Perturbation-Driven Anonymization of Trajectories

Mehmet Ercan Nergiz
CS Dept., Purdue University
mnergiz@cs.purdue.edu

Maurizio Atzori
KDD Laboratory, ISTI-CNR
atzori@di.unipi.it

Yucel Saygin
CS Prog., Sabanci University
ysaygin@sabanciuniv.edu

Abstract

Trajectory datasets are becoming more and more popular due to the massive usage of GPS devices. In this paper, we address privacy issues regarding the identification of individuals in static trajectory datasets. We provide privacy protection by (1) first enforcing k-anonymity, meaning every released information refers to at least k users/trajectories, (2) then reconstructing randomly a representation of the original dataset from the anonymization. We present a utility metric that maximizes the probability of a good representation and propose trajectory anonymization techniques to address time and space sensitive applications. The experimental results over synthetic trajectory datasets show the effectiveness of the proposed approach.

1. Introduction

Data publishing is essential for providing resources for research and for the transparency of government institutions and companies. However, data publishing is also risky since published data may contain sensitive information. Therefore, the first step before data publishing is to remove the personally identifying information. But it has been shown that removing personally identifying information is not enough to protect the privacy. This is due to the fact that the released database can be linked to public databases through a set of common attributes which are called quasi-identifiers. For example in US the combination of zip code, and birth date is unique for 87% of the citizens[21]. This figure increases as more attributes are added to the combination. Sweeney et al showed that they could re-identify the supposedly anonymous health records via linking them to voters registration list that can be purchased from, and they were actually able to recover the health records of the government of Massachusets. This striking result increased the concerns and research efforts for privacy and anonymization in published databases. The problem of linkage becomes even more complicated in our highly connected world as the number and variety of data sources increase.

One of the important data sources in todays ubiquitous environments are mobile devices and RFID tags. Mobile service providers can now predict the location of mobile users via triangulation with a high precision. Coupled with applications such as location based services that are enabled by GPS enabled mobile devices, it is now very easy to track the location of individuals voluntarily or non-voluntarily over a period of time. The time and location information of a person (or a moving object in general) collected over a period of time forms a trajectory which can be thought of as a set of spatio-temporal data points spanning a time interval. Trajectory data sets contain valuable information which can be harvested by data mining tools to obtain models for applications such as city traffic planning, or determining emergency evacuation routes. However, time and location is also very sensitive information, therefore personally identifying information needs to be removed from trajectories before they can be released. Even after de-identification, trajectory data sets are still prone to linkage attacks since space and time attributes are very powerful quasi-identifiers. For example, for a trajectory that starts at a specific location every weekday in the morning and reaches another location in an hour, it is very easy to infer that the starting location in the morning is home, and the location reached after an hour is the work place. What an adversary can do is to look at a phone directory to search for home addresses and work addresses to link the trajectories with their owners.

The solution in general to prevent linkage attacks in de-identified data sets is anonymization [21, 20]. k-Anonymity was proposed as a standard for privacy which can be summarized as safety in numbers and ensures that every entity in the dataset is indistinguishable from $k-1$ other entities. Achieving optimal k-anonymity was proven to be NP-Hard, therefore heuristic algorithms have been proposed in the literature to k-anonymize data sets. In case of spatio-temporal trajectories the problem of anonymization is even harder since consecutive points in a trajectory are dependent to each other. Therefore anonymization should consider every trajectory as a whole when anonymizing. In this paper, we concentrate on spatio-temporal trajectories. We first extend the notion of k-anonymity for trajectories and then describe a heuristic method for achieving k-anonymity of trajectories. Trajectories are published by only releasing a representative trajectory to further protect the privacy.

2. Related Work

2.1. k-Anonymity and Privacy over Databases

Addressing privacy concerns when releasing person specific datasets is well studied in the literature. [20, 15, 3, 14, 17] Simply removing uniquely identifying information (SSN, name) from data is not sufficient to prevent identification because partially identifying information (quasi-identifiers (QI); age, sex, city ...) can still be mapped to individuals by using external knowledge[21]. k-Anonymity is defined in [20], to protect against identification of individuals in person specific datasets.

Definition 1 (k-Anonymity) *A table T^* is k-anonymous w.r.t. a set of attributes QI if each record in $T^*[QI]$ appears at least k times.*

k-Anonymity property ensures that a given set of quasi identifiers can only be mapped to at least k entities in the dataset. The most common technique being used to anonymize a given dataset is value generalizations and suppressions. In multidimensional space, the counter part of these operations is replacing a set of points with the minimum bounding box that covers the points. It should be noted that k-anonymization preserves the truth of the data.

Entities in trajectory datasets are more complex than those studied by classical k-anonymity approaches. Anonymization of complex entities was proposed in [19] where data about private entities reside in multiple datasets of a relational database. Even though trajectory datasets can be represented in relational databases, order of points over a given trajectory matters due to the linear time property. Work in [19] does not assume any ordering between points. Also applications over trajectory databases are very specific and require different cost metrics and different anonymization techniques.

2.2. Privacy-preserving Location-based Services and Trajectories

There has been many work on privacy issues regarding the use of LBSs by mobile users. Most work defined the privacy risk as linking of requests and locations to specific mobile users. Works in [7, 12] used perturbation and obfuscation techniques to deidentify a given request or a location and they differ from this work in the privacy constraints they enforce. Anonymization based privacy protection was used in [8, 4, 9, 10]. In [10], anonymity was enforced on sensitive locations other than user location points or trajectories. In [8, 9], individual location points belonging to a user is assumed to be unlinked and points of the users are anonymized other than the trajectories. The work in [4] is the closest work to trajectory anonymization. Anonymization process enforces points refering to same set of users to be anonymized together always. However work assumes anonymization per request other than whole trajectory anonymization and heuristic to specify groups of users is quite simple.

