2012 IEEE 13th International Conference on Mobile Data Management

# Privacy-preserving sharing of sensitive semantic locations under road-network constraints

Emre Yigitoglu
TOBB University
Ankara, Turkey
eyigitoglu@etu.edu.tr

Maria Luisa Damiani
University of Milan
Milan, Italy
mdamiani@dico.unimi.it

Osman Abul
TOBB University
Ankara, Turkey
osmanabul@etu.edu.tr

Claudio Silvestri
Università Ca' Foscari Venezia
Venice, Italy
silvestri@dsi.unive.it

***Abstract*—This paper presents a privacy-preserving framework for the protection of sensitive positions in real time trajectories. We assume a scenario in which the sensitivity of user's positions is space-varying, and so depends on the spatial context, while the user's movement is confined to road networks and places. Typical users are the non-anonymous members of a geo-social network who agree to share their exact position whenever such position does not fall within a sensitive place, e.g. a hospital. Suspending location sharing while the user is inside a sensitive place is not an appropriate solution because the user's stopovers can be easily inferred from the user's trace. In this paper we present an extension of the *semantic location cloaking* model [1] originally developed for the cloaking of non-correlated positions in an unconstrained space. We investigate different algorithms for the generation of cloaked regions over the graph representing the urban setting. We also integrate methods to prevent velocity-based linkage attacks. Finally we evaluate experimentally the algorithms using a real data set.**

## I. Introduction

Location sharing is an increasingly popular location-based information service (LBS), available for example in geo-social networking applications, such as Google Latitude and Glympse, to enable users equipped with a location-aware client to share their position with *friends*. Position is typically computed by a third party, the network location provider (e.g. Skyhook Wireless), based on the contextual information sent by the client, e.g. the wifi networks in the vicinity. In dense urban areas, individuals can be tracked both in indoor and outdoor spaces with a spatial accuracy of a few tens of meters. Moreover, following common practices, the requesters of the location service are not anonymous.

In this paper we focus on the issue of protecting the users of a location sharing application, located in an urban setting, against the risk of *semantic location identification* [1]. The problem is to prevent the disclosure of users' positions to untrusted LBS providers and friends, when users stop in some sensitive semantic location (or place) along the way. A sensitive place is a bounded place within which any position is considered as sensitive information, e.g. a hospital. Following the advances in positioning technology, identifying the places in which users stay is becoming more and more easy [2].

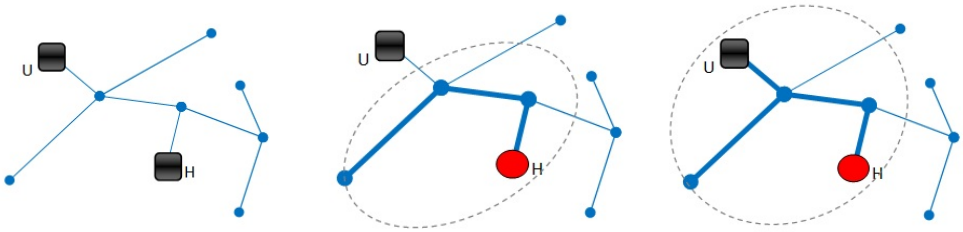
An example of urban setting is shown in Figure 1. The map shows a number of places in Milan[1]: the premises of the Policlinico hospital, the University of Milan, a few religious buildings, various private buildings, and the road network. Assume that the user Bob connects to the location sharing service through a mobile device, e.g. a smartphone, requesting the location service to a trusted network location provider. Bob is driving his car when in the proximity of the Policlinico hospital, Bob stops in a parking area and steps onto the hospital premises where he remains for a few hours for a medical visit, before again taking the car to reach his friends in a pub in downtown. During this time, Bob's position is continuously reported to the LBS provider as well as his friends, therefore the route and the places in which Bob stops as well as the time spent in each of those places are made known to the untrusted parties (i.e. the adversary), including the hospital that Bob considers a sensitive place. Of course, Bob could decide to disconnect himself from the location sharing service. However that would prevent Bob from being in touch with his friends, unless suspending and then resuming the service which would create considerable burden to Bob.


Fig. 1. Urban setting including a hospital (H) and a university campus (U).

To overcome this problem, policy-based solutions specifying privacy rules such as "do not disclose the position if I am nearby a hospital" are not really helpful, because the adversary could infer the destination from the analysis of the user's trace. Also the approaches based on the mix-zone and location anonimyzation paradigms, such as [3], [4] are not appropriate, because of the assumption that users are not anonymous.

[1]The map is drawn from http://www.openstreetmap.org

978-0-7695-4713-8/12 $26.00 © 2012 IEEE
DOI 10.1109/MDM.2012.48




(a) (b) (c)

Fig. 2. (a) Graph-based representation of the urban setting with two places, Hospital and University; (b) a naive CR (in bold inside the circle) ; (c) the CR (in bold inside the circle) comprising the two places.

A more robust approach is semantic location cloaking [1], [5]. Relevant features of this solution are: a) privacy can be personalized, i.e. a privacy profile specifies sensitive place types and the desired degree of privacy for each of those types; b) *location cloaking* algorithms generate cloaked regions (CRs) which cover sensitive places while satisfying the preferences in the privacy profile independently of the user's actual position. These algorithms, which can be defined as *location oblivious*, prevent any inference on the possible correlation between the users' position and the CR; c) finally the *position transformation* operation matches users' position against the set of CRs. If such a position falls into one of those CRs then that CR becomes the location which is shared, otherwise the actual position is disclosed.

Unfortunately, semantic location cloaking methods have been designed to work only in unconstrained spaces in which users can move without restrictions, while in an urban setting the movement is confined to road networks (such as local streets, railways, highways) and places. This calls for a different model of CR grounded on the urban topology. Such scenario also brings to the forefront another major requirement. In particular a CR should not only blur the actual position but also possible stopovers in sensitive places.

As an example, Figure 2.(a) shows the graph-based representation of (a small portion of) the previous urban setting. The nodes of the graph represent places (black rectangles) and road junctions (blue circles), while the edges represent two-ways road segments. This figure shows two places, an hospital (H) and a university (U). For example, a CR blurring the hospital could include, besides the hospital, a number of road segments in proximity (in bold in Figure 2.(b)). Note, however, that if the user remains in this CR for a sufficiently long time, for example longer than the time needed for traversing all the roads, it is very likely that the user stops at the hospital because there is no alternative place in the CR in which the user can reasonably spend much time. The user's stopover is thus disclosed.

