defines the time span of two hours starting at 7am and ending at 9am in a *generic* day. This interval is called unanchored since it does not identifies a specific time interval on the timeline, but an infinite set of intervals, one for each day.

A recurrence formula is associated with each location sequence and it follows the following syntax:

$$r_1.G_1 * r_2.G_2 * \ldots * r_n.G_n$$

where for each $i = 1 \ldots n$, r_i is a positive integer, and G_i is a time granularity, as formally defined in [3].

The intuitive semantics is the following: each sequence must be observed within a single granule of G_1. The value r_1 denotes the *minimum* number of such observations. All the r_1 observations should be within one granule of G_2, and there should be at least r_2 occurrences of these observations. The same semantics clearly extends to n granularities.

Example 2. The LBQID intuitively described in Example 1 can be formally defined as follows:

⟨AreaCondominium [7am,8am], AreaOfficeBldg [8am,9am],
AreaOfficeBldg [4pm,6pm], AreaCondominium [5pm,7pm]⟩
Recurrence: 3.Weekdays * 2.Weeks

Accordingly to the semantics of this expression, each round-trip from home to office and vice versa should be observed in the same weekday, there should be 3 observations in the same week, and for at least 2 weeks.

Considering the given language syntax and semantics, we can observe that any subexpression $1.G_n$ at the end of a recurrence formula can be dropped, since it is implicit. For the formula to be satisfied, it is also implicitly necessary that there are at least r_i granules of G_i, each containing at least r_{i-1} granules of G_{i-1}. If the recurrence formula is empty, it is assumed equivalent to $1.\top$, hence the sequence can actually appear just once at any time.

Note that if user-defined time granularities are allowed, recurrence formulas can also express patterns like "same weekday for at least 3 weeks", or "at least two consecutive days for at least 2 weeks". In the first case we may use the granularities Mondays, Tuesdays, etc., and use a different LBQID for each one of them. The second pattern may require a special granularity having each granule composed of 2 contiguous days.

However, the formalism we propose is only one among many that could be used to represent recurring spatio-temporal sequences. Some of the work on recurring temporal patterns may also be considered [13, 10]. The choice of a particular formalism is not crucial for our approach, as long as there are algorithms to continuously verify if the patterns are matched by the positioning data associated with the users' requests.

Considering the language proposed above, a timed state automata [4] may be used for each LBQID and each user, advancing the state of the automata when the actual location of the user at the request time is within the area specified by one of the current states, and the temporal constraints are satisfied. Details about monitoring LBQIDs are outside the scope of this paper.

For specifying our framework we only need to define the notion of a set of requests *matching* an LBQID.

Definition 2. *If $\langle x_i, y_i, t_i\rangle$ is the exact location and time of a request r_i, as seen by the TS, r_i is said to* match *an element E_j of an LBQID if* ***Area***$_j$ *contains $\langle x_i, y_i\rangle$ and t_i is contained in one of the intervals denoted by* ***U-TimeInterval***$_j$.

Definition 3. *A set of requests R' is said to* match *an LBQID Q if the following conditions hold: (1) each request r_i in R' matches an element E_j of Q, and vice versa each element E_j is matched by a request r_i in R'; (2) if t_i is the time instant of a request r_i matching E_j, the set of t_i's for all $r_i \in R'$ must satisfy the temporal constraints imposed by the recurrence formula of Q.*

5 A Privacy Preservation Framework Based on k-Anonymity

In this section we present the principles defining our framework for privacy preservation. The framework has the main goal of enabling a quantitative evaluation of the effectiveness of privacy preservation solutions.

5.1 The notion of k-anonymity for location-based services

As mentioned earlier, when a user does not want to be recognized when performing an action, like issuing a service a request, making a call, or informing a service provider of the present location, one general solution is to make requests anonymous with pseudonyms. The reason that pseudonyms lead to anonymity is because there exists a set of many individuals, each of whom could have used any given pseudonym. Hence, anonymity can be intuitively described as the property of being indistinguishable among a set of individuals. The *anonymity set* concept was probably first defined in [5] in an analogous context as *the set of participants who could have sent a certain request, as seen by a global observer.* According to this definition the cardinality of the anonymity set gives a measure of anonymity level. The greater the k value, the higher the level of anonymity. In our context, the anonymity of the service requests may be obtained as follows:

Given a measure k of desired anonymity level, algorithms should be applied at the TS to guarantee that the SP will not be able, using location data, to bind a request to an anonymity set with fewer than k users.