To the best of our knowledge, all of the proposed privacy preservation methods on LBSs so far assume a dynamic, real-time environment and methodology being used is based on local decisions. This work differs in this respect since we address the privacy concerns when publishing static trajectory databases and we make use of global decisions. Also no previous work measured the level of distortion due to anonymization in the context of trajectory mining applications which we believe to be the real purpose in data publishing.

3. Problem Formulation

3.1. Notation

We assume the space is discretized into $\epsilon_s \times \epsilon_s$ size grids and a *point* in our domain is actually a grid or set of grids. All space measurements are in units of ϵ_s. We assume *time* is also discretized into buckets of size ϵ_t and domain of time is finite. So datasets act as the snapshots of the world in many time instances. Datasets with continuous time and space domains can be fit into this assumption by the use of interpolations. The level of granularity in discretization does not affect the efficiency of the proposed methodology.

We define a trajectory database in an object-oriented way. A trajectory dataset T is a set of private entities or trajectories(e.g., $T = \{tr_1, \cdots, tr_n\}$). Each private entity tr_i is an ordered set of spatio-temporal 3D volumes (e.g., points) composed of time, x, and y dimensions (e.g., $tr_i = \{p_1, \cdots, p_m\}$ where $p_k = < t_k, x_k, y_k >$). We assume that the t_i, x_i and y_i components are range of values defined as $t_i : [t_i^1 - t_i^2]$ $x_i : [x_i^1 - x_i^2]$ and $y_i : [y_i^1 - y_i^2]$. Each tr_i is ordered by their subtime component t_i^1. tr_is refer to the individuals and each triplet specifies the area location of the individual at some time in the corresponding time interval. We use the following notation for components to express their length; $|x_i| = |x_i^1 - x_i^2|$, $|y_i| = |y_i^1 - y_i^2|$, $|t_i| = |t_i^1 - t_i^2|$. We also use '$\cdot$' operator to refer to a specific component of a bigger set. (e.g., $tr_i.p_j$: jth point of the ith trajectory)

We say a trajectory tr_1 is a *subset* of another trajectory tr_2 and write $tr_1 \subset tr_2$ if for each point $p_j \in tr_2$, we have some unique $p_i \in tr_1$ such that $t_i^1 \leq t_j^1$, $t_i^2 \geq t_j^2$, $x_i^1 \leq x_j^1$, $x_i^2 \geq x_j^2$, $y_i^1 \leq y_j^1$, $y_i^2 \geq y_j^2$. We say a trajectory tr is atomic if $|x_i| = |y_i| = |t_i| = 1$ for every $p_i \in tr$. We use the notation BB_P for the 3D point with minimum volume that covers all points inside set P (e.g., minimum bounding box).

3.2. Problem Definition

We assume that prior to release, the trajectory database is complete and static. No uniquely identifying information is released. However we assume that we have adversaries that may

- already know some portion of the trajectory of an individual in the dataset and may be interested in the rest. (e.g., adversary knows that a particular person lives in a particular house. He also knows that she leaves the house and comes back home at specified times. He is interested in finding the locations she visited.)
- already know the whole trajectory of an individual but be interested in some sensitive information about the individual. This is a concern if some sensitive info is also released, as part of the database, for some of the spatio-temporal triplets or for some individuals. Sensitive info may be in the form of the requests done by the individual to location based services.

We protect privacy of the individuals against the above adversary by using the following techniques

- k-Anonymity: anonymize the dataset so that every trajectory is indistinguishable from $k-1$ other trajectories.
- Reconstruction: release atomic trajectories sampled randomly from the area covered by anonymized trajectories.

k-Anonymity limits adversary's ability in linking any information to an individual. Reconstruction further prevents leakage due to anonymization. Both techniques are discussed in Sections 4 and 5.

Since reconstruction is just sampling from anonymized data, the amount of privacy-utility depends only on the anonymization. As an anonymization is required to satisfy the privacy constraints, it also needs to maximize the utilization. An anonymization with a reconstruction that better explains the data is considered to be highly utilized. However the amount of utilization also depends on the target applications. Although there may be many classes of target applications, in this work, we consider two of them:

Time Sensitive Applications: This class covers the applications in which the time component is crucial compared to space components. Trajectories that have similar paths in space, but occur in different time periods are considered to be far away from each other. Such applications include mining traffic data to monitor traffic jams, anomaly detection when timely access control constraints are in place, etc.

Space Sensitive Applications: Similarities are calculated w.r.t. space. *Time shifted* trajectories or trajectories with different velocities can be considered to be close. Target applications include mining the world for region popularity to make business decisions, measuring road erosion caused by vehicles for maintenance, etc.

Section 5 discusses that some anonymization tr^* of tr minimizing the following equation (*log cost metric*) also maximizes the probability of generating the exact dataset.

$$LCM(tr^*) = \sum_{p_i \in tr^*} [w_s(\log|x_i| + \log|y_i|) + w_t \log|t_i|] + (|tr| - |tr^*|) \cdot (w_s \log S + w_t \log T) \quad (1)$$

where w_s and w_t are weights to adjust sensitivity to space and time respectively and, S and T are universal space and time according to their domain.

From now on, our objective is to minimize Equation 3 while respecting k-anonymity in anonymizations.

4. k-Anonymity in Trajectory Databases

In this section, we redefine the k-anonymity notion for sets of trajectories. Next, we use a condensation based approach to form groups of *similar* trajectories. Last, we show how to k-anonymize trajectories in a given group. Anonymization process will be dependent on the selection of metric parameters being used for grouping.