To forestall this privacy breach an approach is to define CRs which contain one or more non-sensitive places. For example the CR in Figure 2.(c) contains also the university which is a highly frequented place. An individual located in the CR can be either in U or in H, or simply driving along the roads. In any case, should the user stop at some place, such place would be uncertain. The higher the popularity of the non-sensitive places in the CR, the lower the chances of linking the user with a sensitive place. This calls for solutions which use background knowledge on places to generate appropriate CRs.

Another requirement to address regards the aforementioned operation of position transformation, i.e. the operation matching the actual user's position against the set of CRs. Whenever the user's position is frequently updated, an adversary can use the information on the speed of the user to prune the CR and thus more precisely localize the user inside the CR. This inference is called velocity-based linkage attack [6]. Since this kind of privacy breach can compromise the effectiveness of the cloaking strategy, countermeasures tailored on the urban setting are to be integrated into the privacy protection framework. In this paper we address all these requirements.

In summary, the major contributions of this paper are:

- We present a comprehensive approach to the problem of safeguarding sensitive positions in an urban setting. The approach extends the semantic location cloaking model and integrates countermeasures against the velocity-based linkage attack.
- We specify two different cloaking techniques for the offline generation of CRs (*static location cloaking*). These techniques generate non-overlapping and overlapping CRs, respectively. This is a major novelty because existing algorithms are only capable of generating non-overlapping CRs. We also adapt these algorithms to the case in which CRs are generated at run-time (*dynamic cloaking*).
- We evaluate those methods bases on a real data set (OpenStreetMap dataset). We show that the generation of overlapping CRs is significantly more efficient and provides more accurate CRs than the generation of non-overlapping CRs.

The remainder of the paper is organized as follows. The next section overviews related work. Section III develops the problem formulation, formally defining the privacy requirements. Algorithms are provided in Section IV, after which we detail an experimental evaluation of our proposal in Section V. Finally, Section VI concludes and presents our agenda for the future work.

## II. Related work

This work relates to two main streams of research, concerning the modeling of semantic trajectories and location privacy, respectively.

*a) Semantic trajectories:* a semantic trajectory is an annotated representation of the trace of a moving object. Although it is not rigorously defined, this notion is used in a variety of applications, such as recommender systems suggesting popular places and tours [2], navigation services in indoor settings [7], compression of raw trajectories [8]. A conceptual data model for the representation of semantic trajectories has been defined by Spaccapietra et al. [9]. In such a model, a semantic trajectory is a sequence of *stops* and *moves* where a stop can represent a place, e.g. home, while a move represents the path, e.g. the sequence of road segments, between two consecutive places. The overall purpose is to provide a way for representing the behavior of moving objects. In this view, blurring sensitive places paves the way to the definition of privacy-preserving semantic trajectories.

*b) Location privacy:* In recent work, Chow & Mokbel [10] survey trajectory privacy techniques in the context of *continuous* LBS (as opposed to *snapshot* LBS), such as [11], [12], and trajectory data publishing, such as [13], [14]. Tacit assumption is that the privacy goal is to safeguard identity privacy [15], because position can act as *quasi-identifier* [16] and thus identity privacy is at stake if users are to be anonymous. In reality, position information can also play, in alternative or in addition to the role of quasi-identifier, the role of *sensitive attribute*. Moreover, the position can have the granularity of place (instead of coordinated point) and be defined in a symbolic way. A number of approaches adopt this viewpoint. For example, Bamba et al. present PrivacyGrid, a system that supplements *location k-anonymity* with *location l-diversity* [17]. In this approach, a cloaked region is a region containing $k$ mobile users and $l$ places (here called static objects, e.g. churches and clinics). There is no distinction between sensitive and non-sensitive places. A similar notion of location l-diversity has been used with linear objects for anonymizing the positions of LBS users driving along a road network [18]. In this case, the exact user position is replaced by a set of segments. The number of segments in the cloaked region defines the degree of diversity. In the previous approaches, the cloaked region (or the set of segments) is $l$-occurrence diverse but not $l$-type diverse. For example, the $l$ places may all be of the same type (e.g. l hospitals). Xue et al. [19] defined location-diversity as the number of different types of places.

In all these solutions, the degree of diversity is measured by counting the number of occurrences or types inside the cloaked region. The fact that places can be differently frequented and so have a different degree of popularity, is not taken into account. An approach that overcomes this limitation while providing guarantees of location diversity in a space of non-uniformly distributed positions in which there are sensitive and non-sensitive places is Probe [5]. All these approaches, however, target snapshots LBS. A different solution which targets the protection of both sensitive positions and identity privacy is presented by Monreale et al. [20], but in a different application context, i.e. trajectory data publishing. We are not aware, instead, of techniques enabling the protection of sensitive positions in continuous LBS under road network constraints, which is the focus of this paper.

## III. Problem Formulation

### A. Background knowledge model and definitions

Let us denote with $PT$ and $P$ the set of place types (e.g. hospital, mall) and places (i.e. Policlinico, Carrefour) in a bi-dimensional coordinate space. We introduce the concept of *annotated city network* to model the background knowledge on the urban setting.

*Definition 1 (Annotated city network):* An annotated city network is a connected and undirected weighted graph $G$=$(V, E, pop, pt, tt)$ where:

i) $V = V_P \bigcup V_j$ is the set of vertices with $v \in V_P$ representing a place and $v \in V_j$ a road junction[2]
ii) $E \subseteq V \times V$ is the non-empty set of edges where edge $(u, v) \in E$ denotes a road segment connecting two road junctions or, alternatively, one road junction and one place. Every pair of places are connected through a path which does not include intermediate places, that is all places are reachable through a sequence of road segments.
iii) Each place has a popularity and a type, expressed by the functions $pop : V_P \rightarrow (0, 1)$ and $pt : V_P \rightarrow PT$, respectively
iv) Every edge $e = (u, v) \in E$ is assigned a weight of travel time, i.e. $tt : E \rightarrow \mathbb{R}$, denoting the minimum time needed to travel from $u$ to $v$, and vice versa. $\Diamond$

Note that the popularity of a place is intended to represent the prior probability that a random user is located in that place. Places having popularity 0 are places that are not reachable and thus are not relevant for our model. We assume that a mapping exists between the points in the coordinate space and the graph elements in $V \cup E$. Accordingly, a true position $(x, y)$ is mapped onto either an edge or a place.

