# Movement Data Anonymity through Generalization

Anna Monreale<sup>2,3</sup>, Gennady Andrienko<sup>1</sup>, Natalia Andrienko<sup>1</sup>, Fosca Giannotti<sup>2,4</sup>, Dino Pedreschi<sup>3,4</sup>, Salvatore Rinzivillo<sup>2</sup>, Stefan Wrobel<sup>1</sup>

<sup>1</sup>Fraunhofer IAIS, Sankt Augustin, Germany.

<sup>2</sup> KddLab ISTI-CNR, Pisa, Italy.

<sup>3</sup> KddLab Computer Science Department, University of Pisa, Italy.

<sup>4</sup> Center for Complex Network Research, Northeastern University, Boston, MA

E-mail:

{gennady.andrienko, natalia.andrienko, stefan.wrobel}@iais.fraunhofer.de,
{fosca.giannotti,rinzivillo}@isti.cnr.it,{annam,pedre}@di.unipi.it

Abstract. Wireless networks and mobile devices, such as mobile phones and GPS receivers, sense and track the movements of people and vehicles, producing society-wide mobility databases. This is a challenging scenario for data analysis and mining. On the one hand, exciting opportunities arise out of discovering new knowledge about human mobile behavior, and thus fuel intelligent info-mobility applications. On other hand, new privacy concerns arise when mobility data are published. The risk is particularly high for GPS trajectories, which represent movement of a very high precision and spatio-temporal resolution: the de-identification of such trajectories (i.e., forgetting the ID of their associated owners) is only a weak protection, as generally it is possible to re-identify a person by observing her routine movements. In this paper we propose a method for achieving true anonymity in a dataset of published trajectories, by defining a transformation of the original GPS trajectories based on spatial generalization and k-anonymity. The proposed method offers a formal data protection safeguard, quantified as a theoretical upper bound to the probability of re-identification. We conduct a thorough study on a real-life GPS trajectory dataset, and provide strong empirical evidence that the proposed anonymity techniques achieve the conflicting goals of data utility and data privacy. In practice, the achieved anonymity protection is much stronger than the theoretical worst case, while the quality of the cluster analysis on the trajectory data is preserved.

Keywords. k-anonymity, Privacy, Spatio-temporal Clustering

## 1 Introduction

In recent years, many Knowledge Discovery techniques have been developed that provide a new means of improving personalized services through the discovery of patterns which

<sup>\*</sup>This article is an extended version of a paper presented at the 2nd SIGSPATIAL ACM GIS 2009 International Workshop on Security and Privacy in GIS and LBS (SPRINGL 2009), Seattle WA, USA, Nov. 3, 2009.

#### Anna Monreale, Gennady Andrienko, Natalia Andrienko, Fosca Giannotti, Dino Pedreschi, Salvatore Rinzivillo, Stefan Wrobel

represent typical or unexpected customer and user behavior. The collection and the disclosure of personal, often sensitive, information increase the risk of violating a citizen's privacy. Much research thus focused on privacy-preserving data mining [2, 25, 11, 15]. These approaches enables knowledge to be extracted at the same time trying to protect the privacy of the individuals represented in the dataset. Some of these techniques return anonymous data mining results, while others provide anonymous datasets to the companies/research institutions responsible for their analysis. In the last few years, spatio-temporal and moving objects databases have gained considerable interest, due to the diffusion of locationaware devices, such as PDAs, cell phones with GPS technology, and RFID devices, which enable a huge number of traces left by moving objects to be collected. Clearly, in this context privacy is a concern: location data enable inferences which may help an attacker to discover personal and sensitive information such us the habits and preferences of individuals. Hiding car identifiers, for example by replacing them with pseudonyms as shown in [25], is insufficient to guarantee anonymity, since the location could still lead to the identification of the individual. Sensitive information about individuals can be uncovered with the use of visual analytical methods [4]. Therefore, in all cases when privacy concerns are relevant, such methods must not be applied to original movement data. The data must be anonymized, i.e., transformed in such a way that sensitive private information can no longer be retrieved.

In this paper we present a method for the anonymization of movement data combining the notions of spatial generalization and *k*-anonymity. The main idea is to hide locations by means of generalization, specifically, replacing exact positions in the trajectories by approximate positions, i.e. points by centroids of areas. The main steps involved in the proposed methods are: (i) to construct a suitable tessellation of the geographical area into sub-areas that depends on the input trajectory dataset; (ii) to apply a spatial generalization of the original trajectories; (iii) to transform the dataset of generalized trajectories to ensure that it satisfies the notion of *k*-anonymity.

In the literature, most anonymization approaches proposed in a spatio-temporal context are based on randomization techniques, space translations of points, and the suppression of various portions of a trajectory. To the best of our knowledge only [27] uses spatial generalization to achieve anonymity for trajectory datasets; however, the authors used a fixed grid hierarchy to discretize the spatial dimension. In contrast, the novelty of our approach lies in finding a suitable tessellation of the geographical area into sub-areas dependent on the input trajectory dataset. The concept of spatial generalization has also been used in studies on privacy in location-based services [10, 19, 20], where the goal is the online anonymization of individual location-based queries. Our aim, on the other hand, is privacy-preserving data publishing, which requires the anonymization of each entire trajectory. A detailed discussion appears in Section 2.

In this paper, we make the following contributions:

92

- We formally define an adversary's model of attack by clarifying the exact background knowledge that the adversary may possess. We also give a formal statement of the problem studied. Our anonymous dataset is based on the notion of *k*-anonymity.
- We develop an anonymization algorithm based on a spatial generalization and *k*-anonymity. In order to guarantee *k*-anonymity, we propose two strategies for the anonymization step called KAM-CUT and KAM-REC.
- We conduct a formal analysis based on our attack model and prove that our approaches guarantee that the probability of re-identification can always be controlled

to below the threshold chosen by the data owner, by setting the threshold k.

• We conduct a detailed analysis of our approach using a real data set. The dataset consists of a set of 5707 trajectories of GPS tracked cars moving in the area of Milan (Italy). The dataset was provided by the European project GeoPKDD. Our results show that we obtain anonymous trajectories with a good analytical utility. We show how the results of the clustering analysis are preserved by analyzing the quality of the clusters both using a visual analysis and analytical measurements.