4.1. k-Anonymity for Trajectory Databases

Original k-anonymity prevents an adversary from identifying a given QI to be in a set with less than k elements in the anonymized dataset. Since we assume adversaries know about all or some of the spatio-temporal points about an individual, the set of all points corresponding to a trajectory becomes the quasi identifiers in our domain. To enforce k-anonymity against such an adversary, we require the following property to hold in a given anonymization T^* of trajectory dataset T:

$$|\{tr^* \in T^* \mid tr \subset tr^*\}| \geq k \qquad \forall\, tr \in T$$

This implies that a given trajectory in the original db can at best be linked to at least k trajectories in the anonymized db. It can be shown easily that the following definition for

k-anonymity satisfies the requirement and also preserves the truth of the original dataset:

Definition 2 (k-Anonymity for Trajectory Databases) *A trajectory database T^* is a k-anonymization of a trajectory dataset T if*

- *for every trajectory in T^*, there are at least $k-1$ other trajectories with exactly the same set of points.*
- *there is a one to one relation between the trajectories $tr \in T$ and trajectories $tr^* \in T^*$ such that $tr \subset tr^*$.*

Following definitions show how to create anonymization of a set of trajectories.

Definition 3 (Point Link and Matching) *A point link between a set of trajectories $TR = \{tr_1, \cdots, tr_n\}$ is an ordered set of points $PL = \{p_1, \cdots, p_n\}$ such that $p_i \in tr_i$. An ordered set of point links between trajectories in TR, $PM = \{PL_1, \cdots, PL_m\}$, is a point matching between the trajectories if for all $i < j$ and all possible k, $PL_i.t_k^1 < PL_j.t_k^1$.*

Figure 1.d shows a point matching between trajectories tr_1, tr_2, and tr_3. Note that point links are ordered, do not overlap and there may be unmatched points in any of the trajectories.

Theorem 1 *Let $TR = \{tr_1, \cdots, tr_n\}$ be a set of trajectories and $PM = \{PL_1, \cdots, PL_m\}$ be a valid point matching between them. Let $TR^* = \{tr_1^*, \cdots, tr_n^*\}$ be another set such that $tr_1^*.p_i = \cdots = tr_n^*.p_i = BB_{PL_i}$. Then TR^* is an n-anonymization of TR.*

Proof 1 *Since all the n elements in TR^* are the same, the first requirement of anonymity trivially holds. Since each point in tr_j^* is a bounding box for some point in tr_j; $tr_j \subset tr_j^*$. The second requirement also holds.*

Figure 1.c shows the 3-anonymization of tr_1, tr_2, and tr_3 through the point matching in d. Unmatched points are suppressed in the anonymization.

Theorem 1 states that any matching between the points of a given set of trajectories can be used to anonymize the trajectories. Although there are many possible matchings, the aim of the anonymization is to find the one that will minimize the log cost of the output anonymization.

4.2. Trajectory Grouping

Although there are numerous k-anonymity algorithms proposed for single table datasets, a grouping based approach is discussed to be more suitable for the anonymization of complex structures, due to the direct identification of private entities (trajectories in our case) being anonymized [19]. Most clustering algorithms can easily be modified for k-anonymity by enforcing that the size of the clusters should be more than k [2, 18, 6, 1]. The only challenge at this stage is to define a distance metric between trajectories. Since our objective is to minimize the log cost metric, we can define the distance of two trajectories as the cost of their *optimal* anonymization. Having said that the problem reduces to finding the cost optimal anonymization given two trajectories.

Finding the optimal anonymization of two trajectories is the same as finding the point matching between the two trajectories such that anonymizing the trajectories through the matching minimizes the log cost. A similar *alignment* problem is well studied for strings (where the goal is to find an alignment of strings such that total pairwise edit distance between the strings is minimized) in the context of DNA comparisons. Alignment problem for two trajectories is polynomial and can be solved by using a dynamic programming approach. The equation that solves the alignment problem for optimizing against a given incremental function σ is given in Table 1. The log cost function is also incremental and defines σ as follows:

$$\sigma_{LCS}(p_1, p_2) \begin{cases} \log U, & p_2 = \perp; \\ \log BB_{\{p_1,p_2\}}, & \text{otherwise.} \end{cases}$$

where U is the volume of the universal space. So the distance between two trajectories t_1 and t_2 is given by

$$DST(t_1, t_2) = OPT_{\sigma_{LCS}}(t_1, t_2)$$

In this work, we adapted and slightly modified the condensation based grouping algorithm given in [1] for trajectory k-anonymity. Algorithm *multi TGA*, in each iteration, creates an empty group G, randomly samples one trajectory $tr \in TR$, puts tr into G, sets the group representative $rep_G = tr$. Next, the closest trajectory $tr' \in TR - G$ to rep_G is specified (line 6). tr' is added into g and group representative rep_G is updated as the anonymization of rep_G and tr' (line 8). Update of rep_G and G with new trajectories continues until G contains k trajectories. At the end of each iteration, a new group of k trajectories is formed, and is removed from TR. Trajectories in every group is anonymized with each other (details are in the next subsection.). Iteration stops when there are less than k trajectories remaining in TR.

The costly operation in the grouping algorithm is finding the closest trajectory to the group representative (line 6). This nearest neighbor operation needs to be done $|TR|$ many times and it is difficult to speed up each operation by indexing (this is because our distance metric does not satisfy triangular inequality). To decrease the number of operations, we also try another version of algorithm 1 (*fast*

