Privacy-Preserving Trajectory Data Publishing by Local Suppression

Rui Chen[a], Benjamin C. M. Fung[a,*], Noman Mohammed[a], Bipin C. Desai[a], Ke Wang[b]

[a] *Concordia University, Montreal, Quebec, Canada, H3G 1M8*
[b] *Simon Fraser University, Burnaby, British Columbia, Canada, V5A 1S6*

Abstract

The pervasiveness of location-aware devices has spawned extensive research in trajectory data mining, resulting in many important real-life applications. Yet, the privacy issue in sharing trajectory data among different parties often creates an obstacle for effective data mining. In this paper, we study the challenges of anonymizing trajectory data: high dimensionality, sparseness, and sequentiality. Employing traditional privacy models and anonymization methods often leads to low data utility in the resulting data and ineffective data mining. In addressing these challenges, this is the first paper to introduce local suppression to achieve a tailored privacy model for trajectory data anonymization. The framework allows the adoption of various data utility metrics for different data mining tasks. As an illustration, we aim at preserving both instances of location-time doublets and frequent sequences in a trajectory database, both being the foundation of many trajectory data

*Corresponding author
Email addresses: ru_che@encs.concordia.ca (Rui Chen), fung@ciise.concordia.ca (Benjamin C. M. Fung), no_moham@encs.concordia.ca (Noman Mohammed), bcdesai@cs.concordia.ca (Bipin C. Desai)

mining tasks. Our experiments on both synthetic and real-life data sets suggest that the framework is effective and efficient to overcome the challenges in trajectory data anonymization. In particular, compared with the previous works in the literature, our proposed local suppression method can significantly improve the data utility in anonymous trajectory data.
Keywords:
Privacy preservation, trajectory data, local suppression, frequent sequence

1. Introduction

Recently, the prevalence of various location-aware devices, such as RFID tags, cell phones, GPS navigation systems, and point of sale terminals, has made trajectory data ubiquitous in various domains. The fact has stimulated extensive trajectory data mining research [11][14][15], resulting in many important real-life applications, such as city traffic management [20], homeland security [19], and location-based advertising [35].

Having access to high-quality trajectory data is the prerequisite for effective data mining. However, trajectory data often contain detailed information about individuals, and disclosing such information may reveal their lifestyles, preferences, and sensitive personal information. Moreover, for many applications, trajectory data need to be published with other attributes, including sensitive ones, thus incurring the privacy concern of inferring individuals' sensitive information via trajectory data. This emerging data publishing scenario, however, has not been well studied in existing works. Such privacy concerns often limit trajectory data holders' enthusiasm in providing data for further research and applications. Example 1.1

illustrates the potential privacy threats due to trajectory data publishing.

Example 1.1. A hospital has employed a RFID patient tagging system in which patients' trajectory data, personal data, and medical data are stored in a central database [27]. The hospital intends to release such data (Table 1) to data miners for research purposes. A trajectory is a sequence of spatio-temporal *doublets* in the form of $(loc_i t_i)$. For example, *Record*#3 indicates that the tagged patient visited locations b, c, and e at timestamps 3, 7, and 8, respectively, and has hepatitis (other information is omitted for the purpose of illustration). With adequate background knowledge, an adversary can perform two kinds of privacy attacks on the trajectory database.

Identity linkage attack: If a trajectory in the database is so specific that not many patients can match it, there is a chance that with the help of background knowledge an adversary could uniquely identify the victim's record and, therefore, his sensitive information. Suppose an adversary knows that the record of the target victim, Claude, is in Table 1, and that Claude visited locations d and e at timestamps 2 and 4, respectively. The adversary can associate *Record*#1 with Claude and in turn identify Claude as an HIV patient because *Record*#1 is the only record containing both $d2$ and $e4$.

Attribute linkage attack: If a sensitive value occurs frequently with some sequences of doublets, it is possible to infer the sensitive value from these sequences even though the record of the victim cannot be uniquely identified. Suppose the adversary knows that another victim, Bill, visited $a1$ and $f6$. The adversary can infer that Bill has HIV with $2/3 = 67\%$ confidence because two of the three records (*Records*#$1, 5, 8$) containing $a1$ and $f6$ have the sensitive value HIV. ■

Table 1: Raw trajectory database T

Rec. #	**Path**	**Diagnosis**	**...**
1	$a1 \rightarrow d2 \rightarrow b3 \rightarrow e4 \rightarrow f6 \rightarrow e8$	HIV	...
2	$d2 \rightarrow c5 \rightarrow f6 \rightarrow c7 \rightarrow e9$	Fever	...
3	$b3 \rightarrow c7 \rightarrow e8$	Hepatitis	...
4	$b3 \rightarrow e4 \rightarrow f6 \rightarrow e8$	Flu	...
5	$a1 \rightarrow d2 \rightarrow c5 \rightarrow f6 \rightarrow c7$	HIV	...
6	$c5 \rightarrow f6 \rightarrow e9$	Hepatitis	...
7	$f6 \rightarrow c7 \rightarrow e8$	Fever	...
8	$a1 \rightarrow d2 \rightarrow f6 \rightarrow c7 \rightarrow e9$	Flu	...

A trajectory database (e.g., Table 1) may contain other attributes, such as gender, age, and nationality. Although they are not explicit identifiers, an adversary may utilize combinations of these attributes, called *quasi-identifiers* (*QIDs*), to identify the records and sensitive information of target victims. To thwart privacy threats due to QIDs, many privacy models, such as k-anonymity [30], ℓ-diversity [22], and confidence bounding [34], have been proposed in the context of relational data. These privacy models are effective for relational data anonymization; however, they fail to address the new challenges of trajectory data anonymization, as described below.

High dimensionality: Trajectory data are usually high-dimensional and cannot be effectively handled by traditional k-anonymity and its extensions due to the *curse of high dimensionality* [2]. Consider a transit system with 300 stations operating 24 hours a day. The corresponding trajectory database would have $300 \times 24 = 7200$ dimensions, because a trajectory could be represented in a tabular format with 7200 attributes filled with 0/1 values. Since k-anonymity and its extensions require every trajectory to be shared by at least k records and/or impose the diversity of sensitive values in every tra-

jectory group, most data have to be suppressed in order to meet these kinds of restrictive privacy requirements.

Sparseness: Trajectory data are usually sparse. Consider passengers in transit systems. Among all available locations, they may visit only a few, making the trajectory of each individual relatively short. Anonymizing such short trajectories in a high-dimensional space poses great challenges for traditional anonymization techniques because the trajectories may have little overlap. Enforcing k-anonymity could lower the data utility significantly.

Sequentiality: Time contains important information for trajectory data mining, but it also brings new privacy threats. Consider two trajectories $b3 \rightarrow e6$ and $e3 \rightarrow b6$. They have the same locations and timestamps but in a different order and, thus, are different from each other. An adversary could exploit such difference in order to increase the chance of a successful linkage attack. Therefore, traditional k-anonymity is not applicable to trajectory data, and anonymizing trajectory data requires additional efforts.

1.1. Trade-off between Privacy and Utility

One common assumption of k-anonymity and its extensions is that an adversary may use any or *even all* attributes in QIDs to perform linkage attacks. Yet this common assumption may be overly restrictive in the context of trajectory data. In a real-life attack, it is very unlikely that an adversary can identify all the visited locations along with the timestamps of a victim because it requires significant efforts to collect every piece of such background information. If the adversary is able to learn all such information, it is also possible that he can learn the victim's sensitive information. Thus, in the context of trajectory data, it is reasonable to derive a practical privacy model

Table 2: $(2, 50\%)_2$-privacy preserved database T'

Rec. #	**Path**	**Diagnosis**	**...**
1	$b3 \rightarrow e4 \rightarrow f6 \rightarrow e8$	HIV	...
2	$d2 \rightarrow c5 \rightarrow f6 \rightarrow c7 \rightarrow e9$	Fever	...
3	$c7 \rightarrow e8$	Hepatitis	...
4	$b3 \rightarrow e4 \rightarrow f6 \rightarrow e8$	Flu	...
5	$d2 \rightarrow c5 \rightarrow f6 \rightarrow c7$	HIV	...
6	$c5 \rightarrow f6 \rightarrow e9$	Hepatitis	...
7	$f6 \rightarrow c7 \rightarrow e8$	Fever	...
8	$d2 \rightarrow f6 \rightarrow c7 \rightarrow e9$	Flu	...

based on the assumption that an adversary's background knowledge on a target victim is bounded by at most L location-time doublets. We call such bounded background knowledge L-*knowledge*.