The rest of the paper is organized as follows. Section 2 discusses the relevant related studies on privacy issues in spatio-temporal data. In Section 3 some background information and some notions on the clustering analysis are given. Section 4 introduces the attack model and the background knowledge of the adversary. Section 5 states the problem. In Section 6 we describe our anonymization method. In Section 7 we present the privacy analysis. The experimental results of the application of our methods on the real-world moving object dataset are presented and discussed in Section 8. Finally, Section 9 provides the conclusion.

# 2 Related Work

Many research studies have focused on techniques for privacy-preserving data mining [2] and for privacy-preserving data publishing. The first operation before data publishing is to replace personal identifiers with pseudonyms. In [25] Samarati and Sweeney showed that this simple operation is insufficient to protect privacy. They proposed *k*-anonymity to make each record indistinguishable with at least k - 1 other records. In recent years many algorithms for *k*-anonymity have been developed [14, 11, 8, 15]. Although it has been shown that finding an optimal *k*-anonymization is NP-hard [18] and that *k*-anonymity has some limitations [17, 16], this framework is still very relevant.

*K*-anonymity has been used both for data publishing and the publication of data mining results, such as patterns extracted from the data [7]. Moreover, this technique is the most popular method for the anonymization of spatio-temporal data. It is often used both in the studies on privacy issues in location-based services (LBSs) and on the anonymity of trajectories.

In an LBS context, a trusted server usually has to handle the requests of users and to pass them on to the service providers. Essentially, it has to provide an on-line service without compromising the anonymity of the user. The various systems proposed in the literature to make the requests indistinguishable from other k-1 requests use a space generalization, called *spatial-cloaking* [10, 19, 20]. In our approach the anonymization process is off-line, as we want to anonymize a static database of trajectories. To the best of our knowledge only three studies have addressed the problem of the k-anonymity of moving objects using a data publishing perspective [1, 22, 27]. In [1], the authors studied the privacy-preserving publication of a moving object database. They propose the notion of  $(k, \delta)$ -anonymity for moving object databases, where  $\delta$  represents the possible location imprecision. This is an innovative concept of k-anonymity based on co-localization, which exploits the inherent uncertainty of the whereabouts of the moving objects. The authors also proposed an approach, called *Never Walk Alone* based on trajectory clustering and spatial translation. In [22] Nergiz et al. addressed privacy issues regarding the identification of individuals in static trajectory datasets. They provide privacy protection by: (1) first enforcing k-anonymity, i.e. all released information refers to at least k users/trajectories,

(2) randomly reconstructing a representation of the original dataset from the anonymization. Yarovoy et al. in [27] study the *k*-anonymization of moving object databases in order to publish them. Different objects in this context may have different quasi-identifiers and thus, anonymization groups associated with different objects may not be disjoint. Therefore, an innovative notion of *k*-anonymity based on spatial generalization is provided. In fact, the authors proposed two approaches in order to generate anonymity groups that satisfy the novel notion of *k*-anonymity. These approaches are called *Extreme Union* and *Symmetric Anonymization*.

Another approach based on the concept of k-anonymity is proposed in [23], where a framework for the k-anonymization of sequences of regions/locations is presented. The authors also propose an approach that is an instance of their framework, which enables protected datasets to be published while preserving the data utility for sequential pattern mining tasks. This approach, called *BF-P2kA*, uses a prefix tree to represent the dataset in a compact way. Given a threshold k it generates a k-anonymous dataset while preserving the sequential pattern mining results.

Lastly, in [26], a suppression-based algorithm is suggested. Given the head of the trajectories, it reduces the probability of disclosing the tail of the trajectories. It is based on the assumption that different attackers know different and disjoint portions of the trajectories and the data publisher knows the attacker's knowledge. Thus, the solution is to suppress all the dangerous observations.

# 3 Preliminaries

A moving object dataset is a collection of trajectories  $\mathcal{D} = \{T_1, T_2, ..., T_m\}$  where each  $T_i$  is a trajectory represented by a sequence of spatio-temporal points.

**Definition 1** (Trajectory). A Trajectory or spatio-temporal sequence is a sequence of triples  $T = \langle x_1, y_1, t_1 \rangle, \ldots, \langle x_n, y_n, t_n \rangle$ , where  $t_i$   $(i = 1 \dots n)$  denotes a timestamp such that  $\forall_{1 \leq i < n} t_i < t_{i+1}$  and  $(x_i, y_i)$  are points in  $\mathbb{R}^2$ .

Intuitively, each triple  $\langle x_i, y_i, t_i \rangle$  indicates that the object is in the position  $(x_i, y_i)$  at time  $t_i$ .

**Definition 2** (Sub-Trajectory). Let  $T = \langle x_1, y_1, t_1 \rangle, \ldots, \langle x_n, y_n, t_n \rangle$  be a trajectory. A trajectory  $S = \langle x'_1, y'_1, t'_1 \rangle, \ldots, \langle x'_m, y'_m, t'_m \rangle$  is a *sub-trajectory* of T or *is contained* in T ( $S \leq T$ ) if there exist integers  $1 \leq i_1 < \ldots < i_m \leq n$  such that  $\forall 1 \leq j \leq m < x'_j, y'_j, t'_j \rangle = \langle x_{i_j}, y_{i_j}, t_{i_j} \rangle$ .

We refer to the number of trajectories in  $\mathcal{D}$  containing a sub-trajectory S as support of S and denote it by  $supp_{\mathcal{D}}(S)$ , more formally  $supp_{\mathcal{D}}(S) = |\{T \in \mathcal{D} | S \leq T\}|$ .

