Protecting Location Privacy with Personalized k-Anonymity: Architecture and Algorithms

Buğra Gedik
Thomas J. Watson Research Center
IBM Research

Ling Liu
CERCS, College of Computing
Georgia Tech

Abstract

Continued advances in mobile networks and positioning technologies have created a strong market push for location-based applications. Examples include location-aware emergency response, location-based advertisement, and location-based entertainment. An important challenge in wide deployment of location-based services (LBSs) is the privacy-aware management of location information, providing safeguards for location privacy of mobile clients against vulnerabilities for abuse. This paper describes a scalable architecture for protecting location privacy from various privacy threats resulting from uncontrolled usage of LBSs. This architecture includes the development of a personalized location anonymization model and a suite of location perturbation algorithms. A unique characteristic of our location privacy architecture is the use of a flexible privacy personalization framework to support *location k-anonymity* for a wide range of mobile clients with context-sensitive privacy requirements. This framework enables each mobile client to specify the *minimum level of anonymity* it desires and the *maximum temporal and spatial tolerances* it is willing to accept when requesting for k-anonymity preserving LBSs. We devise an efficient *message perturbation engine* to implement the proposed location privacy framework. The prototype we develop is designed to be run by the *anonymity server* on a trusted platform and performs location anonymization on LBS request messages of mobile clients, such as identity removal and spatio-temporal cloaking of location information. We study the effectiveness of our location cloaking algorithms under various conditions using realistic location data that is synthetically generated from real road maps and traffic volume data. Our experiments show that the personalized location k-anonymity model together with our location perturbation engine can achieve high resilience to location privacy threats without introducing any significant performance penalty.

Keywords: k-anonymity, Location Privacy, Location-based Applications, Mobile Computing Systems

1 Introduction

In his famous novel *1984* [1], George Orwell envisioned a world in which everyone is being watched, practically at all times and places. Although, as of now, the state of affairs has not come to such a totalitarian control, projects like DARPA's recently dropped LifeLog [2], which has stimulated serious privacy concerns, attest that continuously tracking where individuals go and what they do is not only in the range of today's technological advances but also raises major personal privacy issues regardless of many beneficial applications it can provide.

According to the report by the Computer Science and Telecommunications Board on *IT Roadmap to a Geospatial Future* [3], 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. Such services are already being developed and deployed in commercial and research worlds. For instance, the NextBus [4] service provides location-based transportation data, the CyberGuide [5] project investigates context-aware location-based electronic guide assistants, and the FCC's Phase II E911 requires wireless carriers to provide precise location information within 125 meters in most cases for emergency purposes [6].

1.1 Location Privacy Risks

Advances in global positioning and wireless communication technologies create new opportunities for location-based mobile applications, but they also create significant privacy risks. Although with LBSs mobile clients can obtain a wide variety of location-based information services, and businesses can extend their competitive edges in mobile commerce and ubiquitous service provisions, extensive deployment of LBSs can open doors for adversaries to endanger the location privacy of mobile clients and to expose LBSs to significant vulnerabilities for abuse [7].

A major privacy threat specific to LBS usage is the location privacy breaches represented by space or time correlated inference attacks. Such breaches take place when a party that is not trusted gets access to information which reveals the locations visited by the individual as well as the times during which these visits took place. An adversary can utilize such location information to infer details about the private life of an individual, such as political affiliations, alternative lifestyles, or medical problems of an individual [8], or the private businesses of an organization, such as new business initiatives and partnerships.

Consider a mobile client which receives real-time traffic and roadside information service from an LBS provider. If a user submits her service request messages with raw position information, the privacy of the user can be compromised in several ways, assuming that the LBS providers are not trusted but semi-honest. For instance, if the LBS provider has access to information that associates location with identity such as

person $\mathcal{A}$ lives in location $\mathcal{L}$, and if it observes that all request messages within location $\mathcal{L}$ are from a single user, then it can infer that the identity of the user requesting the roadside information service is $\mathcal{A}$. Once the identity of the user is revealed, further tracking of future positions can be performed by using a simple connect-the-dots approach. In literature, this type of attack is called *Restricted Space Identification* [8]. Another possible attack is to reveal identity by relating some external observation on location-identity binding to a message. For instance, if person $\mathcal{A}$ was reported to visit location $\mathcal{L}$ during time interval τ, and if the LBS provider observed that all request messages during the time interval τ came from a single user within location $\mathcal{L}$, then it can infer that the identity of the user in question is $\mathcal{A}$. The second type of attack is called *Observation Identification* [8] in the literature.

In order to protect location information from third parties that are semi-honest but not completely trusted, we define a *security perimeter* around the mobile client. In this paper, the security perimeter includes the mobile client of the user, the trusted *anonymity server*, and a secure channel where the communication between the two is secured through encryption (see Figure 1). By *semi-honest*, we mean that the third party LBS providers are honest and can correctly process and respond to messages, but curious such that they may attempt to determine the identity of a user based on what they "see", which includes information in the physical world that can lead to location-identity binding or association. We do not consider the case where LBS providers are malicious. Thus we do not consider the attack scenarios in which the LBSs can inject a large number of colluding dummy users into the system.

1.2 Architecture Overview

We show the system architecture in Figure 1. Mobile clients communicate with third party LBS providers through the anonymity server. The anonymity server is a secure gateway to the semi-honest LBS providers for the mobile clients. It runs a *message perturbation engine*, which performs *location perturbation* on the messages received from the mobile clients before forwarding them to the LBS provider. Each message sent to an LBS provider contains location information of the mobile client, a timestamp, in addition to service specific information. Upon receiving a message from a mobile client, the anonymity server 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. Spatial cloaking refers to replacing a two

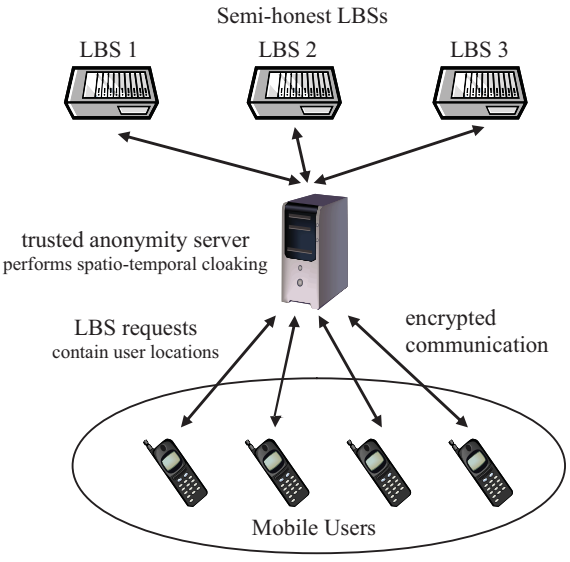


Figure 1: System Architecture

dimensional point location by a spatial range, where the original point location lies anywhere within the range. Temporal cloaking refers to replacing a time point associated with the location point with a time interval that includes the original time point. These terms were introduced by Gruteser and Grunwald [8]. In our work, the term *location perturbation* refers to the combination of spatial cloaking and temporal cloaking.

1.3 k-anonymity and Location k-anonymity

There are two popular approaches to protect location privacy in the context of LBS usage: policy-based [9] and anonymity-based approaches [8]. In policy-based approaches mobile clients specify their location privacy preferences as policies and completely trust that the third party LBS providers adhere to these policies. In anonymity-based approaches, LBS providers are assumed to be semi-honest instead of completely trusted. We advocate k-anonymity preserving management of location information by developing efficient and scalable system-level facilities for protecting location privacy through ensuring location k-anonymity. We assume that anonymous location-based applications do not require user identities for providing service. A discussion on pseudonymous and non-anonymous LBSs is provided in Section 7.

The concept of k-anonymity was originally introduced in the context of relational data privacy [10, 11]. It addresses the question of "How can a data holder release its private data with guarantees that the individual subjects of the data cannot be identified while the data remain practically useful" [12]. For instance, a medical institution may want to release a table of medical records with the names of the individuals replaced with dummy identifiers. However, some set of attributes can still lead to identity breaches. These attributes are referred to as the *quasi-identifier*. For instance, the combination of birth date, zip code and the gender attributes in the disclosed table can uniquely determine an individual. By joining such a medical record table with some publicly available information source, like a voters list table, the medical information can be easily linked to individuals. k-anonymity prevents such privacy breach by ensuring that each individual record can only be released if there are at least $k - 1$ distinct individuals whose associated records are indistinguishable from the former in terms of their quasi-identifier values.

In the context of LBSs and mobile clients, location k-anonymity refers to k-anonymous usage of location information. A subject is considered location k-anonymous if and only if the location information sent from a mobile client to an LBS is indistinguishable from the location information of at least $k - 1$ other mobile clients [8]. This paper argues that location perturbation is an effective technique for supporting location k-anonymity and dealing with location privacy breaches exemplified by the location inference attack scenarios discussed in Section 1.1. If the location information sent by each mobile client is perturbed by replacing the position of the mobile client with a coarser grained spatial range, such that there are $k - 1$ other mobile

clients within that range ($k > 1$), then the adversary will have uncertainty in matching the mobile client to a known location-identity association or an external observation of location-identity binding. This uncertainty increases with the increasing value of k, providing higher degree of privacy for mobile clients. Referring back to the roadside information service example of Section 1.1, now even though the LBS provider knows that person $\mathcal{A}$ was reported to visit location $\mathcal{L}$, it cannot match a message to this user with certainty. This is because there are at least k different mobile clients within the same location $\mathcal{L}$. As a result, it can not perform further tracking without ambiguity.