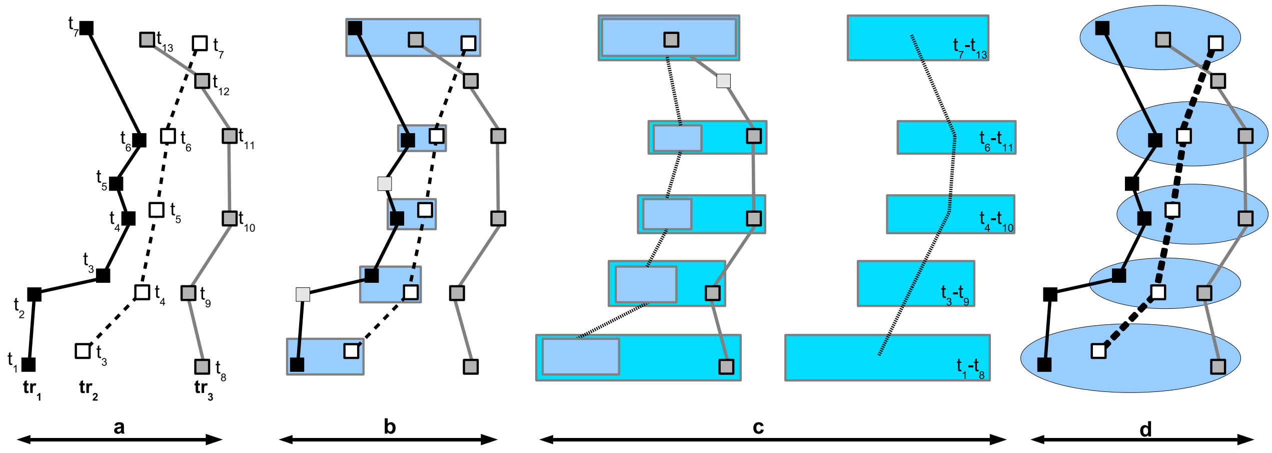

Figure 1. Anonymization Process
a. trajectories tr_1,tr_2, and tr_3; b. anonymization tr^* of tr_1 and tr_2; c. anonymization of tr^* and tr_3; d. point matching

Table 1. Optimal Alignment Optimizing Against Metric σ

$$
OPT_\sigma(tr_1, tr_2)\begin{cases}
\sum_{p_i \in tr_1} \sigma(p_i, \perp), & |tr_2| = 0; \\
\sum_{p_i \in tr_2} \sigma(p_i, \perp), & |tr_1| = 0; \\
min\{OPT_\sigma(tr_1 - tr_1.p_1\ ,\ tr_2 - tr_2.p_1) + \sigma(tr_1.p_1, tr_2.p_1), & \\
OPT_\sigma(tr_1\ ,\ tr_2 - tr_2.p_1) + \sigma(tr_2.p_1, \perp), & \\
OPT_\sigma(tr_1 - tr_1.p_1\ ,\ tr_2) + \sigma(tr_1.p_1, \perp)\}, & |tr_1|, |tr_2| > 0.
\end{cases}
$$

TGA) by skipping the update of group representative (e.g., removal of line 8). In this case, $k-1$ closest trajectories to the group representative can be found in one pass so the number of nearest neighbor operations will be $\frac{|TR|}{k}$. The resultant algorithm is faster by a factor of k but expected to be less utilized since it does not directly optimize against log cost function. We will experiment the time/utility relations between fast and multi TGA algorithms in Section 6.

4.3. Anonymization of Trajectories

Once the groups are formed, the trajectories inside each group needs to be anonymized. As mentioned before, the anonymization process needs to specify the optimal point matching that will minimize the log cost. Finding the optimal matching between two trajectories is easy. Algorithm specifies the point pairs between the trajectories by tracing $OPT_{\sigma_{LGM}}$ and anonymize the paired points w.r.t. each other (by replacing the points with the minimum bounding box that covers the points). Any unmatched points are suppressed.

The real challenge is to find the optimal point matching between $n > 2$ trajectories. Similar versions of the problem on strings were proved to be NP-Hard [13]. Trajectory alignment and its complexity is not yet studied. We, now, formalize and prove the NP-Hardness of the trajectory

Algorithm 1 multi TGA(TR, k)

```
Require: Set of trajectories $TR$, integer $k > 1$, the log
    distance metric
Ensure: return $k$-anonymization of the trajectories in $TR$.
 1: repeat
 2:   Let $G$ be an empty group with group representative
      $rep_G$
 3:   Let $tr \in TR$ be a randomly selected trajectory.
 4:   $G = \{tr\}, rep_G = tr$.
 5:   repeat
 6:     Let $tr' \in TR - G$ be the closest trajectory to $rep_G$.
 7:     $G+ = tr'$,
 8:     $rep_G = anonTraj(rep_G, tr')$.
 9:   until $|G| = k$
10:   $anonTraj(G)$
11:   $TR- = G$
12: until $|TR| < k$
13: Suppress remaining trajectories in $TR$.
```

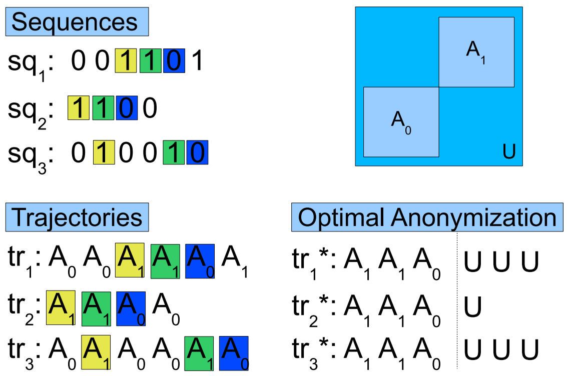

Figure 2. NP-Hardness Reduction Construction

alignment problem:

Definition 4 (Decision Trajectory Alignment (DTA)) *Given a set of trajectories $TR = \{tr_1, \cdots, tr_n\}$ for arbitrary $n > 2$, is there a point matching PM between the trajectories in TR such that the log cost (with arbitrary weights w_s and w_t) of anonymizing TR through PM is at most c?(i.e., is $DTA(TR) \leq c$?)*

Theorem 2 *DTA problem is NP-Hard*

Proof 2 *We first assume the log cost function has parameters $w_s = 1$,$w_t = 0$. We extend the proof for cost functions with arbitrary weight parameters at the end. We reduce DTA to longest common subsequence problem (LCS) which is proved to be NP-Hard for a sequence alphabet of size 2 [16]:*

Definition 5 (Decision Longest Common Subsequence (LCS)) *Given an integer ℓ and a set of sequences $SQ = \{sq_1, \cdots, sq_n\}$ where each $sq_i = \{s_1, \cdots, s_m\}$ is an ordered set of strings from the alphabet $\sum = \{0, 1\}$; is there a common subsequence of sequences in SQ with length at least ℓ? (i.e., is $LCS(SQ) \geq \ell$?)*

For an instance (ℓ, SQ) of LCS, we create the set of input trajectories $TR_{SQ} = \{tr_1, \cdots, tr_n\}$ for DTA, as follows: setting $|tr_i| = |sq_i|$

$$tr_i.p_j = \begin{cases} <[j-j+1],[0-1],[0-1]>, & sq_i.s_j = 0; \\ <[j-j+1],[1-2],[1-2]>, & sq_i.s_j = 1. \end{cases}$$

Figure 2 shows an example trajectory construction for a given set of sequences.