Clustering analysis is one of the general approaches to exploring and analyzing large amounts of data. Here we consider the problem of clustering a large dataset of trajectories. The aim of a clustering method is to partition a set of trajectories into smaller groups, where each group (or cluster) contains objects that are more similar to each other than to objects in other clusters. The methods to determine this partition may use different strategies [12]. An exploration of the clustering search space is driven by the concept of "similarity". When dealing with complex objects, such as trajectories, the approaches to clustering can be summarized into two main groups: (*i*) distance-based clustering, where the concept of "similarity" is encapsulated within the definition of a distance function, and (*ii*) methods

tailored for the specific domain [9, 13, 3]. Following strategy (*i*), the problem of clustering a set of trajectories can be reduced to selecting a particular clustering method and defining a similarity function for the specific domains. In this paper we focus on the application of a density-based clustering algorithm for trajectories, namely OPTICS [6] and a specific distance function, i.e., "Route Similarity" [4]. The density-based clustering methods have been successfully applied for the clustering of trajectories [21], since they are very robust to noisy data and do not make any assumption on the number of clusters and their shape (i.e., they are not limited to the discovery of convex shaped clusters).

The OPTICS algorithm uses a density threshold around each object to identify interesting data items (i.e., the trajectories that form a cluster) from the noise. The density threshold for a trajectory T is expressed by means of two parameters: a distance threshold *MaxDistance* from T, defining the maximum neighborhood extension, and a population threshold *MinNeighbors*, defining the minimum number of items within the neighborhood. The role of these two parameters as follows: (1) if an object has at least *MinNeighbours* objects lying within the distance *MaxDistance*, this is a *core object* of a cluster; (2) if an object lies within the distance *MaxDistance* from a core object of a cluster, it also belongs to this cluster; the other objects are marked as noise. The algorithm then identifies all the *core trajectories* and group them together into clusters. For each trajectory T, the algorithm checks if T is a *core trajectory*. If it is a core trajectory, a new cluster is initiated from this first element and it is enlarged by checking the *core condition* for all its neighbors. The enlargement of the cluster continues until all the neighbors of the cluster are noise objects, i.e., the cluster is separated from the other elements in the dataset by the noise. The algorithm the visits the next unvisited object of  $\mathcal{D}$ , if there is one.

During the visit, each trajectory is only tested for the *core condition* once. The outcome of the tests for all the trajectories is summarized in a *reachability plot*: the objects are ordered according to the visiting order and, for each item in the plot, the *reachability distance* is specified. Intuitively, the *reachability distance* of a trajectory T corresponds to the minimum distance of T from any other trajectory in its cluster. If T is a noise object, its distance is set by default to  $\infty$ . As a consequence, a high mean value of the *reachability distance* in a cluster denotes a sparse set of objects, while a low value is associated with very dense clusters.

# 4 Privacy Model

Let  $\mathcal{D}$  denote the original dataset of moving objects. The dataset owner applies an anonymization function to transform  $\mathcal{D}$  into  $\mathcal{D}^*$ , the anonymized dataset.

Our anonymization schemes are based on: (a) generating a partition in areas of the territory covered by the trajectories; (b) applying a function for the spatial generalization to all the trajectories in order to transform them into sequences of points that are centroids of specific areas; (c) transforming the generalized trajectories to guarantee privacy.

We use g to denote the function that applies the spatial generalization to a trajectory. Given a trajectory  $T \in \mathcal{D}$ , this function generates the generalized trajectory g(T), i.e. the centroid sequence of areas crossed by T.

**Definition 3** (Generalized Trajectory). Let  $T = \langle x_1, y_1, t_1 \rangle, \ldots, \langle x_n, y_n, t_n \rangle$  a trajectory. A generalized version of T is a sequence of pairs  $T_g = \langle x_{c_1}, y_{c_1} \rangle, \ldots, \langle x_{c_m}, y_{c_m} \rangle$  with  $m \leq n$  where each  $x_{c_i}, y_{c_i}$  is the centroid of an area crossed by T.

**Definition 4** (Generalized Sub-Trajectory). Let  $T_g = \langle x_1, y_1 \rangle, \ldots, \langle x_n, y_n \rangle$  be a generalized trajectory. A generalized trajectory  $S_g = \langle x'_1, y'_1 \rangle, \ldots, \langle x'_m, y'_m \rangle$  is a generalized

sub-trajectory of  $T_g$  or is contained in  $T_g$  if there exist integers  $1 \le i_1 < \ldots < i_m \le n$  such that  $\forall 1 \le j \le m < x'_j, y'_j > = < x_{i_j}, y_{i_j} >$ .

We refer to the number of generalized trajectories in a dataset  $\mathcal{D}^{\mathcal{G}}$  containing a sub-trajectory  $S_g$  as support of  $S_g$  and denote it by  $supp_{\mathcal{D}^{\mathcal{G}}}(S_g)$ , where  $supp_{\mathcal{D}^{\mathcal{G}}}(S_g) = |\{T_g \in \mathcal{D}^{\mathcal{G}} | S_g \preceq T_g\}|$ .

#### 4.1 Adversary Knowledge

An intruder who gains access to  $D^*$  may possess some background knowledge allowing him/her to conduct attacks that may allows him/her to make inferences on the dataset. We generically refer to any of these agents as an attacker. An attacker may know a sub-trajectory of the trajectory of some specific person, for example, by shadowing that person for some time, and could use this information to retrieve the complete trajectory of the same person in the released dataset. Thus, we assume the following adversary knowledge.

**Definition 5** (Adversary Knowledge). The attacker has access to the anonymized dataset  $\mathcal{D}^*$  and knows: (a) the details of the schema used to anonymize the data, (b) the fact that a given user *U* is in the dataset  $\mathcal{D}$  and (c) a sub-trajectory *S* relative to *U*.

#### 4.2 Attack Model

The ability to link the published data to external information, which enables various respondents associated with the data to be re-identified is known as *linking attack model*. In relational data, linking is made possible by using a combination of attributes that can uniquely identify individuals, such as birth date and gender; these attributes are called *quasi-identifiers*. The remaining attributes, called *sensitive*, represent the private information, which may be disclosed by the linking attack, and thus has to be protected. In privacypreserving data publishing techniques, such as *k*-anonymity, the goal is to protect personal sensitive data against this kind of attack by suppression or generalization of quasi-identifier attributes.