A location-based anonymization algorithm based on this notion was first proposed in [11]. The main idea is to forward a request to the SP only when at least k different subjects have been in the space defined by **Area** in anyone of the subintervals of **TimeInterval**. Since any of the subjects may have issued the request, even if the SP had access to a direct observation of the area, the SP may not determine which of the subjects actually issued the request.

Note that this approach has an implicit assumption (not mentioned in [11]): There is a very low probability that all the individuals in the anonymity set will actually make exactly the same request in the same time interval. Indeed, if this is not the case, the cardinality of the set is irrelevant, since it would be sufficient to know that an individual belongs to the set in order to know that he/she actually issued the request. A similar assumption is made when dealing with k-anonymity in relational databases. In the following we also make this assumption and we assume that it has been validated by a data statistical analysis. Note that [9] requires that

In this paper we also take into account sequences of requests issued by the same individual. These sequences are usually identified by service providers when the service requires authentication, since each request is explicitly associated with a userid. We remind that in our model, a **UserPseudonym** is included in each request and it is also used to authenticate the user. Central to our framework are the notions of *Service Request Linkability* and *Historical k-anonymity.*

5.2 Service Request Linkability

Intuitively, service request linkability (*linkability* for shortness) is a measure of the possibility, by looking at the set of service requests issued to a service provider, to guess that two requests have been issued by the same user. Any two requests with the same UserPseudonym are clearly linkable, since we assume that pseudonyms are not shared by different individuals. Using different pseudonyms for the same individual may not be sufficient to make his/her requests unlinkable. In general computing linkability is not a trivial task, since various techniques can be used to link two requests. The issue has been investigated in [12] considering multi target tracking techinques to associate the location of a new request with an existing trace; if this association succeeds, the new request is considered linkable (with a certain probability) to all the requests used in the trajectory. However, many other techniques could also be applied, including pattern matching of traces (to guess, for example, recurring traces), or probability-based techniques considering most common trajectories based on physical constraints like roads, crossings, etc.. While it is out of scope in this paper to consider specific linking techniques, we assume the TS can replicate the techniques used by a possible attacker, hence computing a likelihood value for the linkability of any pair of issued requests.

Definition 4. *Given the set $R = \{r_1, \ldots, r_n\}$ of all requests issued to a certain SP, linkability is represented by a partial function $Link()$ from $R \times R$ to $[0, 1]$, intuitively defining for a pair of requests r_i and r_j in R the likelihood value of the two requests being issued by the same individual.*

The $Link()$ function is assumed to have some simple properties: $Link(r_i, r_j) = Link(r_j, r_i)$ (symmetricity), and $Link(r_i, r_i) = 1$ (reflexivity).

For each request we can define the set of all requests linkable to it as follows.

Definition 5. *Let R be the set of all the requests issued to a certain SP and $R' \subseteq R$. Then we say R' is* link-connected with likelihood Θ *if for each pair r_i and r_j of requests in R', there exist $r_{i_1}, \ldots, r_{i_k}$ in R', where $r_{i_1} = r_i$ and $r_{i_k} = r_j$, such that $Link(r_{i_l}, r_{i_{l+1}}) \geq \Theta$ for all $l = 1, \ldots, k-1$.*

Note that we may say that the $Link()$ function is *correct* if, for each set R' of requests, the following holds: all the requests of R' belong to the same user if and only if R' is link-connected with $\Theta = 1$.

5.3 Historical k-Anonymity

In order to define historical k-Anonymity we need some preliminary definitions.

The trusted server not only stores in its database the set of requests that are issued by each user, but also stores for each user the sequence of his/her location updates. We call this sequence *Personal History of Locations.*

Definition 6. *The sequence of spatio-temporal data associated with a certain user in the TS database is called his/her Personal History of Locations (PHL), and it is represented as a sequence of 3D points $(\langle x_1, y_1, t_1\rangle, \ldots, \langle x_m, y_m, t_m\rangle)$, where $\langle x_i, y_i\rangle$, for $i = 1, \ldots, m$, represents the position of the user (in two-dimensional space) at the time instant t_i.*

Note that a location update may be received by the TS even if the user did not make a request when being at that location. Hence, for each request r_i there must be an element in the PHL of $User(r_i)$, but the vice versa does not hold. This has an intuitive motivation in the fact that the anonymity set for a certain area and a certain time interval is the set of users who were in that area in that time interval, and who could *potentially* make a request.

