Location Privacy in Mobile Systems: A Personalized Anonymization Model

Buğra Gedik
College of Computing, GaTech
bgedik@cc.gatech.edu

Ling Liu
College of Computing, GaTech
lingliu@cc.gatech.edu

Abstract

This paper describes a personalized k-anonymity model for protecting location privacy against various privacy threats through location information sharing. Our model has two unique features. First, we provide a unified privacy personalization framework to support *location k-anonymity* for a wide range of users with context-sensitive personalized privacy requirements. This framework enables each mobile node to specify the *minimum level of anonymity* it desires as well as the *maximum temporal and spatial resolutions* it is willing to tolerate when requesting for k-anonymity preserving location-based services (LBSs). Second, we devise an efficient *message perturbation engine* which runs by the location protection broker on a trusted server and performs location anonymization on mobile users' LBS request messages, such as identity removal and spatio-temporal cloaking of location information. We develop a suite of scalable and yet efficient spatio-temporal cloaking algorithms, called $CliqueCloak$ algorithms, to provide high quality personalized location k-anonymity, aiming at avoiding or reducing known location privacy threats before forwarding requests to LBS provider(s). The effectiveness of our $CliqueCloak$ algorithms is studied under various conditions using realistic location data synthetically generated using real road maps and traffic volume data.

1 Introduction

Advances in sensing and tracking technologies create new opportunities for location-based applications but they also create significant privacy risks. According to the report by Computer Science and Telecommunications Board on *IT Roadmap to a Geospatial Future* [2], location based services (LBSs) are expected to form an important part of the future computing environments that will seamlessly and ubiquitously integrate into our life (examples include NextBus [8], CyberGuide [1], or FCC's Phase II E911 Rules). Although with LBSs mobile users can obtain wide variety of location-based information services, and businesses can extend their competitive edges in mobile commerce and ubiquitous service provisions, extensive deployment of location based services may open doors for adversaries to endanger location privacy of mobile users and to expose LBSs to significant vulnerabilities for abuse [16]. Location privacy threats describe the risk that an adversary learns the locations that a subject visited, as well as times during which these visits took place. Through these locations, the adversary can receive clues about private information such as political affiliations, alternative lifestyles, or medical problems. The two classes of most popular privacy threats to LBSs are communication privacy threats, exemplified by the passive-logging based attacks, and location privacy threats, represented by space or time correlated inference attacks. Even when a subject does not disclose her identity at a private location, an adversary may still gain this information through location tracking or space and time correlation inference. In case that a subject is identified at any point, her complete movements can also be exposed.

One way to reduce location privacy risks is to promote k-anonymity preserving management of location information and develop efficient and scalable system-level facilities for protecting location privacy with location k-anonymity. Anonymity can be seen as "a state of being not identifiable within a set of subjects, the anonymity set" [9]. The concept of k-anonymity is originally introduced in the context of relational data privacy research [11]. It addresses the question of "How can a data holder release a version of its private data with scientific guarantees that the individuals who are the subjects of the data cannot be re-identified while the data remain practically useful" [14].

In the context of LBSs and mobile users, location k-anonymity refers to k-anonymous usage of location information. A subject is considered k-anonymous with respect to location information if and only if the location information sent from a mobile user to a LBS is indistinguishable from the location information of at least $k-1$ other subjects (e.g. different mobile nodes) [6]. Location perturbation is an effective technique in dealing with location privacy breaches exemplified by the above cases and is effective for supporting location k-anonymity. If the location information sent by each mobile node is perturbed by replacing the position of the mobile node with a coarser grained spatial range, such that there are several other mobile nodes within that range, say k of them, then the adversary will have uncertainty in matching the mobile node to a location-identity association she has obtained through external observation or knowledge. This uncertainty will increase with the increasing value of k, providing better privacy.

In this paper, we describe a personalized k-anonymity model for protecting location privacy against various privacy threats through location information sharing. There is a close synergy between location privacy and k-anonymity. Larger k in location anonymity usually implies higher guarantees for location privacy. Therefore, to ensure that a subject is k anonymous, one can perturb the location information by replacing it with a relatively large spatial region (range) or by delaying the message long enough. However, this has two downsides. First, low spatial resolution in location perturbation may lead the LBS provider to provide more coarse grained location-dependent information to the mobile user, which may deteriorate the quality of service; or it may result in sending more than required information back to the mobile user, which is going to be filtered out by the mobile node, resulting in communication and processing overhead. Second, the extra delay introduced through temporal cloaking of location information may decrease the perceived service quality of the mobile user.

The development of our location privacy model exhibits two distinct features. First, it enables each mobile node to specify the *minimum level of anonymity* it desires as well as the *maximum temporal and spatial resolutions* it is willing to tolerate when requesting for k-anonymity preserving location-based services. Concretely, instead of imposing a uniformed k for all mobile users, we provide efficient algorithms and system level facilities to support personalized k at per-user level. Each user can specify a different k-anonymity level, and can change this specification at per-message granularity. Furthermore, each user can specify her preferred *spatial and temporal tolerance* values that should to be respected while maintaining the desired level of location k-anonymity. We call such tolerance specification (service quality) and preference of k value (location privacy), the *anonymization constraint* of the message. By providing a unified framework to support location k-anonymity with variable anonymization constraints, we allow a wide range of users to benefit from the location privacy protection with personalized privacy and quality requirements.

Second, we devise an efficient *message perturbation engine* which runs by the location protection broker on a trusted server and performs location anonymization on mobile users' LBS request messages, such as identity removal and spatio-temporal cloaking of location information. We develop a suite of scalable and yet efficient spatio-temporal cloaking algorithms, called $CliqueCloak$ algorithms, taking into account the tradeoffs between location privacy and quality of service. Our location perturbation engine can continuously process a stream of messages for location k-anonymity, and can work with different $CliqueCloak$ algorithms to perturb the location information contained in the messages sent from mobile users by performing spatio-temporal cloaking. The resulting three dimensional box, called the *spatio-temporal cloaking box*, indicates the acceptable decrease of the spatial resolution of location information and the tolerable delay of the message in an effort to meet the specified anonymity level. Our experiments show that the proposed personalized location k-anonymity model, through the use of our perturbation engine and its $CliqueCloak$ algorithms, can achieve high guarantee of k-anonymity and high resilience to location privacy threats without introducing significant performance penalty.