The movement data have a sequential nature; in the case of sequential data the dichotomy of attributes into quasi-identifiers (QI) and private information (PI) does not hold any longer. Thus, in the case of spatio-temporal data a sub-trajectory can play both the role of QI and PI. In a linking attack conducted by a sub-trajectory known by the attacker the entire trajectory is the PI that is disclosed after the re-identification of the respondent, while the sub-trajectory serves as QI.

In this paper, we consider the following attack:

**Definition 6** (Attack Model). Given the anonymized dataset  $\mathcal{D}^*$  and a sub-trajectory S relative to a user U, the attacker: (*i*) generates the partition of the territory starting from the trajectories in  $\mathcal{D}^*$ ; (*ii*) computes g(S) generating the sequence of centroids of the areas containing the points of S; (*iii*) constructs a set of candidate trajectories in  $\mathcal{D}^*$  containing the generalized sub-trajectory g(S) and tries to identify the whole trajectory relative to U. The probability of identifying the whole trajectory by a sub-trajectory S is denoted by prob(S).

From the point of view of data protection, minimizing the probabilities of re-identification is desirable. Intuitively, the set of candidate trajectories corresponding to a given sub-trajectory S should be as large as possible. A good solution would be to minimize the

probabilities of re-identification and maximize data utility by minimizing the transformation of the original data. We propose a *k*-anonymity setting as a compromise. The general idea is to control the probability of the re-identification of any trajectory to below the threshold  $\frac{1}{k}$  chosen by the data owner. Thus, our goal is to find an anonymous version of the original dataset  $\mathcal{D}$ , such that, on the one hand, it is still useful for analysis, when published, and on the other, a suitable version of *k*-anonymity is satisfied.

The crucial point of our attack model is that it can be performed by using any subtrajectory in  $\mathcal{D}$ : a sub-trajectory occurring only a few times in the dataset (but at least once) enables a few specific complete trajectories to be selected, and thus the probability that the sequence linking attack succeeds is very high. On the other hand, a sub-trajectory occurring so many times that it is compatible with too many subjects reduces the probability of a successful attack. We will therefore consider the first type of trajectories as *harmful* in terms of privacy, and the second as *non-harmful*.

**Definition 7.** Given a moving object dataset  $\mathcal{D}$  and a trajectory  $T \in \mathcal{D}$  we denote by  $P_{\mathcal{D}}(T)$  the set of all the trajectories  $T' \in \mathcal{D}$  such that  $g(T) \preceq g(T')$ .

**Definition 8.** Given a moving object dataset  $\mathcal{D}$  and an anonymity threshold k > 1, a trajectory T is *k*-harmful (in  $\mathcal{D}$ ) iff  $0 < |P_{\mathcal{D}}(T)| < k$ .

The above notion allows us to define a *k*-anonymous version of a moving object dataset D, parametric w.r.t. an anonymity threshold k > 1.

**Definition 9.** Given an anonymity threshold k > 1, a moving object dataset  $\mathcal{D}$  and a generalized moving object dataset  $\mathcal{D}^*$ , we can say that  $\mathcal{D}^*$  is a *k*-anonymous version of  $\mathcal{D}$  iff for each *k*-harmful trajectory  $T \in \mathcal{D}$  either  $supp_{\mathcal{D}^*}(g(T)) = 0$  or  $supp_{\mathcal{D}^*}(g(T)) \ge k$ , where g(T) is the spatial generalization of T.

The parameter k is the minimal acceptable cardinality of a set of indistinguishable generalized trajectories, which is considered as sufficient protection of anonymity in the given situation. Note that, the Definition 9 does not apply any constraints to non-harmful trajectories in the original dataset: these may have any support value in the anonymous version. The following theorem proves that in the k-anonymous version of a dataset the probability of re-identification has an upper bound of  $\frac{1}{k}$ .

**Theorem 10.** Given a k-anonymous version  $\mathcal{D}^*$  of a moving object dataset  $\mathcal{D}$ , we have that, for any QI trajectory T,  $prob(T) \leq \frac{1}{k}$ .

Proof. Two cases arise.

- **Case 1:** if *T* is a *k*-harmful trajectory in  $\mathcal{D}$ , then, by Def. 9, either  $supp_{\mathcal{D}^*}(g(T)) = 0$ , which implies prob(T) = 0, or  $supp_{\mathcal{D}^*}(g(T)) \ge k$ , which implies  $prob(T) = \frac{1}{supp_{\mathcal{D}^*}(g(T))} \le \frac{1}{k}$ .
- **Case 2:** if *T* is not a *k*-harmful trajectory in  $\mathcal{D}$ , then, by Def. 8 we have  $|\mathcal{P}_{\mathcal{D}}(T)| = M \ge k$ and by Def. 9, g(T) can have an arbitrary support in  $\mathcal{D}^*$ . If  $supp_{\mathcal{D}^*}(g(T)) = 0$  or  $supp_{\mathcal{D}^*}(g(T)) \ge k$ , then the same reasoning as Case 1 applies. If  $0 < supp_{\mathcal{D}^*}(g(T)) < k$ , then the probability of success of the linking attack via *T* to person *U* is the probability that *U* is present in  $\mathcal{D}^*$  times the probability of picking *U* in  $\mathcal{D}^*$ , i.e.,

$$prob(T) = \frac{supp_{\mathcal{D}^*}(g(T))}{M} \times \frac{1}{supp_{\mathcal{D}^*}(g(T))} = \frac{1}{M} \le \frac{1}{k}$$

where the final inequality is justified by the fact that U is present in  $\mathcal{D}$  by the assumption of the linking attack, and by the fact that  $M \ge k$  due to the hypothesis that T is not k-harmful in  $\mathcal{D}$ . This concludes the proof.

The above theorem provides a formal mechanism for quantifying the probability of success of a successfully attack defined in Definition 6. Note that, given a dataset D, this may have many possible *k*-anonymous versions, corresponding to different ways of dealing with the *k*-harmful trajectories. However, from a statistics/data mining point of view, we are only interested in the *k*-anonymous versions that preserve various interesting analytical properties of the original dataset.