Theorem 3 *For a sequence $SQ = \{sq_1, \cdots, sq_n\}$, $LCS(SQ) \geq \ell$ if and only if $DTA(TR_{SQ}) \leq (t - n \cdot \ell) \cdot \log 4$ where $t = \sum_i |tr_i|$*

Proof 3 *($\overset{onlyif}{\longleftarrow}$) Suppose $sq' = \{s'_1, \cdots, s'_\ell\}$ is one common subsequence, and let in^i_j returns the index of s'_i in sq_j. Observe that $PM = \{PL_1, \cdots, PL_\ell\}$ where $PL_i.p_j = tr_j.p_{in^i_j}$ is a valid point matching for TR_{SQ}. Since $sq_1.s_{in^i_1} = \cdots = sq_n.s_{in^i_n} = s'_i$; we have, using the notation $\overset{S}{=}$ as an equality operator for points having the same spatial components, $PL_i.p_1 \overset{S}{=} \cdots \overset{S}{=} PL_i.p_n$ for every $1 \leq\ i \leq \ell$. This implies that every point in a point link has the same spatial components. So anonymizing TR_{SQ} through PM will match ℓ space-similar points. The final anonymization will have a unit (1 $\times$ 1) area in ℓ positions. Assuming the worst anonymization (in this case, an area of 2 $\times$ 2) for the $t - n \cdot \ell$ points, we have a log cost at most $(t - n \cdot \ell) \log 4 + n \cdot \ell \log 1 = (t - n \cdot \ell) \log 4$.*

($\overset{if}{\longrightarrow}$) Let $PM = \{PL_1, \cdots, PL_r\}$ be the point matching resulting in at most $(t - n \cdot \ell) \log 4$ log cost. Let $PM^0 = \{PL_i \in PM \mid PL_i.p_1 \overset{S}{=} \cdots \overset{S}{=} PL_i.p_n\}$ and $PM^1 = PM - PM^0$. (PM^0 contains the point links that connect space similar points. Every link in PM^1 contains at least two spatially different points.) Since we have only two points in our domain, the points in PM^1 will add a log cost of the whole space (an area of 2×2). The same cost applies also for points unmatched (suppressed). However the points in PM^0 will have unit (1 $\times$ 1) area. Since the total number of points in PM^0 is $n|PM^0|$, we have;

$$\begin{aligned} LCM(TR^*_{SQ}) &\leq (t - n \cdot \ell) \log 4 \\ n|PM^0| \log 1 + (t - n|PM^0|) \log 4 &\leq (t - n \cdot \ell) \log 4 \\ (t - n|PM^0|) \log 4 &\leq (t - n \cdot \ell) \log 4 \\ |PM^0| &\geq \ell \end{aligned}$$

This means that we have a possible matching of size at least ℓ where the points linked to each other are space-similar. The reverse construction of the ($\overset{onlyif}{\longleftarrow}$) proof states that such a matching implies a common subsequence of length at least ℓ.

We ignored the effect of time component in the log cost function ($w_t = 0$) in the above construction. However, the proof can be modified to prove NP-Hardness of any fixed log cost function with any selection of weight parameters. The intuition is to prevent the effect of time component on finding the optimal matching. (The same matching needs to be optimal regardless of the value of w_t.) This can be done by adjusting the domains of space and time components such that increase in cost due to time generalizations will be negligible compared to the cost due to space generalizations. ($w_s \log S >> n \cdot w_t \log T$ where S and T are the universal space and time respectively.)

Given the similar nature of the string and trajectory alignment problems, we adopted the string alignment

Table 2. Reconstruction tr^R of tr^*

$$Pr(tr^R = tr') = \begin{cases} \dfrac{1}{\prod\limits_{p_i \in tr^*} (|x_i| \cdot |y_i| \cdot |t_i|)}, & \text{atomic } tr' \subset tr^*; \\ 0, & \text{otherwise.} \end{cases}$$

heuristic given in [11](where an upper bound on the total pairwise distance for the output alignment is guaranteed.) for trajectory alignment problem. Algorithm given in Alg. 2 uses the following heuristic to come up with a possible alignment of points. Algorithm first identifies the trajectory tr_m whose total pairwise log cost distance with other trajectories is minimum and marks tr_m as done. At each step, $OPT_{\sigma_{LGM}}$ finds the optimal matching between the points of one unmarked trajectory tr_{new} and the current anonymization of the marked trajectories, and marks tr_{new}. Each matching creates links between the points. Point suppressions and generalizations are applied according to the matching. (Figure 1 shows an example anonymization of three trajectories.) In later sections, we show experimentally that alignment heuristic works in practice.

```
Algorithm 2 anonTraj($G$)
Require: a (set) group of trajectories $G$.
Ensure: anonymize the trajectories inside $G$.
 1: let $tr_m \in G$ be the trajectory whose total pairwise distance with other trajectories is minimum.
 2: let set of trajectories $M$ contains initially $tr_m$.
 3: repeat
 4:     let $tr^*$ be the anonymization of trajectories in $M$ through linked points.
 5:     let $tr_{new} \in G - M$ be a randomly chosen trajectory
 6:     run $OPT_{\sigma_{LCM}}$ to find a min cost matching between the points in $tr_{new}$ and $tr^*$
 7:     create links between the points matched by $OPT_{\sigma_{LCM}}$.
 8:     suppress all unmatched points and all points directly or indirectly linked to unmatched points.
 9:     $M = M + tr_{new}$
10: until $M = G$
11: for all unsuppressed point $p$ of each $tr \in M$ do
12:     let $PL$ be the point link containing $p$.
13:     $p = BB_{PL}$
14: end for
```

5. A Randomization Approach to Trajectory k-Anonymity

Trajectory anonymization techniques preserve the truth of the data while providing protection against certain adversaries. However the approach suffers from the following shortcomings.