In this model, a *region* is a connected subgraph of the city network, denoted $G' = (V', E')$ with $V' \subseteq V$ and $E' \subseteq E$. The simplest region consists of a single place. In that case the graph degenerates in a singleton graph. In case needed, it is trivial to get an areal representation of the region, by finding the minimum bounding rectangle of the geo-spatial extension of the subgraph. Moreover, as we are in an urban setting, the elements in the region can be also identified by their street address.

Given a region $r$, we define the popularity of a place type $pt$ in $r$, denoted $pop_r(pt)$, as the aggregated popularity of the places of that type located in $r$. Conventionally, $pop_r(.)$ denotes the popularity of the region, i.e. $\sum_{pt_i \in PT} pop_r(pt_i)$.

[2]To simplify the terminology, we use the term place for both the elements in $V_P$ and the corresponding locations

For example the popularity of a region which only encloses roads (and no places) is 0.

Finally, the *real time trajectory* of a user over a city network is a sequence of timestamped regions, i.e. $T = \{(r_1, t_1), (r_2, t_2), \ldots, (r_n, t_n)\}$ with $t_i < t_{i+1}$. The snapshot position $(r_i, t_i)$ means that at time $t_i$ the user is located in the subgraph of region $r_i$ where the subgraph can also be a singleton graph; $(r_n, t_n)$ is the current position. We refer to the real time trajectory which is disclosed to the LBS provider as shared trajectory.

### B. Privacy requirements

We adopt the computational model defined in [1]. Let us introduce the set $PT_S \subseteq PT$ of user-defined sensitive place types. We recall that, in such a model, a set of CRs blurring the sensitive places are first generated, then each user's position is possibly replaced by the CR containing that position. Transposed into our domain, a CR is a region of the city network, i.e. a subgraph, satisfying a set of privacy requirements. We consider two kinds of privacy requirements: the requirements on the single CR, and the requirements over sequences of CRs.

Replacing the true position with a CR impacts the quality of the position information. We measure the quality of the position resulting from the cloaking operation using the following metric, the average diameter of the CRs subgraphs $\{G_1, G_2, \ldots, G_n\}$, i.e.

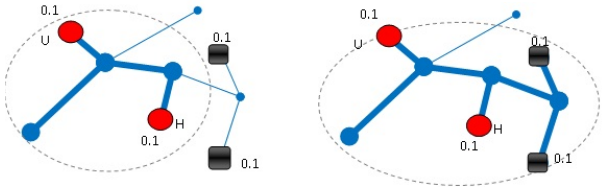
$$QS_{CR} = \frac{1}{n} \sum_{i}^{n} diameter(G_i)$$

We refer the reader to [6] for additional metrics that characterize the loss in service quality due to the protection against the velocity-based attacks. Those metrics can be straightforwardly transposed to our domain.

**Privacy requirements on single CRs**. The privacy profile specifies for each place type $pt_i \in PT$ a user-defined threshold value $\tau_i$ indicating the maximum allowed probability of association between a user and a place of such type. We rule out the case in which $\tau_i = 1$ because it means that the place $pt_i$ is not sensitive. Formally, the pair $(pt_i, \tau_i)$ prescribes that in any CR the posterior probability that a user is in a sensitive place of type $pt_i$ must not exceed the user-defined threshold. The privacy requirement is:

$$\frac{pop_r(pt_i)}{pop_r(.)} \leq \tau_i \tag{1}$$

Consider the example reported in Figure 3. The graph shows two sensitive places (red circles) of type U and H respectively and two non-sensitive places (black rectangles). All places have the same popularity (0.1). Assume a privacy profile consisting of two constraints: $(U, 0.5)$ and $(H, 0.5)$. The CR in Figure 3(a) satisfies the two constraints because the posterior probability that the user is in U (in H) is 0.5. However, it is easy to see that if the user stops in a place within the region (and that can be inferred from the time spent in the region) such place is certainly sensitive. That leads to a privacy breach. In general, this privacy breach occurs if the probability of associating the user with a sensitive place (of any type) exceeds the highest threshold specified in the privacy profile. Formally, this additional privacy requirement can be expressed as follows:




(a) (b)

Fig. 3. (a) Cloaking of two sensitive places of different type. The privacy requirements specified in the privacy profile are satisfied, but not the minimal dislcosure; (b) A strongly cloaked region satisfies also the minimal disclosure requirement.

$$\sum_{pt_i \in PT_S} \frac{pop_r(pt_i)}{pop_r(.)} \leq \max_i\{\tau_i\} \tag{2}$$

We refer to this privacy requirement as *minimal disclosure*. Now we introduce the notion of *strongly cloaked region* and show that it satisfies the minimal disclosure requirement and generalizes the definition given in [1].

*Definition 2 (Strongly cloaked region):* A strongly cloaked region $r$, for a given privacy profile, is a region satisfying the following conditions:

- $r$ contains at least one sensitive place
- The popularity of $r$ satisfies the following inequality :

$$\sum_{pt_i \in PT_S} \frac{pop_r(pt_i)}{\tau_i} \leq pop_r(.) \tag{3}$$

$\Diamond$

*Property 1:* A strongly cloaked region has the following properties:

(1) It satisfies the privacy requirements of the privacy profile
(2) It satisfies the minimal disclosure requirement
(3) It contains at least one place which is not sensitive. In particular, by rewriting inequality 3 as:

$$\sum_{pt_i \in PT_S} pop_r(pt_i) \frac{(1 - \tau_i)}{\tau_i} \leq \sum_{pt_j \in PT_{NS}} pop_r(pt_j) \tag{4}$$

we obtain the condition that must be satisfied by the set of non sensitive places ($PT_{NS}$) in the region.

*Proof sketch.* From inequality 3:

(1) For every place type, it holds that $\frac{pop_r(pt_i)}{\tau_i} \leq pop_r(.)$;
(2) It holds:

$$\sum_{pt_i \in PT_S} \frac{pop_r(pt_i)}{\max_i\{\tau_i\}} \leq \sum_{pt_i \in PT_S} \frac{pop_r(pt_i)}{\tau_i} \leq pop_r(.)$$

(3) By definition a CR must contain at least one sensitive place whose popularity cannot be 0. Therefore the left

member of inequality 4 is greater than 0 and thus also the popularity of non-sensitive places. ◊

An example of strongly cloaked region is illustrated in Figure 3(b). The CR contains, in addition to the sensitive places, also two non-sensitive places (the black rectangles). Therefore the posterior probability that the user is in some sensitive place is 0.5 and thus the minimal disclosure requirement is satisfied.