# 5 Problem Statement

We will now tackle the problem of constructing a *k*-anonymous version of  $\mathcal{D}$ . The proposed approaches are based on two main steps: (i) applying a spatial generalization of the trajectories in  $\mathcal{D}$ , and (ii) checking the *k*-anonymity property on the generalized dataset and transforming the data when this property is not satisfied. Our approaches guarantee the privacy requirements and also preserve the clustering analysis.

**Definition 11** (Problem Definition). Given a moving object dataset  $\mathcal{D}$ , and an anonymity threshold k > 1, find a *k*-anonymous version  $\mathcal{D}^*$  of  $\mathcal{D}$  such that clustering results are preserved in  $\mathcal{D}^*$ .

In Section 8 we assess the preservation of the clustering results after the anonymization by using both analytical and visual comparisons.

# 6 Anonymization Approaches

In this section we will present our approach, which enables us to anonymize a dataset of moving objects  $\mathcal{D}$ , by combining spatial generalization of trajectories and *k*-anonymization. The details of the two main steps (see Algorithm 1) are described below. Note that we propose two different strategies for the *k*-anonymization step.

#### **Algorithm 1**: PPSG(*D*, *k*, *r*, *p*)

```
Input: A moving object database \mathcal{D}, an integer k, the radius r, a percentage p

Output: A k-anonymous version \mathcal{D}^{*'} of the database \mathcal{D}

// Step I: Trajectory Generalization

< \mathcal{D}^{\mathcal{G}'}, V, C >= SpatialGeneralization(\mathcal{D}, r);

// Apply progressive generalization to the Voronoi tessellation (see Algorithm 3)

< \mathcal{D}^{\mathcal{G}}, V^*, C^* >= ProgressiveGeneralization(\mathcal{D}^{\mathcal{G}'}, V, C, k);

// Step II: Trajectory k-Anonymization

\mathcal{D}^* = k-Anonymization(\mathcal{D}^{\mathcal{G}}, k, p);

return \mathcal{D}^*
```

## 6.1 Trajectory Generalization

The generalization method (see Algorithm 2) extracts characteristic points from the trajectories, groups them into spatial clusters, and uses the centroids of the clusters as generating points for Voronoi tessellation of the geographical area. Each trajectory is then transformed into a sequence of visits of the resulting Voronoi cells. The generalized version of the trajectory is built from the centroids of the visited cells. The degree of generalization depends on the sizes of the cells which, in turn, depend on the spatial extents of the point clusters. The desired spatial extent (radius) is a parameter of the method.

<b>Algorithm 2</b> : SpatialGeneralization( $D, r$ )							
<b>Input</b> : A moving object database $\mathcal{D}$ , the radius $r$							
<b>Output</b> : A generalized moving object database $\mathcal{D}^{\mathcal{G}}$ , set of Voronoi cells <i>V</i> , set of their							
centroids C							
// Extract P - the set of characteristic points from ${\cal D}$							
$P = CharacteristicPoints(\mathcal{D});$							
// Group the points of $P$ in spatial clusters with desired radius $r$							
S = SpatialClusters(P, r);							
// Compute the centroids of the spatial clusters							
C = Centroids(S);							
// Generate Voronoi cells around the centroids							
V = VoronoiTessellation(C);							
// Generalize the trajectories							
$\mathcal{D}^{\mathcal{G}} = GeneralizedTrj(\mathcal{D}, V, C);$							
return $\mathcal{D}^{\mathcal{G}}, V, C$							

The algorithms for extracting characteristic points from trajectories, the spatial clustering of the points and the division of the territory are described in [5]. The use of the resulting division for producing generalized trajectories is straightforward.

A special note should be made concerning the time stamps of the positions in the generalized trajectories. The generalization method associates each position with a time interval  $[t_1, t_2]$ , where  $t_1$  is the moment of entering and  $t_2$  is the moment of exiting the respective area. This can be considered as a temporal generalization. However, the current version of the anonymization algorithm ignores the temporal component of the data.

Figure 1 outlines the generalization process. We used a small sample consisting of 54 trajectories of cars (Figure 1A). Figure 1B demonstrates one of the trajectories with its characteristic points, which include the start and end points, the points of significant turns, the points of significant stops, and representative points from long straight segments. The extraction of the characteristic points is controlled by the user-specified thresholds: minimum angle of a turn, minimum duration of a stop, and maximum distance between two extracted points. In the example given, we used the values 45°, 5 minutes, and 3500 meters. Figure 1C shows the characteristic points extracted from all trajectories of the sample. The points are shown by circles with 30% opacity, so that the concentrations of points are visible. The points were grouped into spatial clusters with the desired radius 3500m (we use a large value to have a clearer picture for the figure). The rhombus-shaped symbols mark the centroids of the clusters. Figure 1D shows the results of using the centroids for the Voronoi tessellation of the territory. The generalized trajectories generated on the basis of this tessellation are presented in Figure 1E. The trajectories are drawn on the map with 10% opacity. The shapes of most of the trajectories coincide exactly; therefore, they appear just as a single line on the map. Figure 1F provides an insight into the number of coinciding trajectories. For each pair of neighboring areas, there is a pair of directed lines, called flow symbols. The thickness of a symbol is proportional to the number of trajectories go-

ing in the direction indicated by the arrow pointer. In this figure, the maximum thickness

Figure 1: Illustration of the process of trajectories generalization. A) Subset of trajectories. B) One of the trajectories and the characteristic points extracted from it. C) Characteristic points extracted from all trajectories (represented by circles) and the centroids of the spatial clusters of the points (represented by rhombuses). D) Voronoi tessellation of the territory. E) Generalized trajectories. F) A summarized representation of the trajectories. The thickness of each flow symbol (directed line) is proportional to the number of trajectories going between the respective pair of areas in the indicated direction.

#### 6.1.1 Spatial Density of Trajectories

The density of trajectories is not the same throughout the territory as depicted in Figure 2. On the left, car trajectories are shown with 10% opacity. It can be seen that the density on the belt roads around the city and on the major radial streets is much higher than in the other places. On the right, the differences in the densities are represented by shading various compartments. The number of trajectories passing an area varies from 1 to 386.