1.4 Contributions and Scope of the Paper

This paper describes a personalized k-anonymity model for protecting location privacy . Our design and development are motivated by an important observation: *Location Privacy Demands Personalization*. In other words, location privacy is a personalized requirement and is context sensitive. An individual may have different privacy requirements in different contexts and different individuals may require different levels of privacy in the same context. Unfortunately, existing anonymization solutions to location privacy threats in accessing LBSs, such as the earlier work on location k-anonymity [8], are essentially "one size fits all" − users do not have the ability to tailor the personalization capability to meet their individual privacy preferences. Most of the LBSs today promise ubiquitous access in an increasingly connected world. However, they do not take into account the possibility that a user's willingness to share location data may depend on a range of factors, such as various contextual information about the user, e.g., environmental context, task context, social context, etc. One way to support personalized location k-anonymity is to allow users to specify different k at different times.

There is a close synergy between location privacy and location k-anonymity. Larger k in location anonymity implies higher guarantees for location privacy. However, larger k values make it necessary to perform additional spatial and temporal cloaking for successful message anonymization, resulting in low spatial and temporal resolution for the anonymized messages. This in turn, may lead to degradation in the quality of service (QoS) of LBS applications, such as taking a longer time to serve the user's request or sending more than required information back to the mobile client as a result of the inaccuracy associated with the user's location after applying location cloaking.

These observations illustrate that there is a trade-off between the desired level of location privacy and the resulting loss of QoS from LBSs. The "one size fits all" approach to location privacy negatively impacts the quality of service for mobile clients with lower privacy requirements. This calls for a framework that can handle users with different location privacy requirements, and allows users to specify their preferred

balance between the degree of privacy and the quality of service (QoS).

Our location privacy model exhibits two distinct features. The first characteristic of our model is its ability to enable each mobile client to specify the *minimum level of anonymity* it desires and the *maximum temporal and spatial tolerances* it is willing to accept when requesting for k-anonymity preserving LBSs. Concretely, instead of imposing a uniform k for all mobile clients, 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 based on her specific privacy requirement, and can change this specification at per-message granularity. Furthermore, each user can specify her preferred spatial and temporal tolerances that should be respected by the location perturbation engine while maintaining the desired level of location k-anonymity. We call such tolerance specification and preference of the k value the *anonymization constraint* of the message. The preference of the k value defines the desired level of privacy protection that a mobile client wishes to have, whereas the temporal and spatial tolerance specifications define the acceptable level of loss in QoS from the LBS applications.

The second distinctive characteristic of our location privacy model is the development of an efficient *message perturbation engine*, which is run by the trusted anonymization server, and performs location anonymization on mobile clients' LBS request messages, such as identity removal and spatio-temporal cloaking of location information. We develop a suite of scalable and efficient spatio-temporal location cloaking algorithms, taking into account the tradeoff 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 cloaking algorithms to perturb the location information contained in the messages sent from mobile clients, before forwarding any request messages to the LBS provider(s).

We conduct a series of experimental evaluations. Our results show that the proposed personalized location k-anonymity model, together with our message perturbation engine and location cloaking algorithms, can achieve high guarantees of k-anonymity and high resilience to location privacy threats without introducing significant performance penalties. To the best of our knowledge, previous work on location privacy has not addressed these issues. An earlier version of this paper appeared in a conference [13], which focuses on the basic concept of personalized location k-anonymity and the design ideas of our base algorithm for performing personalized location cloaking. In contrast, this paper presents the complete framework for supporting personalized location privacy, which includes the formal definition of personalized location k-anonymity, the theoretical framework for the proposed base algorithm with respect to its compliance with personalized location k-anonymity, the optimized algorithms that enhance the base algorithm, and an extensive experimental study illustrating the effectiveness of the newly proposed algorithms.

2 Personalized Location k-anonymity: Terminology and Definitions

2.1 Message Anonymization Basics

In order to capture varying location privacy requirements and ensure different levels of service quality, each mobile client 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 clients into a new message that can be safely (k-anonymously) forwarded to the LBS provider. The key idea that underlies 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 clients present in the same spatial area. This approach is called spatial cloaking. Second, one can achieve location anonymity by delaying the message until k mobile clients have visited the same area located by the message sender. This approach is called temporal cloaking. For reference convenience, we provide Table 1 to summarize the important notations used in this section and throughout the rest of the paper. The last three notations will be introduced in Section 3.

We denote the set of messages received from the mobile clients as S. We formally define the messages in the set S as follows:

$$m_s \in S\colon \langle \underbrace{u_{id}, r_{no}}_{\substack{\text{sender id}\\ \text{message no}}}, \underbrace{\{t, x, y\}}_{\substack{\text{spatio-temporal}\\ \text{point}}}, \underbrace{k}_{\substack{\text{anonymity}\\ \text{level}}}, \underbrace{\{d_t, d_x, d_y\}}_{\substack{\text{temporal and spatial}\\ \text{tolerances}}}, C\rangle$$

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 client 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 client in the two dimensional space (x-axis and y-axis), and the timestamp t refers to the time point at which the mobile client was present at that position (temporal dimension: t-axis).

Notation	Meaning
S	Source message set
T	Transformed message set
m_s	A message in set S
m_t	A message in set T
$R(m_s)$	Transformed format of m_s
k	Anonymity level
u_{id}, r_{no}	Sender id, message no
d_t and d_x, d_y	Temporal and spatial tolerances
$L(m_s) = (x, y, t)$	Spatio-temporal point of m_s
$B_{cn}(m_s)$	Constraint box of m_s
$B_{cl}(m_t)$	Cloaking box of m_t
$B_m(S')$	MBR of a set of source messages
$G_m(S, E)$	Constraint graph
$nbr(m_s, G_m)$	Neighbors of m_s in G_m

Table 1: Notation Reference Table

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 client. Thus, larger k values imply higher degrees of privacy. One way to determine the appropriate k value is to assess the certainty with which an adversary can link the message with a location/identity association or binding. This certainty is given by $1/k$.

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 results to location-dependent service requests and larger temporal tolerances may result in higher latencies of the messages. The d_t value also defines a *deadline* for the message, such that a message should be anonymized until time $m_s.t + m_s.d_t$. If not, the message cannot be anonymized according to its constraints and it is dropped. 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. The messages in T are defined as follows:

$$m_t \in T\colon \langle u_{id}, r_{no}, \underbrace{\{X : [x_s, x_e], Y : [y_s, y_e], I : [t_s, t_e]\}}_{B_{cl}(m_t),\quad \text{spatio-temporal cloaking box}}, C\rangle$$

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 surjection (onto mapping) from S to $T \cup \{null\}$. Concretely, if $R(m_s) = m_t$, then $m_t.u_{id} = m_s.u_{id}$ and $m_t.r_{no} = m_s.r_{no}$. If $R(m_s) = null$, then the message m_s is not anonymized. The (u_{id}, r_{no}) fields of a message in T are replaced with a randomly generated identifier 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.

Basic Service Requirements

The following basic properties must hold between a raw message m_s in S and its perturbed format m_t in T, where $A \sqsubset B$ notation is used in the rest of the paper to express that the n-dimensional rectangular region A is contained in the n-dimensional rectangular region B:

i) *Spatial Containment*: $m_s.x \in m_t.X$, $m_s.y \in m_t.Y$

ii) *Spatial Resolution*: $m_t.X \sqsubset \Phi(m_s.x, m_s.d_x)$, $m_t.Y \sqsubset \Phi(m_s.y, m_s.d_y)$

iii) *Temporal Containment*: $m_s.t \in m_t.I$

iv) *Temporal Resolution*: $m_t.I \sqsubset \Phi(m_s.t, m_s.d_t)$

v) *Content Preservation*: $m_s.C = m_t.C$

Spatial containment and temporal containment requirements state that the cloaking box of the perturbed message, $B_{cl}(m_t)$, should contain the spatio-temporal point $L(m_s)$ of the original message m_s. Spatial resolution and temporal resolution requirements amount to say 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 desired maximum tolerance value specified in the original message. This is equivalent to stating that the cloaking box of the perturbed message should be contained within the constraint box of the original message, i.e., $B_{cl}(m_t) \sqsubset B_{cn}(m_s)$. Content preservation property ensures that the message content remains as it is.

Anonymity Requirement

We formally capture the essence of 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 client. 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.

The following formal definition summarizes personalized location k-anonymity:

Definition 2.1. *Personalized location k-anonymity:*
The set T of anonymized messages respect personalized location k-anonymity if and only if the following conditions are met for each message m_t in T and its corresponding message m_s in S where $m_t = R(m_s)$: (i) Spatial Containment, (ii) Spatial Resolution, (iii) Temporal Containment, (iv) Temporal Resolution, (v) Content Preservation, and (vi) Location k-anonymity.

2.2 Location k-Anonymity: Privacy Value and Performance Implications

The personalized location k-anonymity model presented in this paper strives to strike a balance between providing effective location privacy and maintaining desired quality of service and performance characteristics. In the rest of this section, we first describe the privacy value of our approach and then discuss the impact of location perturbation on the performance and QoS of the location-based applications.

Privacy Value of Location k-Anonymity

We illustrate the privacy value of our location k-anonymity model by comparing it with the strategy that only masks the sources of the LBS requests. Assuming that the sources of service request messages of mobile clients can be masked from the LBS servers through the use of a trusted anonymization server as a mix [14, 15], we want to show that the location information retained in these messages can still create a threat against location privacy, especially when combined with information obtained from external observation or knowledge.