Based on this observation, we adopt a new privacy model called $(K, C)_L$-*privacy* that requires any subsequence q of any adversary's L-knowledge to be shared by either 0 or at least K records in a trajectory database T and the confidence of inferring any sensitive value in S from q to be at most C, where L and K are positive integer thresholds, C is a real number threshold in the range of $[0, 1]$, and S is a set of sensitive values specified by the data holder. $(K, C)_L$-privacy guarantees that the probability of succeeding in an identity linkage attack is $\leq 1/K$ and the probability of succeeding in an attribute linkage attack is $\leq C$. Table 2 presents an example of an anonymous database satisfying $(2, 50\%)_2$-privacy from Table 1, in which every sequence q with maximum length 2 is shared by at least 2 records and the confidence of inferring any sensitive value in $S = \{HIV, Hepatitis\}$ from q is $\leq 50\%$.

Protecting privacy is one aspect of anonymizing trajectory data. Another aspect is preserving data utility in the anonymous data for data mining. The

anonymized data may be used for different data mining tasks; therefore, we propose a generic framework to accommodate different utility requirements. As an illustration, in this paper we aim to preserve both instances of location-time doublets and frequent sequences in a trajectory database. The ratio of suppressed instances is a general measure of anonymized data quality for a wide range of trajectory data mining tasks [14][15]; the ratio of suppressed frequent sequences is a direct indication of anonymized data quality for trajectory pattern mining [11].

Generalization, *bucketization*, and *suppression* are the most widely used anonymization mechanisms. Generalization requires the use of taxonomy trees, which are highly specific to a particular application [3]. In many trajectory data applications, such domain specific taxonomy trees are not available. This fact largely hinders generalization's applicability on trajectory data anonymization. Bucketization merely breaks the correlation between trajectory data and sensitive attributes, and publishes trajectory data without any modification, which fails to protect identity linkage attacks on trajectory data. In addition, a *condensation* approach [3] is proposed for multi-dimensional data publishing. However, it does not prevent from attribute linkage attacks in general. Specifically, for trajectory data, its complexity grows exponentially due to the high dimensionality. Furthermore, there lacks a way of measuring the similarity of trajectories, which is essential to the condensation approach. Therefore, in this paper, we employ suppression, both local and global suppressions, to eliminate privacy threats from a trajectory database. The introduction of local suppression results in significant data utility improvements for trajectory data anonymization. In

global suppression, if a location-time doublet p is selected to be suppressed from a trajectory database T, then all instances of p are removed from T, whereas in local suppression, some instances of p may remain intact in T while other instances are removed. Global suppression punishes all records containing p even if the privacy leakage is caused by only one instance of p in one record. In contrast, local suppression eliminates the exact instances that cause privacy breaches without penalizing others. Thus, local suppression preserves much better data utility compared to global suppression.

1.2. Contributions

In this paper, we acknowledge the emerging data publishing scenario, in which trajectory data need to be published with sensitive attributes. This naturally requires to prevent from both identity linkage attacks and attribute linkage attacks, which has not been studied in existing works. Based on the practical assumption that an adversary has only limited background knowledge on a target victim, we adopt $(K, C)_L$-privacy model for trajectory data anonymization, which takes into consideration not only identity linkage attacks on trajectory data, but also attribute linkage attacks via trajectory data. We present an anonymization framework that supports both local suppression and global suppression with the goal of preserving data utility for data mining. This is the first study introducing local suppression to trajectory data anonymization. In this paper, we tailor our anonymization framework to preserve both instances of location-time doublets and frequent sequences in trajectory data. The framework itself is open to different data mining workloads by incorporating different data utility metrics. We provide comprehensive experimental evaluations on both synthetic and real-life tra-

jectory data sets. The experimental results demonstrate that our proposed algorithm is both effective and efficient to address the special challenges in trajectory data anonymization. In particular, local suppression is shown to be essential to enhance the resulting data utility when combined with $(K, C)_L$-privacy.

2. Related Work

2.1. Anonymizing Relational & Statistical Data

k-anonymity [30] prevents identity linkage attacks by requiring every qid group (a.k.a. equivalent class) in a relational data table T to contain at least k records. ℓ-diversity [22] and confidence bounding [34] aim at preventing attribute linkage attacks. ℓ-diversity requires every qid group to contain at least ℓ "well-represented" sensitive values, while confidence bounding limits an adversary's confidence of inferring a sensitive value in any qid group to a certain threshold. (α, k)-anonymity [36] incorporates both k-anonymity and confidence bounding into a single privacy model. Li and Li [17] model an adversary's background knowledge by mining negative association rules from the data and then use them in the anonymization process. Kisilevich et al. [13] make use of the decision tree built on an original data set to perform multi-dimensional suppression for achieving k-anonymity. Matatov et al. [23] partition a data set into several k-anonymous projections, train a classifier on each of them, and then conduct classification tasks by combining the classifications of all such classifiers. Traditional privacy models for relational data suffer from the curse of high dimensionality [2] and, therefore, may render high-dimensional data totally useless for further data mining tasks. Recently,

Mohammed et al. [26] propose the LKC-privacy model for high-dimensional relational data, which assumes that the adversary's prior knowledge is limited to at most L attributes in QID. They achieve the LKC-privacy model based on global generalization. In contrast, this paper focuses on trajectory data anonymization by local suppression.

Dwork [5] proposes an insightful privacy notion based on the principle that the risk to a record owner's privacy should not substantially increase as a result of participating in a statistical database. Consequently, Dwork [5] introduces a privacy model called ϵ-*differential privacy* to ensure that the removal or addition of a single record does not have a significant effect on the outcome of any analysis. Differential privacy does not prevent identity and attribute linkages studied in this paper, but assures record owners that nothing can be discovered by comparing the databases with and without their records. Most works on differential privacy focus on relational data and are still limited to a very few primitive data mining tasks. Machanava et al. [21] further indicate that differential privacy can only be achieved by randomized mechanisms, for example, adding noise. Therefore, it cannot preserve data truthfulness, which is important if the data will be examined by human users for the purposes of auditing, data interpretation, or visual data mining.

2.2. Anonymizing Transaction Data

Recently, there is more focus on anonymizing high-dimensional transaction data [10][12][32][38][39], in which sequentiality is not a concern. Ghinita et al. [10] propose a permutation method that groups transactions with close proximity and then associates each group to a set of diversified sensitive values. Terrovitis et al. [32] propose an algorithm to k-anonymize transac-

tions by generalization according to some given taxonomy trees. He and Naughton [12] extend [32] by introducing local generalization, which gains better utility. Neither [12] nor [32] addresses attribute linkage attacks. In real-life trajectory databases, however, taxonomy trees may not be available or a logical one for locations may not exist. Moreover, Fung et al. [8] point out that if the taxonomy tree tends to be flat and fans out, which is the case of trajectory data, employing generalization loses more information than employing suppression because generalization has to merge all siblings of a selected node to their parent node, whereas suppression only removes the selected child node. Xu et al. [38][39] extend the k-anonymity model by assuming that an adversary knows at most a certain number of transaction items of a target victim, which is similar to our assumption of limited background knowledge of an adversary. Though their method addresses the high dimensionality concern, it does not consider the sequential property of trajectory data and, therefore, is not applicable to trajectory data anonymization. Furthermore, Xu et al. [38][39] achieve their privacy model by global suppression, which significantly hinders data utility on trajectory data.

2.3. Anonymizing Trajectory Data

Some recent works [1][6][7][24][29][31][40] study anonymization of trajectory data from different perspectives. Abul et al. [1] propose (k, δ)-anonymity based on the imprecision of sampling and positioning systems, where δ represents the possible location imprecision. Based on *space translation*, the general idea is to modify the paths of trajectories so that k different trajectories co-exist in a cylinder of the radius δ. However, the imprecision assumption may not hold in some sources of trajectory data, such as transit

data, RFID data, and purchase records.

Due to the high dimensionality of trajectory data, [29] and [31] study the anonymization problem on a simplified form of trajectory data, in which only temporal sequentiality is considered, known as *sequential data*. Pensa et al. [29] propose a variant of k-anonymity model for sequential data, with the goal of preserving frequent sequential patterns. Similar to the space translation method in [1], Pensa et al. [29] transform a sequence into another form by inserting, deleting, or substituting some items. Terrovitis et al. [31] further assume that different adversaries may possess different background knowledge and that the data holder has to be aware of *all* such adversarial knowledge. The objective is to prevent adversaries from gaining further information from the published sequential data. The assumption of knowing *all* adversarial knowledge before publishing the data is possible in the specific scenario described in their paper, but it is not applicable in the context of trajectory data in general. The simplification from trajectory data to sequential data does help overcome the high dimensionality of trajectory data. However, for many trajectory data mining tasks, the time information is indispensable. Therefore, these approaches fail to satisfy the information requirements of the data mining tasks. Yarovoy et al. [40] present a novel notion of k-anonymity in the context of *moving object databases* (MOD) based on the assumption that different moving objects may have different QIDs. Specifically, they consider timestamps as the QIDs, with moving objects' locations forming their values. Adversaries are assumed to conduct privacy attacks based on an *attack graph*. An underlying assumption of [40] is that the data holder must be aware of the QIDs of all moving objects. However,

the paper leaves the problem of the acquisition of QIDs for a data holder unsolved.