Figure 2: (a) A subset of car trajectories made in Milan in the morning of a working day. The lines are drawn with 10% opacity. (b) The tessellation of the territory based on spatial clusters of characteristic points with approximate radius 500m. The shading of the areas shows the variation of the density of the trajectories over the territory

corresponds to 32 trajectories.

The areas with a low density of trajectories (less than k) are not very suitable for the purposes of anonymization, because the probability of identifying of the trajectories passing these areas will be more than  $\frac{1}{k}$ . These trajectories will need to be removed in the second step of Algorithm 1, which increases the loss of information. However, it is possible to handle the areas with a low density of trajectories at the generalization stage. The idea is to increase the sizes of the areas in the low density regions so as to increase the number of trajectories passing these areas. This means that the level of spatial generalization should vary depending on the density of the trajectories. The distortion of the trajectories will be higher in sparse regions than in dense regions; however, fewer trajectories will need to be removed from the dataset to ensure k-anonymity.

Furthermore, not only is the number of trajectories passing each area important but also the connections between areas. Thus, if there are less than k trajectories going from A to B, the risk of their identification will exceed  $\frac{1}{k}$ , although each of the areas A and B may be passed by more than k trajectories.

#### 6.1.2 Progressive Generalization

In this section we describe how to adapt the generalization method in order to consider the spatial density of the trajectories and the anonymity threshold *k*.

At the generalization stage, it is hard to account for the connections of all possible pairs of places whereas connections between neighboring places are an effective way to adapt the generalization level to the data density. Specifically, if A and B are neighboring areas and there are less than k (but more than 0) trajectories going from A to B or from B to A, these areas should be joined into a single area. This will hide the sensitive segments of the trajectories.

However, directly joining the irregularly shaped areas may produce non-convex shapes, which are computationally costly and inconvenient to use. Therefore, we apply another approach. Let a and b be two areas that need to be joined, C is the set of centroids of all areas,  $c_a \in C$  is the centroid of a and  $c_b \in C$  is the centroid of b. We extract all trajectory points contained in a and in b, and compute  $c_{a+b}$  as the centroid or medoid of these points (using the centroid distorts the data more than using the medoid). Then, we remove  $c_a$  and  $c_b$  from C and put  $c_{a+b}$  instead. The modified set C is used to generate a new Voronoi tessellation. The whole algorithm, called *Progressive Generalization*, can be described as in Algorithm 3.

Formally, Algorithm 3 terminates in one of the following two cases: either there are no neighboring areas connected by less than k trajectories or less than two areas remain. However, achieving one of these termination conditions, especially the second one, may not be desirable because increasing the generalization level decreases the quality of the resulting generalized trajectories. Andrienko and Andrienko in [5] introduce numeric measures of the generalization quality derived from the displacements of trajectory points, i.e. the distances from the original points to their representatives in the generalized trajectories, which are the centroids of the areas. There are local and global quality measures. The former are the average and maximum displacements in an area and the latter are the sum, average, and maximum of the displacements over the whole territory. It is obvious that increasing the sizes of the areas increases the displacements of the trajectory points. A very high level of generalization can make the data unsuitable for analysis. Therefore, it may be reasonable to stop the generalization process before reaching the formal termination conditions.

The numeric measures of the generalization quality allow us to modify Algorithm 3 in a way that the quality of the resulting generalized trajectories can be controlled by the

#### **Algorithm 3**: ProgressiveGeneralization( $\mathcal{D}^{\mathcal{G}}$ , V, C, k)

**Input**: A generalized moving object database  $\mathcal{D}^{\mathcal{G}}$ , set of Voronoi cells V, set of their centroids C, integer k

**Output**: modified set of Voronoi cells  $V^*$ , set of their centroids  $C^*$ 

1 foreach pair of neighboring cells from V do

2 Count the number of trajectories going between them in each of the two directions;

Take the minimum of the two counts if both are positive or the maximum otherwise; end

<sup>5</sup> Create a set *P* including all pairs of areas such that the count of the connecting trajectories is positive but less than *k*;

```
6 if P == \emptyset then
       V^* = V;
 7
       C^* = C;
 8
       go to line 23;
 9
10 else
       Sort the set P in the order of increasing count;
11
       while P \neq \emptyset do
12
            Take the first pair (a, b) from P;
13
14
            Remove all other pairs including a or b from P;
           Generate C^* computing c_{a+b} and replacing c_a and c_b in C by c_{a+b};
15
       end
16
       Generate a new set of Voronoi cells V^* using the points from C^*;
17
       if V^* contains more than 2 areas then
18
            V = V^*;
19
            C = C^*;
20
           go to line 1;
21
22
       else
           // Generalize the trajectories using the modified tessellation
           \mathcal{D}^{\mathcal{G}'} = GeneralizedTrj(\mathcal{D}^{\mathcal{G}}, V^*, C^*);
23
           return \mathcal{D}^{\mathcal{G}'}, V^*, C^*
24
25
       end
26 end
```

user. For this purpose, we introduce an additional parameter for the minimum acceptable quality, which is expressed as the maximum local displacement allowed, denoted *MaxDisplacement*. Let us assume that a pair of areas (a, b) is chosen for joining in Algorithm 3 (see line 5). Before performing the joining, it is possible to compute the expected displacement in the resulting area a + b. First, the centroid of this area  $c_{a+b}$  is computed. Next, the expected local displacement is derived from the distances of the trajectory points contained in areas a and b to the point  $c_{a+b}$ . The areas a and b are joined only if the expected value of the local displacement in a + b is not higher than *MaxDisplacement*. Hence, we modify Algorithm 3 in the following way: when a candidate pair of areas is selected for joining (line 5 of Algorithm 3), the expected displacement in the area resulting from their joining is computed. If the expected displacement exceeds *MaxDisplacement*, the pair is removed from the list *P*.