We also need to define the following relationship between PHLs and sets of requests issued to the SP.

Definition 7. *A PHL* $(\langle x_1, y_1, t_1\rangle, \ldots, \langle x_m, y_m, t_m\rangle)$ *is said to be* location-time-consistent*, or* LT-consistent *for short, with a set of requests* $r_1, \ldots, r_n$ *issued to an SP if for each request* r_i *there exists an element* $\langle x_j, y_j, t_j\rangle$ *in the PHL such that the area of* r_i *contains the location identified by the point* x_j, y_j *and the time interval of* r_i *contains the instant* t_j*.*

We can now define Historical k-Anonymity.

Definition 8. *Given the set* R *of all requests issued to a certain SP, a subset of requests* $R' = \{r_1, \ldots, r_m\}$ *issued by the same user* U *is said to satisfy* Historical k-Anonymity *if there exist* $k-1$ *PHLs* $P_1, \ldots, P_{k-1}$ *for* $k-1$ *users different from* U*, such that each* P_j*,* $j = 1, \ldots, k-1$*, is LT-consistent with* R'*.*

What we want to do below is to make sure that if a set of requests R' matches an LBQID and is link connected with a certain likelihood, then R' satisfies historical k-anonymity. This means that if an SP can successfully track the requests of a user through all the elements of an LBQID, then there would be at least $k-1$ other users whose personal history of locations is consistent with these requests; in other words, from the SP perspective, there would be at least k users who may have issued those requests.

From the above definitions it should be clear that the two main parameters defining a "level of privacy concern" in our framework are k, the anonymity value, and Θ, the linkability likelihood.

6 Preserving Historical k-anonymity

While several strategies may be devised for privacy preserving in our framework, here we illustrate a simple approach that may be used as a starting point to develop more sophisticated techniques.

6.1 A simple strategy

We assume that each location-based service has some *tolerance constraints* that define the coarsest spatial and temporal granularity for the service to still be useful. For example, consider a service that returns information on the closest hospital. For the service to be useful, it should receive as input a user location that is at most in the range of a few square miles, and a time-window for the actual request time of at most a few minutes. On the contrary, a service providing localized news may even work reasonably with much coarser spatial and temporal granularities.

Our strategy can be summarized as follows:

1. The TS monitors all incoming user requests for the possible release of LBQIDs. While the TS knows the exact position and time ($\langle x, y, t \rangle$) associated with a request r, they are generalized when r is forwarded to the SP in the following case: r matches an element E_j of an LBQID for $User(r)$ such that (1) either E_j is the first element or (2) a previous request r' by the same user has matched E_{j-1} and the time instants associated with r' and r satisfy the temporal constraints specified in Q between E_{j-1} and E_j. The generalization essentially consists in enlarging the Area and TimeInterval, hence increasing the uncertainty about the real user location and time of request. Generalization is performed by an algorithm that tries to preserve Historical k-Anonymity of the set of requests that have matched the current partial LBQID.
2. If, for a particular request, generalization fails, (i.e., historical k-anonymity is violated), the system will try to unlink future requests from the previous ones by changing the pseudonym of the user. If unlinking succeeds before a complete LBQID is matched, all partially matched patterns based on old pseudonym for that user are reset. Otherwise, the user is considered at risk of identification, and notified about it so that he may refrain from sending sensitive information, disrupt the service, or take other actions.

Clearly, if the generalization algorithm succeeds for each LBQID sub-pattern, k-historical anonymity is satisfied. If the algorithm fails, it does not mean that an LBQID would be revealed. Indeed, if Step 2 succeeds in changing the pseudonym of the user before a complete LBQID is matched, no further requests with the old pseudonym will occur and the partially matched patterns would have no chance to be completed.

We now show the generalization algorithm explaining how it can guarantee Historical k-Anonymity. For simplicity, we make the assumption that each request can match an element in only one of the LBQIDs defined for a certain user. The algorithm can be easily extended to consider multiple LBQIDs.