**Privacy requirements on sequences of CRs.** The effectiveness of the cloaking method can be compromised by the velocity-based linkage attack [6], i.e. an adversary can leverage the information on the maximum velocity to delimit the user's position within the CRs reported in the shared trajectory. We recall that the edges of the city network are weighted with travel time, expressing the minimum time (i.e. maximum velocity) to traverse an edge and that such information is publicly known. To prevent this privacy breach, we redefine the *safety* condition that must hold between CRs for a shared trajectory not to be susceptible to velocity-based linkage attacks [6].

Accordingly, we define the node-pairwise distance $d_{pp}(G_1, G_2)$ between the two CRs $G_1$=$(V_1, E_1)$ and $G_2$=$(V_2, E_2)$ as the longest shortest path between any node in $G_1$ and any node in $G_2$, i.e. $d_{pp}(G_1, G_2)$ = $\max_{v \in_l V_1} \max_{w \in_l V_2}$ *ShortestPath(v,w)*. Notice that the distance along the graph is measured in time units. The safety requirement is as follows: $G_1$ and $G_2$ are safe to disclose if the node-pairwise distance between them is lower than the time $t$ spent by the user to reach $G_2$ from $G_1$ (or vice versa), i.e.:

$$d_{pp}(G_1, G_2) < t \tag{5}$$

**Problem formulation.** In summary the problem can be formulated as follows. Assume the adversary knows (i) the city network, (ii) the user's privacy profile, (iii) the privacy algorithms and (iv) all the previous and current reported positions. The problem is to generate strongly cloaked regions and ensure that those regions are safe at run time, while limiting the loss of quality of the location sharing service.

### C. Architecture

We consider two kinds of architecture: *offline* and *online* . a) In the offline architecture, all the cloaked regions are precomputed, possibly by the client itself, if the device is properly equipped, or by some other party, and recorded on the client. Service requests (i.e. location sharing services) are checked for privacy breach (against the velocity-based linkage attacks) if the respective cloaked region is disclosed. If no, the respective cloaked region is disclosed to the LBS provider, otherwise a transformation is needed. We consider two kinds of transformations: time delay and postdating.

In the time delay mode, the request is postponed in time domain. In the postdating mode, instead of disclosing the actual cloaked region $r_j$, a previous safe position is disclosed. The time delay mode introduces temporal error while the postdating mode introduces spatial error, both measured in time metrics. Unless the time delay is not greater than the acceptable time delay threshold, we prefer time delay over space error. Otherwise, we apply postdating.

b) In the online architecture, both the region cloaking and transformation are done at client side when the services are requested. Hence, this is more computationally demanding than the offline cloaking. However, online cloaking is advantageous for constantly changing city networks (e.g. the popularity of places during the day) and user privacy requirements.

## IV. Algorithms

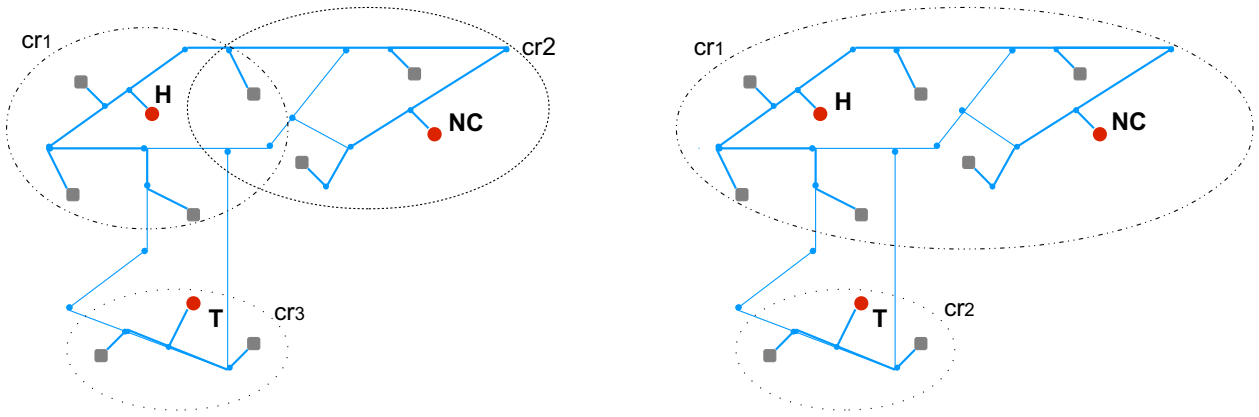
In this section we propose algorithms that satisfy privacy preference for each LBS user according to respective privacy profile. To do so, we first generate cloaked regions and then transform the current location by taking into consideration both the previously reported location sequence and the velocity-based linkage attack. We have two kinds of algorithms, *offline cloaking* and *online cloaking*.

### A. Offline Cloaking

Offline cloaking operates in two stages, (i) offline static cloaking of sensitive places, and (ii) online transformation which ensures no privacy breach against velocity attacks. We consider two cloaking methods: *disjoint* and *overlapping*. Disjoint cloaking allows no overlap between cloaked regions but allows more than one sensitive place to be co-located into a single cloaked region. On the other hand, overlapping cloaking allows overlaps between cloaked regions and assigns only one sensitive place per cloaked region. Note that assigning one sensitive place per cloaked region is possible as all the places are terminal nodes. During the LBS request, in case a privacy breach is detected a transformation (either time delay or postdating) is applied. Figure 4 shows a sample overlapping and disjoint cloaking.

Since a single place can fall in multiple cloaked regions with overlapping cloaking (e.g. $cr1$ and $cr2$ in Figure 4(a)), care must be taken while picking the cloaked region to be reported among alternatives. This is simply because, we have assumed that the attacker knows our algorithm (hence our cloaking strategy). One trivial solution is randomly picking anyone among the alternatives. A potential danger, however, may be if the user stays too long in the same place and issues service requests constantly, the attacker can conclude that the user is indeed at the intersection of randomly reported cloaked regions. In such case, it suffices to keep sending the initially randomly selected region all the time.

*1) Generating Cloaking Map:* The pseudo-code of overlapping and disjoint cloaking algorithms are given in Algorithm 1 and Algorithm 2, respectively. Starting from the sensitive seed node, the algorithms do a breadth-first search (BFS) to extend the subgraph for the respective cloaked region. BFS is preferred because it tends to output compact (small diameter) subgraphs. After the BFS traversal is completed, we include all the original edges between the vertices in the




(a) Overlapping cloaking

(b) Disjoint cloaking

Fig. 4. Overlapping cloaking and Disjoint cloaking.

resulting BFS tree. This is particularly important to preserve the shortest paths among the vertices of the subgraph. We call the output of the algorithms as the cloaking map, consisting of a number of cloaked regions per profile (not per person, as many individuals may be assigned to the same profile). Note that both of the algorithms enforce the privacy requirements for a particular profile. Also note that, the cloaking map is produced without any reference to the velocity attack, which is handled during the transformation stage.