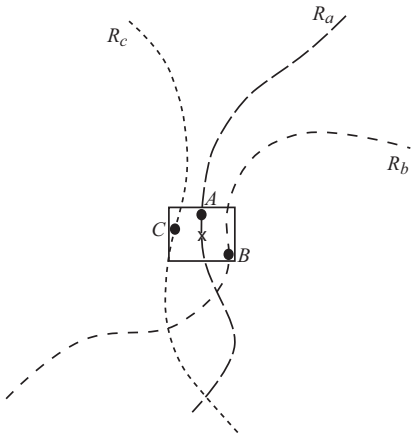


Figure 2: Linking attack

As an example consider the scenario in Figure 2, where three mobile clients, labeled as A, B, and C, and their trajectories (routes), denoted as R_a, R_b, and R_c, are depicted. If no location k-anonymization is performed at the anonymity server, then the following linking attack can be performed by an adversary: First, the adversary can construct an approximation of the trajectories of the mobile clients using the location information exposed in the service request messages of the mobile clients. Second, it can obtain a location/identity binding through external observation [8], such as A has been spotted at position x at time t. Finally, the adversary can link this binding with the trajectories it has at hand, finding the correct trajectory of node A. In Figure 2, if node A has been spotted at the position marked as x, then the trajectory R_a can be associated with A, since no other trajectories cross the point x. However, such a linking attack is not effective if proper location perturbation is performed by the trusted anonymity server. For instance, assume that the location information in the service request messages of nodes A, B, and C are perturbed according to location k-anonymity with $k = 3$. The location k-anonymization replaces the point position information x included in the request messages by a location cloaking box, as shown in Figure 2. Now, even if the adversary can infer approximate trajectories, it cannot link the location/identity binding with one of the trajectories with certainty, because the point x now links with three different request messages, each with probability $k^{-1} = 1/3$. In summary, the higher the k value is, the better the protection one can achieve against linking attacks.

QoS and Performance Implications of Location k-Anonymity

Achieving location k-anonymity with higher k, thus with higher location privacy, typically requires assigning larger cloaking boxes to perturbed messages. However, arbitrarily large cloaking boxes can potentially result in decreased level of QoS or performance with respect to the target location-based application. Here we give an example scenario to illustrate the negative impacts of using large cloaking boxes.

Assume that our LBS application provides *resource location service*, such as providing answers to continuous queries asking for nearest resources with certain types and properties during a specified time interval, e.g. nearest gas station offering gas under $\$2/gal$ during next half an hour. Figure 3 shows four objects o_1, o_2, o_3, and o_4 that are valid gas stations for the result of such a query posed by a mobile client A, where the aim of the LBS is to provide A with the nearest gas station, that is o_1. The figure also shows a sample cloaking box assigned to one of the service request messages of node A by the anonymity server. Since the LBS is unable to tell the exact location of node A within the cloaking box, and since the cloaking box covers all of the four objects, the LBS will be unable to determine the exact result, i.e., it cannot decide which of the four objects is the closest one. There are different possible ways of handling this problem, each with a different shortcoming. First, the server may choose to report only one result by making a random decision. In this case the QoS of the application, as perceived by the user, will degrade. The larger the cloaking box is, the higher the expected distance between the actual result and the reported result will be. Second, the server may choose to report all valid resources within the cloaking box. In this case several side-effects arise. For instance, one side-effect is the larger the cloaking box is, the larger the size of the result set will be, and thus the higher the communication and processing costs are, which degrades the performance. Moreover, the result set requires further filter processing at the mobile client side to satisfy the application semantics. This means each LBS may need to ship some amount of post-processing to the mobile client side in order to complete query processing, and results in running untrusted code at the mobile client side. On the other hand, the LBS may choose to skip the post processing step, which leaves the filtering to the user of the system and thus decreases the QoS. An alternative for the LBS is to use probabilistic query processing techniques [16] to rank the results and provide hints to the filtering process to alleviate the degradation in QoS.

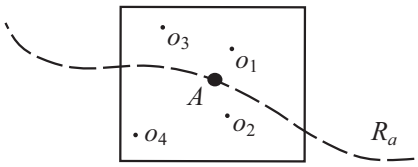


Figure 3: Loss of QoS/performance

In summary, we argue that there is a need for adjusting the balance between the level of protection provided by location k-anonymity and the level of performance degradation in terms of the QoS of LBSs due to location cloaking. Such a balance should be application driven. Our personalized location-anonymity model makes it possible to adjust this balance at per-user level with the granularity of individual messages.

3 Message Perturbation: Design and Algorithms

In this section, we first give an overview of the message perturbation engine. Then we establish a theoretical basis for performing spatio-temporal cloaking. This is followed by an in-depth description of the perturbation engine and the algorithms involved.

3.1 Engine Overview

The message perturbation engine processes each incoming message m_s from mobile clients 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* have passed, and thus should be dropped. The deadline of a message is the high point along the temporal dimension of its spatio-temporal constraint box and it is bounded by the user-specified temporal tolerance level.

In order to guide the process of finding the particular set of messages that should be anonymized as a group, we develop the $CliqueCloak$ theorem (see Section 3.2). We refer to cloaking algorithms that make their decisions based on this theorem as $CliqueCloak$ algorithms. The perturbation engine, which is driven by the *local-k search* as the main component of the detection step, is our base $CliqueCloak$ algorithm. Other $CliqueCloak$ algorithms are discussed in Section 4.

3.2 Grouping Messages for Anonymization

A key objective for location anonymization is to develop efficient location cloaking algorithms for providing personalized privacy protection while maintaining the desired quality of service. 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 client, 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 2.1 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[†]) 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 $\{L(m_s)|m_s \in M\}$, where $L(m_s)$ denote the spatio-temporal location point of the message m_s.

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:

Definition 3.1. *Constraint Graph*
Let $G(S, E)$ be an undirected graph where S is the set of vertices, each representing a message received at the trusted location 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 clients and their spatio-temporal points are contained in each other's constraint boxes defined by their tolerance values.

Theorem 3.1. *$CliqueCloak$ Theorem*
Let $G(S, E)$ be a constraint graph, $M = \{m_{s_1}, m_{s_2}, \ldots, m_{s_l}\} \subset S$, and $\forall_{1 \leq i \leq l}$, $m_{t_i} = \langle m_{s_i}.u_{id}, m_{s_i}.r_{no}, B_m(M), m_{s_i}.C\rangle$. Then, $\forall_{1 \leq i \leq l}, m_{t_i}$ is a valid k-anonymous perturbation of m_{s_i}, i.e., $m_{t_i} = R(m_{s_i})$, if and only if the set M of messages form an l-clique in the constraint graph $G(S, E)$ such that $\forall_{1 \leq i \leq l}, m_{s_i}.k \leq l$.

Proof. First we show that the left hand side holds if we assume that the right hand side holds. Spatial and temporal containment requirements are satisfied as we have $\forall_{1 \leq i \leq l}, L(m_{s_i}) \in B_m(M) = B_{cl}(m_{t_i})$ from definition of an MBR. k-anonymity is also satisfied, as for any message $m_{s_i} \in M$ there exists $l \geq m_{s_i}.k$ messages $\{m_{t_1}, m_{t_2}, \ldots, m_{t_l}\} \subset T$ s.t. $\forall_{1 \leq j \leq l}, B_m(M) = B_{cl}(m_{t_j}) = B_{cl}(m_{t_i})$ and $\forall_{1 \leq i \neq j \leq l}, m_{t_i}.u_{id} \neq$

[†]The MBR of a set of points is the smallest rectangular region that would enclose all the points.

$m_{t_j}.u_{id}$. The latter follows as M forms an l-clique and due to condition (iii) two messages m_{s_i} and m_{s_j} do not have an edge between them in $G(S, E)$ if $m_{s_i}.u_{id} = m_{s_j}.u_{id}$ and we have $\forall_{1\leq i\leq l}, m_{s_i}.u_{id} = m_{t_i}.u_{id}$.

It remains to prove that spatial and temporal resolution constraints are satisfied. To see this, consider one of any three dimensions in our spatio-temporal space, without loss of generality, say x-dimension. Let $x_{min} = min_{1\leq i\leq l}\ m_{s_i}.x$ and let $x_{max} = max_{1\leq i\leq l}\ m_{s_i}.x$. Since M forms an l-clique in $G(S, E)$, from condition (i) and (ii) we have $\forall_{1\leq i\leq l}, \{x_{min}, x_{max}\} \sqsubset \Phi(m_{s_i}.x, m_{s_i}.d_x)$ and thus $\forall_{1\leq i\leq l}, [x_{min}, x_{max}] \sqsubset \Phi(m_{s_i}.x, m_{s_i}.d_x)$ from convexity. Using a similar argument for other dimensions and noting that $B_m(M) = ([x_{min}, x_{max}], [y_{min}, y_{max}], [t_{min}, t_{max}])$, we have $\forall_{1\leq i\leq l}, B_m(M) \sqsubset B_{cn}(m_{s_i})$. Now we show that the right hand side holds if we assume that the left hand side holds. Since $\forall_{1\leq i\leq l}, m_{t_i} = R(m_{s_i})$, from definition of k-anonymity, we must have $\forall_{1\leq i\leq l}, m_{s_i}.k \leq l$. From spatial and temporal containment requirements, we have $\forall_{1\leq i\leq l}, L(m_{s_i}) \in B_m(M)$ and from spatial and temporal resolution constraints we have $\forall_{1\leq i\leq l}, B_m(M) \sqsubset B_{cn}(m_{s_i})$. These two imply $\forall_{1\leq i,j\leq l}, L(m_{s_i}) \in B_{cn}(m_{s_j})$ satisfying conditions (i) and (ii); and again from k-anonymity we have $\forall_{1\leq i\neq j\leq l}, m_{s_i}.u_{id} \neq m_{s_j}.u_{id}$ satisfying condition (iii). Thus S forms an l-clique in $G(S, E)$, completing the proof. □

