A Customizable k-Anonymity Model for Protecting Location Privacy

Buğra Gedik
College of Computing
Georgia Institute of Technology
bgedik@cc.gatech.edu

Ling Liu
College of Computing
Georgia Institute of Technology
lingliu@cc.gatech.edu

Abstract

Continued advances in mobile networks and positioning technologies have created a strong market push for location-based services (LBSs). Examples include location-aware emergency services, location based service advertisement, and location sensitive billing. One of the big challenges in wide deployment of LBS systems is the privacy-preserving management of location-based data. Without safeguards, extensive deployment of location based services endangers location privacy of mobile users and exhibits significant vulnerabilities for abuse.

In this paper, we describe a customizable k-anonymity model for protecting privacy of location data. Our model has two unique features. First, we provide a customizable framework to support k-anonymity with variable k, allowing a wide range of users to benefit from the location privacy protection with personalized privacy requirements. Second, we design and develop a novel spatio-temporal cloaking algorithm, called $CliqueCloak$, which provides location k-anonymity *for mobile users of a LBS provider. The cloaking algorithm is run by the location protection broker on a trusted server, which anonymizes messages from the mobile nodes by* cloaking *the location information contained in the messages to reduce or avoid privacy threats before forwarding them to the LBS provider(s). Our model enables each message sent from a mobile node to specify the desired level of anonymity as well as the maximum temporal and spatial tolerances for maintaining the required anonymity. We study the effectiveness of the cloaking algorithm under various conditions using realistic location data synthetically generated using real road maps and traffic volume data. Our experiments show that the location k-anonymity model with multi-dimensional cloaking and tunable k parameter can achieve high guarantee of k anonymity and high resilience to location privacy threats without significant performance penalty.*

1 Introduction

In his famous novel *1984* [15], George Orwell has 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 [12], 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 may provide.

According to the report by Computer Science and Telecommunications Board on *IT Roadmap to a Geospatial Future* [5], location based services 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 [14] service provides location based transportation data, the CyberGuide [1] project investigates context-aware location-based electronic guide assistants, and FCC's Phase II E911 requires wireless carriers to provide precise location information within 50 to 100 meters in most cases for emergency purposes.

Although location privacy is widely recognized as a significant concern for the pervasive use of LBSs, there has been little work in this area until recently. The concept of *location k-anonymity* is first introduced in [10] as a natural extension of the k-anonymity model for relational data records [20], and it deals with the anonymous release of real-time location data to LBSs with certain anonymity guarantees.

1.1 General k-anonymity

k-anonymity is a model that addresses the question, "How can a data holder release a version of its private data with scientific guarantees that the individuals who are the subjects of the data cannot be re-identified while the data remain practically useful?" [20]. For instance, a medical institution may want to release a table of medical records. Even though the names of the individuals can be replaced with dummy identifiers, some set of attributes (so called the quasi-identifier) can leak confidential information. For instance, the birth date, zip code and the gender attributes in the disclosed table can uniquely determine an individual. Joining such a table with some other publicly available information source, like a voters list table, which consists of records containing the attributes that make up the quasi-identifier as well as the identities of individuals, the medical information can be easily linked to individuals. k-anonymity prevents such a privacy breach by ensuring that each individual record can only be released if there is at least $k-1$ other (distinct) individuals whose associated records are indistinguishable from the former in terms of their quasi-identifier values.

1.2 Location k-anonymity

In the context of LBSs and mobile users, location k-anonymity demands that location information contained in a message sent from a mobile user to a LBS should be indistinguishable from at least $k-1$ other messages from different mobile nodes [10]. Generally speaking, anonymity in LBSs depends on the trustworthiness of the entities involved and needs to be addressed at multiple layers in the network stack. In this paper we tackle the problem of location k anonymity at the application layer by giving LBS providers access to anonymous location information.

The location protection algorithm uses *spatio-temporal cloaking* to transform each original message from a mobile node into a privacy protected message with the k-anonymity guarantee.

As shown in [10], location k-anonymity can be used to prevent attacks such as *Restricted Space Identification* and *Observation Identification*. The former reveals identity by relating a known location-person association to a message, whereas the later reveals identity by joining location information from external observation to a message.

1.3 Contributions and Scope of the Paper

In this paper, we describe a customizable k-anonymity model for protecting privacy of location data. *Conceptually*, instead of using a uniformed k for all messages [10], we provide efficient algorithms and system level facilities to support customizable k at per-message level. Each message can specify a different k anonymity value based on its specific privacy requirement. Furthermore, each message can specify its preferred *spatial and temporal tolerance* level in order to maintain the desired k anonymity property. We call such tolerance specification and preference of k value, the anonymity constraint of the message. By providing a customizable framework to support k-anonymity with variable anonymity constraints, we allow a wide range of users to benefit from the location privacy protection with personalized privacy requirements.

Algorithmically, in order to anonymize a message originated by a mobile user, the spatial position information of the mobile user contained in the message is converted into a two dimensional spatial box, and the timestamp of the message is converted into a temporal interval by the cloaking algorithm according to the anonymity constraint specification of the message. The resulting three dimensional box, called the *spatio-temporal cloaking box* of the message, indicates the acceptable decrease of the spatial resolution of location information and the tolerable delay of the message in the effort to meet the specified k-anonymity requirement of the message, namely we need to find a minimal spatio-temporal cloaking box under the specified spatial and temporal tolerance constraints, such that there are at least $k-1$ other messages from other mobile users with the same spatio-temporal cloaking box. We call the transformed message the k-anonymized message.

There are a number of challenges for supporting a customizable k-anonymity model. The first key challenge is to design a spatio-temporal cloaking algorithm that is capable of handling variable k anonymity requirements. The second key challenge is to find the minimal spatio-temporal cloaking box for each k-anonymized message such that k-anonymity can be satisfied with higher (close to optimal) spatio-temporal resolution than the acceptable spatio-temporal tolerance specified by the anonymity constraint of the original message.