**Algorithm 1** Overlapping cloaking

**Input:** Annotated city network $G = (V, E, pop, pt, tt)$, privacy profile $PP = \{(pt_i, \tau_i)\}_{i \in [1,n]}$
**Output:** Cloaked region map

1: $map \leftarrow \emptyset$
2: **for all** $u \in V$ s.t. $u.pt \in PT_S$ **do**
3: $cr \leftarrow \emptyset$
4: $totalPop \leftarrow u.pop$
5: **while** $(true)$ **do**
6: $v \leftarrow$ next move from BFS($u$)
7: **if** $v.pt \in PT_{NS}$ **then**
8: $cr.addEdge(edge(parent(v), v))$
9: $totalPop \leftarrow totalPop + v.pop$
10: **if** $\frac{u.pop}{pop_{cr}(\cdot)} \leq \tau_i$ where $u.pt = pt_i$ **then**
11: **break**
12: $map \leftarrow map \bigcup \{cr\}$

Consider Figure 4, where rounded rectangles show the nonsensitive places while the bigger solid circles show the sensitive places, one from each of Hospital, Night Club and Temple place types, also shown is the roads and road crossings. The figure illustrates the progress of the overlapping and disjoint cloaking algorithms. In the overlapping cloaking, a separate cloaking is started from each of the sensitive place to result in three cloaked regions, $cr1$, $cr2$ and $cr3$ (Figure 4(a)). Note that $cr1$ and $cr2$ overlap and if the user is located in the intersection, either $cr1$ or $cr2$ can be reported as the cloaked region. In the disjoint cloaking, similarly a separate cloaking is started from each of sensitive place. However, whenever an overlap is detected the regions are merged into one to result in disjoint cloaked regions of $cr1$ and $cr2$ (Figure 4(b)).

**Algorithm 2** Disjoint cloaking

**Input:** Annotated city network $G = (V, E, pop, pt, tt)$, privacy profile $PP = \{(pt_i, \tau_i)\}_{i \in [1,n]}$
**Output:** Cloaked region map

1: $map \leftarrow \emptyset$
2: **for all** $u \in V$ s.t. $u.pt \in PT_S$ **do**
3: $cr \leftarrow \emptyset$
4: $necesNonSenPop \leftarrow 0$
5: **while** $(true)$ **do**
6: $v \leftarrow$ next move from BFS($u$)
7: **if** $v.pt \in PT_{NS}$ **then**
8: $cr.addEdge(edge(parent(v), v))$
9: $cv \leftarrow cloakingRegionOf(v)$
10: $necesNonSenPop$ += $v.pop\frac{(1-\tau_i)}{\tau_i}$ where $v.pt = pt_i$
11: **if** $cv \neq \perp$ **then**
12: $cr.merge(cv)$
13: **for all** $w \in cv.V$ s.t. $w.pt \in PT_S$ **do**
14: $necesNonSenPop$ += $w.pop\frac{(1-\tau_i)}{\tau_i}$ where $w.pt = pt_i$
15: **if** $pop_{cr}(PT_{NS}) \geq necesNonSenPop$ **then**
16: **break**
17: $map \leftarrow map \bigcup \{cr\}$

*2) Transformation:* When the user requests to use LBS with his last position, then velocity attack should be voided. Algorithm 3 gives the pseudo-code for the transformation needed to guard against the velocity attack. First the algorithm checks whether it is possible to convey the request under the velocity attack; if so, the current position is said to be safe

with respect to the previous position and is conveyed (with or without cloaking depending on the location of the user w.r.t. the cloaking map). If the current position opens up a privacy breach, computed in line 5 according to Equation 5, then two alternatives (time delay and postdating) are evaluated and time delay is preferred in case the best time delay is less than a predefined maximum delay parameter. In case it is impossible, postdating remains the only possibility and we do a regression along the path to the previously reported location and report the first position not causing a privacy breach. In the algorithm, for the sake of simplicity, we treated user positions not inside cloaked regions as a degenerate cloaked region consisting of a single point.

In case the postdating introduces too much spatial error, then it might be preferable to drop the service request rather than reporting an obsolete location. For a fixed postdating threshold the success rate (a quality metric) can be measured.

**Algorithm 3** Transformation

**Input:** Annotated city network $G = (V, E, pop, pt, tt)$, cloaking map $map$, request timestamp $t_q$, location $loc$ of user $U$

**Output:** Cloaked region/point and issuance time

1: Let $A$ to be last issued cr/point with issuance time $t_A$
2: $CRsU \leftarrow \{cr \in map : loc \in cr\}$
3: **if** $CRsU = \emptyset$ **then**
4: $\quad CRsU \leftarrow loc \quad \triangleright$ *a single point cr*
5: $\overline{CRsU} \leftarrow \{cr \in CRsU : cr\ is\ safe\ w.r.t.\ A\}$
6: **if** $\overline{CRsU} \neq \emptyset$ **then**
7: $\quad$ **return** a random $cr \in \overline{CRsU}$ and $t_q$
8: $mindelay \leftarrow min_{cr \in CRsU}\{delay\ needed\ for\ cr\}$
9: **if** $mindelay \leq MAX\_DELAY$ **then**
10: $\quad \triangleright$ *time delay*
11: $\quad cr_{min} \leftarrow argmin_{cr \in CRsU}\{delay\ needed\ for\ cr\}$
12: $\quad$ **return** $cr_{min}$ and $t_q + mindelay$
13: **else**
14: $\quad \triangleright$ *postdate*
15: $\quad cr_f \leftarrow$ first safe $cr$ (w.r.t. $A$) along regressing $path(loc, A)$
16: $\quad$ **return** $cr_f$ and $t_q$

### B. Online Cloaking

We consider online cloaking more appropriate for the cases where the popularity of places are not static and change significantly depending on the time of day, day of week and so on. Just consider for instance that night clubs are more frequented in nights and almost vacant in day time. So, any offline cloaking is a potential privacy breach for such situations.