6.2 A Spatio-temporal Generalization Algorithm

Algorithm 1 is composed of two main steps (lines 1 and 8 in the listing). If the current request matches the first element in an LBQID, then steps at lines 5 and 6 are executed selecting from all the user PHLs the k points (each one from a different user) that are closest to the point corresponding to the request. For requests matching intermediate elements in an LBQID, Steps at lines 2 and 3 are performed using the PHLs of the k users selected when the request matching the first element of the LBQID has been processed. The second main step (line 8) simply checks if the generalization satisfies the tolerance constraints. If it doesn't, in order not to disrupt the service, the computed area and time interval are shrunk as much as required to satisfy the constraints. A *False* value is returned for the variable HK-anonymity to notify the failure in proper generalization.

The most time consuming step is the one at line 5. This can be performed using a brute-force algorithm by simply considering the nearest neighbor in the

Algorithm 1 The generalization algorithm

Input: $\langle x, y, t\rangle$ as position and time of request r, k user-ids (if r matches the initial element of an LBQID) or a parameter k, tolerance constraints;

Output: $\langle$ **Area**, **TimeInterval** $\rangle$, boolean value for HK-anonymity, k user-ids (if r matches the initial element of an LBQID);

Method:

```
 1: if $k$ user-ids are given as part of the Input then
 2:    For each of the $k$ user-ids, find the 3D point in its PHL closest to $\langle x, y, t\rangle$.
 3:    Compute $\langle$ Area, TimeInterval $\rangle$ as the smallest 3D space containing these
       points
 4: else
 5:    Compute $\langle$ Area, TimeInterval $\rangle$ as the smallest 3D space (2D area + time)
       containing $\langle x, y, t\rangle$ and crossed by $k$ trajectories (each one for a different user)
 6:    Store the ids of the $k$ users.
 7: end if
 8: if $\langle$ Area, TimeInterval $\rangle$ satisfies the tolerance constraints then
 9:    HK-anonymity := True
10: else
11:    HK-anonymity := False
12:    Area and TimeInterval are uniformly reduced to satisfy the tolerance constraints
13: end if
```

PHL of each user and then taking the closest k points. In this case, the worst case complexity of this step is $O(k * n)$ where n is the number of location points in the TS. Optimizations may be inspired by the work on indexing moving objects. The computation necessary for steps at lines 2 and 3 is quite simple, considering that it is restricted to the traces of k users, and that this number is usually much smaller than the total number of users.

In order to make the algorithm practical several issues still have to be addressed. The most relevant one is the trade-off between quality of service (i.e., how strict tolerance constraints should be), degree of anonymity (i.e., choice of k), and frequency of unlinking (i.e., number of possible interruptions of the service). These parameters must be considered carefully, possibly based on the user policies. Based on specific objective functions, several techniques can then be applied to improve the algorithm. For example, if we want to ensure historical k-anonymity, we should probably use an initial parameter k' larger than k. Indeed, the longer the trace, the less are the probabilities that the same k individuals will move along the same trace (even considering generalizations along the space and time dimensions). Starting with a larger k and decreasing its value at each point in the trace, until k is reached, should increase the probability to maintain historical k-anonymity for longer traces. Guidance on the choice of k' and on the value by which it should be decremented at each step should come from the analysis of historical data.

6.3 Unlinking techniques

Unlinking is performed by changing the pseudonym of the user, possibly doing it when he crosses a mix-zone [2] (see Section 2), in order for the SP not to be able of binding the different pseudonyms to the same user. Mix-zones have been defined as “natural” locations where no service is available to anybody and specific conditions are satisfied such that it becomes very difficult for an SP to link two requests from the same user if the user has crossed the mix-zone. The definition is partly due to the fact that the class of services considered in [2] are specific to certain areas (e.g., all branches of a department store), leaving most of the remaining space unserviced.

We are interested in defining mix-zones on-demand, for example temporarily disabling the use of the service for a number of users in the same area for the time sufficient to confuse the SP. Technically, we may define the problem as that of finding, given a specific point in space, k diverging trajectories (each one for a different user) that are sufficiently close to the point. The “diverging” feature should capture the intuitive idea that these users, once out of the mix-zone, will take very different trajectories.