2 Personalized Location k-anonymity

We assume that the LBS system consists of mobile nodes, a wireless network, anonymity servers, and LBS servers. Location information is typically determined by a location information source, such as GPS receiver in a vehicle. We assume that location information includes temporal information (when the subject was present at the location) in addition to spatial information. Mobile nodes communicate with third party LBS providers through one or a collection of anonymity servers located at trusted computing bases. The mobile nodes establish communication with an anonymity server through an authenticated and encrypted connection. Each message destined to an LBS provider contains location information of the mobile node, a timestamp, in addition to service specific information. Upon receiving a message from a mobile node, the anonymity server decrypts the message and removes any identifiers, such as IP addresses, and perturbs the location information through spatio-temporal cloaking, and then forwards the anonymized message to the LBS provider.

In order to capture varying location privacy requirements and ensure different levels of service quality, each mobile node specifies its *anonymity level* (k value), *spatial tolerance*, and *temporal tolerance*. The main task of a location anonymity server is to transform each message received from mobile nodes into a new message that can be safely (k-anonymously) forwarded to the LBS provider. The key idea underlying the location k-anonymity model is two-fold. First, a given degree of location anonymity can be maintained, regardless of population density, by decreasing the location accuracy through enlarging the exposed spatial area, such that there are other $k-1$ mobile nodes present in the same spatial area. This approach is called spatial cloaking. Second, one can achieve the location anonymity by delaying the message until k mobile nodes have visited the same area located by the message sender. This approach is called temporal cloaking.

We denote the set of messages received from the mobile nodes as S. We formally define a messages m_s in the set S as follows: $\langle u_{id}, r_{no}, \{t, x, y\}, k, \{d_t, d_x, d_y\}, C\rangle$. Messages are uniquely identifiable by the sender's identifier, message reference number pairs, (u_{id}, r_{no}), within the set S. Messages from the same mobile node have same sender identifiers but different reference numbers. In a received message, $x, y,$ and t together form the three dimensional *spatio-temporal location point* of the message, denoted as $L(m_s)$. The coordinate (x, y) refers to the spatial position of the mobile node in the two dimensional space (i.e., x-axis and y-axis), and the timestamp t refers to the time point at which the mobile node was present at that position (temporal dimension: t-axis of the message). The k value of the message specifies the desired minimum *anonymity level*. A value of $k = 1$ means that anonymity is not required for the message. A value of $k > 1$ means that the perturbed message will be assigned a spatio-temporal cloaking box that is indistinguishable from at least $k - 1$ other perturbed messages, each from a different mobile node. Thus, larger k values imply higher degree of privacy. One way to determine the appropriate k value is to assess the certainty with which an adversary can associate the message with an external location/identity binding. This certainty is given by, $1/k$. This means that P% privacy requires to set the k value to be $(1 - P/100)^{-1}$. The d_t value of the message represents the temporal tolerance specified by the user. It means that, the perturbed message should have a spatio-temporal cloaking box whose projection on the temporal dimension does not contain any point more than d_t distance away from t. Similarly, d_x and d_y specify the tolerances with respect to the spatial dimensions. The values of these three parameters are dependent on the requirements of the external LBS and users' preferences with regard to quality of service. For instance, larger spatial tolerances may result in less accurate answers to location-dependent service requests and larger temporal tolerances may result in higher latencies of the messages. Let $\Phi(v, d) = [v - d, v + d]$ be a function that extends a numerical value v to a range by amount d. Then, we denote the *spatio-temporal constraint box* of a message m_s as $B_{cn}(m_s)$ and define it as $(\Phi(m_s.x, m_s.d_x)$, $\Phi(m_s.y, m_s.d_y)$, $\Phi(m_s.t, m_s.d_t))$. The field C in m_s denotes the message content.

We denote the set of perturbed (*anonymized*) messages as T. We formally define a messages m_t in the set T as follows: $\langle u_{id}, r_{no}, \{X : [x_s, x_e], Y : [y_s, y_e], I : [t_s, t_e]\}, C\rangle$. For each message m_s in S, there exists at most one corresponding message m_t in T. We call the message m_t, the *perturbed format* of message m_s, denoted as $m_t = R(m_s)$. The function R defines a one-to-one and onto mapping from S to T. Concretely, if $m_t = R(m_s)$, then $m_t.u_{id} = m_s.u_{id}$ and $m_t.r_{no} = m_s.r_{no}$. If $R(m_s) = \emptyset$, then the message m_s is not anonymized. The (u_{id}, r_{no}) fields of a message in T should be replaced with a dummy identifier (e.g., with $h(u_{id}||r_{no})$, where h is a secure hash function) before the message can be safely forwarded to the LBS provider. In a perturbed message, $X : [x_s, x_e]$ denotes the extent of the spatio-temporal cloaking box of the message on the x-axis, with x_s and x_e denoting the two end points of the interval. The definitions of $Y : [y_s, y_e]$ and $I : [t_s, t_e]$ are similar with y-axis and t-axis replacing the x-axis, respectively. We denote the *spatio-temporal cloaking box* of a perturbed message as $B_{cl}(m_t)$ and define it as $(m_t.X : [x_s, x_e], m_t.Y : [y_s, y_e], m_t.I : [t_s, t_e])$. The field C in m_t denotes the message content. We now describe how the fields of a perturbed message in set T relates to its counterpart in set S.

There are three basic properties that must hold between a raw message m_s in S and its perturbed format m_t in T. These are: (*i*) *Spatio-temporal Containment*, which states that the cloaking box $B_{cl}(m_t)$ of the perturbed message should contain the spatio-temporal point $L(m_s)$ of the original message m_s. (*ii*) *Spatio-temporal Resolution*, which states that for each of the three dimensions, the extent of the spatio-temporal cloaking box of the perturbed message on that dimension should be contained within the interval defined by the maximum tolerance value specified in the original message. This is equivalent to stating that the cloaking box $B_{cl}(m_t)$ of the perturbed message, should be contained within the constraint box $B_{cn}(m_s)$ of the original message m_s. (*iii*) *Content Preservation*, which ensures that the message content remains as it is, i.e. $m_s.C = m_t.C$.

We formally capture the essence of the location k-anonymity by the following requirement, which states that, for a message m_s in S and its perturbed format m_t in T, the following condition must hold:

- *Location k-anonymity*:
$\exists T' \subset T$, s.t. $m_t \in T', |T'| \geq m_s.k$,
$\forall_{\{m_{t_i}, m_{t_j}\} \subset T'},\ m_{t_i}.u_{id} \neq m_{t_j}.u_{id}$ and
$\forall_{m_{t_i} \in T'}, B_{cl}(m_{t_i}) = B_{cl}(m_t)$

The k-anonymity requirement demands that, for each perturbed message $m_t = R(m_s)$, there exist at least $m_s.k - 1$ other perturbed messages with the same spatio-temporal cloaking box, each from a different mobile node. A key challenge for the cloaking algorithms employed by the message perturbation engine is to find a set of messages within a minimal spatio-temporal cloaking box that satisfies the above conditions.