All these works [1][29][31][40] are limited to privacy protection on only identity linkage attacks over trajectory data, whereas our method prevents not only identity linkage attacks, but also attribute linkage attacks via trajectory data in order to accommodate the emerging trajectory data publishing scenario. A line of very recent papers [6][7][24] have launched studies on protecting both identity and attribute linkages. However, such papers are limited to global suppression, which results in less desirable utility. To enhance the resulting data utility, local suppression is utilized for the first time in the context of trajectory data. In addition, all these works are effective in some specific scenarios, while our proposed framework has fewer constraints, and therefore, is applicable for different trajectory data sources and different data mining workloads.

3. Problem Definition

3.1. Trajectory Database

A typical trajectory system generates a sequence of sensory data records of the general form $\langle ID, loc, t \rangle$, where each record indicates that the record owner (or the object) having the unique identifier ID was detected in location loc at time t. For example, in transportation systems, a record represents that a passenger was present in station loc at time t, where ID could be the passenger's transportation card number. Different types of trajectory data can be easily converted into the general form by pre-processing steps. For example, GPS data, a typical type of trajectory data, is of the form

$\langle ID, (X\ coordinate, Y\ coordinate), timestamp\rangle$, which can be converted by substituting the grid ID/name containing a point for $(X\ coordinate, Y\ coordinate)$. By selecting proper granularity of such grids, this general form is suitable to represent various kinds of trajectory data for different data mining tasks.

The trajectory of a specific record owner, representing the owner's movement history, is composed of a sequence of (loc, t) *doublets*. A *trajectory*, denoted by $(loc_1t_1) \rightarrow \ldots \rightarrow (loc_nt_n)$, can be constructed by grouping the sensory data records $\langle ID, loc, t\rangle$ by ID and sorting them by the timestamps. The timestamps in a trajectory are always increasing.

In addition to trajectory data, a *trajectory database* may also contain other attributes that are associated with the record owners. Formally, a trajectory database contains a collection of data records in the form of

$$(loc_1t_1) \rightarrow \ldots \rightarrow (loc_nt_n) : s_1, \ldots, s_p : d_1, \ldots, d_m$$

where $(loc_1t_1) \rightarrow \ldots \rightarrow (loc_nt_n)$ is a trajectory, $s_i \in S_i$ are the sensitive attributes with values from the domain S_i, and $d_i \in D_i$ are the quasi-identifiers (QIDs) of the record owner with the values from the domain D_i. Given a trajectory database, an adversary can perform privacy attacks via either trajectories or QID attributes. Anonymization on relational QID attributes has been extensively studied in previous works [9][16][22][30][37]. This paper focuses on addressing the privacy threats posed by trajectories.

3.2. Privacy Threats

Suppose a data holder wants to publish a trajectory database T to some recipients for data mining. Explicit identifiers, e.g., name, SSN, and ID, have been removed. One recipient, the adversary, seeks to identify the record or

sensitive values of some target victim V in T. As explained in Section 1, we assume that the adversary knows at most L spatio-temporal doublets that the victim V has previously visited. Such background knowledge about the victim V is denoted by $\kappa_V = (loc_1 t_1) \rightarrow \ldots \rightarrow (loc_z t_z)$, where $z \leq L$. Using the background knowledge κ_V, the adversary could identify a group of records in T, denoted by $T(\kappa_V)$, that "matches" κ_V. A record *matches* κ_V if κ_V is a subsequence of the trajectory in the record. For example, in Table 1, if $\kappa_V = d2 \rightarrow e4$, then $Record\#1$ matches κ_V, but $Record\#2$ does not. Given the background knowledge κ_V, an adversary could identify and utilize $T(\kappa_V)$ to perform two types of privacy attacks:

1. *Identity linkage attack*: $T(\kappa_V)$ is a set of candidate records that contains the victim V's record. If the group size of $T(\kappa_V)$, denoted by $|T(\kappa_V)|$, is small, then the adversary may identify V's record from $T(\kappa_V)$ and, therefore, V's sensitive value.
2. *Attribute linkage attack*: Given $T(\kappa_V)$, the adversary may infer that V has sensitive value s with confidence $Conf(s|T(\kappa_V)) = \frac{|T(\kappa_V \bigcup s)|}{|T(\kappa_V)|}$, where $T(\kappa_V \bigcup s)$ denotes the set of records containing both κ_V and s. $Conf(s|T(\kappa_V))$ is the percentage of the records in $T(\kappa_V)$ containing s. The privacy of V is at risk if $Conf(s|T(\kappa_V))$ is high.

Example 1.1 illustrates these two types of attacks.

3.3. Privacy Requirement

An adversary's background knowledge κ could be any non-empty subsequence q with $|q| \leq L$ of any trajectory in the trajectory database T. Intuitively, $(K, C)_L$-*privacy* requires that every subsequence q with $|q| \leq L$

in T is shared by at least a certain number of records, and that the confidence of inferring any sensitive value via q cannot be too high.

Definition 3.1 ($(K,C)_L$-privacy). Let L be the maximum length of the background knowledge. Let S be a set of sensitive values of the sensitive attributes of a trajectory database T selected by the data holder. T satisfies $(K,C)_L$-*privacy* if and only if for any subsequence q in T with $0 < |q| \leq L$,

1. $|T(q)| \geq K$, where K is a positive integer specifying the anonymity threshold, *and*
2. $Conf(s|T(q)) \leq C$ for any $s \in S$, where $0 \leq C \leq 1$ is a real number specifying the confidence threshold. ■

The $(K,C)_L$-privacy model has several desirable properties. First, it is a generalized version of several existing privacy models: k-anonymity [30] is a special case of the $(K,C)_L$-privacy model with $L = |d|$ and $C = 100\%$, where $|d|$ is the number of dimensions in a given database. ℓ-diversity [22] is a special case of $(K,C)_L$-privacy model with $L = |d|$, and $\ell = 1/C$. Confidence bounding [34] is a special case of the $(K,C)_L$-privacy model with $L = |d|$ and $K = 1$. (α,k)-anonymity [36] is also a special case of $(K,C)_L$-privacy with $L = |d|$, $K = k$, and $C = \alpha$. Second, it is intuitive for a data holder to impose different types and levels of privacy protection by specifying different L, K, and C thresholds.

It is worth noting that $(K,C)_L$-privacy is a stronger privacy notion than other existing privacy models for trajectory data [1][29][31][40] in the sense that $(K,C)_L$-privacy thwarts both identity linkages on trajectory data and attribute linkages via trajectory data. It is vital to thwart attribute linkage

attacks in trajectory data publishing because more and more trajectory data mining tasks will resort to both trajectory data and other personal information. For example, Utsunomiya et al. [33] conducted an interesting passenger classification analysis using both passengers' trajectory data and personal information. A recent investigation [28] further indicates that there is a need to enrich trajectory data by incorporating sociodemographic data for data mining tasks.

3.4. Utility Requirement

Since we aim at presenting a framework that allows the adoption of various data utility metrics for different data mining tasks, we illustrate the preservation of two different kinds of utility metrics, both *instances of location-time doublets* and *frequent sequences* in a trajectory database. The ratio of suppressed instances is a general measure of the usefulness of anonymized data for a wide range of trajectory data mining tasks [14][15]. In addition, previous works [9][18] suggest that anonymization algorithms can be tailored to better preserve utility if the utility requirement is known in advance. We also preserve frequent sequences specifically for trajectory pattern mining [11]. However, extracting all possible frequent sequences in a trajectory database is computationally expensive. It is even exacerbated when dealing with large data sets with long frequent sequences because all subsequences of a frequent sequence are also frequent. A more feasible solution is to preserve *maximal frequent sequences* (*MFS*).

Definition 3.2 (Maximal frequent sequence). For a given minimum support threshold $K' > 0$, a sequence q is *maximal frequent* in a trajectory database T if q is frequent and no super sequence of q is frequent in T. ■

The set of MFS in T, denoted by $U(T)$, is much smaller than the set of frequent sequences (FS) in T given the same K', but still contains the essential information of FS. Any subsequence of an MFS is also an FS. Once all the MFS have been determined, the support count of any particular FS can be computed by scanning $U(T)$ once.

We emphasize that although in this paper we aim at preserving instances and MFS, the $(K, C)_L$-privacy model and the anonymization framework presented in Section 4 are independent of the underlying utility metric and are flexible enough to serve other utility requirements. The only change is to replace the greedy function guiding the anonymization process, which will be further explained in Section 4.2.