3.3 Illustration of the Theorem

We demonstrate the application of this theorem with an example. Figure 4 shows four messages, m_1, m_2, m_3, and m_4. Each message is from a different mobile client[‡]. We omitted the time domain in this example for ease of explanation, but the extension to spatio-temporal space is straightforward. Initially, the 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 other's 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 cannot be anonymized and are removed from the graph. This is because $m_2.k = 3$ and as a result the clique does not satisfy the

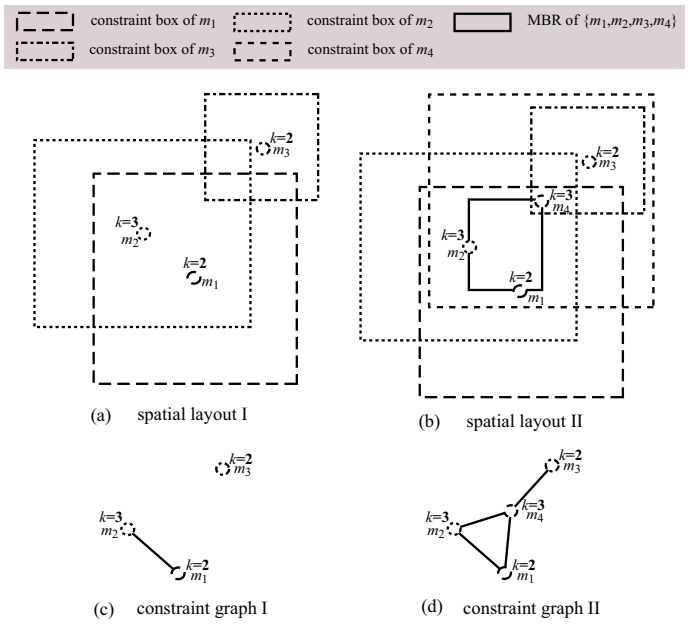


Figure 4: Illustration of the $CliqueCloak$ theorem

[‡]If a node sends a message while it already has one in the system, the new message can be easily detected

$CliqueCloak$ 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 4(b) clearly shows that the MBR is contained by the spatial constraint boxes of all messages within the clique. Once these messages are anonymized, the remaining message m_3 is not necessarily dropped from the system. It may later be picked up by some other new message and anonymized together with it or it may be dropped if it cannot be anonymized until its deadline specified by its temporal tolerance constraint.

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 $CliqueCloak$ theorem. We address the questions of *what* kind of cliques to search, *when* and *how* to search for such cliques, in detail in Section 4.

3.4 Data Structures

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

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

The *Multi-dimensional Index* I_m 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 the 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 [17] in our system.

The *Constraint Graph* G_m 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 3.2. The multi-dimensional index I_m is mainly used to speed-up the maintenance of the constraint graph G_m, which is updated when new messages arrive or when messages are anonymized or expired.

The *Expiration Heap* H_m 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

[§]It is memory-wise more efficient to store only identifiers of messages as data in I_m, G_m, and H_m

is used to detect expired messages that cannot be successfully anonymized, so that they can be dropped and removed from the system.

3.5 Engine Algorithms

Upon arrival of a new message, the perturbation 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 perturbation engine is given in Algorithm 1.

MSGPERTENGINE()

The engine runs in its own thread, as long as the variable $engine\_running$ remains set to **true**. This variable is initialized to **true** when the engine is turned on, and is set to **false** when the engine is closed down upon an explicit command.

```
(1) while engine_running = true
(2)   if Q_m ≠ ∅
(3)     m_sc ← Pop the first item in Q_m
(4)     Add m_sc into I_m with L(m_sc)
(5)     Add m_sc into H_m with (m_sc.t + m_sc.d_t)
(6)     Add the message m_sc into G_m as a node
(7)     N ← Range search I_m using B_cn(m_sc)
(8)     foreach m_s ∈ N, m_s ≠ m_sc
(9)       if L(m_sc) ∈ B_cn(m_s)
(10)        Add the edge (m_sc, m_s) into G_m
(11)    G'_m ← Subgraph of G_m consisting of messages in N
(12)    M ← LOCAL-k-SEARCH(m_sc.k, m_sc, G'_m)
(13)    if M ≠ ∅
(14)      Randomize the order of messages in M
(15)      foreach m_s in M
(16)        Output perturbed message
                m_t ← ⟨m_s.u_id, m_s.r_no, B_m(M), m_s.C⟩
(17)        Remove the message m_s from G_m, I_m, H_m
(18)  while true
(19)    m_s ← Topmost item in H_m
(20)    if m_s.t + m_s.d_t < now
(21)      Remove the message m_s from G_m, I_m
(22)      Pop the topmost element in H_m
(23)    else break
```

Algorithm 1: Message Perturbation Engine

Phase 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 all the messages pending for perturbation and locate the messages that should be assigned as neighbors to it in the constraint 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. 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 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 3-11 in the pseudo code.

Phase 2: Detection − In this step we apply the *local-k search* algorithm 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 $nbr(m_{s_c}, G_m)$. Formally $nbr(m_{s_c}, G_m)$ is defined as $\{m_s|(m_{s_c}, m_s)$ is an edge in $G_m\}$. 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 $CliqueCloak$ 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 1-2). 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 3-8). At each filtering step, every message $m_s \in U$ is checked to see 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.

LOCAL-k_SEARCH(k, m_{s_c}, G'_m)
(1) $U \leftarrow \{m_s | m_s \in nbr(m_{s_c}, G'_m)$ and $m_s.k \leq k\}$
(2) **if** $|U| < k-1$ **then return** $\emptyset$
(3) $l \leftarrow 0$
(4) **while** $l \neq |U|$
(5) $\quad l \leftarrow |U|$
(6) $\quad$ **foreach** $m_s \in U$
(7) $\quad\quad$ **if** $(|nbr(m_s, G'_m) \cap U| < k-2)$
(8) $\quad\quad\quad U \leftarrow U \backslash \{m_s\}$
(9) Find any subset $M \subset U$, s.t.
$\quad |M| = k-1$ and $M \cup \{m_{s_c}\}$ forms a clique
(10) **if** M found **then return** M
(11) **else return** $\emptyset$

Algorithm 2: local-k Search Algorithm

Phase 3: 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 first randomized to prevent temporal correlations with message arrival times. Then they are anonymized by assigning $B_m(M)$ (the MBR of the spatio-temporal points of the messages in the clique), as their cloaking box. Then they 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 13-17. 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.

Phase 4: Expiration − A message is considered to be expired if the current time is beyond the high point along the temporal dimension of the message's spatio-temporal constraint box, which means that the message has delayed beyond its deadline. 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 have 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 18-23 of the pseudo code deals with expiration.

4 Improved $CliqueCloak$ Algorithms

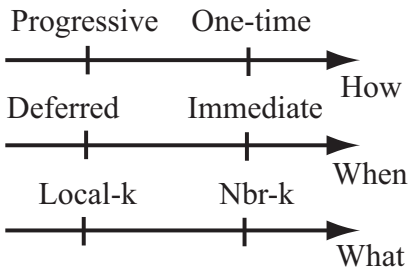


Figure 5: Algorithmic Dimensions

In this section, we describe several $CliqueCloak$ algorithms that improve the performance of the base algorithm described in Section 3.5. These variations are introduced through configurations along the three dimensions shown in Figure 5. These three dimensions represent the following three critical aspects of the clique search performed for locating a group of messages that can be anonymized together: (i) what sizes of message groups are searched (ii) when the search is performed, (iii) how the search is performed.

In the rest of this section, we discuss various optimizations we propose along these three dimensions to improve the basic algorithm. All of the proposed optimizations are heuristic in nature. We would like to note that the general problem of optimal k-anonymization is shown to be NP-hard [18, 19].

4.1 What Size Cliques to Search: Nbr-k vs. Local-k

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} in the graph, it must contain m_{s_c}. Thus, m_{s_c} is a good starting position. In addition, we want to look for bigger cliques that include m_{s_c}, instead of searching for a clique with size $m_{s_c}.k$, provided that the k value of each message within the clique is smaller than or equal to the size of the clique. There are two strong motivations behind this approach. First, by anonymizing a larger number of messages at once, we can minimize the number of messages that have to wait for later arriving messages in order to be anonymized. Second, by anonymizing messages in larger groups, we can provide better privacy protection against linking attacks. We develop the *nbr-k* search algorithm based on these design guidelines. Its pseudo code is given in Algorithm 3.

NBR-k_SEARCH(m_{s_c}, G'_m)
(1) **if** $|nbr(m_{s_c}, G'_m)| < m_{s_c}.k - 1$ **then return** $\emptyset$
(2) $V \leftarrow \{m_s.k | m_s = m_{s_c} \vee m_s \in nbr(m_{s_c}, G'_m)\}$
(3) **foreach** distinct $k \in V$ in decreasing order
(4) **if** $k < m_{s_c}.k$ **then return** $\emptyset$
(5) $M \leftarrow$ LOCAL-k_SEARCH(k, m_{s_c}, G'_m)
(6) **if** $M \neq \emptyset$ **then return** M
(7) **return** $\emptyset$

Algorithm 3: nbr-k Search Algorithm

Nbr-k search 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 V in the pseudo code. The k values in V 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 $k \in V$ considered, a clique of size k is searched by calling the local-k_Search function with appropriate parameters (see line 5). If such a clique can be found, the messages within the clique are returned. To integrate nbr-k search into the message perturbation engine, we can replace line 12 of Algorithm 1 with the call to nbr-k_Search function.

4.2 When to Search for Cliques: Deferred vs. Immediate

We have described the algorithm of searching for cliques upon the arrival of a new message, and refer to this type of search as *immediate* search. The immediate search is not beneficial when the constraint graph is sparse around the new message and the anonymization is less likely to be successful. Thus it may result in increased number of unsuccessful searches and deteriorate the performance.