3 Message Perturbation Engine

The message perturbation engine processes each incoming message m_s from mobile nodes in four steps. The first step, called *zoom-in*, involves locating a subset of all messages currently pending in the engine. This subset contains messages that are potentially useful for anonymizing the newly received message m_s. The second step, called *detection*, is responsible for finding the particular group of messages within the set of messages located in the zoom-in step, such that this group of messages can be anonymized together with the newly received message m_s. If such a group of messages is found, then the perturbation is performed over these messages in the third step, called *perturbation*, and the perturbed messages are forwarded to the LBS provider. The last step, called *expiration*, checks for pending messages whose *deadlines* has passed, and thus are dropped. The deadline of a message is the high point along the temporal dimension of its spatio-temporal constraint box.

3.1 Message Anonymization

A main technical challenge for developing an efficient cloaking algorithm is to find the smallest spatio-temporal cloaking box, for each message $m_s \in S$, within its specified spatial and temporal tolerances, such that there exist at least $m_s.k - 1$ other messages, each from a different mobile node, with the same minimal cloaking box. Let us consider this problem in two steps (in reverse order): (1) given a set M of messages that can be anonymized together, how to find the minimal cloaking box in which all messages in M reside; and (2) for a message $m_s \in S$, how to find the set M containing m_s and the group of messages that can be anonymized together with m_s. A set $M \subset S$ of messages are said to be anonymized together if they are assigned the same cloaking box and all the requirements defined in Section 2 are satisfied for all messages in M.

Consider a set $M \subset S$ of messages that can be anonymized together. The best strategy to find a minimal cloaking box for all messages in M is to use the minimum bounding rectangle (MBR [1]) of the spatio-temporal points of the messages in M as the minimal cloaking box. This definition of minimal cloaking box also ensures that the cloaking box is contained in the constraint boxes of all other messages in M. We denote *the minimum spatio-temporal cloaking box* of a set $M \subset S$ of messages that can be anonymized together as $B_m(M)$, and define it to be equal to the MBR of the points in the set $\{L(m_s)|m_s \in M\}$.

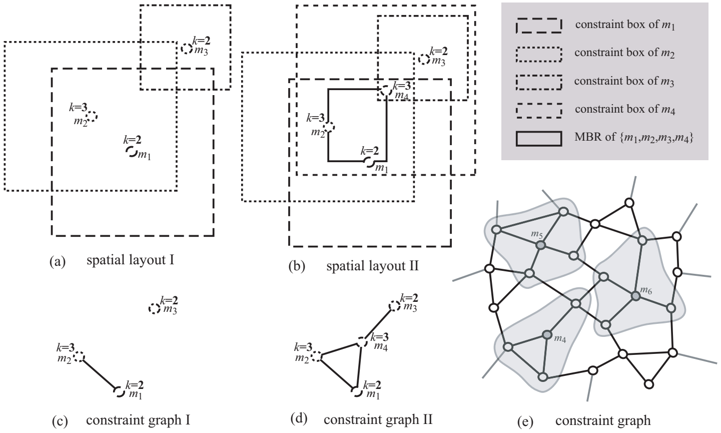
Now let us consider the second step: given a message $m_s \in S$, how to find the set M containing m_s and the group of messages that can be anonymized together with m_s. Based on the above analysis and observations, one way to tackle this problem is to model the anonymization constraints of all messages in S as a constraint graph defined below and translate the problem into the problem of finding cliques that satisfy certain conditions in the constraint graph: Let $G(S, E)$ be an undirected graph where S is the set of vertices, each representing a message received at the message perturbation engine, and E is the set of edges. There exists an edge $e = (m_{s_i}, m_{s_j}) \in E$ between two vertices m_{s_i} and m_{s_j}, if and only if the following conditions hold: (*i*) $L(m_{s_i}) \in B_{cn}(m_{s_j})$, (*ii*) $L(m_{s_j}) \in B_{cn}(m_{s_i})$, (*iii*) $m_{s_i}.u_{id} \neq m_{s_j}.u_{id}$. We call this graph the *constraint graph*. The conditions (*i*), (*ii*), and (*iii*) together state that, two messages are connected in the constraint graph if and only if they originate from different mobile nodes and their spatio-temporal points are contained in each other's constraint boxes defined by their tolerance values.

Given the definition of constraint graph, the following property holds: Let $M = \{m_{s_1}, m_{s_2}, \ldots, m_{s_l}\}$ be a set of messages in S. For each message m_{s_i} in M, we define $m_{t_i} = \langle m_{s_i}.u_{id}, m_{s_i}.r_{no}, B_m(M), m_{s_i}.C\rangle$. Then m_{t_i}, $1 \leq i \leq l$, is a valid perturbed format of m_{s_i} *if and only if* the set M of messages form an l-clique in the constraint graph $G(S, E)$ with the additional condition that for any message m_{s_i} in S, we have $m_{s_i}.k \leq l$ (i.e. m_{s_i}'s user specified k value is not larger than the cardinality of the set M). See our technical report [4] for a formal theorem (refered to as $ClickCloak$ Theorem) governing this property.

We demonstrate the application of this property with an example. Figure 1 shows four messages, m_1, m_2, m_3, and m_4. We assume that each message is from a different mobile node. We omitted the time domain in this example for ease of explanation, but the extension to spatio-temporal space is straightforward. Initially, first three of these messages are inside the system. *Spatial layout I* shows how these three messages spatially relate to each other. It also depicts the spatial constraint boxes of the messages. *Constraint graph I* shows how these messages are connected to each other in the constraint graph. Since the spatial locations of messages m_1 and m_2 are mutually contained in each others spatial constraint box, they are connected in the constraint graph and m_3 lies apart by itself. Although m_1 and m_2 form a 2-clique, they can not be anonymized and removed from the graph. This is because $m_2.k = 3$ and as a result the clique does not satisfy the Clique-Cloak theorem. *Spatial layout II* shows the situation after m_4 arrives and *constraint graph II* shows the corresponding status of the constraint graph. With the inclusion of m_4, there exists only one clique whose size is at least equal to the maximum k value of the messages it contains. This clique is $\{m_1, m_2, m_4\}$. We can compute the MBR of the messages within the clique and use it as the spatio-temporal cloaking box of the perturbed messages and then safely remove this clique. Figure 1(b) clearly shows that the MBR is contained by the spatial constraint boxes of all messages within the clique.

Although in the described example we have found a single clique immediately after m_4 was received, we could have had cliques of different sizes to choose from. For instance, if $m_4.k$ was 2, then $\{m_3, m_4\}$ would have also formed a valid clique according to the Clique-Cloak theorem. We address the questions of *what* kind of cliques to search and *when* to search for such cliques, in more detail in Section 4.