3.5. Problem Statement

To achieve $(K, C)_L$-privacy for a given trajectory database T, our proposed framework conducts a sequence of local and global suppressions to remove all privacy threats from T while preserving as much data utility as possible. *Global suppression* eliminates *all* instances of a doublet p from T if some instances of p cause privacy breaches, while *local suppression* eliminates only the instances of p that cause privacy breaches and leaves others intact. Finding an optimal solution based on suppression for $(K, C)_L$-privacy, however, is *NP-hard* (see Section 4.2 for proof). Thus, we propose a greedy algorithm to efficiently identify a reasonably "good" sub-optimal solution.

Definition 3.3 (Trajectory data anonymization). Given a trajectory database T, a $(K, C)_L$-privacy requirement, a utility metric, and a set of sensitive values S, the task of *trajectory data anonymization* is to generate

a transformed version of T that satisfies $(K, C)_L$-privacy while maintaining the maximum utility with respect to the utility metric by a sequence of local and global suppressions. ■

4. The Anonymization Algorithm

The proposed anonymization algorithm consists of two phases. First, identify all violating sequences that breach a given $(K, C)_L$-privacy requirement in a trajectory database. Second, perform a sequence of local and global suppressions to anonymize the trajectory database while maintaining as much data utility as possible.

4.1. Identifying Violating Sequences

An adversary may use any non-empty sequence with length not greater than L as background knowledge to launch a linkage attack. Thus, given a $(K, C)_L$-privacy requirement, any subsequence q with $0 < |q| \leq L$ in a trajectory database T is a *violating sequence* if its group $T(q)$ does not satisfy Condition 1, Condition 2, or both in $(K, C)_L$-privacy in Definition 3.1.

Definition 4.1 (Violating sequence). Let q be a subsequence of a trajectory in T with $0 < |q| \leq L$. q is a *violating sequence* with respect to a $(K, C)_L$-privacy requirement if $|T(q)| < K$ *or* $Conf(s|T(q)) > C$ for any sensitive value $s \in S$. ■

Example 4.1. Given $L = 2$, $K = 2$, $C = 50\%$, and the sensitive value set $S = \{HIV, Hepatitis\}$. In Table 1, the sequence $q_1 = a1 \rightarrow b3$ is a violating sequence because $|T(q_1)| = 1 < K$; the sequence $q_2 = a1 \rightarrow d2$

is also a violating sequence because $Conf(HIV|T(q_2)) = 2/3 = 67\% > C$. However, the sequence $q_3 = b3 \rightarrow c7 \rightarrow e8$ is *not* a violating sequence even though $|T(q_3)| = 1 < K$ and $Conf(Hepatitis|T(q_3)) = 100\% > C$ because $|q_3| = 3 > L$. ■

To satisfy a given $(K, C)_L$-privacy requirement on a trajectory database T, it is sufficient if all violating sequences in T with respect to the privacy requirement are removed, because all possible channels for identity and attribute linkages are eliminated. A naive approach is to first enumerate all possible violating sequences and then remove them. This approach is infeasible because of the huge number of violating sequences. Consider a violating sequence q with $|T(q)| < K$. Any super sequence of q, denoted by q'', with $|T(q'')| > 0$ in T is also a violating sequence because $|T(q'')| \leq |T(q)| < K$. To overcome the bottleneck of violating sequence enumeration, our insight is that a few "minimal" violating sequences exist among the violating sequences, and it is sufficient to achieve $(K, C)_L$-privacy by removing *only* the minimal violating sequences.

Definition 4.2 (Minimal violating sequence). A violating sequence q is a *minimal violating sequence* (MVS) if every proper subsequence of q is not a violating sequence. ■

Example 4.2. Given $L = 2$, $K = 2$, $C = 50\%$, and $S = \{HIV, Hepatitis\}$. In Table 1, the sequence $q_1 = d2 \rightarrow e4$ is an MVS because $|T(q_1)| = 1 < K$, and none of its proper subsequences, $d2$ and $e4$, is a violating sequence. In contrast, the sequence $q_2 = a1 \rightarrow d2$ is a violating sequence, but *not* an MVS, because one of its proper subsequences, $a1$, is a violating sequence. ■

The set of MVS is much smaller than the set of violating sequences; therefore, we can efficiently identify all privacy threats by generating all MVS. A trajectory database T satisfies $(K, C)_L$-privacy if and only if T contains no MVS.

Theorem 4.1. A trajectory database T satisfies $(K, C)_L$-privacy if and only if T contains no minimal violating sequence.

Proof. Suppose a database T does not satisfy $(K, C)_L$-privacy even if T contains no MVS. By Definition 3.1, T must contain some violating sequences. According to Definition 4.2, a violating sequence must be an MVS itself or contain an MVS, which contradicts the initial assumption. Therefore, T must satisfy $(K, C)_L$-privacy. ■

Hence, our first step is to efficiently identify all the MVS, $V(T)$, in the given trajectory database T. Procedure 1 presents the details of generating $V(T)$. Based on Definition 4.2, we generate all MVS of size $i+1$, denoted by V_{i+1}, by incrementally extending non-violating sequences of size i, denoted by U_i, with an additional doublet. *This needs to take into consideration the sequentiality of trajectory data.* Line 1 loads all distinct doublets in T as the initial candidate set C_1. Line 4 scans T once to compute $|T(q)|$ and $Conf(s|T(q))$ for every sequence $q \in C_i$, and for every sensitive value $s \in S$. If a sequence q is not violating, it is added to the non-violating sequence set U_i for generating the next candidate set C_{i+1} (Line 7); otherwise, q is added to the MVS set (Line 9). The next candidate set C_{i+1} is generated in two steps. First, conduct a self-join of U_i (Line 11). Second, remove all super sequences of the identified MVS from C_{i+1} (Lines 12-14). The second

Procedure 1 Identify Minimal Violating Sequences (MVS)

Input: Raw trajectory database T

Input: Thresholds L, K, C, and sensitive values S

Output: Minimal violating sequences $V(T)$

```
1: C_1 ← all distinct doublets in T;
2: i = 1;
3: while i ≤ L and C_i ≠ ∅ do
4:    Scan T once to compute |T(q)| and Conf(s|T(q)), for ∀q ∈ C_i, ∀s ∈ S;
5:    for each sequence q ∈ C_i with |T(q)| > 0 do
6:       if |T(q)| ≥ K and Conf(s|T(q)) ≤ C for all s ∈ S then
7:             Add q to U_i;
8:       else
9:             Add q to V_i;
10:   i++;
11:   Generate candidate set C_i by U_{i-1} ⋈ U_{i-1};
12:   for each sequence q ∈ C_i do
13:      if q is a super sequence of any v ∈ V_{i-1} then
14:            Remove q from C_i;
15: return V(T) = V_1 ∪ ··· ∪ V_{i-1};
```

step significantly reduces the minimal violating sequence search space. Two sequences $q_x = (loc_1^x t_1^x) \to \ldots \to (loc_i^x t_i^x)$ and $q_y = (loc_1^y t_1^y) \to \ldots \to (loc_i^y t_i^y)$ can be joined if the first $i-1$ doublets are identical and $t_i^x < t_i^y$. The joined result is $(loc_1^x t_1^x) \to \ldots \to (loc_i^x t_i^x) \to (loc_i^y t_i^y)$. The definition of join-compatibility makes sure that every potential candidate sequence would be generated exactly once.

Example 4.3. Given $L = 2$, $K = 2$, $C = 50\%$, and the sensitive value set $S = \{HIV, Hepatitis\}$, the MVS set generated from Table 1 is $V(T) = \{a1, d2 \to b3, d2 \to e4, d2 \to e8, b3 \to c7\}$. ■

4.2. Removing Violating Sequences

The second step is to remove all identified minimal violating sequences using suppression with the goal of preserving as much data utility as possible. However, finding an optimal solution is *NP-hard*.

Theorem 4.2. Given a trajectory database T and a $(K, C)_L$-privacy requirement, it is NP-hard to find the optimal anonymization solution.

Proof. The problem of finding the optimal anonymization solution can be converted into the *vertex cover problem*. The vertex cover problem is a well-known problem in which, given an undirected graph $G = (V, E)$, it is NP-hard to find the smallest set of vertices S such that each edge has at least one endpoint in S. To reduce our problem into the vertex cover problem, we only consider the set of MVS of length 2. Then, the set of candidate doublets represents the set of vertices V and the set of MVS is analogous to the set of edges E. Hence, the optimal vertex cover, S, means finding the smallest set of candidate doublets that must be suppressed to obtain the optimal anonymous data set T'. Given that it is NP-hard to determine S, it is also NP-hard to find the optimal set of candidate doublets for suppression. ■

Therefore, we propose a greedy algorithm that employs both local and global suppressions to eliminate all identified MVS, $V(T)$, with respect to the given $(K, C)_L$-privacy requirement in order to efficiently identify a reasonably "good" solution. Generally, suppressing a doublet p from $V(T)$ increases privacy and decreases data utility. So our goal is to design a greedy function, $Score(p)$, that guides us to find the sub-optimal trade-off between privacy

and data utility. In this paper, we define our greedy function as follow:

$$Score(p) = \frac{PrivGain(p)}{UtilityLoss(p) + 1}$$

where $PrivGain(p)$ is the number of MVS that can be eliminated by suppressing p, and $UtilityLoss(p)$ is the number of either instances or MFS that are lost due to suppressing p, depending on the given utility metric. Since suppressing p may not cause utility loss in terms of MFS, we add 1 to the denominator to avoid the division by zero error. The function considers both privacy and utility simultaneously by selecting the anonymization operation with the maximum privacy gain per unit of utility loss. Considering only privacy gain or utility loss would lead to inferior performances according to our tests. Again, our anonymization algorithm is independent of the underlying data utility metric. To optimize the data utility for other data mining workloads, we can simply re-design the meaning of $UtilityLoss(p)$.