We develop a novel spatio-temporal cloaking algorithm, called $CliqueCloak$, to implement quality and performance optimizations for the spatio-temporal cloaking. To our knowledge, the proposed *ClicqueCloaking* algorithm has two distinct characteristics. First, it supports location k-anonymity with customizable k as well as spatial and temporal tolerance constraints. Second, it can continuously process a stream of messages for location k-anonymity. We conduct a series of experimental evaluations on the effectiveness of the cloaking algorithm under various conditions using realistic location data synthetically generated using real road maps and traffic volume data. Our experiments show that the location k-anonymity model with multi-dimensional cloaking and tunable k parameter can achieve high guarantee of k anonymity and high resilience to location privacy threats without significant performance penalty.

2 Related Work

Location Anonymity. The work presented in this paper is highly inspired by [10]. The main contribution of [10] is to introduce the concept of location k-anonymity in the context of LBSs, the quadtree-based algorithm for performing spatial and temporal cloaking, and the analysis of privacy threats through location information. However, the location k-anonymity model proposed in [10] suffers from several assumptions. First, it assumes a system-wide static k value for all messages, which is unrealistic in practice as mobile users tend to have varying privacy protection requirements under different contexts and on different subjects. Second, the quadtree-based algorithm anonymizes the messages by dividing the quadtree cells until the number of messages in each cell falls below k and returning the previous quadrant for each cell as the spatial cloaking box of the messages under that cell. This approach fails to provide any quality of service guarantees with respect to the sizes of the cloaking boxes produced and is highly dependent on the existence of a system wide static k value. It is also unclear how the quadtree-based algorithm can be extended to work with a stream of incoming messages. In comparison, our customizable framework for location k-anonymity captures the desired degree of anonymity on per-message base, supporting mobile users with diverse context-dependent location privacy requirements. Our $CliqueCloak$ algorithm is efficient and can anonymize a stream of messages, where each message can specify an independent k value, as well as customized spatial and temporal tolerance values to restrict the size of the cloaking box produced by the cloaking algorithm.

Anonymity Support in Databases. Samarati and Sweeney have developed a k-anonymity model [20] for protecting data privacy and a set of generalization and suppression techniques [21] for safeguarding the anonymity of individuals whose information is recorded in database tables. There exists large amount of work in the subject of statistical databases, with regard to providing security to statistical databases 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.

Based on the survey by Adam and Wortman [2], 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 [8], controlling the overlap among the successive queries from each user [7], keeping up-to-date logs of all queries made by each user and constantly checking for possible compromise whenever a new query is issued [4], and clustering individual entities in a number of mutually exclusive subsets and restricting the queries to the statistical properties of these subsets [18]. 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 [16, 11], or by perturbing the values of the attributes in the database [22]. 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 [6] or by introducing a varying perturbation (not permanent) to the data that are used to compute the result of a given query [3].

Our work, although it is in a different context, can be viewed as perturbation of the attribute values of the messages sent by mobile nodes communicating with external LBS providers through a trusted anonymity server.

3 Customizable k-anonymity Model

We assume a model in which mobile nodes communicate with external LBS providers through one or a collection of central anonymity servers located at trusted computing bases. The mobile nodes initialize the communication with the anonymity server through an authenticated and encrypted connection. Each message from a mobile node contains location information regarding the mobile node and a timestamp, in addition to service specific information. Upon receiving a message from a mobile node, the anonymity server decrypts the message and removes any identifiers (such as IP addresses) and perturbs the position data through spatio-temporal cloaking according to our $CliqueCloak$ algorithm, and then exports the anonymized message to the LBS provider.

3.1 k-Anonymous Location Information

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

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

$$m_s \in S \colon \langle u_{id}, r_{no}, \{t, x, y\}, k, \{d_t, d_x, d_y\}, C\rangle$$

Messages are uniquely identifiable by the sender's identifier, message reference number pairs, (u_{id}, r_{no}), within the set S. Messages from the same mobile node have same sender identifiers but different reference numbers. In a received message, x, y, and t together form the three dimensional *spatio-temporal point* of the message, denoted as $P(m_s)$. The coordinate (x, y) refers to the spatial position of the mobile node in the two dimensional space (i.e., x-axis and y-axis), and the timestamp t refers to the time point at which the mobile node was present at that position (temporal dimension: t-axis of the message). The k value of the message specifies the desired minimum *anonymity level*. A value of $k = 1$ means that anonymity is not required for the message. A value of $k > 1$ means that the transformed message will be assigned a spatio-temporal cloaking box that is indistinguishable from at least $k-1$ other transformed messages, each from a different mobile node. Larger k values imply higher degree of anonymity. The d_t value of the message represents the temporal tolerance specified by the user. It means that, the transformed 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. 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 *anonymity 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 transformed (*anonymized*) messages as T. We formally define the messages in the set T as follows:

$$m_t \in T \colon \langle u_{id}, r_{no}, \{X \colon [x_s, x_e], Y \colon [y_s, y_e], I \colon [t_s, t_e]\}, C\rangle$$

For each message m_s in S, there exists at most one corresponding message m_t in T. We call the message m_t, the *transformed format* of message m_s, denoted as $m_t = R(m_s)$. Concretely, if $m_t = R(m_s)$, then $m_t.u_{id} = m_s.u_{id}$ and $m_t.r_{no} = m_s.r_{no}$. The (u_{id}, r_{no}) fields of a message in T should be replaced with a dummy identifier before the message can be safely exported to the LBS provider. In a transformed message, $X \colon [x_s, x_e]$ denotes the extent of the *spatio-temporal cloaking box* of the transformed 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 transformed message as $B_{cl}(m_t)$ and define it as $(m_t.X, m_t.Y, m_t.I)$. The field C in m_t denotes the message content. We describe how the fields of a transformed message in set T relates to its counterpart in set S, in the following subsection.