There are three key points that makes the application of *Clique-Cloak* theorem effective in practice: (i) Successful anonymization of a message m_{s_c} results in anonymization of *at least* $m_{s_c}.k - 1$ other messages. (ii) The search performed on the constraint graph for the purpose of anonymizing a message m_{s_c} only deals with a small subgraph that consists of m_{s_c} and its neighbors (illustrated in Figure 1 for m_4, m_5, and m_6), and the cost of this step does not depend on



(a) spatial layout I

(b) spatial layout II

(c) constraint graph I

(d) constraint graph II

(e) constraint graph

Figure 1: Illustration of the Clique-Cloak Algorithm

[1] MBR of a set of points is the smallest rectangular region enclosing all the points

the scale of the complete constraint graph. (iii) For messages whose subgraphs on which the search is performed do not share any messages (exemplified by m_4, m_5, and m_6 in Figure 1 (e)), the anonymization process can be efficiently parallelized. When overlaps exist, simple locking strategies can be employed to achieve effective parallelization (See [4]).

3.2 Data Structures

We briefly describe the four main data structures that are used in the message perturbation engine.

Message Queue, Q_m: Message queue is a simple FIFO queue, which collects the messages sent from the mobile nodes in the order they are received. The messages are popped from this queue by the message perturbation engine in order to be processed.

Multi-dimensional Index, I_m: The multi-dimensional index is used to allow efficient search on the spatio-temporal points of the messages. For each message, say m_s, in the set of messages that are not yet anonymized and are not yet dropped according to expiration condition (specified by the temporal tolerance), I_m contains a three dimensional point $L(m_s)$ as a key, together with the message m_s as data. The index is implemented using an in-memory R*-tree in our system.

Constraint Graph, G_m: The constraint graph is a dynamic in-memory graph, which contains the messages that are not yet anonymized and not yet dropped due to expiration. The structure of the constraint graph is already defined in Section 3.1. The multi-dimensional index I_m is mainly used to speedup the maintenance of the constraint graph G_m, which is updated when new messages arrive or when messages get anonymized or expired.

Expiration Heap, H_m: Expiration heap is a mean-heap, sorted based on the deadline of the messages. For each message, say m_s, in the set of messages that are not yet anonymized and are not yet dropped due to expiration, H_m contains a deadline $m_s.t + m_s.d_t$ as the key, together with the message m_s as the data. Expiration heap is used to detect expired messages (i.e. messages that cannot be successfully anonymized), so that they can be dropped and removed from the system.

3.3 Perturbation Engine Algorithms

Upon arrival of a new message, the message engine will update the message queue (FIFO) to include this message. The message perturbation process works by continuously popping messages from the message queue and processing them for k-anonymity in four steps. The pseudo code of the message perturbation engine is given in Algorithm 1.

Zoom-in $-$ In this step we update the data structures with the new message from the message queue, and integrate the new message into the constraint graph, i.e., search the constraint graph containing all the messages pending for perturbation and locate the messages that should be assigned as neighbors to it in the graph (zoom-in). Concretely, when a message m_{s_c} is popped from the message queue, it is inserted into the index I_m using $L(m_{s_c})$, inserted into the heap H_m using $m_{s_c}.t + m_{s_c}.d_t$, and inserted into the graph G_m as a node. Then the edges incident upon vertex m_{s_c} are constructed in the constraint graph G_m by searching the multi-dimensional index I_m using the spatio-temporal constraint box of the message, i.e. $B_{cn}(m_{s_c})$, as the range search condition. The messages whose spatio-temporal points are contained in $B_{cn}(m_{s_c})$ are candidates for being m_{s_c}'s neighbors in the constraint graph. These messages (denoted as N in the pseudo code) are filtered based on whether their spatio-temporal constraint boxes contain $L(m_{s_c})$. The ones that pass the filtering step and are different from m_{s_c} become neighbors of m_{s_c}. We call the subgraph that contains m_{s_c} and its neighbors the focused subgraph, denoted by $G^{'}_m$. See lines 3-10 in the pseudo code.

Algorithm 1: Message Perturbation Engine

```
MsgPertEngine()
(1)  while true
(2)      if $Q_m \neq \emptyset$
(3)          $m_{s_c} \leftarrow$ Pop the first item in $Q_m$
(4)          Add $m_{s_c}$ into $I_m$ with $L(m_{s_c})$
(5)          Add $m_{s_c}$ into $H_m$ with $(m_{s_c}.t + m_{s_c}.d_t)$
(6)          Add the message $m_{s_c}$ into $G_m$ as a node
(7)          $N \leftarrow$ Range search $I_m$ using $B_{cn}(m_{s_c})$
(8)          foreach $m_s \in N, m_s \neq m_{s_c}$
(9)              if $L(m_{s_c}) \in B_{cn}(m_s)$
(10)                 Add the edge $(m_{s_c}, m_s)$ into $G_m$
(11)         $G^{'}_m \leftarrow$ Subgraph of $G_m$ consisting of messages in $N$
(12)         $M \leftarrow$ local-$k$_Search($m_{s_c}.k, m_{s_c}, G^{'}_m$)
(13)         if $M \neq \emptyset$
(14)             foreach $m_s$ in $M$
(15)                 Output perturbed message $m_t \leftarrow$
(16)                     $\langle h(m_s.u_{id}||m_s.r_{no}), B_m(M), m_s.C\rangle$
(17)                 Remove the message $m_s$ from $G_m$
(18)                 Remove the message $m_s$ from $I_m$
(19)                 Pop the topmost element in $H_m$
(20)     while true
(21)         $m_s \leftarrow$ Topmost item in $H_m$
(22)         if $m_s.t + m_s.d_t < now$
(23)             Remove the message $m_s$ from $G_m$
(24)             Remove the message $m_s$ from $I_m$
(25)             Pop the topmost element in $H_m$
(26)         else
(27)             break
```

Algorithm 2: local-k Search Algorithm

```
local-$k$_Search($k, m_{s_c}, G^{'}_m$)
U ← $\{m_s | m_s \in nbr(m_{s_c}, G^{'}_m)$ and $m_s.k \leq k\}$
if $|U| < k - 1$
    return $\emptyset$
$l \leftarrow 0$
while $l \neq |U|$
    $l \leftarrow |U|$
    foreach $m_s \in U$
        if $(|nbr(m_s, G^{'}_m) \cap U| < k - 2)$
            $U \leftarrow U \backslash \{m_s\}$
Find any subset $M \subset U$, s.t. $|M| = k - 1$ and $M \cup \{m_{s_c}\}$ forms a clique
return $M$
```