Instead of immediately searching for a clique for each newly arrived message, we can defer this processing. One extreme approach to deferred search is to postpone the clique search for every message and perform clique search for a deferred message at the time of its expiration if it has not yet anonymized together with other messages. However, this will definitely increase the delay for all messages. An alternative way to deferred search is to postpone the search only if the new message does not have enough neighbors around and thus the constraint graph around this new message is sparse. Concretely, we introduce a system parameter $\alpha \geq 1$ to adjust the amount of messages for which the clique search is deferred, and perform the clique search for a new message m_{s_c} only if the number of neighbors this new message has at its arrival time is larger than or equal to $\alpha * m_{s_c}.k$. Smaller α values push the algorithm toward immediate processing. We can set the α value 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 refer to this variation of the clique search algorithm as *deferred* $CliqueCloak$. The main idea of deferred search is to wait until certain message density is reached and then perform the clique search. Thus, the deferred approach performs less number of clique searches at the cost of larger storage and data structure maintenance overhead. The deferred search can improve the overall performance when clique searches dominate the running time when compared to the performance of index search and update.

4.3 How to Search Cliques: Progressive vs. One-time

Another important aspect that we can optimize the clique search performance is from the how to search perspective. It is interesting to note that when the constraint boxes of messages are large, messages are more likely to be anonymized since the constraints are relaxed. However, when the constraint boxes of messages are large, the clique searches do not terminate early and incur a high performance penalty. We observe that this is due to the increased search space of the clique search phase, which is a direct consequence of the fact that large constraint boxes result in large number of neighbors around the messages in the constraint graph. This inefficiency becomes more prominent with increasing k due to the combinatorial nature of the search. An obvious way to improve the search is to first consider neighbors that are spatially close by, which allows us to terminate our search quickly and avoid or reduce the processing time spent on the neighbors that are

spatially far away and potentially less useful for anonymization.

The *progressive search* technique builds upon this insight. It first sorts (in increasing order) the neighbors of a message m_{s_c}, based on the distance of their spatio-temporal point to m_{s_c}'s spatio-temporal point. Then the search (either local-k or nbr-k) is performed iteratively over the set of sorted messages using a progressively enlarging subgraph that consists of progressively increasing number of messages from the sorted set of messages. Initially the subgraph consists of only $z * m_{s_c}.k$ messages, where $z = 2$. If a proper clique cannot be found, we increase z by one and apply the search over the enlarged subgraph again. This process repeats until the complete set of sorted messages is exhausted. Algorithm 4 gives a sketch of the progressive search. For the purpose of comparison, we refer to the non-progressive search as *one-step* search.

PROGRESSIVESEARCH(m_{s_c}, G'_m)
(1) $U \leftarrow$ Sort $nbr(m_{s_c}, G'_m)$ based on Euclidean distance between $L(m_s)$ and $L(m_{s_c})$, $m_s \in nbr(m_{s_c}, G'_m)$
(2) $z \leftarrow 1$
(3) **repeat**
(4) $z \leftarrow z + 1$
(5) $v \leftarrow$ MIN(z*$m_{s_c}.k$, $|U|$)
(6) $G''_m \leftarrow$ Subgraph of G'_m containing m_{s_c} and the first $v - 1$ messages in U
(7) $M \leftarrow$ LOCAL-k_SEARCH(k, m_{s_c}, G''_m) // or NBR-k_SEARCH(m_{s_c}, G''_m)
(8) **if** $M \neq \emptyset$ **then return** M
(9) **until** $v < |U|$
(10) **return** $\emptyset$

Algorithm 4: Progressive Search Algorithm

5 Evaluation Metrics

In this section we discuss several evaluation metrics for system level control of the balance between privacy value and performance implication in terms of QoS. These metrics can be used to evaluate the effectiveness and the efficiency of the message perturbation engine.

Success Rate is an important measure for evaluating the effectiveness of the proposed location k-anonymity model. Concretely, the primary goal of the cloaking algorithm is to maximize the number of messages perturbed successfully in accordance with their anonymization constraints. In other words, we want to maximize $|T|$. 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. It is important to remember that the primary evaluation metric for our algorithms is the success rate. This is because our approach always guarantees that the anonymization constraints are respected for the messages that are successfully anonymized. When comparing two approaches that have similar success rates, then one that provides better relative anonymity level or relative spatial/temporal resolution is preferred.

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. Higher relative anonymity levels mean that on the average messages are getting anonymized with larger k values than the user-specified minimum k-anonymity levels. Due to the inherent trade-off between the anonymity level and the spatial and temporal resolutions, a user may have to specify a lower k value than what she actually desires, in order to maintain a certain amount of spatial resolution and/or temporal resolution for the service request messages. In these cases, we will prefer algorithms that can provide higher relative anonymity levels.

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 $||.||$ is applied to an interval and gives its length. The numerator in the equation is the area of the constraint box for the source message, whereas the denominator is the area of the cloaking box for its transformed format. Higher relative spatial resolution values imply that anonymization is performed with smaller spatial cloaking regions relative to the constraint boxes specified.

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 that anonymization is performed with smaller temporal cloaking intervals and thus with smaller delays due to perturbation. Relative spatial and temporal resolutions cannot 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 the experiments of Section 6, we use the average CPU time needed to process 10^3 messages as the message processing time.

6 Experimental Study

We break up the experimental evaluation into three components. The first two components demonstrate the effectiveness of the $CliqueCloak$ algorithms in realizing the proposed personalized location k-anonymity model, in terms of the success rate, relative anonymity level, and relative spatial/temporal resolution. The third component studies the scalability of the algorithms under extreme cases, in terms of the running-

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 2: Message generation parameters

road type	expressway	arterial	collector
mean of car speeds (km/h)	90	60	50
std.dev. in car speeds (km/h)	20	15	10
traffic volume data (cars/h)	2916.6	916.6	250

Table 3: Car movement parameters

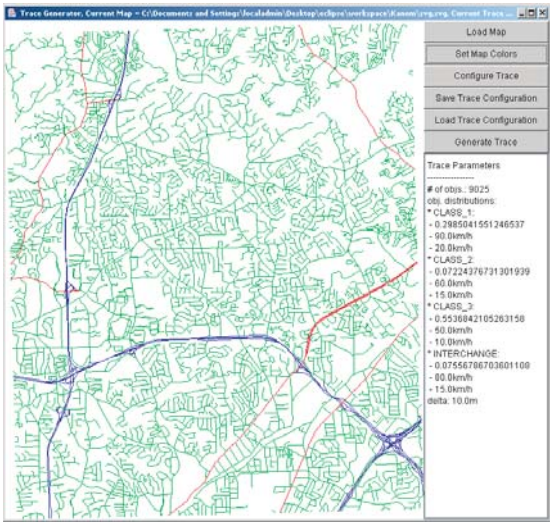


Figure 6: The trace generator

time performance. 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.

6.1 Experimental Setup

We have developed a trace generator (shown in Figure 6), 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 the National Mapping Division of the United States Geological Survey (USGS) [20] in SDTS [21] format. We use transportation layer of 1:24K Digital Line Graphs (DLGs) as road data. We convert the graphs into Scalable Vector Graphic [22] format using the Global Mapper [23] 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 for different road classes. The total number of cars on a certain class of roads is proportional to the total length of the roads for that class, the traffic volume for that class, and is inversely proportional to the average speed of cars for that class. 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. A car changes its speed at each joint based on a normal distribution whose mean is equal to the average speed for the particular class of roads that the car is on.

We used a map from the Chamblee region of the state of Georgia in the USA to generate the trace used in this paper. Figure 6 shows this map loaded into the trace generator. The map covers a region of $\approx 160km^2$. In terms of the length of roads, class 1 roads constitute 7.3% of the total, whereas class 2 and

3 roads constitute 5.4% and 87.3%, respectively. The mean speeds and standard deviations for each road type are given in Table 3. The traffic volume data is taken from Gruteser and Grunwald [8] and is also listed in Table 2. These settings result in approximately 10,000 cars, 32% of which are on class 1 roads, 13% are on class 2 roads, and 55% are on class 3 roads. The trace has the duration of one hour. Each car generates several messages during the simulation. All evaluation metrics are calculated over these messages generated within the one hour period ¶ (over one million messages). The experimental results are averages of large number of messages. Each message specifies an anonymity level in the form of a k value, which is picked from the list $\{5, 4, 3, 2\}$ using a Zipf distribution with parameter 0.6. The setting of $k = 5$ is the most popular one, and $k = 2$ is the least popular one based on the Zipf distribution. In certain experiments we extend this list up to $k = 12$, keeping the highest k value as the most popular anonymity level. This enables us to model a population which prefers higher privacy in general. We show that even for such a workload, the personalized k-anonymity model provides significant gains.

The spatial and temporal tolerance values of the messages are selected using normal distributions whose default parameters are given in Table 2. 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 listed in Table 2.

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 [6]. 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. Our results show that the personalized location k-anonymity approach presented in this paper is a promising solution. Although there are many aspects of the experimental design, such as car movement patterns, privacy and QoS requirements of users, message generation rates, etc., which can effect the results of our experiments, we believe that our results provide a necessary and informative first step to understand the fundamental characteristics of this personalized location privacy model. We hope that the results presented in this paper will stimulate more research to comprehensively evaluate the applicability of personalized location privacy in real-world LBS applications.

¶From running 15 minutes traces 12 times we have observed insignificant standard deviation values (≈ 0.4 for mean success rate of ≈ 70) and thus decided to report results from a single 1 hour trace.

6.2 Effectiveness of Personalized k-Anonymity Model

We first study the effectiveness of our personalized location k anonymity model with respect to (1) different k requirements from individual users and (2) the uniform k-anonymity model.