A key to an efficient solution is to ensure that no new MVS will be generated in the anonymizing process. Upon satisfying the requirement, the identified MVS $V(T)$ always decreases monotonically. A suppression-based algorithm is guaranteed to achieve $(K, C)_L$-privacy within less than $|V(T)|$ iterations. One nice property of *global suppression* is that it does not generate any new MVS during the anonymizing process.

Theorem 4.3. A global suppression does not generate any *new* minimal violating sequence with respect to a $(K, C)_L$-privacy requirement.

Proof. Suppose a doublet p is globally suppressed from a given trajectory database T. The database after the global suppression is denoted by T'.

- For any sequence q in T not containing an instance of p, we have

$|T'(q)| = |T(q)|$ and $Conf(s|T'(q)) = Conf(s|T(q))$. Identically, for any subsequence q' of q, which does not contain p either, we have $|T'(q')| = |T(q')|$ and $Conf(s|T'(q')) = Conf(s|T(q'))$. So q cannot be a new minimal violating sequence in T'.

- For any sequence q in T that contains an instance of p, q no longer exists in T', so q cannot be a new minimal violating sequence.

Therefore, no sequence in T will become a new MVS in T'. ■

However, *local suppression* does not share the same property. For example, locally suppressing $c7$ from $Record\#3$ in Table 1 will generate a new MVS $c7 \rightarrow e8$ because in the resulting database T', $|T'(c7 \rightarrow e8)| = 1 < K$. Identifying the values of all newly generated MVS requires expensive computational cost. Moreover, there is no guarantee that the anonymization algorithm can converge within a bounded number of iterations, $|V(T)|$. Therefore, it is beneficial to perform local suppressions only when no new MVS will be generated. Such a local suppression is called a *valid* local suppression.

Definition 4.3 (Valid local suppression). A local suppression over a trajectory database is *valid* if it does not generate any *new* MVS. ■

An intuitive way to check if a local suppression is valid is to re-invoke Procedure 1 and compare $V(T)$ and $V(T')$. However, it is extremely costly. Instead, Procedure 2 presents an efficient approach to avoid the computational cost of calculating the values of all newly generated MVS. It significantly narrows down the checking space to a very small set of sequences that may be affected by a local suppression by carefully using the properties of MVS.

Procedure 2 Check if a local suppression is valid

Input: Trajectory database T

Input: Thresholds L, K, C, and sensitive values S

Input: A doublet p in an MVS m

Output: A *boolean* value indicating if locally suppressing p from m is valid

1: $P \leftarrow$ distinct doublet p' such that $p' \in T(m) \wedge p' \in (T(p) - T(m))$;
2: $V' \leftarrow$ all size-one MVS and the MVS containing p, $V(p)$;
3: Remove all doublets, except p, in V' from P;
4: $Q \leftarrow$ all possible sequences with size $\leq L$ generated from P after removing super sequences of the sequences in $V(T) - V(p)$;
5: Scan $T(p) - T(m)$ once to compute $|q|$ and $Conf(s|T(q))$ for each sequence $q \in Q$ and for every sensitive value $s \in S'$, where S' is the subset of S in $T(p) - T(m)$;
6: **for** each sequence q with $|q| > 0$ **do**
7: **if** $|q| < K$ or $Conf(s|T(q)) > C$ for any $s \in S'$ **then**
8: **return** $false$;
9: **return** $true$;

Theorem 4.4. Procedure 2 is sufficient to check if a local suppression is valid.

Proof. Suppose a doublet p in an MVS m is locally suppressed from a given trajectory database T. The resulting database is denoted by T'. For any sequence q in T not containing an instance of p, we have $|T'(q)| = |T(q)|$ and $Conf(s|T'(q)) = Conf(s|T(q))$. Identically, for any subsequence q' of q, we have $|T'(q')| = |T(q')|$ and $Conf(s|T'(q')) = Conf(s|T(q'))$. So q cannot be a new MVS in T'. If there is a new MVS, it must contain p. Since p is eliminated from the records containing m, $T(m)$, we only need to consider the sequences in $T(p) - T(m)$, where $T(p)$ denotes the records containing p. For a sequence q in T containing an instance of p, if $q \notin T(m)$, we have $|T'(q)| = |T(q)|$ and $Conf(s|T'(q)) = Conf(s|T(q))$ and, therefore, such q

cannot be a new MVS. q is possible to be a new MVS only if $q \in T(m)$ *and* $q \in (T(p) - T(m))$ (Line 1). Since we only care about *new* MVS, we could further filter out all identified MVS and their super sequences. For the remaining sequences, if none of them is a violating sequence, it is sufficient to ensure that there is no new MVS by Definition 4.2 (Lines 4-9). ■

Example 4.4. Consider Table 1 with $L = 2$, $K = 2$, $C = 50\%$, and the sensitive value set $S = \{HIV, Hepatitis\}$. For the local suppression of $d2$ in MVS $d2 \rightarrow e4$, we get $P = \{d2, f6\}$ and $V' = \{a1, d2 \rightarrow b3, d2 \rightarrow e4, d2 \rightarrow e8\}$. Since all sequences in $Q = \{d2, f6, d2 \rightarrow f6\}$ are not violating sequences, this local suppression is *valid*.

Algorithm 1 presents the entire anonymization algorithm. Line 1 calls Procedure 1 to generate the MVS set $V(T)$. For preserving MFS, Line 2 is needed, which calls the MFS mining algorithm to build a MFS-tree with a UL table that keeps track of the occurrences of all candidate doublets in the MFS-tree. We adapt *MAFIA* [4], originally designed for mining maximal frequent itemsets, to mine MFS. For all instances of all doublets in $V(T)$, their scores for local and global suppressions are calculated and stored in *Score* table based on Procedure 2 (Line 3). Different instances of a doublet in $V(T)$ have different entries in *Score* table. Only valid local suppressions are assigned scores. The global suppression scores of all instances of a doublet are the same. Lines 4-15 iteratively select a doublet p with the highest score in *Score* table to suppress. According to whether the highest score is obtained from *local suppression* or *global suppression*, our algorithm performs different strategies. For local suppression, the algorithm identifies the set of MVS, denoted by V', that will be eliminated due to locally suppressing

Algorithm 1 Trajectory Database Anonymizer

Input: Raw trajectory database T

Input: Thresholds L, K, C, (K'), and sensitive values S

Output: Anonymous T' satisfying the given $(K, C)_L$-privacy requirement

1: Generate $V(T)$ by Procedure 1;
2: Generate MFS by MFS algorithm and build MFS-tree;
3: Build *Score* table by Procedure 2;
4: **while** *Score* table $\neq \emptyset$ **do**
5: Select a doublet p with the highest score from its MVS m;
6: **if** p is obtained from *local suppression* **then**
7: $V' \leftarrow$ each MVS m' such that $p \in m' \wedge T(m') = T(m)$;
8: Suppress the instances of p from $T(m)$;
9: Delete the MFS containing p if their supports are $< K'$ after the suppression, otherwise update their supports;
10: **else**
11: $V' \leftarrow V(p)$;
12: Suppress all instances of p in T;
13: Delete all MFS containing p from MFS-tree;
14: Update the $Score(p')$ if both p and p' are in V' (or in the same MFS);
15: $V(T) = V(T) - V'$;
16: **return** the suppressed T as T';

p, and removes the instances of p from the records $T(m)$. One extra step is performed for MFS to update the supports of MFS in the MFS-tree (Line 9). For global suppression, the algorithm removes all the MVS containing p, and suppresses all instances of p from T. For preserving MFS, the MFS containing p are removed from the MFS tree (Line 13). Line 14 updates the *Score* table, which requires two tasks: 1) checking if the doublets affected by the current suppression are valid for future local suppressions; and 2) calculating the scores for such doublets. Specifically, for preserving MFS, a special data structure, *MFS-tree*, is created to facilitate the anonymization.