We abstract the above into an action called “Unlinking with a likelihood parameter Θ”. This action will make sure that two requests, when unlinked, will (1) have two pseudonyms pID_1 and pID_2, and (2) $Link(r_1, r_2) < \Theta$ for all requests r_1 and r_2 having pID_1 and pID_2, respectively.

We can now state our correctness result for our algorithm:

Theorem 1. *If we apply our strategy with Algorithm 1, and we assume we can always perform Unlinking for a certain likelihood parameter Θ, then, given an anonymity value k, any set of requests issued to an SP by a certain user that matches one of his/her LBQIDs and is link connected with likelihood Θ, will satisfy Historical k-anonymity.*

By choosing an appropriate k, Theorem 1 ensures that no SP may use an LBQID to personally identify a user.

7 Conclusions and Open Issues

In this paper we have formally defined the problem of the personal identification of sensitive data in location-based services. We believe that the formal framework we have defined can be used for two very different purposes:

(a) to enforce a certain level of privacy – possibly disabling the service when the level cannot be guaranteed –, and

(b) to evaluate if the privacy policies that a location-based service guarantees are sufficient to deploy the service in a certain area. This may be achieved by considering, for example, the typical density of users, their movement patterns, their concerns about privacy, as well as the spatio-temporal tolerance constraints of the service and the presence of natural mix-zones [1] in the area.

While in this paper we presented preliminary results about (a), we consider (b) as another promising research direction.

Regarding a) we already pointed out several issues that deserve further investigation, including monitoring multiple LBQIDs, efficient generalization algorithms and unlinking techniques. In addition, randomization should be used as part of the TS strategy to prevent inference attacks.

Another interesting open issue regards user interfaces. On one side, very simple tools should be provided to define LBQIDs and verify them based on statistical data. On the other side, simple and effective interfaces are needed to specify the level of anonymity required by the user, as well as to notify when identification is at risk. Graphical solutions, like the open and closed lock in an internet browser should be considered.

References

1. A. Beresford, F. Stajano, Mix Zones: User Privacy in Location-aware Services. In Proc. IEEE Workshop on Pervasive Computing and Communication Security (PerSec), pp. 127-131, IEEE, 2004.
2. A. Beresford, F. Stajano. Location Privacy in Pervasive Computing. IEEE Pervasive Computing, 2(1):46-55, 2003.
3. C. Bettini, S. Jajodia, X.S. Wang, Time Granularities in Databases, Data Mining, and Temporal Reasoning, Springer, 2000.
4. C. Bettini, X. Wang, and S. Jajodia. Testing complex temporal relationships involving multiple granularities and its application to data mining, in Proc. of ACM Symposium in Principles of Database Systems (PODS), ACM press, 1996.
5. D. Chaum, The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. Journal of Cryptology 1(1): 65-75, 1988.
6. D. Chaum, Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Communications of the ACM, 24(2): 84-88, 1981.
7. J. Cuellar, J. Morris, and D. Mulligan. Internet Engineering task force geopriv requirements. http://www.ietf.org/html.charters/geopriv-charter.html, 2002.
8. T. Dalenius. Finding a needle in a haystack – or identifying anonymous census record. Journal of Official Statistics, 2(3):329–336, 1986.
9. B. Gedik, L. Liu. A Customizable k-Anonymity Model for Protecting Location Privacy. The 25th International Conference on Distributed Computing Systems (IEEE ICDCS 2005).
10. I. Goralwalla, Y. Leontiev, M. Özsu, D. Szafron, Carlo Combi, Temporal Granularity: Completing the Puzzle, J. Intell. Inf. Syst. 16(1): 41-63, 2001.
11. M. Gruteser, D. Grunwald, Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In Proc. of MobiSys 2003.
12. M. Gruteser, B. Hoh, On the Anonymity of Periodic Location Samples In Proc. of 2nd International Conference on Security in Pervasive Computing, LNCS series, Springer, 2005.
13. L. Khatib, R. Morris Generating Scenarios for Periodic Events with Binary Constraints, In Proc. of TIME, pp. 67–72, IEEE, 1999.
14. P. Samarati, Protecting Respondents' Identities in Microdata Release, IEEE Trans. Knowl. Data Eng. 13(6): 1010–1027, 2001.