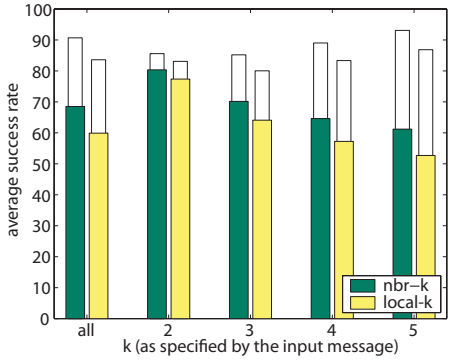
Table 4 shows the advantage of using variable k, compared to using a uniform k value ($= 5$) independent of the individual k values specified in the messages. We observe that the variable k approach provides 33% higher success rate, 110% better relative spatial resolution, and 30% better relative temporal resolution for messages with $k = 2$. The improvements are higher for messages with smaller k values, which imply that the variable k location anonymity approach does not unnecessarily penalize users with low privacy requirements, when compared to the uniform k approach. The amount of improvement in terms of the evaluation metrics decreases as k approaches to its maximum value of 5.

	with variable k				with fixed $k\,(=5)$
	2	3	4	5	
success rate	79.1	70.1	64.2	59.8	59.4
relative spatial res.	6.3	4.0	3.6	3.3	3.0
relative temporal res.	15.9	14.0	13.4	13.0	12.2

Table 4: Success rate and relative spatial/temporal resolutions, fixed k compared to variable k

6.3 Results on Success Rate

Figure 7 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 $CliqueCloak$ 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 $U = \{m_{s_i} | m_{s_i} \in S \wedge L(m_{s_i}) \in B_{cn}(m_s) \wedge m_{s_i}.t \in \Phi(m_s.t, m_s.d_t)\}$ 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 during its lifetime 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 anonymizing 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 suffi-



The thin bars represent a lower bound on the percentage of the messages that cannot be successfully anonymized, due to insufficient spatio-temporal message density and not because of the suboptimality of the anonymization algorithm.

Figure 7: Success rates for different k values

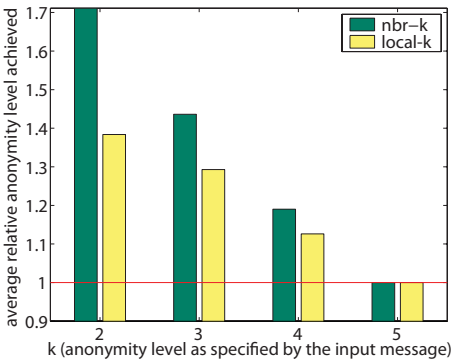


Figure 8: Relative anonymity levels for different k values

cient 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 off-line. 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 anonymized.

There are three observations from Figure 7. 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 of a way to construct the optimal algorithm with a reasonable time and space complexity given full knowledge of the trace, we could have gotten 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 8 shows the relative anonymity level (the higher, the better) for nbr-k and local-k. 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 algorithms do not attempt to search cliques of sizes larger than the maximum k value in the system. The difference in the relative anonymity level between nbr-k and local-k shows that the nbr-k approach is able to anonymize messages with smaller k values together with the ones with higher k values. This is particularly beneficial for messages with higher k values, as they are harder to anonymize. This also explains why nbr-k results in a better success rate.

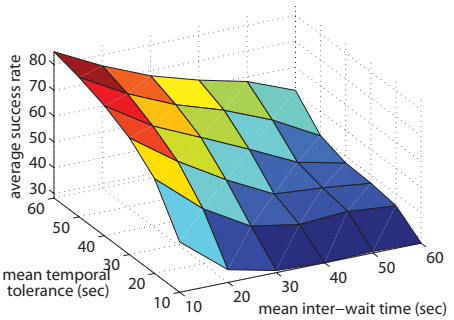


Figure 9: Success rate wrt. temporal tolerance and inter-wait time

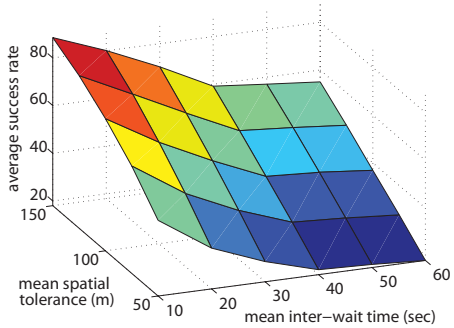


Figure 10: Success rate wrt. to spatial tolerance and inter-wait time

We also studied the average success rate and the message processing time for nbr-k and local-k search approaches with immediate or deferred processing mode. The results can be found in our technical report [13]. In summary, we found that the immediate approach provides better success rate than the deferred

approach, and that the deferred approach does not provide improvement in terms of the message processing time even though it decreases the number of times the clique search is performed. The reason that the deferred approach performs worse in terms of the total processing time is that, for $k \leq 10$ the index update dominates the cost of processing the messages 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 and thus the clique search phase dominates the cost.

Figure 9 plots the average success rate as a function of mean inter-wait time and mean temporal tolerance. Similarly, Figure 10 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 have tremendous effect on the success rate. Although high success rates (around 85) are achieved with high temporal and spatial tolerances, we will show in the next section that the relative temporal and spatial resolutions are much larger than 1 in such cases.

6.4 Results on Spatial/Temporal Resolution

In Section 6.3, we 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 11(a) plots the frequency distribution (y-axis) of the relative temporal resolutions (x-axis) of the anonymized messages. For a specific resolution value ν on the x-axis, the corresponding value on the y-axis represents the frequency of messages having a relative temporal resolution value of ν. Figure 11 shows that in 75% of the cases the provided relative temporal resolution is > 3.25, thus an average temporal accuracy of roughly $< 10s$ (recal 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

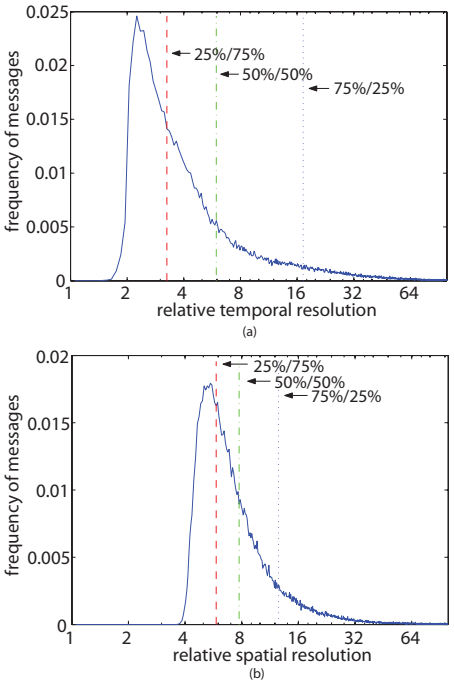


Figure 11: Relative temporal and spatial resolution distributions

points out that the observed performance with regard to temporal resolutions is much better than the worst case specified by the temporal tolerances. Moreover, this property of the algorithm holds under different settings of the mean and variance values for the spatial and temporal tolerances [13].

Figure 11(b) plots the frequency distribution (y-axis) of the relative spatial resolutions (x-axis) of the anonymized messages. Figure 11 shows that in 75% of the cases the provided relative spatial resolution is > 5.85, thus an average spatial accuracy of roughly $< 18m$ (recal 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. Moreover, this property of the algorithm holds under different settings of the mean and variance values for the spatial and temporal tolerances [13].

6.5 Results on Message Processing Time

We now evaluate the scalability of our algorithms with respect to message processing time and message success rate. We measure how the $CliqueCloak$ algorithm performs in extreme conditions with large k and large subgraphs on which the search is performed. Given that larger constraint boxes lead to larger subgraphs, we multiply the default values of the spatial and temporal tolerances that define the constraint boxes with a *scaling factor*. Larger values of the scaling factor represent more relaxed constraints with respect to spatial and temporal tolerances, and thus larger constraint boxes.

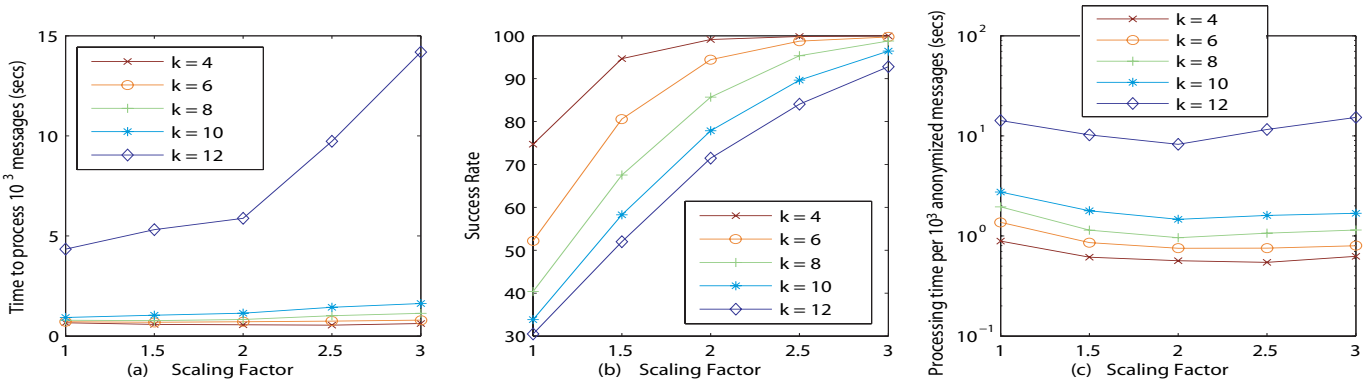


Figure 12: Scalability with respect to size of constraint boxes and k

Figure 12 (a) plots the time to process 10^3 messages as a function of the scaling factor, for various k values going up to 12. All experiments reported in Figure 12 use [nbr-k, immediate, one-time] configuration. We make two observations from Figure 12 (a). First, the message processing time shows a smaller increase with increasing k initially, which is replaced by an exponential increase after $k = 10$. We have omitted cases where $k > 12$ due to their very high message processing times. This observation is in line with our previous claim that the clique search part does not dominate the message processing cost until k gets close