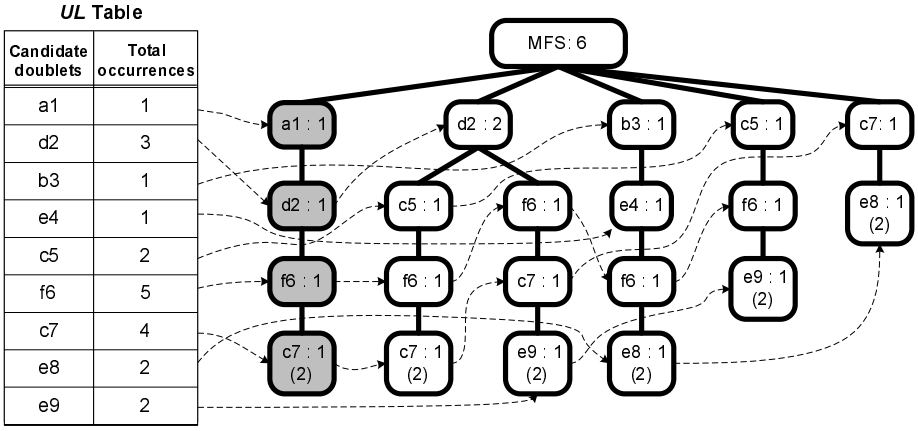


Figure 1: MFS-tree for efficient *Score* updates

Definition 4.4 (MFS-tree). MFS-tree is a tree structure that represents each MFS as a tree path from root to leaf. The support of each MFS is stored at its leaf node. Each node keeps track of a count of MFS sharing the same prefix. The count at the root is the total number of MFS. MFS-tree has a UL table that keeps the total occurrences of every candidate doublet p. Each candidate doublet p in the UL table has a link, denoted by $Link_p$, that links up all the nodes in MFS-tree containing p. ■

Example 4.5. Figure 1 presents the MFS-tree generated from Table 1 with $K' = 2$. To find all the MFS containing $f6$, simply follow $Link_{f6}$, starting from the $f6$ entry in the UL table. ■

4.3. Complexity Analysis

Our anonymization algorithm consists of two steps. In the first step, we identify all MVS. The most expensive operation is scanning the raw trajectory database T once for all sequences in each candidate set C_i. The cost is $\sum_{i=1}^{L} |C_i|i$, where $|C_i|$ is the size of candidate set C_i. The size of C_1 is the number of distinct doublets in T whose upper limit is $|d|$, the number of

dimensions. Since C_2 is generated by self-joining all doublets in U_1, whose size is less than or equal to $|C_1|$, its upper bound is $|d|(|d|-1)/2$. However, when $i \geq 3$, the sizes of the candidate sets do not increase significantly for two reasons: 1) All candidates are generated by self-joining, which requires that only if two sequences share the same prefix, their resulting sequence can be considered a future candidate. When i is relatively large, the chance of finding two such sequences decreases significantly. 2) The pruning process in Procedure 1 also greatly reduces the candidate search space. Therefore, a good approximation is $C \approx |d|^2$. However, in the worst case, the computational cost of the first step is bounded by $O(|d|^L|T|)$, where $|T|$ is the number of records in T. In the second step, we construct the *Score* table, and then remove all MVS iteratively. The most costly operation is to check if the instances of the doublets in $V(T)$ are valid to be locally suppressed. The number of instances of doublets in $V(T)$ is less than $\sum_{i=1}^{L} |C_i|i$, and thus also bounded by $|d|^L$. For every instance in $V(T)$, we need to invoke Procedure 2 *at most* twice. For each invocation, in the worst case, it has to go through all records in T. So the cost of the second step is still bounded by $O(|d|^L|T|)$. By incorporating both steps, the complexity of the entire algorithm is $O(|d|^L|T|)$. The scalability of our algorithm is further demonstrated in Section 5.2.

5. Experimental Evaluation

In this section, we examine the performance of our anonymization framework in terms of *utility loss* due to the anonymization and *scalability* for handling large data sets. For preserving *instances*, the utility loss is defined

Table 3: Experimental data set statistics

Data sets	Records $\|T\|$	Dimensions $\|d\|$	Data size (K bytes)	Sensitive set cardinality	Data type
City80K	80,000	624	2,297	1/5	Synthetic
STM460K	462,483	3,264	9,810	6/24	Real-life

as $\frac{N(T)-N(T')}{N(T)}$, where $N(T)$ and $N(T')$ are the numbers of instances of doublets in the original data set T and the anonymous data set T' respectively; for preserving *MFS*, the utility loss is defined as $\frac{|U(T)|-|U(T')|}{|U(T)|}$, where $|U(T)|$ and $|U(T')|$ are the numbers of MFS in T and T' respectively. The formulas respectively measure the percentage of instances and MFS that are lost due to suppressions. Lower utility loss implies better resulting data quality. We cannot directly compare our algorithm with previous works [1][29][31][40] on trajectory data anonymization because none of them can prevent from both identity and attribute linkage attacks. Instead, we compare our local suppression method with the global suppression method described in our technical report [25]. In the following experiments, we show that applying local suppression along with $(K, C)_L$-privacy would significantly lower utility loss in the context of trajectory data.

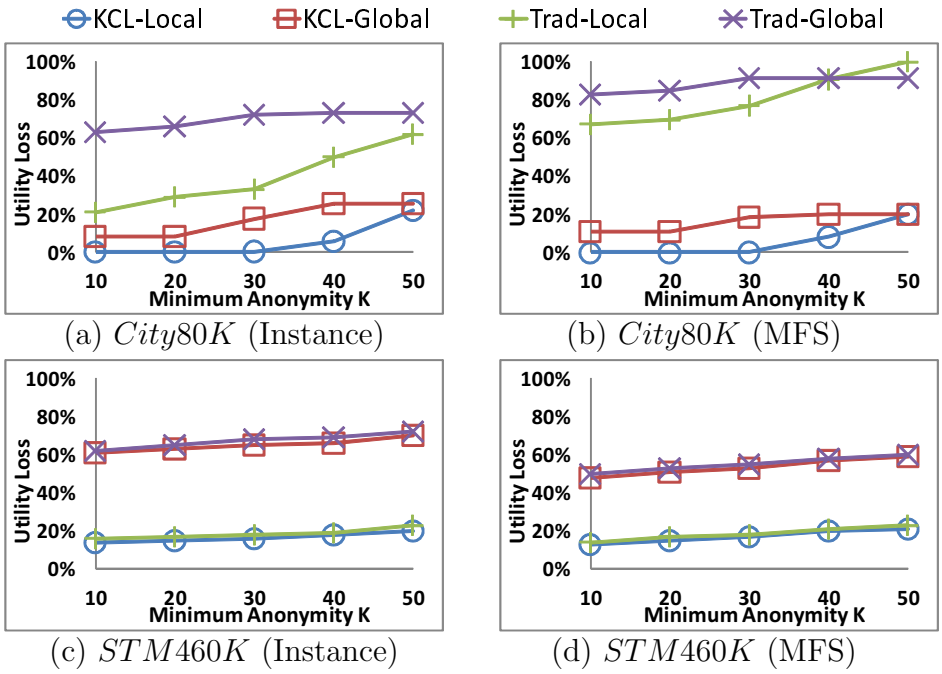
Two data sets, $City80K$ and $STM460K$, are used in the experiments. $City80K$ is a *synthetic* data set simulating the routes of 80,000 pedestrians roaming in a metropolitan area of 26 blocks in 24 hours. The sensitive attribute of $City80K$ contains a total of five possible values, one of which is considered as sensitive. $STM460K$ is a *real-life* data set provided by *Société de transport de Montréal* (STM), the public transit agency in Montréal. It contains the transit data of 462,483 passengers among 68 subway stations within 48 hours, where the time granularity is set to hour level.

The passengers' fare types are currently considered as the sensitive attribute. It contains 24 distinct values and 6 of them are considered as sensitive. The properties of the two experimental data sets are summarized in Table 3.

5.1. Utility Loss

To fully study the effectiveness of our anonymization algorithm, we evaluate the utility loss in terms of varying K, C, L values. Specifically, for preserving MFS, we also study the effect of varying K' values. Instead of examining the effect of L separately, we show the benefit of a reasonable L value over the traditional k-anonymity (confidence bounding) in combination with other parameters. In Figures 2-4, the following legends are used: *KCL-Local* uses local suppression for $(K, C)_L$-privacy; *KCL-Global* uses global suppression for $(K, C)_L$-privacy [25]; *Trad-Local* uses local suppression for traditional k-anonymity (confidence bounding); *Trad-Global* uses global suppression for traditional k-anonymity (confidence bounding).