Another possibility is to do progressive generalization in an interactive way. This may be

102

preferable when the user has difficulties in choosing an appropriate value for the parameter *MaxDisplacement*. We have implemented an interactive version of the algorithm that allows the user to see the result of each iteration step on a map and stop the process when the visible distortion becomes too high. In this case, the most recent result is discarded, and the previous result is taken as final.

The result of each step in the progressive generalization is shown to the user using flow symbols (Figure 1F). As an example, we applied the progressive generalization to the set of approximately 6000 car trajectories in the Milan area - see Figure 2(a). We used Algorithm 2 to produce the initial tessellation with the desired cell radius 500m, as shown in Figure 2(b). Then we applied Algorithm 3 with k = 5. Figures 3(a) & 3(b) present the outcomes of 11 and 12 iteration steps, respectively. Figures 3(a) & 3(b) demonstrate that the progressive generalization method tends to preserve the representative points (cell centroids) lying on the major roads where the density of the trajectories is high. Hence, the distortions in the corresponding segments of trajectories after the generalization are low.



Figure 3: The results of progressive generalization after 11 (a), 12 (b) and 36 (c) iteration steps.

To provide a comparison of the outcomes of two consecutive generalization steps, their results can be shown in one map by superimposing one map layer on the other. The flow symbols are drawn in a semi-transparent mode, which allows the user to see where the two results coincide and where they differ. The user may also superimpose the results of the generalization on the map layer with the original trajectories (Figure 2(a)). For the example presented in Figure 3(b), the user may feel that the outcome of 12 steps distorts the original trajectories too much and decide to keep the previous result. In the original generalization, there were 1908 pair-wise connections with less than 5 trajectories between areas. After 11 steps of the algorithm's work, there are 233 such connections, which is a substantial improvement in terms of preserving privacy. At the next stage, the *k*-anonymization method is applied to the generalized trajectories.

Figure 3(c) presents the result of the progressive generalization algorithm when not interrupted by the user. The algorithm has made 36 iteration steps and removed 1102 of the original 1245 cells. The resulting generalization level is very high, which decreases the usefulness of the generalized trajectories for analysis.

One more approach to controlling the generalization quality is a combination of the interactive and automatic approaches, which allows the user to overcome the difficulty of choosing a suitable value of *MaxDisplacement* in advance. In this approach, Algorithm 3

TRANSACTIONS ON DATA PRIVACY 3 (2010)

performs a certain (user-chosen) number of generalization steps and presents the result to the user. If the user is fully satisfied with the quality, he/she allows Algorithm 3 to perform a given number of further steps. If the user finds the quality in some areas to be unacceptable, he/she interactively marks these areas on the map. Then, the local displacements in all areas are computed. The maximum among the local displacements in the non-marked areas that is lower than the smallest local displacement among the marked areas is taken as the value of the parameter *MaxDisplacement*. After this, the modified Algorithm 3 is fulfilled.

### 6.2 Generalization vs *k*-anonymity

The approach described in Section 6.1, given a dataset of trajectories enables us to generate a generalized version, denoted by  $\mathcal{D}^{\mathcal{G}}$ . In order to complete the anonymization of movement data, a further step is required. It is necessary to ensure that the disclosed dataset is a *k*-anonymous version of the original dataset. In other words, we want to ensure that for any sub-trajectory used by the attacker, the re-identification probability is always controlled below a given threshold  $\frac{1}{k}$ . In the database  $\mathcal{D}^G$  we do not have this guarantee, in fact we could have a single generalized trajectory representing a single trajectory (or in general few trajectories) in  $\mathcal{D}$ . Clearly, there are many ways to make a generalized dataset *k*-anonymous and each one corresponds to a different implementation of the *k*-Anonymization function in the Algorithm 1. Therefore, we now introduce two variants of a method based on the work introduced in [23], KAM\_CUT and KAM\_REC, which apply a transformation to the generalized moving object dataset  $\mathcal{D}^{\mathcal{G}}$  in order to obtain a *k*-anonymous version of  $\mathcal{D}$ . They are based on the well-known data structure called prefix tree, which allows us to represent the list of generalized trajectories in  $\mathcal{D}^{\mathcal{G}}$  in a more compact way.

#### 6.2.1 KAM\_CUT Algorithm

The main steps of the Algorithm 4 are as follows:

- 1. The generalized trajectories in the input dataset  $\mathcal{D}^{\mathcal{G}}$  are used to build a prefix tree  $\mathcal{PT}$ .
- 2. Given an anonymity threshold *k*, the prefix tree is anonymized, i.e., all the trajectories whose support is less than *k* are pruned from the prefix tree.
- 3. The anonymized prefix tree, as obtained in the previous step, is post-processed to generate the anonymized dataset of trajectories  $\mathcal{D}^*$ .

We will now describe the details of each step to better understand the transformation applied to the generalized dataset.

**Prefix Tree Construction.** The first step of the KAM\_CUT algorithm (Algorithm 4) is the *PrefixTreeConstruction* function. It builds a prefix tree  $\mathcal{PT}$ , representing the list of generalized trajectories in  $\mathcal{D}^{\mathcal{G}}$  in a more compact way. The prefix tree is defined as a triple  $\mathcal{PT} = (\mathcal{N}, \mathcal{E}, Root(\mathcal{PT}))$ , where  $\mathcal{N}$  is a finite set of labeled nodes,  $\mathcal{E}$  is a set of edges and  $Root(\mathcal{PT}) \in \mathcal{N}$  is a fictitious node that represents the root of the tree. Each node of the tree (except the root) has exactly one parent and it can be reached through a path, which is a sequence of edges starting from the root node. Each node  $v \in \mathcal{N}$ , except  $Root(\mathcal{PT})$ , has entries in the form  $\langle id, point, support, children \rangle$  where *id* is the identifier of the node v, *point* represents a point of a trajectory, *support* is the support of the trajectory represented by the path from  $Root(\mathcal{PT})$  to v, and *children* is the list of child nodes of v.