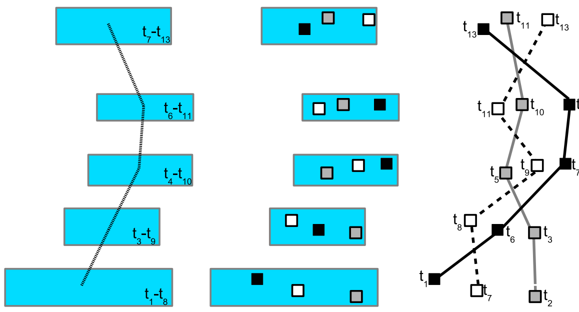

Figure 3. Reconstruction Process

1. Use of minimum bounding boxes in anonymization discloses uncontrolled information about exact locations of the points. (e.g., in the case of two trajectories, two non-adjacent corners give out the exact locations) This information may be critical for applications where existence of a trajectory in a dataset is sensitive (e.g., δ-presence [17]).

2. It is challenging to take full advantage of information contained in anonymizations. Most data mining and statistical applications work on atomic trajectories

The first problem can be weakened by applying some cloaking on the sides of the rectangle or by partitioning the space into grids and returning set of grids covering all points.

The second problem is more tricky as it is a common problem for heterogenous anonymizations with large output domain (most clustering based anonymity algorithms suffer from the same problem.). One proposed technique to solve this issue is reconstruction [18, 1] where an atomic dataset is recreated from the anonymized dataset by uniformly selecting atomic points from anonymized regions. It is experimentally shown in [18] that reconstruction is sufficiently successful in learning from anonymized data.

In this work, we adapt reconstruction approach as a means for privacy protection (as in [1]) rather than information retrieval and release reconstructed data rather than anonymized data. The intuition behind is that reconstruction serves, to the best of our knowledge, as the only solution to learn from the heterogeneous anonymized datasets and it also greatly weakens the first problem without requiring a user input. We define the reconstruction tr^R of trajectory tr^* in Table 2.

An example reconstruction is shown in Figure 3. The output after reconstruction is atomic and suitable for any trajectory applications.

The success of the anonymization heavily depends on the success of the reconstructed data in explaining the original data. Since we have $tr \subset tr^*$ between original trajectory tr and its anonymization tr^*, the probability of generating the original trajectory is non-zero and given by the constant denominator in Table 2 case 1. A good anonymization would maximize this probability.

$$\begin{aligned} & \arg\max_{tr^*} \prod_{p_i \in tr^*} \frac{1}{|x_i|} \cdot \frac{1}{|y_i|} \cdot \frac{1}{|t_i|} \\ = \; & \arg\min_{tr^*} \Big(\sum_{p_i \in tr^*} \log|x_i| + \log|y_i| + \log|t_i| \Big) \quad (2) \end{aligned}$$

The Equation 2 equally weights the effects of time and space on the reconstruction. This is not desirable if we have the class of target applications given in Section 4. So instead, we weight the log cost metric;

$$\sum_{p_i \in tr^*} w_s(\log|x_i| + \log|y_i|) + w_t \log|t_i| \quad (3)$$

Since a given anonymization tr^* of tr does not contain the points suppressed in tr, Equation 3 does not add any log cost regarding those suppressed points. However a suppressed point can be safely thought as a point covering the whole universal space. The final weighted log cost function is given by;

$$\begin{aligned} LCM(tr^*) = & \sum_{p_i \in tr^*} [w_s(\log|x_i| + \log|y_i|) + w_t \log|t_i|] \\ & + (|tr| - |tr^*|) \cdot (w_s \log S + w_t \log T) \quad (4) \end{aligned}$$

6. Experiments

We run a set of experiments on a trajectory dataset generated by using the state-of-the-art Brinkhoff generator[1]. It contains 1000 spatio-temporal trajectories with an average length of 70 points, for a total of 70118 spatio-temporal points. The spatial projection of the dataset is shown in Figure 4. For a qualitative understanding of the log distance behavior, we also show 3 randomly-chosen groups of trajectories obtained by using $k = 2$. Trajectories in the same group are clearly close in space and also similar in length (although not shown, also time intervals are similar.) Experiments focus on (1) measuring the amount of utility preserved after anonymization and perturbation processes, and (2) time performance.

[1]`http://fh-oow.de/institute/iapg/personen/brinkhoff/generator/`

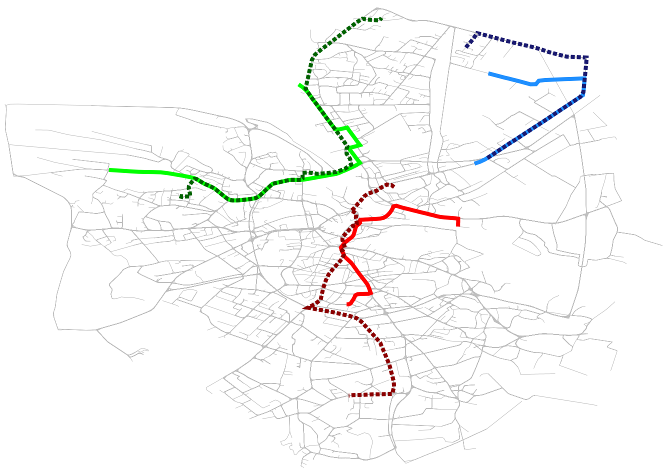

Figure 4. Map of the city with 3 groups each containing 2 trajectories

6.1. Utility

We compared the anonymized datasets (by varying k and the anonymization heuristics) against the original one, measuring how much different they are according to a number of metrics.