Algorithm 4 presents our online cloaking method. First of all the algorithm differs from the offline cloaking method by combining the cloaking map generation and transformation stages into a single stage. The function $Subgraph(G, A, t_q - t_A)$ returns the subgraph $G'$ of the annotated city network $G$. The subgraph contains $A$ and its reachable vertices/edges within time $t_q - t_A$. The function $CloakingRegions(G', PP)$ returns the cloaked regions (local $map$) for the subgraph $G'$. The function can invoke either of Algorithm 1 or 2 with $G'$. After we find the local $map'$, the $Transformation$ algorithm (Algorithm 3) is invoked to give the cloaked region/point and issuance time to be reported to the LBS. Note that online cloaking combines the two stages of offline cloaking on the local subgraph $G'$.

**Algorithm 4** Online cloaking

**Input:** Annotated city network $G = (V, E, pop, pt, tt)$, privacy profile $PP = \{(pt_i, \tau_i)\}_{i \in [1,n]}$, request timestamp $t_q$, location $loc$ of the user $U$

**Output:** cr/point and issuance time

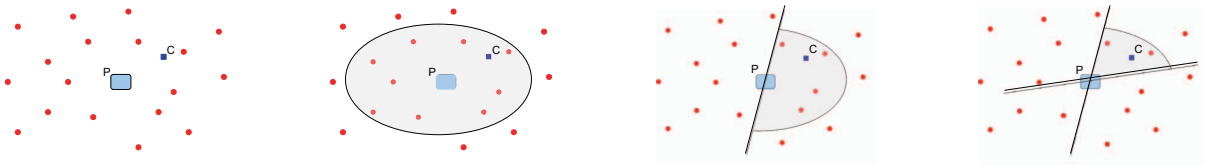
1: Let $A$ to be the last issued cr/point with issuance time $t_A$
2: $G' \leftarrow Subgraph(G, A, t_q - t_A)$
3: $map' \leftarrow CloakingRegions(G', PP)$
4: **return** $Transformation(G', map', t_q, loc)$

*1) Performance improvement:* Although the online cloaking algorithm is conceptually simple, it may become not practical due to the online performance requirements. The main bottleneck is the size of the subgraph for the annotated city network and number of cloaked regions within it, since both of them have to be computed online, i.e. following the service request. To guard against velocity attack, one can be tempted to pick a cloaked region among a few cloaked regions closest to the actual user location. However, this approach comes with a privacy breach regardless of the selection process, whether deterministic or random. Note that any deterministic strategy is not private due to the background information of the attacker. Unfortunately, neither the random selection strategy is privacy preserving in most non-trivial cases [6].

Our technique uses the last cloaked region reported to LBS, rather than the actual location/cloaked region. The approach is as follows. First, a cut (passing through the previous reported location/cloaked region) is selected randomly, hence it partitions the region into two, the one containing the subgraph located in clockwise and the another located in counter clockwise directions of the cut. After this step we only maintain the partition containing the actual location and discard the other. Clearly, this step reduces the size of the subgraph $G'$ to almost half. We repeat the process until we get a manageable number of cloaked regions (denoted with $K$) left in $G'$, which is used for transformation. The partitioning procedure is integrated in the function $CloakingRegions(G', PP)$ (line 3) of Algorithm 4. The procedure ensures that the resulting $map'$ contains at most $K$ cloaked regions. The partitioning is illustrated in Figure 5.

## V. Experimental Evaluation

To assess the utility of the proposal we experimented with the OpenStreetMap's Milano Street Map Dataset, and evaluated our offline and online cloaking algorithms.




(a) Previous and the current location (b) Cloaking around the previous location (c) Partitioning the cloaked region (d) Second iteration for partitioning

Fig. 5. Cutting the local subgraph of annotated city network for partitioning.

### A. Dataset

We picked Milano downtown area (from OpenStreetMap[3]) and processed the raw data to obtain the annotated city network according to our definition. In detail, the raw dataset consist of points, lines, polygons, and in some cases an attached semantics (like place types). To get our annotated city network, lines (representing roads), points (representing points of interests like pharmacies), and polygons (representing buildings of places like large hospitals) served as the components. Most of the annotations are added by the users of the application by tagging. We also apply a data cleaning stage to fix wrongly spotted places, for instance a car park with no connection to roads. Table I summarizes the resulting database.

We use the following hypothetical uniform popularity for places of type: $worship \propto 0.09$, $healthcare \propto 0.30$, $education \propto 0.60$, $socialactivities \propto 0.06$, $entertainment \propto 0.15$, $shopping \propto 0.02$ and $others \propto 0.01$. The raw dataset contains travel time information for all the edges. Since our algorithms repeatedly need shortest paths, we pre-compute all the shortest paths using Floyd-Warshall algorithm.

Given the annotated city network, we generate 1000 trajectories each having 100 points to simulate the LBS user requests. Each trajectory on average runs approximately 7 hours.

### B. Experiments

In our preliminary evaluation of two methods for the online cloaking, disjoint method is found to be too slow in comparison to the overlapping method. Since the bottleneck with the online cloaking is efficiency, in the sequel we only provide results for overlapping method for online cloaking.

We are particularly interested in two effectiveness metrics, (i) average cloaked region size measured as mean diameter, and (ii) average total penalty which adds temporal error (time delay) and spatial error (postdate distance) to get a single measure (recall that both are temporal measures). Runtime is the sole metric for efficiency. Figure 6 shows a sample for our cloaking maps.

Figure 7 gives the performance results at various disclosure threshold levels. The reported results belong to two

[3]http://www.openstreetmap.org/

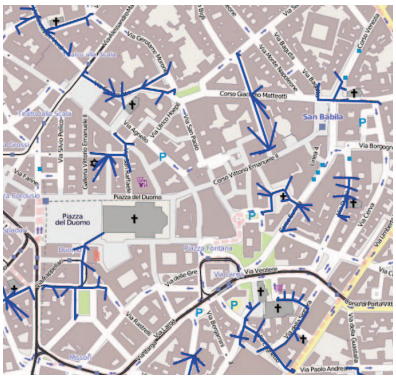



Fig. 6. A sample output from our cloaking methods.

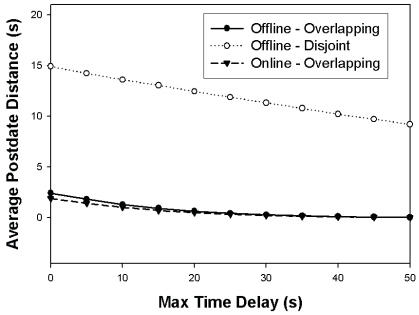



Fig. 8. Time delay versus postdating.

distinct personalized privacy profiles, $PP_1 = \{(worship, \tau = x)\}$ and $PP_2 = \{(worship, \tau = x), (healthcare, \tau = x), (entertainment, \tau = x)\}$ where the value for $x$ is the respective value at x-axis. From the results it is clear that the online method runs quite slow in comparison to offline methods as expected. This is the price paid to accommodate the dynamic nature of popularity. Overlapping method for offline cloaking performs favorably to disjoint method, both in effectiveness and efficiency. As shown in Figure 8 there is a tradeoff between time delay and postdating.