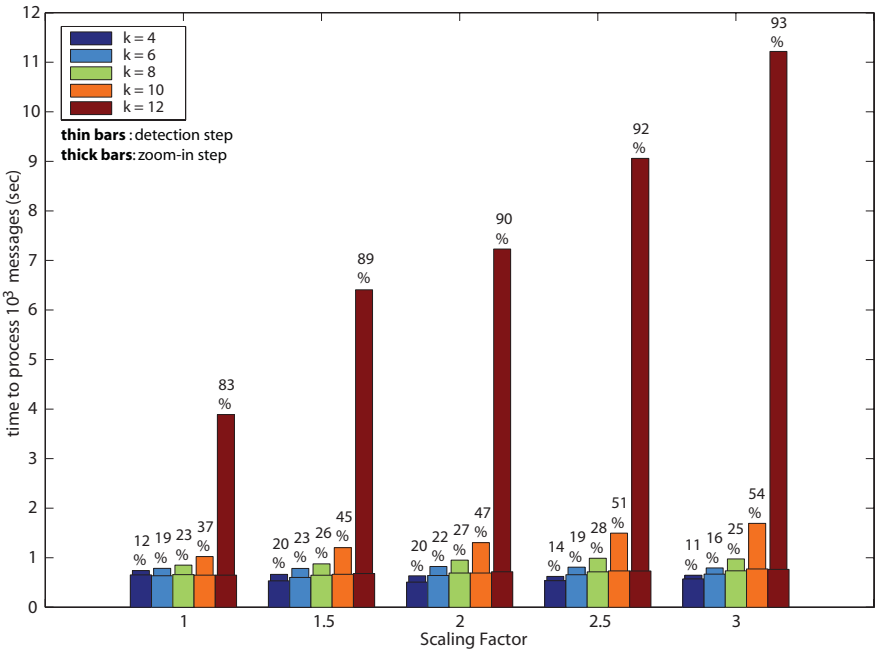


Figure 13: Break-up of message processing time with varying k

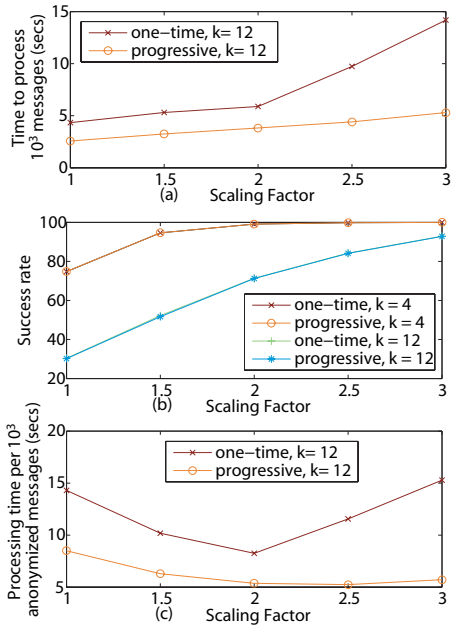


Figure 14: Impact of progressive search on Performance

to 10. This is further backed up by Figure 13, which will be described shortly. The second, and more interesting observation is that, the message processing time shows an increase with increasing constraint box size (scaling factor), especially for large k. Certain amount of increase in message processing time can be described by the fact that the number of successfully anonymized messages increases with increasing scaling factor. This is shown by Figure 12 (b), which plots the success rate as a function of scaling factor for various k values. Although the increase in message processing time can be justified by the extra work done for anonymizing a larger number of messages, the exact evaluation requires a new metric, which we define as the time to process 10^3 messages divided by the fraction of messages successfully anonymized. We call this metric the *processing time per* 10^3 *anonymized messages*. Figure 12 (c) plots processing time per 10^3 anonymized messages as a function of the scaling factor for various k values. Now we can observe a decrease in the evaluation metric with increasing scaling factor, which is intuitive since more relaxed tolerance values (large scaling factor) is expected to improve running-time performance. However, after the scaling factor goes over 2 (2.5 for $k \leq 6$), we see a reverse trend!

Before presenting results on "fixing" this behavior by employing the progressive search technique, we present the breakdown of the message processing cost into its components to show that the clique search starts to dominate the processing time when the scaling factor becomes high, especially for large k. Figure 13 shows the time to process 10^3 messages as a function of scaling factor for various k values as a bar chart, where each bar is divided into two parts. The upper thin bar represents the time spent for the clique search (detection step) and the lower thick bar represents the time spent for searching the spatio-temporal index

(zoom-in step). Percentages are also given over each bar, representing the first part's share. We can observe that for small k the detection step is not dominant. Even for $k = 10$, it is responsible for half the processing time only when the scaling factor is increased to 2.5. For $k = 12$ we observe that the detection step clearly dominates and becomes even more dominant with increasing scaling factor.

Figure 14 (a) plots the time to process 10^3 messages as a function of the scaling factor, for $k = 12$ with two different configurations; progressive search and one-time search. All experiments reported in Figure 14 use [nbr-k, immediate] configuration for the other two dimensions. Figure 14 (a) shows that the progressive approach is able to scale linearly with the tolerance values and provides up to 50% improvement over the one-time seach approach for the particular range of scaling factors used in this experiment. Figure 14 (b) plots the success rate as a function of scaling factor, for $k = 4$ and $k = 12$ with progressive and one-step search. It shows that the improvement in processing time comes at no cost with respect to success rate, both for small k and large k. Finally, Figure 14 (c) plots processing time per 10^3 anonymized messages as a function of the scaling factor, for $k = 12$ with progressive and one-step search. We observe that the progressive search succesfully removes most of the increasing trend seen in one-step search. However, it is not completly removed, as can be seen from the small increase when the scaling factor goes from 2.5 to 3. This also points out that there should be a system specified maximum constraint box to stop the performance degradation with unnecessarily large constraint boxes.

6.6 Summary of Experimental Results

We summarize major findings from our experiments and the insights obtained from the experimental results in four points: *i*) Nbr-k outperforms local-k in both success rate and relative anonymity level metrics, without incurring extra processing overhead. This is due to its ability to anonymize larger groups of messages together at once. *ii*) Deferred search, a technique that aims at decreasing the number of clique searches performed in an effort to increase running time performance, turns out to be inferior to immediate search. This is because, for smaller k values the index search and update cost is dominant over the clique search cost, and the deferred search increases the size of the index due to batching more messages before performing the clique searches. *iii*) Progressive search improves the running time performance of anonymization, especially when constraint boxes and k values are large, without any side-effects on other evaluation metrics. This nature of progressive search is due to its proximity-aware nature $-$ the close-by messages that are more likely to be included in the result of the clique search are considered first with progressive search. *iv*) The $CliqueCloak$ algorithms have the nice property that, for most of the anonymized messages, the cloaking box generated is much smaller than the constraint box of the received message specified by the tol-

erance values, resulting in higher relative spatial and temporal resolutions. In conclusion, the configuration of [Nbr-k, Immediate, Progressive] is superior to other alternatives.

7 Discussions and Future Work

This section discusses some potential improvements, alternatives, and future directions of our work.

Success Rate and QoS vs. Privacy Trade-off

Our personalized k-anonymity model requires mobile clients to specify their desired location anonymity level and their spatial/temporal tolerance constraints. It is possible that the level of privacy and the QoS can be in conflict in a user's specification. When such conflicts occur, the success rate of anonymization will be low for this user's messages. In practice, such conflicts should be checked to determine the need for fine-tuning in the privacy level or QoS. The trade-off between the QoS defined by spatial/temporal tolerance constraints and the level of privacy protection defined by anonymity level k should be adjusted, such that success rate of anonymization is kept close to 1. In this paper, we developed a location anonymization framework and associated system-level facilitates for fine-tuning of the QoS vs. privacy protection trade-off. Due to the space constraint, we did not discuss the application-dependent management of user involved adjustment of this trade-off. We believe that these issues merit an independent study.

Optimality of the $CliqueCloak$ Algorithms

It is important to note that the $CliqueCloak$ algorithms we introduced in this paper are heuristic in nature. Although we do not know the best success rates that can be achieved for various distributions of anonymity constraints, we experimentally showed that for practical scenarios, in the worst case our algorithms drop only 10% of the messages due to non-optimality. Furthermore, since it is extremely hard to accurately predict future patterns of messages, it is difficult to build an on-line optimal algorithm. These two observations lead us to the conclusion that our algorithms will be highly effective in practice. However, it is an open problem to study advanced algorithms that have better optimality and runtime performace.

Pseudonymous and Non-anonymous LBSs

In this paper we assumed that the LBSs are anonymous, i.e., the true identities of mobile clients are not required in the services provided. Services that require the knowledge of user identities or pseudonyms (non-anonymous and pseudonymous LBSs) will make tracking of successive messages from the same users trivial at the LBS side. We beleive that the pseudonymous LBSs can benefit from our solution with some modifications. For instance, one complication may arise when successive location-identity bindings take place, and the set of k messages from the two adjacent bindings share one and only one pseudonym, which can easily lead to a trajectory-identity binding. This type of vulnerabilities can be prevented or mitigated

by setting proper time intervals for changing the pseudonyms associated with mobile clients, without violating the service requirements of the LBSs. Nevertheless, further research is needed for devising effective techniques for performing privacy-preserving pseudonym updates. In the case of non-anonymous LBSs, we believe that the location privacy protection will need to be guaranteed through policy-based solutions managed by LBS providers. Policy-based solutions require mobile clients to completely trust the LBS providers in order to use the services provided.