3.2 k-anonymity Constraints

The following basic properties must hold between a raw message m_s in S and its transformed format m_t in T:

- *Spatial Containment*: $m_s.x \in m_t.X$, $m_s.y \in m_t.Y$
- *Spatial Resolution*: $m_t.X \subset \Phi(m_s.x, m_s.d_x)$ and $m_t.Y \subset \Phi(m_s.y, m_s.d_y)$
- *Temporal Containment*: $m_s.t \in m_t.I$
- *Temporal Resolution*: $m_t.I \subset \Phi(m_s.t, m_s.d_t)$
- *Content Preservation*: $m_s.C = m_t.C$

Spatial containment and temporal containment requirements simply state that the cloaking box of the transformed message, $B_{cl}(m_t)$, should contain the spatio-temporal point $P(m_s)$ 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 transformed message should be contained within the interval defined by the tolerance value specified in the original message. This is equal to stating that the cloaking box of the transformed message should be contained within the constraint box of the original message, i.e. $B_{cl}(m_t) \subset B_{cn}(m_s)$. Content preservation is a trivial property, which ensures that the message content is copied as it is, from the original message to the transformed message.

We formally capture the essence of the location k-anonymity by the following requirement, which states that, for a message m_s in S and its transformed 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 transformed message m_t, there has to be at least $m_s.k - 1$ other transformed messages with the same spatio-temporal cloaking box, each from a different mobile node. A key challenge for the spatio-temporal cloaking algorithm is to find a set of messages within a minimal spatio-temporal cloaking box that satisfies the above conditions.

3.3 Evaluation Metrics

To evaluate the effectiveness of the proposed location k-anonymity model, an important measure is the *success rate*. Concretely, the primary goal of the cloaking algorithm is to maximize the number of messages transformed successfully in accordance with location k-anonymity constraints. In other words, we want to maximize $|T|$. The **success rate** can be defined as the percentage of messages that are successfully anonymized (transformed), i.e. $100 * |T|/|S|$.

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

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

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

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

Message processing time is a measure of the running time performance of the cloaking algorithm. 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 reported in Section 6, we use the average CPU time needed to process 10^3 messages as the message processing time.

3.4 The Clique-Cloak Theorem

A main technical challenge for developing an efficient cloaking algorithm is to find a set of messages and to assign the smallest possible spatio-temporal cloaking box to these messages, such that the k-anonymity requirements are satisfied for all messages in the set.

Consider a set $M \subset S$ of messages that can be anonymized together. This implies that, these messages are from different mobile nodes and the highest k value they have is at most equal to the size of the set M. Then the best strategy, in terms of minimizing the size of the cloaking box of the transformed messages, is to use the minimum bounding rectangle (MBR) of the spatio-temporal points of the messages in M as the cloaking box of the transformed messages. This is because, any cloaking box has to satisfy the spatial and temporal containment requirements for all of the messages in M, thus should cover the MBR. 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 $\{P(m_s)|m_s \in M\}$. Since the messages in M can be anonymized together, the cloaking box assigned to their transformed forms should be covered by the constraint boxes of all messages in M, according to the temporal and spatial resolution requirements. When the cloaking box is selected to be $B_m(M)$, the latter is equivalent to stating that each message's spatio-temporal point should be contained in the constraint boxes of all other messages in M. These observations naturally translate the problem of finding a set of messages that can be anonymized together, into the graph theoretical problem of finding cliques (with certain properties) in the following graph:

Let $G(S, E)$ be an undirected graph where S is the set of vertices 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) $P(m_{s_i}) \in B_{cn}(m_{s_j})$, (ii) $P(m_{s_j}) \in B_{cn}(m_{s_i})$, (iii) $m_{s_i}.u_{id} \neq m_{s_j}.u_{id}$. We call this graph the *constraint graph*. The conditions (i), (ii), and (iii) together state that, two messages are connected in the constraint graph if and only if they originate from different mobile nodes and their spatio-temporal points are contained in each other's constraint boxes defined by their tolerance values.

Theorem 1 *Clique-Cloak Theorem*
Let $M = \{m_{s_1}, m_{s_2}, \ldots, m_{s_l}\} \subset S$ 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 transformed format of m_{s_i}, i.e. $m_{t_i} = R(m_{s_i})$, if and only if the set of messages M form an l-clique in $G(S, E)$ 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 easily satisfied as we have $\forall_{1\leq i\leq l}, P(m_{s_i}) \subset B_m(M) = B_{cl}(m_{t_i})$ from definition of an MBR. It is easy to prove k-anonymity, 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 $\forall_{1\leq i\neq j\leq l}, m_{t_i}.u_{id} \neq m_{t_j}.u_{id}$. 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 $\forall_{1\leq i\leq l}, \{x_{min}, x_{max}\} \subset \Phi(m_{s_i}.x, m_{s_i}.d_x)$ and thus $\forall_{1\leq i\leq l}, [x_{min}, x_{max}] \subset \Phi(m_{s_i}.x, m_{s_i}.d_x)$ 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 $\forall_{1\leq i\leq l}, B_m(M) \subset B_{cl}(m_{s_i})$.

Now we show that the right hand side holds if we assume that the left hand side holds. This part is trivial. 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 $\forall_{1\leq i\leq l}, P(m_{s_i}) \in B_m(M)$ and from spatial and temporal resolution constraints we have $\forall_{1\leq i\leq l}, B_m(M) \subset B_{cn}(m_{s_i})$. These two imply $\forall_{1\leq i,j\leq l}, P(m_{s_i}) \in B_{cn}(m_{s_j})$ 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. □

4 The $CliqueCloak$ Algorithm

We first explain the crux of the $CliqueCloak$ algorithm by illustrating the use of Clique-Cloak theorem. Then we describe the main data structures used to improve the efficiency of the algorithm. We end this section with a pseudo code of the algorithm elaborating on important details.

4.1 Overview

The algorithm works by progressively constructing the constraint graph, finding those cliques that satisfy the anonymity constraints of all messages included in the clique, generating one MBR for the messages in each of such cliques, and removing cliques from the graph, all based on the Clique-Cloak theorem. Messages that can not be anonymized until their *deadline* are dropped. The deadline of a message is the high point along the temporal dimension in its spatio-temporal constraint box.