Detection $-$ In this step we apply the *local-k search* $CliqueCloak$ algorithm (detection) in order to find a suitable clique in the focused subgraph $G^{'}_m$ of G_m, which contains m_{s_c} and its neighbors in G_m, denoted by $nbr(m_{s_c}, G_m)$. In local-k search, we try to find a clique of size $m_{s_c}.k$ that includes the message m_{s_c} and satisfies the *Clique-Cloak* theorem. The pseudo code of this step is given separately in Algorithm 2 as the function local-k_Search. Note that the local-

k_Search function is called within Algorithm 1 (line 12) with parameter k set to $m_{s_c}.k$. Before beginning the search, a set $U \subset nbr(m_{s_c}, G_m^{'})$ is constructed such that for each message $m_s \in U$, we have $m_s.k \leq k$ (lines 1-3). This means that the neighbors of m_{s_c} whose anonymity values are higher than k are simply discarded from U, as they cannot be anonymized with a clique of size k. Then the set U is iteratively filtered until there is no change (lines 4-9). At each filtering step, each message $m_s \in U$ is checked whether it has at least $k-2$ neighbors in U. If not, the message cannot be part of a clique that contains m_{s_c} and has size k, thus the message is removed from U. After the set U is filtered, the possible cliques in $U \cup \{m_{s_c}\}$ that contain m_{s_c} and have size k are enumerated and if one satisfying the k-anonymity requirements is found, the messages in that clique are returned. Up to values of $k = 10$, (where $k = 5$ is considered as a good level of anonymity [6]) the search step does not form a bottleneck. In fact, the subgraph on which we perform the clique search is localized with respect to m_{s_c} and it is very small compared to the complete constraint graph.

Perturbation $-$ In this step we generate the k-anonymized messages to be forwarded to the external LBS providers. If a suitable clique is found in the detection step, then the messages in the clique (denoted as M in the pseudo code) are anonymized by assigning $B_m(M)$ (i.e. the MBR of the spatio-temporal points of the messages in the clique), as their cloaking box (perturbation). Sender's identifier, message reference number pairs are also replaced with their secure hash value before the actual forwarding takes place. Then these messages are removed from the graph G_m, as well as from the index I_m and the heap H_m. This step is detailed in the pseudo code through lines 13-19. In case a clique cannot be found, the message stays inside I_m, G_m, and H_m. *It may be later picked up and anonymized during the processing of a new message* or may be dropped when it expires. We discuss some more advanced ways of searching cliques in Section 4.

Expiration $-$ In this step we take care of the expired messages. After the processing of each message, we check the expiration heap for any messages that has expired. The message on top of the expiration heap is checked and if its deadline has passed, it is removed from I_m, G_m, and H_m. Such a message cannot be anonymized and is dropped. This step is repeated until a message whose deadline is ahead of the current time is reached. Lines 20-27 of the pseudo code deals with message expiration.

4 Discussions on Possible Optimizations

In this section, first we discuss an improved $CliqueCloak$ algorithm, called *nbr-k search*, which utilizes a different criterion in determining *what* kinds of cliques are searched. Then we discuss a variation of the $CliqueCloak$ algorithms discussed so far, that uses a deferred policy with regard to *when* cliques are searched. We also provide a brief discussion on improving the message processing time of $CliqueCloak$ algorithms.

When searching for a clique in the focused subgraph, it is essential to ensure that the newly received message, say m_{s_c}, should be included in the clique. If there is a new clique formed due to the entrance of m_{s_c} into the graph, it must contain m_{s_c}. However, instead of searching a clique with size $m_{s_c}.k$, we can try to find out the biggest clique that includes $m_{s_c}.k$, of course making sure that all messages inside the clique has a k value at most equal to the size of the clique. There are two strong motivations behind the approach. First, by anonymizing a larger number of messages at once, it can provide higher success rate (larger number of messages can be successfully anonymized) which also results in better performance, as the graph will become less crowded. Second, by anonymizing messages that have smaller k's together with messages that have larger k's, it can provide higher relative level of anonymity (meaning that the user perceived anonymity levels of the messages are higher than the user specified anonymity levels). *Nbr-k* search takes this approach. It first collects the set of k values the new message m_{s_c} and its neighbors $nbr(m_{s_c}, G_m^{'})$ have, denoted as L. The k values in L are considered in decreasing order until a clique is found or k becomes smaller than $m_{s_c}.k$ (in which case the search returns empty set). For each $k \in L$ considered, local-k_Search function is called with appropriate parameters. If a non-empty set is returned from the call, the search halts and the messages within the set are returned.

So far we have only considered searching for cliques when each new message arrives. This may result in many unsuccessful searches, thus deteriorate the performance in terms of average time to process a message. Instead of immediately searching for a clique for each message, we can defer this processing. If a deferred message is not already anonymized (together with other messages) at the time of its expiration, we can search for a clique in order to anonymize it before it expires. However, this latter approach will definitely increase user perceived latency. To overcome this, we can only perform the clique search phase for a new message m_{s_c}, if the number of neighbors it has at its arrival is larger than or equal to $\alpha * m_{s_c}.k$. Here, $\alpha \geq 1$ is a system parameter that adjusts the amount of messages for which the clique search is deferred. Smaller values pushes the algorithm toward immediate processing. It can be set statically at compile time based on experimental studies or adaptively during runtime by observing the rate of successful clique searches with different α values. We name this variation of the algorithm as $Deferred\ CliqueCloak$ and the original algorithm as $Immediate\ CliqueCloak$.

There are other dimensions to $CliqueCloak$ algorithms that can improve the running time performance of anonymization (our technical report [4] provides extended study of these dimensions). Here we discuss one such idea that may significantly improve the message processing time for extreme cases where constraint boxes are large. A *progressive search* techniques may be applied, such that when a message is to be processed for anonymization, we use a progressively increasing set of neighbor nodes as the candidate set. If a smaller set is not sufficient to anonymize the message, we can add messages whose spatio-temporal points are further away, after each progressive step. This helps in decreasing the message processing time when the constraint boxes are large, since such large boxes result in large candidate sets, although most of the time anonymization can easily be performed with a much smaller set.

5 Evaluation Metrics

In this subsection we list several evaluation metrics of interest, that can be used to evaluate the effectiveness and the efficiency of the message perturbation engine.

To evaluate the effectiveness of the proposed location k-anonymity model, an important measure is the *success rate*. Concretely, the primary goal of the cloaking algorithm is to maximize the number of messages perturbed successfully in accordance with their anonymization constraints. Success rate can be defined over a set $S' \subset S$ of messages as the percentage of messages that are successfully anonymized (perturbed), i.e. $\frac{|\{m_t | m_t = R(m_s), m_t \in T, m_s \in S'\}|}{100^{-1} * |S'|}$.