Figures 9(e) and 9(f) present the effect of $K$ value on the runtime and request drop counts for online cloaking. With partitioning heuristic, service drops are not due to the lack of solutions rather than to obsolete location reporting. No solution

TABLE I
MILANO OPENSTREETMAP DATASET

| Feature | Value |
|---|---|
| region | Milano downtown with spatial extension of 3km by 3km |
| vertex set | 8263 places of interest plus road intersections |
| # of edges | 34000 (bidirectional) |
| # of places | 3800 |
| place types (counts) | education (22), healthcare (includes hospitals and pharmacy) (27), worship (64), social activities (20), entertainment (29), shopping (40) |

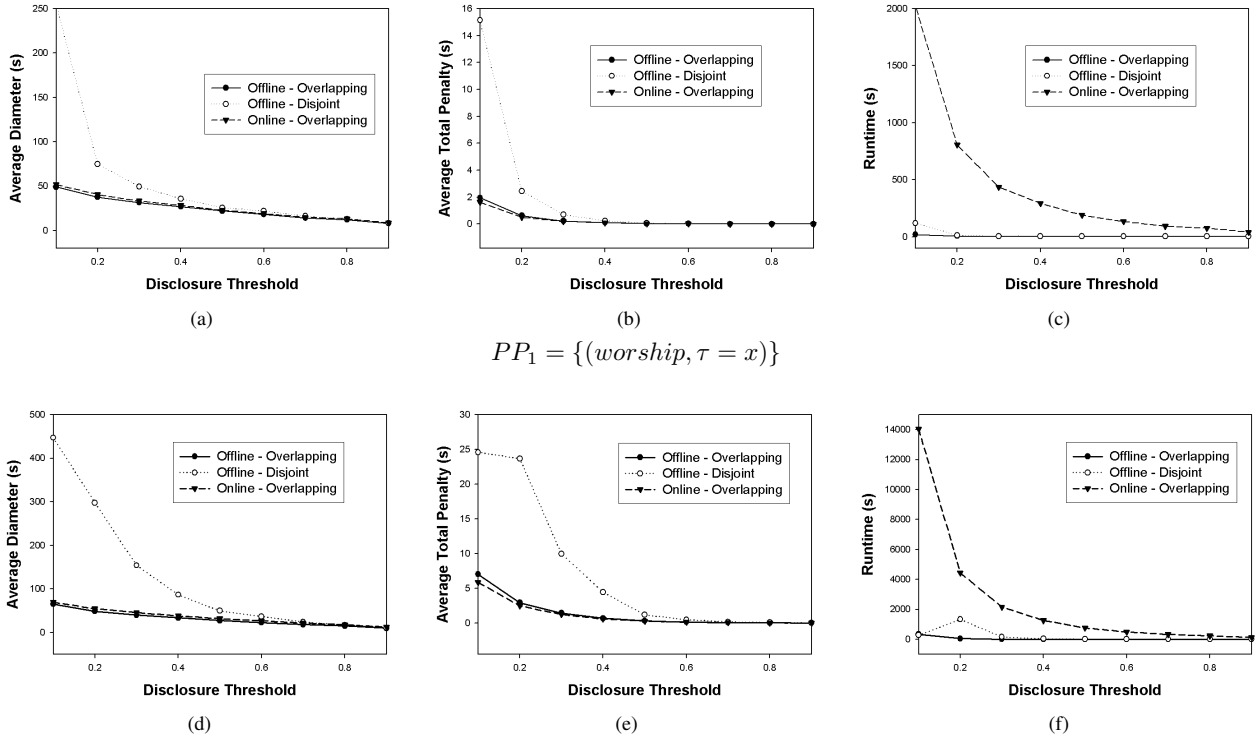


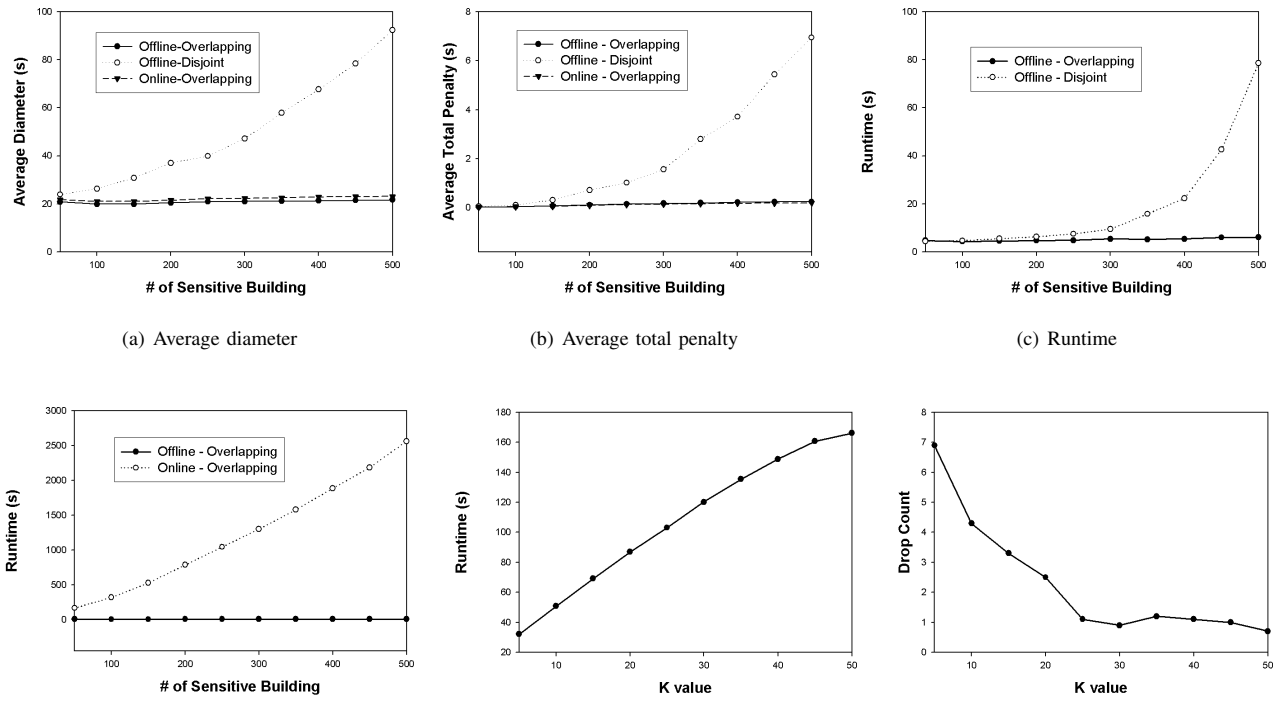

Fig. 7. Performance results at various disclosure thresholds.