Effect of K. We vary the parameter K from 10 to 50 while fixing $L = 3$, $C = 60\%$, and $K' = 800$, on both $City80K$ and $STM460K$ to study the effect of K on $(K, C)_L$-privacy model under the two different utility metrics, the results of which are demonstrated in Figure 2. Recall that k-anonymity is achieved in our framework by setting $L = |d|$ and $C = 100\%$, where $|d|$ is the number of dimensions in the given data set. Comparing the utility loss of the schemes based on $(K, C)_L$-privacy to the ones based on k-anonymity unveils the utility improvement due to the assumption of L-knowledge; comparing the schemes using local suppression to those using only global suppression unveils the utility enhancement due to the employment of local suppression. Overall, *KCL-Local* performs significantly better than *KCL-Global*. In par-



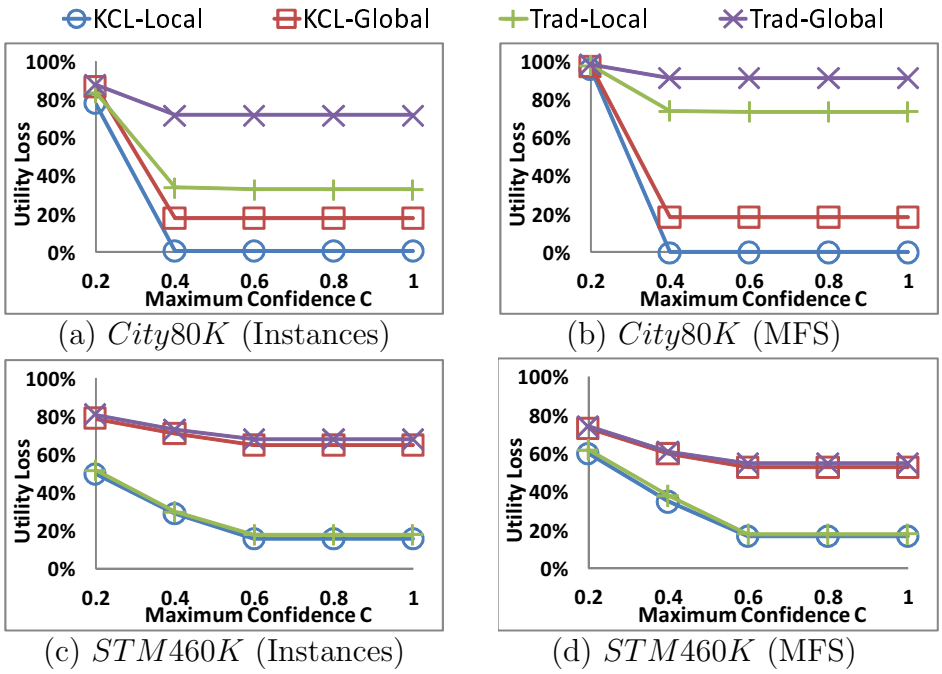
(a) $City80K$ (Instance) (b) $City80K$ (MFS)

(c) $STM460K$ (Instance) (d) $STM460K$ (MFS)

Figure 2: Utility loss vs. K ($L = 3, C = 60\%, K' = 800$)

ticular, it achieves 75% improvement for instance and 68% improvement for MFS on the real data set $STM460K$. However, local suppression itself is not sufficient to guarantee good data utility. When local suppression is applied to k-anonymity, the resulting utility loss is still relatively high on $City80K$. It is interesting to see that on $STM460K$ the utility loss under $(K, C)_L$-privacy and k-anonymity is very close. This is due to the fact that most MVS of $STM460K$ are of size-3 or less. Nevertheless, Figure 2 suggests that when combined with local suppression, $(K, C)_L$-privacy can significantly lower the utility loss than can k-anonymity, in the context of trajectory data.

Effect of C. Figure 3 shows the impact of C on the utility loss while fixing $L = 3$, $K = 30$, and $K' = 800$, which allows us to examine the effect of attribute linkages. Since k-anonymity is unable to prevent attribute linkages,



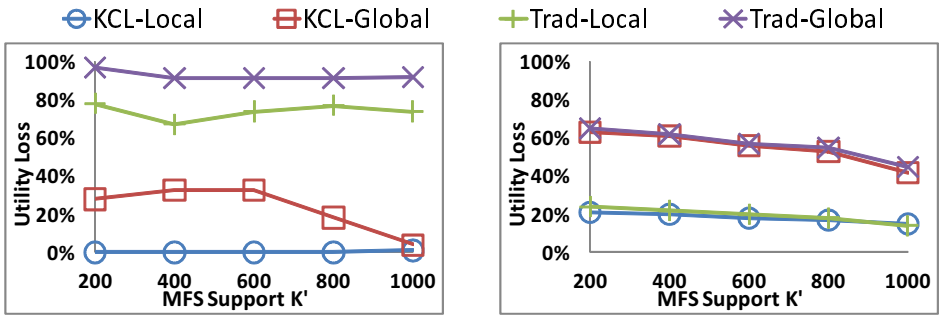
(a) $City80K$ (Instances) (b) $City80K$ (MFS)

(c) $STM460K$ (Instances) (d) $STM460K$ (MFS)

Figure 3: Utility loss vs. C ($L = 3, K = 30, K' = 800$)

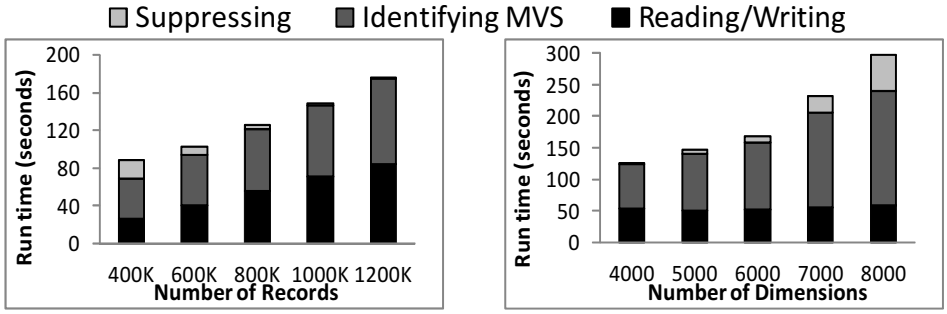
confidence bounding [34] is used to compare with $(K, C)_L$-privacy. Recall that confidence bounding is achieved under $(K, C)_L$-privacy by setting $L = |d|$. When C is small, the utility loss is high for all anonymization schemes because approximately 20% of the records of $City80K$ and 25% of the records of $STM460K$ contain a sensitive value. However, as C increases, the utility loss becomes less sensitive to C. The result also suggests that applying local suppression under $(K, C)_L$-privacy results in substantially lower utility loss.

Effect of K'. For preserving MFS, we study the relationship between K' and the utility loss by fixing $L = 3$, $K = 30$, and $C = 60\%$ in Figure 4. Generally, as K' increases, the utility loss decreases. When K' gets larger, the size of MFS becomes smaller, which, in turn, makes the MFS set and MVS set have less overlap. Hence, suppressions have less influence on MFS.



(a) $City80K$ (b) $STM460K$

Figure 4: Utility loss vs. K' ($L = 3, K = 30, C = 60\%$)



(a) *Runtime vs. # of records* (b) *Runtime vs. # of dimensions*

Figure 5: Scalability

We also observe that local suppression is less sensitive to varying K' values due to the fact that local suppression allows decreasing the support of an MFS rather than always totally eliminating an MFS.

5.2. Scalability

Since the computational complexity of our algorithm is dominated by $|d|$, the number of dimensions, and $|T|$, the number of records, we study the scalability of our anonymization framework in terms of $|d|$ and $|T|$ on relatively large trajectory data sets generated with similar settings as $City80K$. Since using local suppression results in better data utility, we only evaluate the scalability of applying local suppression for preserving MFS (using only

global suppression requires less computing resources), where the following parameters are used: $L = 3$, $K = 30$, $C = 60\%$, and $K' = 800$.

Effect of $|T|$. Figure 5 (a) presents the run time of processing data sets with 4000 dimensions and size ranging from 400,000 to 1,200,000. We can observe that the time spent on reading raw data sets and writing the anonymized data sets is proportional to the data set sizes. The time of identifying MVS sets also increases linearly, which confirms our analysis in Section 4.3. With the increase of the data size, the time spent on suppressions, however, drops substantially. When the number of records increases, there is a much greater chance for a sequence q to satisfy $|T(q)| \geq K$; therefore, the size of MVS decreases significantly, so it takes much less time to perform all suppressions.

Effect of $|d|$. In Figure 5 (b), we increase the dimensions on data sets of 1 million records. The time spent reading raw data and writing anonymized data is insensitive to the number of dimensions of the given data set. However, as the number of dimensions increases, it takes more time to generate the MVS set because the size of each candidate set increases. The size of the resulting MVS set also increases due to the increased sparseness. Thus, the time spent on suppressing all identified MVS also increases.

Overall, our anonymization framework is able to efficiently process large trajectory data sets. The total run time of anonymizing 1 million records with 8000 dimensions is still less than 300 seconds.