Important measures of efficiency include *relative anonymity level*, *relative temporal resolution*, *relative spatial resolution*, and *message processing time*. The first three are measures related with quality of service, whereas the last one is a performance measure.

Relative anonymity level is a measure of the level of anonymity provided by the cloaking algorithm, normalized by the level of anonymity required by the messages. We define relative anonymity level over a set $T' \subset T$ of perturbed messages by $\frac{1}{|T'|} \sum_{m_t = R(m_s) \in T'} \frac{|\{m | m \in T \wedge B_{cl}(m_t) = B_{cl}(m)\}|}{m_s.k}$. Note that relative anonymity level cannot go below 1.

Relative spatial resolution is a measure of the spatial resolution provided by the cloaking algorithm, normalized by the minimum acceptable spatial resolution defined by the spatial tolerances. We define relative spatial resolution over a set of perturbed messages $T' \subset T$ by $\frac{1}{|T'|} \sum_{m_t = R(m_s) \in T'} \sqrt{\frac{2 * m_s.d_x * 2 * m_s.d_y}{||m_t.X|| * ||m_t.Y||}}$, where $||l||$, when applied to an interval l, gives its length. Higher relative spatial resolution values imply more effective cloaking achieved with a smaller spatial cloaking region.

Relative temporal resolution is a measure of the temporal resolution provided by the cloaking algorithm, normalized by the minimum acceptable temporal resolution defined by the temporal tolerances. We define relative temporal resolution over a set of perturbed messages $T' \subset T$ by $\frac{1}{|T'|} \sum_{m_t = R(m_s) \in T'} \frac{2 * m_s.d_t}{||m_t.I||}$. Higher relative temporal resolution values imply more effective cloaking achieved by a smaller temporal cloaking interval and thus with smaller delay due to perturbation. Relative spatial and temporal resolutions can not go below 1.

Message processing time is a measure of the running time performance of the message perturbation engine. The message processing time may become a critical issue, if the computational power at hand is not enough to handle the incoming messages at a high rate. In Section 6, we use the average CPU time needed to process 10^3 messages as the message processing time.

Parameter	Default value
anonymity level range	$\{5, 4, 3, 2\}$
anonymity level zipf param	0.6
mean spatial tolerance	$100m$
variance in spatial tolerance	$40m^2$
mean temporal tolerance	$30s$
variance in temporal tolerance	$12s^2$
mean inter-wait time	$15s$
variance in inter-wait time	$6s^2$

Table 1: Message generation parameters

mean of car speeds for each road type	$\{90, 60, 50\} km/h$
std.dev. in car speeds for each road type	$\{20, 15, 10\} km/h$
traffic volume data	$\{2916.6, 916.6, 250\}$per hour

Table 2: Car movement parameters

6 Experiments

We break up the experimental evaluation into two components: the effectiveness of the personalized k-anonymity model in terms of location privacy quality, and the performance of the location perturbation engine and algorithms in terms of scalability. Due to the space restriction, in this section, we mostly present the first set of experiments that demonstrates the effectiveness of our perturbation engine in terms of location privacy guarantees under different settings with regard to various metrics introduced in Section 5. A smaller set of experiments is included to give a summary characterization of message processing time. We divided the experiments into two, namely *success rate* and *spatial/temporal resolution*. Before presenting our experimental results, we first describe the trace generator used to generate realistic traces that are employed in the experiments and the details of our experimental setup.

We have developed a trace generator, that simulates cars moving on roads and generates requests using the position information from the simulation. The trace generator loads real-world road data, available from National Mapping Division of the United States Geological Survey (USGS) [7] in SDTS [13] format. We use transportation layer of 1:24K Digital Line Graphs (DLGs) as road data. We convert the graphs into Scalable Vector Graphic [12] format using the Global Mapper [5] software and use them as input to our trace generator. We extract three types of roads from the trace graph, class 1 (expressway), class 2 (arterial), and class 3 (collector). The generator uses real traffic volume data to calculate the total number of cars on each road type, as described by [6]. Once the number of cars on each type of road is determined, they are randomly placed into the graph and the simulation begins. Cars move on the roads and take other roads when they reach joints. The simulator tries to keep the fraction of cars on each type of road constant as time progresses. The cars change their speeds at each joint based on a normal distribution whose parameters are also input to the trace generator.

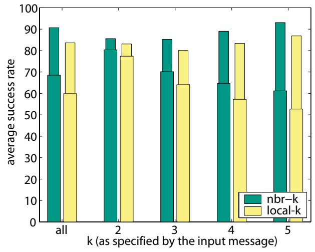


Figure 2: Success rates for different k values

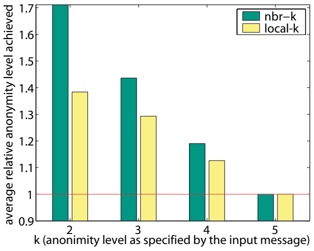


Figure 3: Relative anonymity levels for different k values

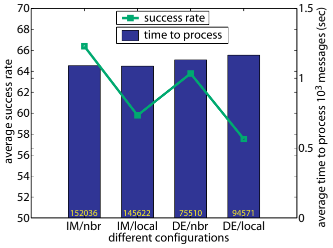


Figure 4: Message processing time and success rate of different approaches

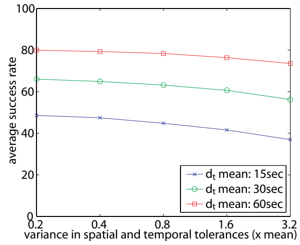


Figure 5: Success rate as a function of variances in spatial and temporal tolerances

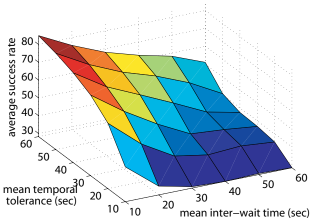


Figure 6: Success rate with respect to temporal tolerance with different inter-wait times

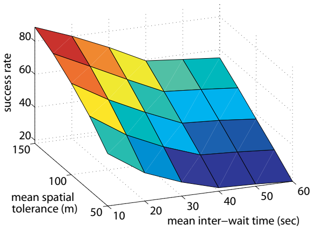


Figure 7: Success rate with respect to spatial tolerance with different inter-wait times

We used a map from Chamblee region of state of Georgia in USA to generate the trace used in this paper. The map covers a region of $\approx 160km^2$. The mean speeds and standard deviations for each road type are given in Table 2. The traffic volume data is taken from [6] and is also listed in Table 1. These settings result in approximately 10,000 cars. The trace has a duration of one hour.