case is possible since some cloaked regions possibly in the original solution are ruled out during partitioning. The request drop counts are average counts per trajectory (e.g. over $1000 * 100$ service requests). The results from Figure 9(e) clearly shows that the runtime decreases with decreasing $K$ values, and hence the utility of the partitioning heuristic. On the other hand, we see from Figure 9(f) that when $K$ is smaller the number of service request drop rates increase as it becomes hard to meet the time delay and postdate requirements with small number of cloaked regions.

Scalability is an important issue when the number of private places are too high. Note that this is quite realistic as we are working in urban settings. For scalability tests, we generated 10 random private place type each with 50 places, and added each of them to the next privacy profile one by one. So, there are 50, 100 and 500 private places in the first, second and tenth tests, respectively. The effectiveness and efficiency performances are provided in Figure 9. In the results, we measure the performance metrics w.r.t. the number of sensitive places (x-axis). In all of the tests, the disclosure threshold is set to 0.1 for all the place types. Online cloaking again runs too slow and its performance degrades with increasing number of sensitive places. In all the tests, offline-overlapping method exhibits a nice scalability profile.

## VI. CONCLUSION

In this paper, we have presented an approach to the privacy-preserving sharing of sensitive positions in urban settings. The reference space is represented by an annotated graph while different techniques for the computation of cloaked regions on this graph have been evaluated. The notion of strongly cloaked region generalizes previous results while novel cloaking methods for the creation of overlapping subgraphs proved to be effective in terms of performance and quality of position information. This method can run on client




(a) Average diameter (b) Average total penalty (c) Runtime

(d) Runtime (e) Runtime (f) Service request drop counts

Fig. 9. Scalability (a),(b),(c),(d) and effect of partitioning on online cloaking (e),(f).

devices, e.g. smartphone, provided that the client can store and efficiently access the annotated city network. Moreover it does not require dedicated costly infrastructure (i.e. anonymizer), and that paves the way to the cost-effective deployment of this solution. Although this work has been developed in the context of LBSs, we imagine that the approach could be extended to the protection of trajectory data in data publishing. In this case, the challenge is to integrate methods for the anonymization of trajectories with solutions for the safe cloaking of sensitive places.

## REFERENCES


[1] M. L. Damiani, C. Silvestri, and E. Bertino, "Fine-Grained Cloaking of Sensitive Positions in Location-Sharing Applications. IEEE Pervasive Computing," *IEEE Pervasive Computing*, vol. 10(4), pp. 64–72, 2011.

[2] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from GPS trajectories," in *Proc. World Wide Web (WWW 2009)*, 2009.

[3] A. R. Beresford and F. Stajano, "Location Privacy in Pervasive Computing," *IEEE Pervasive Computing*, vol. 2, no. 1, pp. 46–55, 2003.

[4] X. Pan, X. Meng, and J. Xu, "Distortion-based anonymity for continuous queries in location-based mobile services," in *Proc. 17th ACM GIS*, 2009.

[5] M. L. Damiani, E. Bertino, and C. Silvestri, "The PROBE Framework for the Personalized Cloaking of Private Locations," *Transactions on Data Privacy*, vol. (3)2, pp. 123–148, 2010.

[6] G. Ghinita, M. L. Damiani, C. Silvestri, and E. Bertino, "Preventing velocity-based linkage attacks in location-aware applications," in *Proc. 17th ACM GIS*, 2009.

[7] H. Hu and D.-L. Lee, "Semantic location modeling for location navigation in mobile environment," in *Proc. MDM*, 2004.

[8] F. Schmid, K.-F. Richter, and P. Laube, "Semantic trajectory compression," in *Proc. 11th SSTD*, 2009.

[9] S. Spaccapietra, C. Parent, M. L. Damiani, J. de Macedo, F. Porto, and C. Vangenot, "A conceptual view on trajectories," *Data Knowl. Eng.*, vol. 65, pp. 126–146, April 2008.

[10] C.-Y. Chow and M. F. Mokbel, "Trajectory privacy in location-based services and data publication," *SIGKDD Explorations*, vol. 13, no. 1, pp. 19–29, 2011.

[11] C. Y. Chow and M. F. Mokbel, "Enabling private continuous queries for revealed user locations," in *SSTD*, 2007.

[12] B. Palanisamy and L. Liu, "Mobimix: Protecting location privacy with mix-zones over road networks," in *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering*, 2011.

[13] O. Abul, F. Bonchi, and M. Nanni, "Never walk alone: Uncertainty for anonymity in moving objects databases." in *Proc. of 24th International Conference on Data Engineering (ICDE)*, 2008.

[14] M. E. Nergiz, M. Atzori, Y. Saygin, and B. Güç, "Towards trajectory anonymization: a generalization-based approach," *Transactions on Data Privacy*, vol. 2, no. 1, pp. 47–75, 2009.

[15] C. S. Jensen, H. Lu, and M. Yiu, "Location Privacy Techniques in Client-Server Architectures," in *Privacy in Location-Based Applications: Research Issues and Emerging Trends*. Springer-Verlag, 2009.

[16] L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression," *Int. Journal on Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 10, pp. 571–588, 2002.

[17] B. Bamba, L. Liu, P. Pesti, and T. Wang, "Supporting Anonymous Location Queries in Mobile Environments with PrivacyGrid," in *Proc. of 17th International World Wide Web Conference (WWW)*, 2008.

[18] T. Wang and L. Liu., "Privacy-aware mobile services over road networks." in *Proc. of VLDB*, 2009.

[19] M. Xue, P. Kalnis, and H. Pung, "Location Diversity: Enhanced Privacy Protection in Location Based Services," in *Proc. of the 4th International Symposium on Location and Context Awareness (LoCA)*, 2009.

[20] A. Monreale, R. Trasarti, D. Pedreschi, C. Renso, and V. Bogorny, "C-safety: a framework for the anonymization of semantic trajectories," *Transactions on Data Privacy*, vol. 4, no. 2, pp. 73–101, 2011.