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

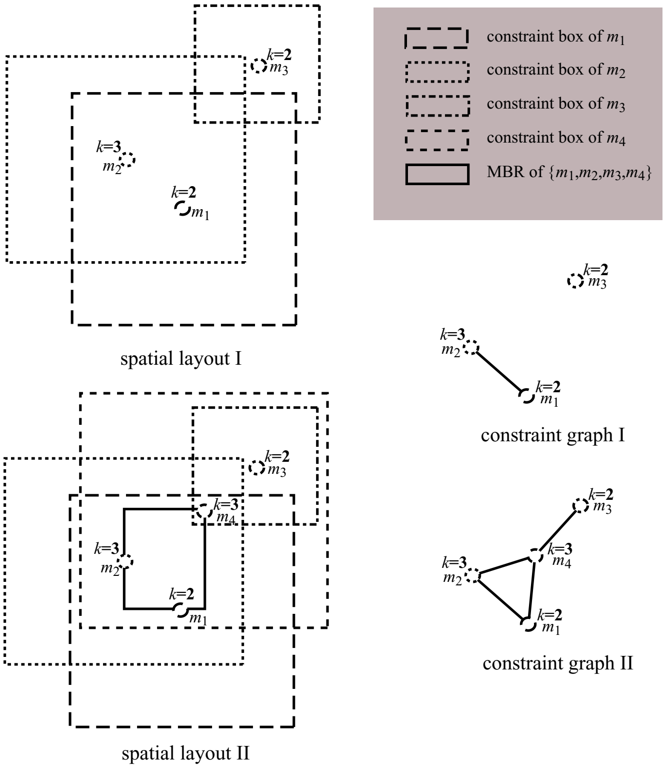


Figure 1: Illustration of the Clique-Cloak Algorithm

Although in the described example we have found a single clique immediately after m_4 was received, we could have had cliques of different sizes to choose from. For instance, if $m_4.k$ was 2, then $\{m_3, m_4\}$ would have also formed a valid clique according to the Clique-Cloak theorem. We address the questions of *what* kind of cliques to search and *when* to search for such cliques, in more detail in Section 5.

4.2 Data Structures

We briefly describe the four main data structures that are used in the $CliqueCloak$ algorithm.

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

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

- *Constraint Graph*, G_m: The constraint graph is a dynamic in-memory graph, which contains the messages that are not yet anonymized and not yet dropped due to expiration. The structure of the constraint graph is already defined in Section 3.4. The multi-dimensional index I_m is mainly used to speedup the maintenance of the constraint graph G_m, which is updated when new messages arrive or when messages get anonymized or expired.

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

[*] It is memory-wise more efficient to store only identifiers of messages as data in I_m, G_m, and H_m; and to keep a hash table to access real message content when needed. We do not reflect this level of detail in our description.

Algorithm 1 Clique-Cloak Algorithm

```
 1: {$G_m$ is the constraint graph}
 2: {$Q_m$ is the queue of incoming messages}
 3: {$I_m$ is the index on spatio-temporal points of messages}
 4: {$H_m$ is the min-heap consisting of message deadlines}
 5: while TRUE do
 6:   if $Q_m \neq \emptyset$ then
 7:     $m_{s_c} \leftarrow$ Pop the first item in $Q_m$
 8:     Add $m_{s_c}$ into $I_m$ with $P(m_{s_c})$
 9:     Add $m_{s_c}$ into $H_m$ with $(m_{s_c}.t + m_{s_c}.d_t)$
10:     Add the message $m_{s_c}$ into $G_m$ as a node
11:     $N \leftarrow$ Range search $I_m$ using $B_{cn}(m_{s_c})$
12:     for all $m_s \in N, m_s \neq m_{s_c}$ do
13:       if $P(m_{s_c}) \in B_{cn}(m_s)$ then
14:         Add the edge $(m_{s_c}, m_s)$ into $G_m$
15:       end if
16:     end for
17:     { Find set of messages $M$ in $G_m$ s.t. $m_{s_c} \in M$, $|M| = m_{s_c}.k$, $\forall_{m_s \in M}, m_s.k \leq |M|$, and $M$ forms a clique in $G_m$ }
18:     $M \leftarrow$ local-$k$_Search($m_{s_c}.k, m_{s_c}, G_m$)
19:     if $M \neq \emptyset$ then
20:       for all $m_s$ in $M$ do
21:         Output transformed message $m_t \leftarrow \langle m_s.u_{id}, m_s.r_{no}, B_m(M), m_s.C \rangle$
22:         Remove the message $m_s$ from $G_m$
23:         Remove the message $m_s$ from $I_m$
24:         Pop the topmost element in $H_m$
25:       end for
26:     end if
27:   end if
28:   while TRUE do
29:     $m_s \leftarrow$ Topmost item in $H_m$
30:     if $m_s.t + m_s.d_t < now$ then
31:       Remove the message $m_s$ from $G_m$
32:       Remove the message $m_s$ from $I_m$
33:       Pop the topmost element in $H_m$
34:     else
35:       break
36:     end if
37:   end while
38: end while
```

Algorithm 2 local-k_Search(k, m_{s_c}, G_m)

```
 1: $U \leftarrow \{m_s | m_s \in nbr(m_{s_c})$ and $m_s.k \leq k\}$
 2: if $|U| < k - 1$ then
 3:   return $\emptyset$
 4: end if
 5: $l \leftarrow 0$
 6: while $l \neq |U|$ do
 7:   $l \leftarrow |U|$
 8:   for all $m_s \in U$ do
 9:     if $(|nbr(m_s) \cap U| < k - 2)$ then
10:       $U \leftarrow U \backslash \{m_s\}$
11:     end if
12:   end for
13: end while
14: Find any subset $M \subset U$, s.t. $|M| = k-1$ and $M \cup \{m_{s_c}\}$ forms a clique
15: return $M$
```

4.3 Algorithmic Details

We assume that the mobile nodes do not send new messages unless their previous messages are anonymized or explicitly dropped by the cloaking algorithm due to expiration. The pseudo code of the $CliqueCloak$ algorithm is given in Algorithm 1. The algorithm works by continuously popping messages from the queue and processing them for k-anonymity in four steps.

The first step is to update the data structures with the new message, and to integrate the new message into the constraint graph. When a message, m_{s_c} is popped from the queue, it is inserted into the index I_m using $P(m_{s_c})$, 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 with vertex m_{s_c} are constructed in the constraint graph G_m by searching the multi-dimensional index I_m using the spatio-temporal constraint box of the message, i.e. $B_{cn}(m_{s_c})$, as the range search condition. The messages whose spatio-temporal points are contained in $B_{cn}(m_{s_c})$ are candidates for being m_{s_c}'s neighbors in the constraint graph. These messages (denoted as N in the pseudo code) are filtered based on whether their spatio-temporal constraint boxes contain $P(m_{s_c})$. The ones that pass the filtering step (excluding m_{s_c} itself) become neighbors of m_{s_c} in the constraint graph. See lines 7-16 in the pseudo code.