Each car generates several messages during the simulation. Each message specifies an anonymity level (k value) from the list $\{5, 4, 3, 2\}$ using a zipf parameter of 0.6, $k = 5$ being the most popular. The spatial and temporal tolerance values of the messages are selected independently using normal distributions whose default parameters are given in Table 1. Whenever a message is generated, the originator of the message waits until the message is anonymized or dropped, after which it waits for a normally distributed amount of time, called the *inter-wait time*, whose default parameters are also listed in Table 1. All parameters take their default values, if not stated otherwise. We change many of these parameters to observe the behavior of the algorithms in different settings.

For spatial points of the messages, the default settings result in anonymizing around 70% of messages with an accuracy of $< 18m$ in 75% of the cases, which we consider to be very good when compared to the E-911 requirement of $125m$ accuracy in 67% of the cases. For temporal point of the messages, the default parameters also result in a delay of $< 10s$ in 75% of the cases and $< 5s$ in 50% of the cases.

6.1 Success Rate

Figure 2 shows the success rate for nbr-k and local-k approaches. The success rate is shown (on y-axis) for different groups of messages, each group representing messages with a certain k value (on x-axis). The two leftmost bars show the success rate for all of the messages. The wider bars show the actual success rate provided by the ClickCloak algorithm. The thinner bars represent a lower bound on the percentage of messages that cannot be anonymized no matter what algorithm is used. This lower bound is calculated as follows. For a message m_s, if the set $U = \{m_{s_i} | m_{s_i} \in S \wedge P(m_{s_i}) \in B_{cn}(m_s)\}$ has size less than $m_s.k$, the message cannot be anonymized. This is because, the total number of messages that ever appear inside m_s's constraint box are less than $m_s.k$. However, if the set U has size of at least $m_s.k$, the message m_s may still not be anonymized under a hypothetical optimal algorithm. This is because, the optimal choice may require to anonymize a subset of U that does not include m_s, together with some other messages not in U. As a result, the remaining messages in U may not be sufficient to anonymize m_s. It is not possible to design an on-line algorithm that is optimal in terms of success rate, due to the fact that such an algorithm will require future knowledge of messages, which is not known beforehand. If a trace of the messages is available, as in this work, the optimal success rate can be computed offline. However, we are not aware of a time and space efficient off-line algorithm for computing the optimal success rate. As a result, we use a lower bound on the number of messages that cannot be anonyimized.

There are three observations from Figure 2. First, the nbr-k approach provides around 15% better average success rate than local-k. Second, the best average success rate achieved is around 70. Out of the 30% dropped messages, at least 65% of them cannot be anonymized, meaning that in the worst case remaining 10% of all messages are dropped due to non-optimality of the algorithm with respect to success rate. If we knew a way to construct the optimal algorithm (with a reasonable time and space complexity) given full knowledge of the trace, we could have got a better bound. Last, messages with larger k values are harder to anonymize. The success rate for messages with $k = 2$ is around 30% higher than the success rate for messages with $k = 5$.

Figure 3 shows the relative anonymity level for nbr-k and local-k approaches. The relative anonymity level is shown (on y-axis) for different groups of messages, each group representing messages with a certain k value (on x-axis). Nbr-k shows a relative anonymity level of 1.7 for messages with $k = 2$, meaning that on the average these messages are anonymized with $k = 3.4$ by the algorithm. Local-k shows a lower relative anonymity level of 1.4 for messages with $k = 2$. This gap between the two approaches vanishes for messages with $k = 5$, since both of the algorithms do not

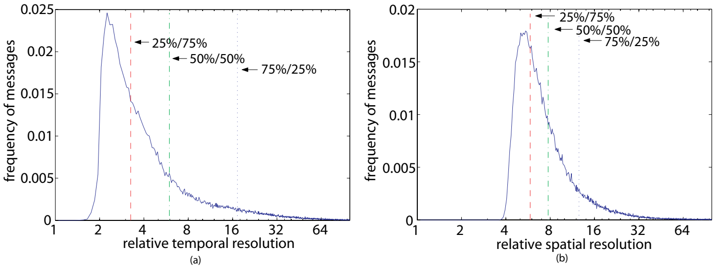


Figure 8: Relative temporal and spatial resolution distributions

attempt to search cliques of sizes larger than the maximum of the k values specified by the messages. The gap in relative anonymity level between nbr-k and local-k shows that the former approach is able to anonymize messages with smaller k values together with the ones with higher k values. This is particularly good for messages with higher k values, as they are harder to anonymize. This also explains why nbr-k results in better success rate.

Figure 4 plots the average success rate (y-axis on the left side) and the message processing time (y-axis on the right side) for nbr-k and local-k search approaches with immediate or deferred processing mode. For deferred processing mode α is taken as 1.4 (as it gave the highest success rate). Other than the immediate approach providing better success rate than the deferred approach, the surprising observation from the figure is that, deferred approach does not provide improvement in terms of message processing time. Figure 4 also shows (above the x-axis) the number of times clique search is performed for different approaches. Although the deferred approach results in slightly higher message processing time, it decreases the number of times the clique search is performed around 50% (for nbr-k). Here is the reason that the deferred approach still performs worse in terms of total processing time: For $k \leq 10$ the index update dominates the cost of processing the message and the deferred approach results in a more crowded index. However, the deferred approach is promising in terms of message processing time, for cases where k values are really large (thus clique search dominates the cost). Another potential enhancement is to design a more efficient multi dimensional index to replace the in-memory R* tree.

Figure 5 plots the success rate for different mean temporal tolerances and different variances in temporal and spatial tolerances. It shows that the algorithm is much less sensitive to the changes in the variances of the spatial and temporal tolerances than the mean temporal tolerance. For instance, when the mean temporal tolerance is $60s$, changing the variance in both spatial and temporal tolerances from 0.2 times their means to 1.6 times their means only decreases the success rate from 80 to 75; whereas decreasing the mean temporal tolerance from $60s$ to $15s$ decreases the success rate by approximately 40% of its success rate (for instance from 80 to 50 when variances are equal to 0.2 times their means).

Figure 6 plots the average success rate as a function of mean inter-wait time and mean temporal tolerance. Similarly, Figure 7 plots the average success rate as a function of mean inter-wait time and mean spatial tolerance. For both of the figures, the variances are always set to 0.4 times the means. We observe that, the smaller the inter-wait time, the higher the success rate. For smaller values of the temporal and spatial tolerances, the decrease in inter-wait time becomes more important, in terms of keeping the success rate high. When the inter-wait time is high, we have a lower rate of messages coming into the system. Thus, it becomes harder to anonymize messages, as the constraint graph becomes sparser. Both spatial and temporal tolerances has tremendous effect on the success rate. Although high success rates (around 85) are achieved with high temporal and spatial tolerances, as we will show in the next section, the relative temporal and spatial resolutions are much larger than 1 in such cases, meaning that the system assigns much smaller spatio-temporal cloaking boxes to the messages compared to the constraint boxes.