Number of Removed Points. The anonymization step allows suppression of points or trajectories, depending on the cost associated to suppression. We used an high cost for suppressions, but notice that since trajectories may have different lengths, suppression may be required to enforce k-anonymity. Figure 5 shows the results on two heuristics used in our experiments: *multi*, i.e. logdistance computed on multiple trajectories; and *fast*, where logdistance

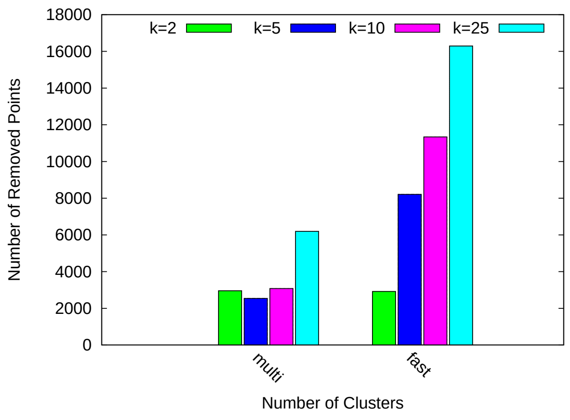

Figure 5. Points removed in the anonymized dataset

has been always computed only on trajectory pairs. As expected, the number of removed points generally increases with k. Notice that *multi* has a low distortion, with less than 9% of points removed even with $k = 25$. On the contrary, *fast* heuristic needs to remove nearly twice or three times the number of points removed by *multi*[2]

Distortion on Clustering. We also analyzed the utility of the anonymized datasets for mining purposes. We measured the deviation from the original clustering results,i.e., we compare clusters obtained from the original trajectory dataset (reference partition) against the clusters obtained from the sanitized dataset (response partition). For the experiments, we used a bottom-up complete-link agglomerative clustering algorithm, coupled with the ERP distance metric [5], which has been specifically developed for trajectories.

As the algorithm requires to specify the number of clusters as input, we ranged from 2 to 60 clusters. Note that due to the large number of experiments and the final complexity of the clustering algorithm we used[3] the whole comparison process required days of computation.

We used a standard approach to evaluate clusters. We considered every pair of trajectories and verified whether both are in the same cluster in the reference partition and whether they are in the response partition. We have therefore four case, namely: true positive (TP), true negative (TN), false positive (FP), false negative (FN). Then we computed the following standard measures:

- $accuracy = (TP + TN)/(TP + FN + FP + TN)$;
- $precision = TP/(TP + FP)$;
- and $recall = TP/(TP + FN)$.

The experiments in Figure 6 show the results computed from the sanitization datasets, by using for different (prearranged) number of clusters. Figures 6(a,b,c) show the behavior of the *multi* heuristic, while on the second row Figures 6(d,e,f) show a similar behavior for the *fast* heuristic. In this set of experiments, we notice therefore that for clustering purposes, fast heuristic has a nice behavior. In order to better understand the values of each measure used in the plots, we also show results of a "random algorithm", i.e. a randomly-selected reference partition of uniformly distributed clusters. For a reasonable number of clusters (e.g., up to 20) all the measures reported good results. We can also notice that smaller k's result in less distortion, although there is not a tight monotonicity due to the randomization steps.

[2]for $k = 2$ the two heuristics are equals, and the only small difference is due to the randomization in the reconstruction of trajectories

[3]Our hierarchical clustering implementation requires $O(n^3)$ distance computations (where n is the number of trajectories), and each ERP computation requires, by using dynamic programming, $O(l^2)$ (where l is the longest trajectory).

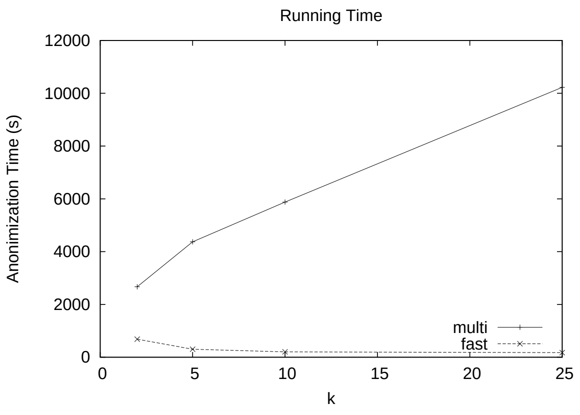

Figure 7. Time performance.

6.2. Time Performance

In Figure 7 we show a plot on time performance. As we can see running time requirements depends linearly on k for *multi* algorithm, while *fast* is almost constant. Also notice that multi required almost 3 hours for $k = 25$; for datasets larger than 3K-4K trajectories, running time may be infeasible for *multi*, while *fast* scales well.

7. Conclusions

We addressed privacy issues regarding the identification of individuals for trajectory datasets sharing. We shifted the notion of k-anonymity from tuples to sequences of spatio-temporal points, and further preserved privacy by releasing only a randomly generated set of representative trajectories. Experiments show that the logdistance and the heuristics proposed are effective for dataset sharing.

References

[1] C. C. Aggarwal and P. S. Yu. A condensation approach to privacy preserving data mining. In *EDBT'04*, pages 183–199, Heraklion, Crete, Greece, Mar. 14 2004.

[2] G. Agrawal, T. Feder, K. Kenthapadi, S. Khuller, R. Panigrahy, D. Thomas, and A. Zhu. Achieving anonymity via clustering. In *PODS '06: Proc. of the 25th ACM Symposium on Principles of database systems*, June 26-28 2006.

[3] M. Atzori. Weak -anonymity: A low-distortion model for protecting privacy. In *ISC*, pages 60–71, 2006.

[4] C. Bettini, X. S. Wang, and S. Jajodia. Protecting privacy against location-based personal identification. In *Secure Data Management*, pages 185–199, 2005.