The second step is to apply the *local-k search* algorithm to find a clique in the constraint graph. In local-k search, we try to find a clique of size $m_{s_c}.k$ that includes the message m_{s_c}. 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 18) with parameter k set to $m_{s_c}.k$. Before beginning the search, a set $U \subset nbr(m_{s_c})$ is constructed such that for each element $m_s \in U$, we have $m_s.k \leq k$ (line 4). 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. Once this is done, the set U is iteratively filtered until there is no change (lines 5-13). At each filtering step, each message $m_s \in U$ is checked whether it has at least $k-2$ neighbors in U. If not, the message cannot be part of a clique that contains m_{s_c} and has size k. 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. Although the general problem of finding cliques in a graph is *NP-Complete*, up to values of $k = 10$, (where $k = 5$ is considered as a good level of anonymity [10]) the search step does not form a bottleneck.

The third step is to generate the k-anonymized messages to be forwarded to the external LBS providers. If a clique can be found, the messages in the clique (denoted as M in the pseudo code) are anonymized by assigning the MBR of the spatio-temporal points of the messages in the clique, $B_m(M)$, 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 19-27. In case a clique cannot be found, the message stays inside the system. It may be later picked up and anonymized during the processing of a new message or may be dropped due to expiration. We discuss some more advanced ways of searching cliques in the next section.

The fourth step is to clean the expiration heap. After the processing of each message, we check the expiration heap for any messages that has expired. The message on top of the expiration heap is checked and if its deadline has passed, it is removed. 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 28-37 of the pseudo code deals with message expiration.

5 Alternative $CliqueCloak$ Algorithms

In this section, we describe an improvement to the clique search part of the algorithm, which makes use of a different criterion in determining what kinds of cliques are searched. We further discuss a variation of the basic algorithm, that uses a deferred policy with regard to when cliques are searched.

5.1 Nbr-k Search

When searching for a clique in the constraint graph, it is essential to ensure that the newly received message, say m_{s_c}, should be included in the clique. If there is a new clique formed due to the entrance of m_{s_c} into the graph, it must contain m_{s_c}. However, instead of searching a clique with size $m_{s_c}.k$, we can try to find out the biggest clique that includes $m_{s_c}.k$, of course making sure that all messages inside the clique has a k value at most equal to the size of the clique. There are two strong motivations behind the approach. First, by anonymizing a larger number of messages at once, it can provide higher success rate which also results in better performance, as the graph will become less crowded. Second, by anonymizing messages that have smaller k's together with messages that have larger k's, it can provide higher relative level of anonymity. *Nbr-k* search takes the latter approach. Its pseudo code is given in Algorithm 3 as the nbr-k_Search function.

Nbr-k search first collects the set of k values the new message m_{s_c} and its neighbors $nbr(m_{s_c})$ have, denoted as L in the pseudo code. The k values in L are considered in decreasing order until a clique is found or k becomes smaller than $m_{s_c}.k$ (in which case the search returns empty set). For each $k \in L$ considered, a clique of size k is searched by calling the local-k_Search function with appropriate parameters (see line 9). If such a clique can be found, the messages within the clique are returned. To integrate nbr-k search into the $CliqueCloak$ algorithm, we can simply replace line 18 of the Algorithm 1 with the call to nbr-k_Search function.

Algorithm 3 nbr-k_Search(m_{s_c}, G_m)

```
 1: if $|nbr(m_{s_c})| < k - 1$ then
 2:     return $\emptyset$
 3: end if
 4: $L \leftarrow \{m_s.k | m_s = m_{s_c} \vee m_s \in nbr(m_{s_c})\}$
 5: for all distinct $k \in L$ in decreasing order do
 6:     if $k < m_{s_c}.k$ then
 7:         return $\emptyset$
 8:     end if
 9:     $M \leftarrow$ local-$k$_Search($k, m_{s_c}, G_m$)
10:     if $M \neq \emptyset$ then
11:         return $M$
12:     end if
13: end for
14: return $\emptyset$
```

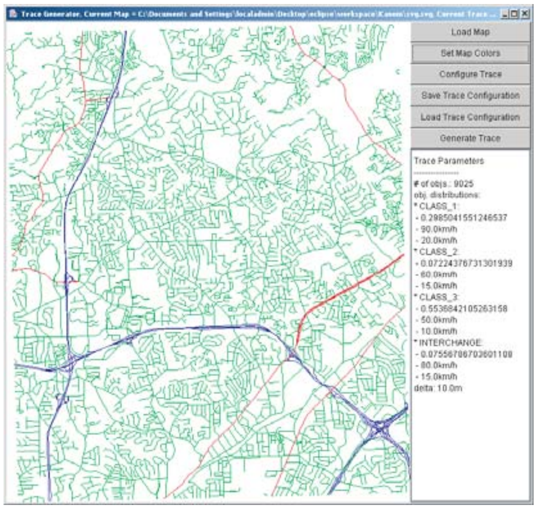
5.2 Deferred $CliqueCloak$ Algorithm

So far we have only considered searching for cliques when each new message arrives. This may result in many unsuccessful searches, thus deteriorate the performance in terms of average time to process a message. Instead of immediately searching for a clique for each message, we can defer this processing. If a deferred message is not already anonymized (together with other messages) at the time of its expiration, we can search for a clique in order to anonymize it before it expires. However, this latter approach will definitely decrease the relative temporal resolution (close to 1). To overcome this, we can only perform the clique search phase for a new message m_{s_c}, if the number of neighbors it has at its arrival is larger than or equal to $\alpha * m_{s_c}.k$. Here, $\alpha \geq 1$ is a system parameter that adjusts the amount of messages for which the clique search is deferred. Smaller values pushes the algorithm toward immediate processing. We name this variation of the algorithm as $Deferred\ CliqueCloak$ and the original algorithm as $Immediate\ CliqueCloak$. Deferred approach is expected to decrease the number of clique searches at the cost of making the data structures more crowded. As a result, its benefit in terms of performance with regard to message processing time is not clear when the required anonymity levels (k values) are not too high (which is the case in this work). However, it can improve performance when clique searches dominate the running time.