8 Related Work

Location Privacy and Anonymity

In the telecommunications domain, policy-based approaches have been proposed for protecting location privacy [9, 24]. Users may use policies to specify their privacy preferences. These policies specify what data about the user can be collected, when and for which purposes it can be used, and how and to whom it can be distributed. Mobile clients have to trust the LBSs that location information is adequately protected.

Another approach to location privacy is a k-anonymity based approach, which depersonalizes data through perturbation techniques before forwarding it to the LBS providers. Location k-anonymity is first studied by Gruteser and Grunwald [8]. 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 clients, which hinders the service quality for those mobile clients whose privacy requirements can be satisfied using smaller k values. Furthermore, this assumption is far from optimal, as mobile clients 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 framework for location k-anonymity captures the desired degree of privacy and service quality on per-user base, supporting mobile clients with diverse context-dependent location privacy requirements. Our message perturbation engine can anonymize a stream of incoming messages with different k anonymization constraints. Unlike earlier work [8], we do not assume knowledge of user positions at the anonymity server at all times. Our work assumes that the user positions are known at the anonymity server side only to the extent they can be deduced from the users' request messages. Our proposed location cloaking algorithms are effective in terms of the success rate of message perturbation and the amount of QoS loss due to location cloaking. They are also flexible in the sense that each user can spec-

ify a personalized k value, as well as spatial and temporal tolerance values at per-message granularity, to adjust the requested level of privacy protection and to bound the amount of loss in spatial resolution and the temporal delay introduced during message perturbation. The spatial and temporal tolerances in our model provide a flexible way of independently adjusting the level of spatial and temporal cloaking performed.

Anonymity Support in Databases

In the database community, there exists a large amount of literature on security control against disclosure of confidential information. Such disclosures may occur if through the answer to one or more queries an adversary can infer the exact value of or an accurate estimate of a *confidential attribute* of an individual. Privacy protection mechanisms suggested in the statistical databases literature can be classified under three general methods, namely *query restriction*, *data perturbation*, and *output perturbation*. In query restriction, the queries are evaluated against the original database, but the results are only reported if the queries meet certain requirements. There are many flavors of query restriction, like restricting the number of entities in the result set [25], controlling the overlap among successive queries [26], keeping up-to-date logs of all queries and checking for compromises whenever a new query is issued [27], and clustering individual entities in mutually exclusive subsets and restricting the queries to the statistical properties of these subsets [28]. In data perturbation, the database is perturbed and the queries are evaluated against the perturbed database. This is usually done by replacing the database with a sample of it [29], or by perturbing the values of the attributes in the database [30]. In output perturbation, the results to the queries are perturbed, whereas the original database is not. This is commonly achieved by sampling the query results [31] or by introducing a varying perturbation (not permanent) to the data that are used to compute the result of a given query [32].

Another piece of related research is computing over encrypted data values in data mining and database queries. One representative work in the recent years is the privacy-preserving indexing technique proposed by Hore, Mehrotra and Tsudik for supporting efficient query evaluation over encrypted data [33]. This work is based on the Database as a Service (DAS) model, where the service providers store encrypted data owned by the content provider in their servers and provide query services over the encrypted data. User queries are translated into two parts, a server-side query that works over the encrypted data, and a client-side query that does decryption and post processing over the results of the server query part. Although the motivation and the problems addressed in our paper is different, our work shares a common assumption with the DAS work in the sense that the content producer does not trust the service providers and thus provides privacy-preserving index instead of the actual data content in the DAS scenario, or the perturbed location data through location anonymizer in the anonymous LBS scenario as discussed in this paper.

Finally, Samarati and Sweeney have developed a k-anonymity model [12, 11, 10] for protecting data

privacy and a set of generalization and suppression techniques for safeguarding the anonymity of individuals whose information is recorded in database tables. Our work makes use of this basic idea of k-anonymity.

9 Conclusion

We proposed a personalized k-anonymity model for providing location privacy. Our model allows mobile clients to define and modify their location privacy specifications at the granularity of single messages, including the minimum anonymity level requirement, and the inaccuracy tolerances along the temporal and spatial dimensions. We developed an efficient message perturbation engine to implement this model. Our message perturbation engine can effectively anonymize messages sent by the mobile clients, in accordance with location k-anonymity, while satisfying the privacy and QoS requirements of the users. Several variations of spatio-temporal cloaking algorithms, collectively called $CliqueCloak$ algorithms, are proposed as the core algorithms of the perturbation engine. We experimentally studied the behavior of our algorithms under various conditions, using realistic workloads synthetically generated from real road maps and traffic volume data. Our work continues along a number of directions, including the investigation of more optimal algorithms under the proposed framework, the study of QoS characteristics of real-world LBS applications, and how QoS requirements impact the maximum achievable anonymity level with reasonable success rate.

References

[1] G. Orwell, *1984*. Everyman's Library, November 1992.

[2] D. W. Gage. (2004, January) Lifelog. [Online]. Available: http://www.darpa.mil/ipto/Programs/lifelog/

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

[4] NextBus Inc. (2004, January) Nextbus web page. [Online]. Available: http://www.nextbus.com/

[5] G. Abowd, C. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton, "CyberGuide: A mobile context-aware tour guide," *ACM Wireless Networks*, vol. 3, no. 5, pp. 421–433, 1997.

[6] J. Reed, K. Krizman, B. Woerner, and T. Rappaport, "Challenges and progress in meeting the E-911 requirement for location service," *IEEE Personal Communications Magazine*, vol. 5, no. 3, pp. 30–37, 1998.

[7] R. Want, A. Hopper, V. Falcao, and J. Gibbons, "The active badge location system," *ACM Transactions on Information Systems*, vol. 10, no. 1, pp. 91–102, 1992.

[8] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *ACM International Conference on Mobile Systems, Applications, and Services, MobiSys*, 2003.

[9] 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.

[10] P. Samarati and L. Sweeney, "Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression," in *IEEE Symposium on Research in Security and Privacy*, 1998.

[11] P. Samarati, "Protecting respondent's privacy in microdata release," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 6, pp. 1010–1027, 2001.

[12] L. Sweeney, "k-Anonymity: A model for protecting privacy," *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[13] B. Gedik and L. Liu, "A customizable k-anonymity model for protecting location privacy," in *IEEE International Conference on Distributed Computing Systems, ICDCS*, 2005, pp. 620–629.

[14] M. Reiter and A. D. Rubin, "Crowds: Anonymity for web transactions," *ACM Transactions on Information and System Security*, vol. 1, no. 1, pp. 66–92, 1998.

[15] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing for anonymous and private internet connections," *Communications of the ACM*, vol. 42, no. 2, pp. 9–41, 1999.

[16] R. Cheng, D. V. Kalashnikov, and S. Prabhakar, "Evaluating probabilistic queries over imprecise data," in *ACM International Conference on Management of Data, SIGMOD*, 2003, pp. 551–562.

[17] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, "R*-Tree: An efficient and robust access method for points and rectangles," in *ACM International Conference on Management of Data, SIGMOD*, 1990, pp. 322–331.

[18] A. Meyerson and R. Williams, "On the complexity of optimal k-anonymity," in *ACM Symposium on Principles of Database Systems, PODS*, 2004, pp. 223–228.

[19] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu, "Anonymizing tables," in *International Conference on Database Theory, ICDT*, 2005, pp. 246–258.

[20] U.S. Department of the Interior. (2003, November) U.S. geological survey web page. [Online]. Available: http://www.usgs.gov/

[21] Mid-Continent Mapping Center. (2003, November) Spatial data transfer format. [Online]. Available: http://mcmcweb.er.usgs.gov/sdts/

[22] SVG Working Group. (2003, November) Scalable vector graphics format. [Online]. Available: http://www.w3.org/Graphics/SVG/

[23] Global Mapper Software LLC. (2003, November) Global mapper web page. [Online]. Available: http://www.globalmapper.com/

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

[25] A. D. Friedman and L. J. Hoffman, "Towards a fail-safe approach to secure databases," in *IEEE Symposium on Security and Privacy*, 1980.

[26] D. Dobkin, A. K. Jones, and R. J. Lipton, "Secure databases: Protection against user influence," *ACM Transactions on Database Systems*, vol. 4, no. 1, pp. 97–106, 1979.

[27] F. Y. Chin and G. Ozsoyoglu, "Auditing and inference control in statistical databases," *IEEE Transactions on Software Engineering*, vol. 8, no. 6, pp. 574–582, 1982.

[28] J. Schlorer, "Information loss in partitioned statistical databases," *The Computer Journal*, vol. 26, no. 3, pp. 218–223, 1983.

[29] S. P. Reiss, "Practical data swapping: The first steps," *ACM Transactions on Database Systems*, vol. 9, no. 1, pp. 20–37, 1984.

[30] J. F. Traub, Y. Yemini, and H. Wozniakowski, "The statistical security of a statistical database," *ACM Transactions on Database Systems*, vol. 9, no. 4, pp. 672–679, 1984.

[31] D. E. Denning, "Secure statistical databases with random sample queries," *ACM Transactions on Database Systems*, vol. 5, no. 3, pp. 291–315, 1980.

[32] L. L. Beck, "A security mechanism for statistical databases," *ACM Transactions on Database Systems*, vol. 5, no. 3, pp. 316–338, 1980.

[33] B. Hore, S. Mehrotra, and G. Tsudik, "A privacy-preserving index for range queries," in *International Conference on Very Large Data Bases, VLDB*, 2004, pp. 720–731.