[5] L. Chen and R. Ng. The marriage of lp-norms and edit distance. In *(e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, Toronto, Canada, August 31 - September 3 2004*, pages 792–803, 2004.

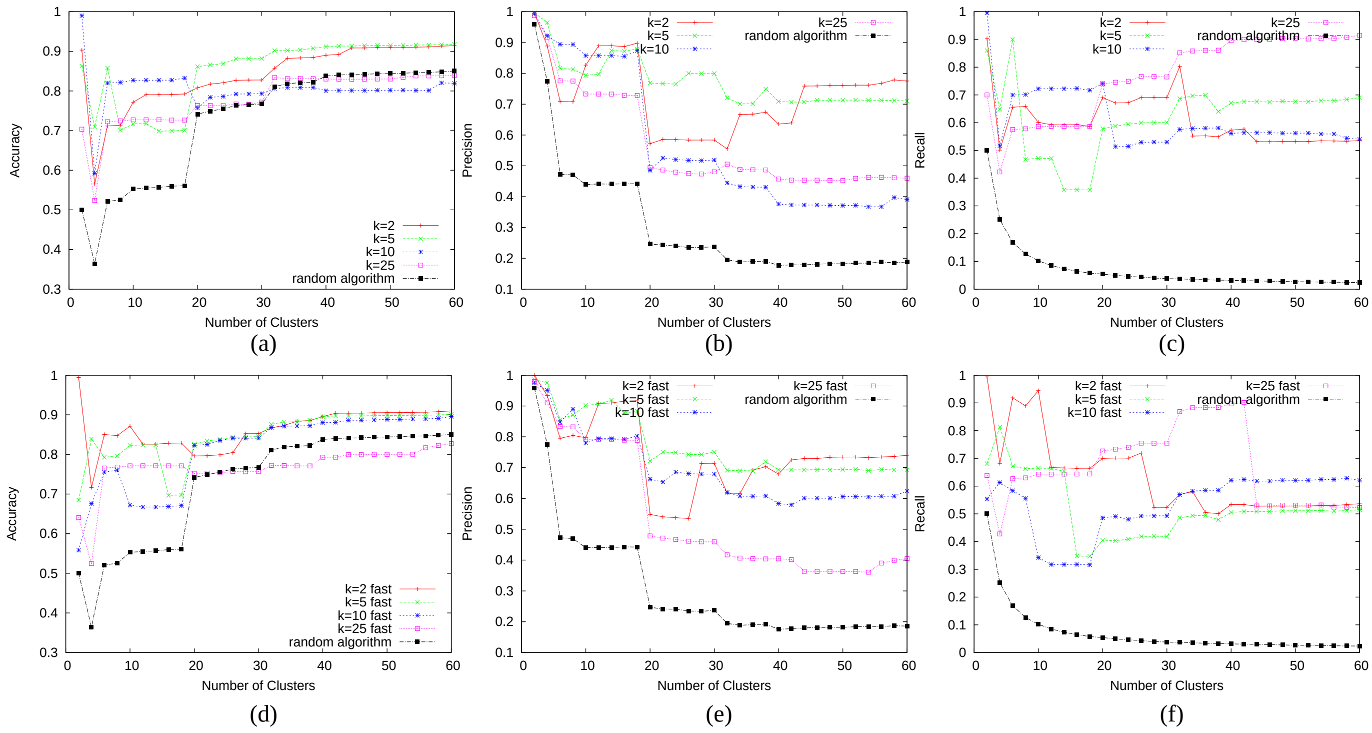

Figure 6. Distortion evaluation on clustering results.

[6] J. Domingo-Ferrer and V. Torra. Ordinal, continuous and heterogeneous k-anonymity through microaggregation. *Data Min. Knowl. Discov.*, 11(2):195–212, 2005.

[7] M. Duckham and L. Kulik. A formal model of obfuscation and negotiation for location privacy. In *Pervasive*, pages 152–170, 2005.

[8] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In *The 25th International Conference on Distributed Computing Systems (ICDCS'05)*, 2005.

[9] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proceedings of the 1st International Conference on Mobile Systems, Applications, and Services*, 2003.

[10] M. Gruteser and X. Liu. Protecting privacy in continuous location-tracking applications. *IEEE Security and Privacy*, 02(2):28–34, 2004.

[11] D. Gusfield. Efficient methods for multiple sequence alignment with guaranteed error bounds. In *Bull. Math. Biol.*, pages 141–154, 1993.

[12] B. Hoh and M. Gruteser. Protecting location privacy through path confusion. In *SECURECOMM '05: Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05)*, pages 194–205, Washington, DC, USA, 2005. IEEE Computer Society.

[13] T. Jiang and L. Wang. On the complexity of multiple sequence alignment. *J. Computer Biologyy*, 1:337–348, 1994.

[14] N. Li and T. Li. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Proceedings of the 23nd International Conference on Data Engineering (ICDE '07)*, Istanbul, Turkey, Apr. 16-20 2007.

[15] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Proc. of the 22nd IEEE Int'l Conf. on Data Engineering (ICDE 2006)*, Atlanta Georgia, Apr. 2006.

[16] D. Maier. The complexity of some problems on subsequences and supersequences. *J. ACM*, 25(2):322–336, 1978.

[17] M. E. Nergiz, M. Atzori, and C. Clifton. Hiding the presence of individuals in shared databases. In *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, Beijing, China, June 11-14 2007.

[18] M. E. Nergiz and C. Clifton. Thoughts on k-anonymization. In *ICDEW '06: Proc. of the 22nd Int'l Conf. on Data Engineering Workshops*, page 96, Washington, DC, USA, 2006. IEEE Computer Society.

[19] M. E. Nergiz and C. Clifton. Multirelational k-anonymity. In *Proceedings of the 23nd International Conference on Data Engineering (ICDE '07)*, Istanbul, Turkey, Apr. 16-20 2007.

[20] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.

[21] L. Sweeney. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, 2002.