6.2 Spatial/Temporal Resolution

In Section 6.1, we have showed that one way to improve success rate is to increase the spatial and temporal tolerance values specified by the messages. In this section, we show that our $CliqueCloak$ algorithms have the nice property that, for most of the anonymized messages, the cloaking box generated by the algorithm is much smaller than the constraint box of the received message (specified by the tolerance values), resulting in higher relative spatial and temporal resolutions.

Figure 8(a) plots the frequency distribution (y-axis) of the relative temporal resolutions (x-axis) of the anonymized messages. Figure 8 shows that in 75% of the cases the provided relative temporal resolution is > 3.25, thus an average temporal accuracy of roughly $< 10s$ (recalling that the default mean temporal tolerance was $30s$). For 50% of the cases it is > 5.95 and for 25% of the cases it is > 17.25. This points out that, the observed performance with regard to temporal resolutions is much better than the worst case specified by the temporal tolerances.

Figure 8(b) plots the frequency distribution (y-axis) of the relative spatial resolutions (x-axis) of the anonymized messages. Figure 8 shows that in 75% of the cases the provided relative spatial resolution is > 5.85, thus an average spatial accuracy of roughly $< 18m$ (recalling that the default mean spatial tolerance was $100m$). In 50% of the cases it is > 7.75 and for 25% of the cases it is > 12.55. This points out that, the observed performance with regard to spatial resolutions is much better than the worst case specified by the spatial tolerances.

7 Related Work

Previous work on location privacy has mostly focused on a policy-based approach [3, 10], specialized in telematics or telecommunication domain. Users may use the policies to specify the privacy preferences. These policies serve as a location information sharing agreement on which data can be collected and shared, when and for what purpose the data can be used, and how and to whom it can be distributed. Mobile users have to trust the LBSs that private location information is adequately protected. Another approach to location privacy is location k-anonymity based approach, which depersonalizes data through perturbation techniques before forwarding it to the LBS. Location k-anonymity is first studied in [6]. Its location perturbation is performed by the quadtree-based algorithm executing spatial and temporal cloaking. However, this work suffers from several drawbacks. First, it assumes a system-wide static k value for all mobile users, which hinders the service quality for those mobile nodes whose privacy requirements can be satisfied using smaller k values. Furthermore, this assumption is unrealistic in practice as mobile users tend to have varying privacy protection requirements under different contexts and on different subjects. Second, their approach fails to provide any quality of service guarantees with respect to the sizes of the cloaking boxes produced. This is because, the quadtree-based algorithm anonymizes the messages by dividing the quadtree cells until the number of messages in each cell falls below k and by returning the previous quadrant for each cell as the spatial cloaking box of the messages under that cell. In comparison, our unified framework for location k-anonymity captures the desired degree of privacy and quality on per-user base, supporting mobile users with diverse context-dependent location privacy requirements. Our message perturbation engine can anonymize a stream of incoming messages through the use of efficient $CliqueCloak$ algorithms, where each message can specify an independent k value, as well as customized spatial and temporal tolerance values to restrict the size of the cloaking box produced as a result of perturbation.

Samarati and Sweeney have developed a k-anonymity model [14] for protecting data privacy and a set of generalization and suppression techniques [15] for safeguarding the anonymity of individuals whose information is recorded in database tables. Our work, although in a different context, can be viewed as location perturbation of the messages sent by mobile nodes communicating with LBS providers through a trusted anonymity server.

8 Conclusion

We have proposed a personalized k-anonymity model for providing location privacy. Our model allows each mobile user to define, and modify this definition at the granularity of single messages, a minimum anonymity level requirement, as well as upper bounds on the inaccuracy to be introduced by the cloaking algorithm in temporal and spatial dimensions. We have developed a novel message perturbation engine to implement this model, that can effectively anonymize messages sent by the mobile nodes, in accordance with location k-anonymity, while satisfying the privacy and quality requirements of the users. Several spatio-temporal cloaking algorithms, called $CliqueCloak$ algorithms, are proposed to work as a part of the perturbation engine. We experimentally studied the behavior of our $CliqueCloak$ algorithms under various conditions, using realistic location data synthetically generated using real road maps and traffic volume data.

Acknowledgement: This work is partially supported by the National Science Foundation under a CNS Grant, an ITR grant, a Research Infrastructure grant, and a DoE SciDAC grant, an IBM SUR grant, an IBM faculty award, and an HP equipment grant.

References

[1] G. Abowd, C. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton. Cyberguide: A mobile context-aware tour guide. *ACM Wireless Networks*, 3, 1997.

[2] Computer Science and Telecommunications Board. *IT Roadmap to a Geospatial Future*. The National Academics Press, November 2003.

[3] S. Duri, M. Gruteser, X. Liu, P. Moskowitz, R. Perez, M. Singh, and J.-M. Tang. Framework for security and privacy in automotive telematics. In *International Workshop on Mobile Commerce*, 2002.

[4] B. Gedik and L. Liu. A customizable k-anonymity model for protecting location privacy. Technical Report GIT-CERCS-04-15, Georgia Institute of Technology, April 2004.

[5] Global mapper. http://www.globalmapper.com/, November 2003.

[6] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *ACM/USENIX MobiSys*, 2003.

[7] National mapping division of the united states geological survey. USGS http://www.usgs.gov/, November 2003.

[8] Nextbus. http://www.nextbus.com/, January 2004.

[9] A. Pfitzmann and M. Koehntopp. Anonymity, unobservability, and pseudonymity - a proposal for terminology. In *Workshop on Design Issues in Anonymity and Unobservability*, 2000.

[10] A. Sahuguet, R. Hull, D. F. Lieuwen, and M. Xiong. Enter once, share everywhere: User profile management in converged networks. In *Conference on Innovative Data Systems Research*, 2003.

[11] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical Report SRI-CSL-98-04, SRI International, 1998.

[12] Scalable vector graphics format. http://www.w3.org/Graphics/SVG/, November 2003.

[13] Spatial data transfer format. http://mcmcweb.er.usgs.gov/sdts/, November 2003.

[14] L. Sweeney. K-anonymity: A model for protecting privacy. *IJUFKS*, 10(5), 2002.

[15] L. Sweeney. k-anonymity privacy protection using generalization and suppression. *IJUFKS*, 10(5), 2002.

[16] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM TOIS*, 10, 1992.