6 Experiments

In this section, we present a set of experiments that demonstrates the performance of the $CliqueCloak$ algorithm under different settings with regard to various metrics introduced in Section 3.3. We divided the experiments into two, namely *success rate* and *spatial/temporal resolution*.

In relation with success rate, we look into five different issues: (1) the effect of nbr-k and local-k on success rate, (2) the effect of nbr-k and local-k on relative anonymity levels and its association to success rate, (3) the effect of immediate and deferred processing on success rate and average message pro-


Trace Generator, Current Map = C:\Documents and Settings\localadmin\Desktop\eclipse\workspace\Kanon\svg.svg, Current Trace ...
Load Map
Set Map Colors
Configure Trace
Save Trace Configuration
Load Trace Configuration
Generate Trace

Trace Parameters

# of objs.: 9025
obj. distributions:
* CLASS_1:
- 0.2985041551246537
- 90.0km/h
- 20.0km/h
* CLASS_2:
- 0.07224376731301939
- 60.0km/h
- 15.0km/h
* CLASS_3:
- 0.5536842105263158
- 50.0km/h
- 10.0km/h
* INTERCHANGE:
- 0.07556786703601108
- 80.0km/h
- 15.0km/h
delta: 10.0m

Figure 2: The trace generator

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

Table 1: Message generation parameters

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

Table 2: Car movement parameters

cessing time, (4) the effect of message arrival rate and average temporal/spatial tolerances of messages on success rate, and (5) the effect of variance in temporal/spatial tolerances of messages on success rate. With regard to spatial/temporal resolution, we look into two issues: (1) relative spatial and relative temporal resolution distributions of anonymized messages, and (2) the effect of spatial and temporal tolerances on relative spatial and relative temporal resolution distributions of anonymized messages. Before presenting our experimental results, we first describe the trace generator used to generate realistic traces that are employed in the experiments and the details of our experimental setup.

We have developed a trace generator (shown in Figure 2), that simulates cars moving on roads and generates requests using the position information from the simulation. The trace generator loads real-world road data, available from National Mapping Division of the United States Geological Survey (USGS) [13] in SDTS [19] format. We use transportation layer of 1:24K Digital Line Graphs (DLGs) as road data. We convert the graphs into Scalable Vector Graphic [17] format using the Global Mapper [9] software and use them as input to our trace generator. We extract three types of roads from the trace graph, class 1 (expressway), class 2 (arterial), and class 3 (collector). The generator uses real traffic volume data to calculate the total number of cars on each road type, as described by [10]. Once the number of cars on each type of road is determined, they are randomly placed into the graph and the simulation begins. Cars move on the roads and take other roads when they reach joints. The simulator tries to keep the fraction of cars on each type of road constant as time progresses. The cars change their speeds at each joint based on a normal distribution whose parameters are also input to the trace generator.

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

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

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

6.1 Success Rate

Figure 3 shows the success rate for nbr-k and local-k approaches. The success rate is shown (on y-axis) for different groups of messages, each group representing messages with a certain k value (on x-axis). The two leftmost bars show the success rate for all of the messages. The wider bars show the actual success rate provided by the ClickCloak algorithm. The thinner bars represent a lower bound on the percentage of messages that cannot be anonymized no matter what algorithm is used. This lower bound is calculated as follows. For a message

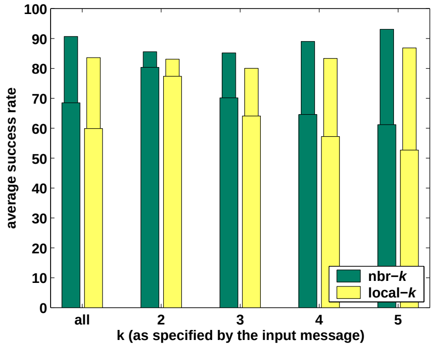


Figure 3: Success rates for different k values

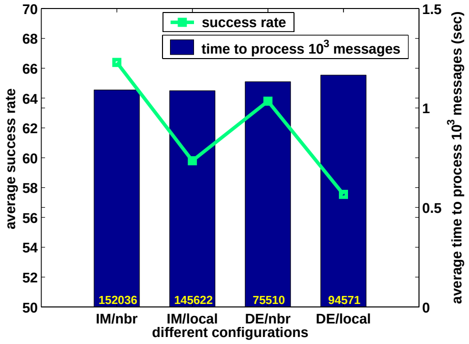


Figure 5: Message processing time and success rate of different approaches

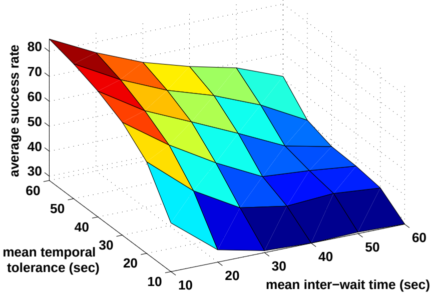


Figure 7: Success rate with respect to temporal tolerance with different inter-wait times

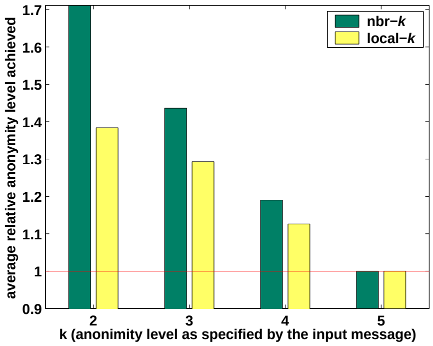


Figure 4: Relative anonymity levels for different k values

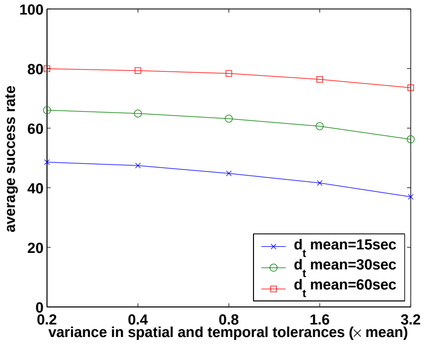


Figure 6: Success rate as a function of variances in spatial and temporal tolerances