6. Conclusions

In this paper, we summarize the special challenges of trajectory data anonymization and show that traditional k-anonymity and its extensions are not effective in the context of trajectory data. Based on the practical assumption of L-knowledge, we achieve a $(K, C)_L$-privacy model on trajectory data without paying extra utility and computation costs due to over-sanitization. This is the first paper that introduces local suppression to trajectory data anonymization to enhance the resulting data utility. Consequently, we propose an anonymization framework that is able to remove all privacy threats from a trajectory database by both local and global suppressions. This framework is independent of the underlying data utility metrics and, therefore, is suitable for different trajectory data mining workloads. Our experimental results on both synthetic and real-life data sets demonstrate that combining $(K, C)_L$-privacy and local suppression is able to significantly improve the anonymized data quality.

Though we adopt a stronger privacy notion than other existing works, in the context of trajectory data, by taking into consideration the possibility of inferring record owners' sensitive information via trajectory data, the specificity of trajectory data enables adversaries to perform other kinds of privacy attacks, especially when they are equipped with different types of background knowledge. These are interesting and open research problems, which are considered as our future research directions.

7. Acknowledgment

The research is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grants and Canada Graduate Scholarships, and Le Fonds québécois de la recherche sur la nature et les technologies (FQRNT) new researchers start-up program.

References

[1] O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *Proc. of the 24th IEEE International Conference on Data Engineering*, pages 376–385, 2008.

[2] C. C. Aggarwal. On k-anonymity and the curse of dimensionality. In *Proc. of the 31st International Conference on Very Large Data Bases*, pages 901–909, 2005.

[3] C. C. Aggarwal and P. S. Yu. A condensation approach to privacy preserving data mining. In *Proc. of the 9th International Conference on Extending Database Technology*, pages 183–199, 2004.

[4] D. Burdick, M. Calimlim, and J. Gehrke. MAFIA: a maximal frequent itemset algorithm for transactional databases. In *Proc. of the 17th IEEE International Conference on Data Engineering*, pages 443–452, 2001.

[5] C. Dwork. Differential privacy. In *Proc. of the 33rd International Colloquium on Automata, Languages and Programming*, pages 1–12, 2006.

[6] B. C. M. Fung, K. Al-Hussaeni, and M. Cao. Preserving RFID data privacy. In *Proc. of the 3rd Annual IEEE International Conference on RFID*, pages 200–207, 2009.

[7] B. C. M. Fung, M. Cao, B. C. Desai, and H. Xu. Privacy protection for RFID data. In *Proc. of the 24th ACM Symposium on Applied Computing*, pages 1528–1535, 2009.

[8] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys*, 42(4):14:1–14:53, June 2010.

[9] B. C. M. Fung, K. Wang, and P. S. Yu. Anonymizing classification data for privacy preservation. *IEEE Transactions on Knowledge and Data Engineering*, 19(5):711–725, 2007.

[10] G. Ghinita, Y. Tao, and P. Kalnis. On the anonymization of sparse high-dimensional data. In *Proc. of the 24th IEEE International Conference on Data Engineering*, pages 715–724, 2008.

[11] F. Giannotti, M. Nanni, D. Pedreschi, and F. Pinelli. Trajectory pattern mining. In *Proc. of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 330–339, 2007.

[12] Y. He and J. F. Naughton. Anonymization of set-valued data via top-down, local generalization. In *Proc. of the 35th International Conference on Very Large Data Bases*, pages 934–945, 2009.

[13] S. Kisilevich, L. Rokach, Y. Elovici, and B. Shapira. Efficient multidimensional suppression for k-anonymity. *IEEE Transactions on Knowledge and Data Engineering*, 22(3):334–347, 2010.

[14] J.-G. Lee, J. Han, X. Li, and H. Gonzalez. Traclass: trajectory classification using hierarchical region-based and trajectory-based clustering. In *Proc. of the 34th International Conference on Very Large Data Bases*, pages 1081–1094, 2008.

[15] J.-G. Lee, J. Han, and K.-Y. Whang. Trajectory clustering: a partition-and-group framework. In *Proc. of the 2007 ACM SIGMOD International Conference on Management of Data*, pages 593–604, 2007.

[16] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *Proc. of the 22nd IEEE International Conference on Data Engineering*, page 25, 2006.

[17] T. Li and N. Li. Injector: Mining background knowledge for data anonymization. In *Proc. of the 24th IEEE International Conference on Data Engineering*, pages 446–455, 2008.

[18] T. Li and N. Li. On the tradeoff between privacy and utility in data publishing. In *Proc. of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 517–526, 2009.

[19] X. Li, J. Han, and S. Kim. Motion-alert: Automatic anomaly detection in massive moving objects. In *Proc. of IEEE International Conference on Intelligence and Security Informatics*, pages 166–177, 2006.

[20] X. Li, J. Han, J.-G. Lee, and H. Gonzalez. Traffic density-based discovery of hot routes in road networks. In *Proc. of the 10th International Symposium on Spatial and Temporal Databases*, pages 441–459, 2007.

[21] A. Machanava, J. Gehrke, and M. Gotz. Data publishing against realistic adversaries. In *Proc. of the 35th International Conference on Very Large Data Bases*, pages 790–801, 2009.

[22] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. ℓ-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data*, 1(1), 2007.

[23] N. Matatov, L. Rokach, and O. Maimon. Privacy-preserving data mining: A feature set partitioning approach. *Information Sciences*, 180(4):2696–2720, 2010.

[24] N. Mohammed, B. C. M. Fung, and M. Debbabi. Walking in the crowd: anonymizing trajectory data for pattern analysis. In *Proc. of the 18th ACM Conference on Information and Knowledge Management*, pages 1441–1444, 2009.

[25] N. Mohammed, B. C. M. Fung, and M. Debbabi. Preserving privacy and utility in RFID data publishing. Technical Report 6850, Concordia University, Montreal, Canada, September 2010.

[26] N. Mohammed, B. C. M. Fung, P. C. K. Hung, and C. K. Lee. Anonymizing healthcare data: a case study on the blood transfusion service. In *Proc. of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1285–1294, 2009.

[27] M. O'Halloran and M. Glavin. RFID patient tagging and database system. In *Proc. of the International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies*, page 162, 2006.

[28] M.-P. Pelletier, M. Trepanier, and C. Morency. Smart card data in public transit planning: a review. Technical Report CIRRELT-2009-46, Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, November 2009.

[29] R. G. Pensa, A. Monreale, F. Pinelli, and D. Pedreschi. Pattern-preserving k-anonymization of sequences and its application to mobility data mining. In *Proc. of the 1st International Workshop on Privacy in Location-Based Applications*, 2008.

[30] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information. In *Proc. of the 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, page 188, 1998.

[31] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In *Proc. of the 9th International Conference on Mobile Data Management*, pages 65–72, 2008.

[32] M. Terrovitis, N. Mamoulis, and P. Kalnis. Privacy-preserving anonymization of set-valued data. In *Proc. of the 34th International Conference on Very Large Data Bases*, pages 115–125, 2008.

[33] M. Utsunomiya, J. Attanucci, and N. Wilson. Potential uses of transit smart card registration and transaction data to improve transit planning. *Transportation Research Record: Journal of the Transportation Research Board*, (1971):119–126, 2006.

[34] K. Wang, B. C. M. Fung, and P. S. Yu. Handicapping attacker's confidence: An alternative to k-anonymization. *Knowledge and Information Systems*, 11(3):345–368, 2007.

[35] D. Wegener, D. Hecker, C. Korner, M. May, and M. Mock. Parallelization of r-programs with gridr in a gps-trajectory mining application. In *Proc. of the 1st Ubiquitous Knowledge Discovery Workshop in conjunction with European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2008.

[36] R. C. W. Wong, J. Li, A. W. C. Fu, and K. Wang. (α,k)-anonymous data publishing. *Journal of Intelligent Information Systems*, 33(2):209–234, October 2009.

[37] X. Xiao and Y. Tao. Personalized privacy preservation. In *Proc. of the 2006 ACM SIGMOD International Conference on Management of Data*, pages 229–240, 2006.

[38] Y. Xu, B. C. M. Fung, K. Wang, A. W. C. Fu, and J. Pei. Publishing sensitive transactions for itemset utility. In *Proc. of the 8th IEEE International Conference on Data Mining*, pages 1109–1114, 2008.

[39] Y. Xu, K. Wang, A. W. C. Fu, and P. S. Yu. Anonymizing transaction databases for publication. In *Proc. of the 14th ACM SIGKDD Inter-*

national Conference on Knowledge Discovery and Data Mining, pages 767–775, 2008.

[40] R. Yarovoy, F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Anonymizing moving objects: How to hide a MOB in a crowd? In *Proc. of the 12th International Conference on Extending Database Technology*, pages 72–83, 2009.