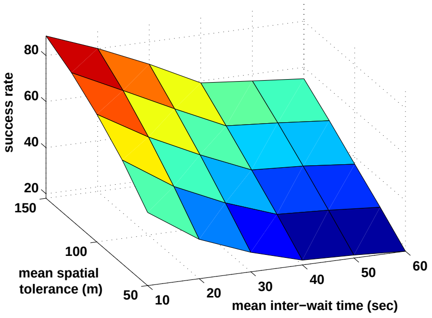


Figure 8: Success rate with respect to spatial tolerance with different inter-wait times

m_s, if the set $U = \{m_{s_i} | m_{s_i} \in S \wedge P(m_{s_i}) \in B_{cn}(m_s)\}$ has size less than $m_s.k$, the message cannot be anonymized. This is because, the total number of messages that ever appear inside m_s's constraint box are less than $m_s.k$. However, if the set U has size of at least $m_s.k$, the message m_s may still not be anonymized under a hypothetical optimal algorithm. This is because, the optimal choice may require to anonymize a subset of U that does not include m_s, together with some other messages not in U. As a result, the remaining messages in U may not be sufficient to anonymize m_s. It is not possible to design an on-line algorithm that is optimal in terms of success rate, due to the fact that such an algorithm will require future knowledge of messages, which is not known beforehand. If a trace of the messages is available, as in this work, the optimal success rate can be computed 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 anonyimized.

There are three observations from Figure 3. First, the nbr-k approach provides around 15% better average success rate than local-k. Second, the best average success rate achieved is around 70. Out of the 30% dropped messages, at least 65% of them cannot be anonymized, meaning that in the worst case remaining 10% of all messages are dropped due to non-optimality of the algorithm with respect to success rate. If we knew a way to construct the optimal algorithm (with a reasonable time and space complexity) given full knowledge of the trace, we could have got a better bound. Last, messages with larger k values are harder to anonymize. The success rate for messages with $k = 2$ is around 30% higher than the success rate for messages with $k = 5$.

Figure 4 shows the relative anonymity level for nbr-k and local-k approaches. The relative anonymity level is shown (on y-axis) for different groups of messages, each group representing messages with a certain k value (on x-axis). Nbr-k shows a relative anonymity level of 1.7 for messages with $k = 2$, meaning that on the average these messages are anonymized with $k = 3.4$ by the algorithm. Local-k shows a lower relative anonymity level of 1.4 for messages with $k = 2$. This gap between the two approaches vanishes for messages with $k = 5$, since both of the algorithms do not attempt to search cliques of sizes larger than the maximum of the k values specified by the messages. The gap in relative anonymity level between nbr-k and local-k shows that the former approach is able to anonymize messages with smaller k values together with the ones with higher k values. This is particularly good for messages with higher k values, as they are harder to anonymize. This also explains why nbr-k results in better success rate.

Figure 5 plots the average success rate (y-axis on the left side) and the message processing time (y-axis on the right side) for

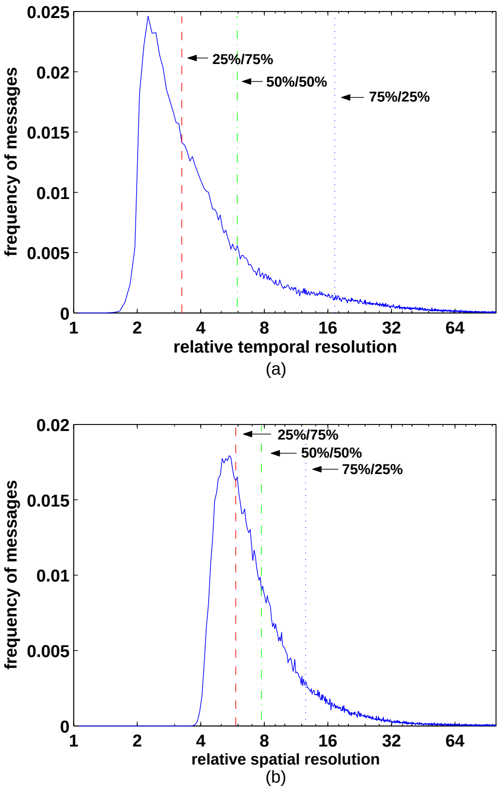


Figure 9: Relative temporal and spatial resolution distributions

nbr-k and local-k search approaches with immediate or deferred processing mode. For deferred processing mode α is taken as 1.4 (as it gave the highest success rate). Other than the immediate approach providing better success rate than the deferred approach, the surprising observation from the figure is that, deferred approach does not provide improvement in terms of message processing time. Figure 5 also shows (above the x-axis) the number of times clique search is performed for different approaches. Although the deferred approach results in slightly higher message processing time, it decreases the number of times the clique search is performed around 50% (for nbr-k). Here is the reason that the deferred approach still performs worse in terms of total processing time: For $k \leq 10$ the index update dominates the cost of processing the message and the deferred approach results in a more crowded index. However, the deferred approach is promising in terms of message processing time, for cases where k values are really large (thus clique search dominates the cost). Another potential enhancement is to design a more efficient multi dimensional index to replace the in-memory R* tree.

Figure 6 plots the success rate for different mean temporal tolerances and different variances in temporal and spatial tolerances. It shows that the algorithm is much less sensitive to the changes in the variances of the spatial and temporal tolerances than the mean temporal tolerance. For instance, when the mean temporal tolerance is $60s$, changing the variance in both spatial and temporal tolerances from 0.2 times their means to 1.6 times their means only decreases the success rate from 80 to 75; whereas decreasing the mean temporal tolerance from $60s$ to $15s$ decreases the success rate by approximately 40% of its success rate (for instance from 80 to 50 when variances are equal to 0.2 times their means).

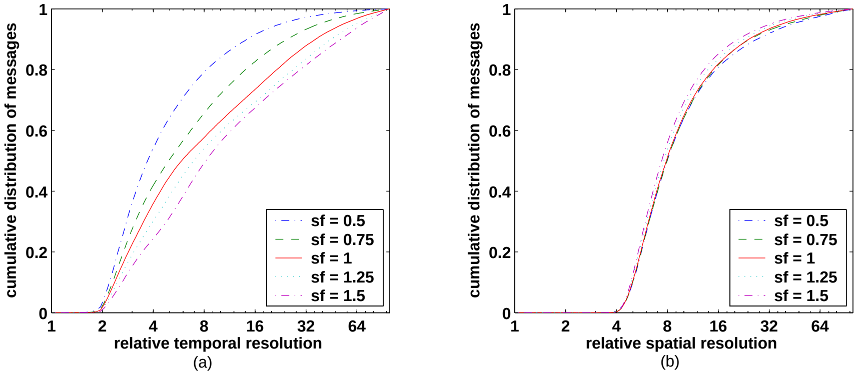


Figure 10: Relative temporal and spatial resolution cdfs for various settings

Figure 7 plots the average success rate as a function of mean inter-wait time and mean temporal tolerance. Similarly, Figure 8 plots the average success rate as a function of mean inter-wait time and mean spatial tolerance. For both of the figures, the variances are always set to 0.4 times the means. We observe that, the smaller the inter-wait time, the higher the success rate. For smaller values of the temporal and spatial tolerances, the decrease in inter-wait time becomes more important, in terms of keeping the success rate high. When the inter-wait time is high, we have a lower rate of messages coming into the system. Thus, it becomes harder to anonymize messages, as the constraint graph becomes sparser. Both spatial and temporal tolerances has tremendous effect on the success rate. Although high success rates (around 85) are achieved with high temporal and spatial tolerances, as we will show in the next section, the relative temporal and spatial resolutions are much larger than 1 in such cases, meaning that the system assigns much smaller spatio-temporal cloaking boxes to the messages compared to the constraint boxes.

6.2 Spatial/Temporal Resolution

In Section 6.1, we have showed that one way to improve success rate is to increase the spatial and temporal tolerance values specified by the messages. In this section, we show that the $CliqueCloak$ algorithm has 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 9(a) plots the frequency distribution (on the y-axis) of the relative temporal resolutions (on the x-axis) of the anonymized messages. Figure 9 shows that in 75% of the cases the provided relative temporal resolution is > 3.25, thus an average temporal accuracy of roughly $< 10s$ (recalling that the default mean temporal tolerance was $30s$). For 50% of the cases it is > 5.95 and for 25% of the cases it is > 17.25. This points out that, the observed performance with regard to temporal resolutions is much better than the worst case specified by the temporal tolerances. Figure 10(a) investigates whether this property of the algorithm holds under different settings. It plots the cumulative distribution (on the y-axis) of the relative temporal resolutions (on the x-axis) of the anonymized messages for different sf values. Here, sf is a *scaling factor*, and denotes that default mean and variance values of spatial and temporal tolerances are multiplied by sf. Figure 10(a) shows that, even for $sf = 0.5$, in approximately 50% of the cases the relative temporal resolution is > 4.

Figure 9(b) plots the frequency distribution (y-axis) of the relative spatial resolutions (x-axis) of the anonymized messages. Figure 9 shows that in 75% of the cases the provided relative spatial resolution is > 5.85, thus an average spatial accuracy of roughly $< 18m$ (recalling that the default mean spatial tolerance was $100m$). In 50% of the cases it is > 7.75 and for 25% of the cases it is > 12.55. This points out that, the observed performance with regard to spatial resolutions is much better than the worst case specified by the spatial tolerances. Figure 10(b) investigates whether this property of the algorithm holds under different settings. It plots the cumulative distribution (y-axis) of the relative spatial resolutions (x-axis) of the anonymized messages for different sf values. Figure 10(b) shows that, the behavior is effected minimally from the changes in the parameters, when compared to the case with temporal resolution.

7 Conclusion

We have proposed a customizable k-anonymity model for providing location privacy. Our model has two unique features. First, it allows each mobile node to define, at the granularity of single messages, its minimum anonymity level requirement, as well as upper bounds on the inaccuracy to be introduced by the cloaking algorithm in temporal and spatial dimensions. Second, it implements the model using a novel spatio-temporal cloaking algorithm, $CliqueCloak$, that can effectively anonymize messages sent by the mobile nodes, in accordance with location k-anonymity, while satisfying the anonymity and accuracy requirements of the messages. We also introduced improvements to the basic $CliqueCloak$ algorithm and experimentally studied the behavior of the algorithms under various conditions, using realistic location data synthetically generated using real road maps and traffic volume data.

References

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

[2] N. R. Adam and J. C. Wortman. Security-control methods for statistical databases. *ACM Computing Surveys*, 21(4), 1989.

[3] L. L. Beck. A security mechanism for statistical databases. *ACM Transactions on Database Systems*, 5(3), 1980.

[4] F. Y. Chin and G. Ozsoyoglu. Auditing and inference control in statistical databases. *IEEE Transactions on Software Engineering*, 8(6), 1982.

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

[6] D. E. Denning. Secure statistical databases with random sample queries. *ACM Transactions on Database Systems*, 5(3), 1980.

[7] D. Dobkin, A. K. Jones, and R. J. Lipton. Secure databases: Protection against user influence. *ACM Transactions on Database Systems*, 4(1), 1979.

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

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

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

[11] C. K. Liew, W. J. Choi, and C. J. Liew. A data distortion by probability distribution. *ACM Transactions on Database Systems*, 10(3), 1985.

[12] Lifelog. http://www.darpa.mil/ipto/Programs/lifelog/, January 2004.

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

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

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

[16] S. P. Reiss. Practical data swapping: The first steps. *ACM Transactions on Database Systems*, 9(1), 1984.

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

[18] J. Schlorer. Information loss in partitioned statistical databases. *The Computer Journal*, 26(3), 1983.

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

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

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

[22] J. F. Traub, Y. Yemini, and H. Wozniakowski. The statistical security of a statistical database. *ACM Transactions on Database Systems*, 9(4), 1984.