Algorithm 4: KAM\_CUT( $\mathcal{D}^{\mathcal{G}}$ , k)

Input: A moving object database  $D^{\mathcal{G}}$ , an integer kOutput: A k-anonymous version  $\mathcal{D}^*$  of the database  $\mathcal{D}^{\mathcal{G}}$ // Step I: Prefix Tree Construction  $\mathcal{PT} = PrefixTreeConstruction(\mathcal{D});$ // Step II: Prefix Tree Anonymization  $\mathcal{L}_{cut} = \emptyset;$ foreach n in  $Root(\mathcal{PT})$ .children do  $| \mathcal{PT}' = TreePruning_{cut}(n, \mathcal{PT}, k);$ end // Step III: Generation of k-anonymous Dataset  $\mathcal{D}^* = AnonymousDataGeneration(\mathcal{PT}');$ return  $\mathcal{D}^*$ 

**Prefix Tree Anonymization.** The *TreePruning*<sub>cut</sub> function prunes the prefix tree with respect to the anonymity threshold k. Specifically, this procedure visits the tree and when the support of a given node n is less than k then it cuts the subtree with root n from the tree.

**Generation of anonymous data.** The third step enables the anonymous dataset  $\mathcal{D}^*$  to be generated. The *AnonymousDataGeneration* procedure visits the anonymized prefix tree and generates all the represented trajectories.

The main characteristic of KAM\_CUT is that it enables us to publish only the *k*-frequent prefixes of the generalized trajectories. For example, given the trajectory  $T_g = \langle x_{c_1}, y_{c_1} \rangle$ , ...,  $\langle x_{c_m}, y_{c_m} \rangle$ ,  $\langle x_{c_{m+1}}, y_{c_{m+1}} \rangle$ , ...,  $\langle x_{c_n}, y_{c_n} \rangle$ , where the initial portion  $\langle x_{c_1}, y_{c_1} \rangle$ , ...,  $\langle x_{c_m}, y_{c_m} \rangle$  occurs less than *k* times in the generalized dataset, while the portion  $\langle x_{c_{m+1}}, y_{c_{m+1}} \rangle$ , ...,  $\langle x_{c_m}, y_{c_m} \rangle$  occurs at least *k* times, the algorithm removes the whole trajectory  $T_g$  from the prefix tree, wasting also the frequent sub-trajectory. Clearly, this method is suitable to anonymize very dense datasets, as in this case the prefix tree considerably compresses the dataset and so the number of lost frequent sub-trajectories tends to be low. Usually, a moving object database is characterized by a low density. It would therefore be useful to have a method capable of recovering portions of trajectories which are frequent at least *k* times. With this aim we developed the algorithm KAM\_REC, described in the next section.

#### 6.2.2 KAM\_REC Algorithm

Algorithm 5, like the previous algorithm, consists of three main steps:

- 1. The generalized trajectories in the input dataset  $\mathcal{D}^{\mathcal{G}}$  are used to build a prefix tree  $\mathcal{PT}$ .
- 2. Given an anonymity threshold *k*, the prefix tree is anonymized, i.e., all the trajectories whose support is less than *k* are pruned from the prefix tree. Part of these infrequent trajectories is then re-appended in the prefix tree.
- 3. The anonymized prefix tree, as obtained in the previous step, is post-processed to generate the anonymized dataset of trajectories  $\mathcal{D}^*$ .

Clearly, the first and the last steps (points 1 and 3) are the same as those described for the algorithm in the previous section. While, the *Anonymization* step (point 2) is different from that of the algorithm KAM\_CUT. Thus we will now describe the details of the second step in order to better understand the transformation applied to the generalized dataset.

Algorithm 5: KAM\_REC( $\mathcal{D}^{\mathcal{G}}$ , k, p)

106

Input: A moving object database  $D^{\mathcal{G}}$ , an integer k, a percentage pOutput: A k-anonymous version  $\mathcal{D}^*$  of the database  $\mathcal{D}^{\mathcal{G}}$ // Step I: Prefix Tree Construction  $\mathcal{P}T = PrefixTreeConstruction(\mathcal{D});$ // Step II: Prefix Tree Anonymization  $\mathcal{L}_{cut} = \emptyset;$ foreach n in  $Root(\mathcal{P}T)$ .children do  $\mid \mathcal{L}_{cut} = \mathcal{L}_{cut} \cup TreePruning(n, \mathcal{P}T, k);$ end  $\mathcal{P}T' = TrjRecovery(\mathcal{P}T, \mathcal{L}_{cut}, p);$ // Step III: Generation of k-anonymous Dataset  $\mathcal{D}^* = AnonymousDataGeneration(\mathcal{P}T');$ return  $\mathcal{D}^*$ 

**Prefix Tree Anonymization.** The first operation performed during this step is to prune the prefix tree with respect to the anonymity threshold k. This operation is executed by the TreePruning function, which modifies the tree by pruning all the infrequent subtrees and decreases the support of the path to the last frequent node. *TreePruning* visits the tree and, when the support of a given node n is less than k, it computes all the trajectories represented by the paths which contain the node n and which start from the root and reach the leaves of the sub-tree with root n. All the computed infrequent trajectories and their supports are inserted into the list  $\mathcal{L}_{cut}$ . Next, the subtree with root *n* is cut from the tree. After the pruning step, the algorithm tries to re-attach part of the generalized trajectories in  $\mathcal{L}_{cut}$  onto the pruned tree, using the *TrjRecovery* function (see Algorithm 6). This function checks if the infrequent trajectories contain generalized sub-trajectories that are frequent at least k times in  $\mathcal{D}^{\mathcal{G}}$ . From each infrequent trajectory it tries to recover the longest subtrajectory that is frequent in the pruned tree  $\mathcal{PT}$  and/or in the list  $\mathcal{L}_{cut}$ . To do this, the function uses the well-known notion of *longest common subsequence*. When the algorithm finds a sub-trajectory which is frequent at least k times then it is appended to the root of the pruned tree only if it contains at least p% (input parameter) of points of the infrequent generalized trajectory.

This algorithm differs from *BF-P2kA* [23] in terms of the *TrjRecovery* step. The algorithm in [23] computes the longest common subsequence only between each infrequent trajectory and the trajectories in the pruned tree, while we also consider in this computation the infrequent trajectories in  $\mathcal{L}_{cut}$ . This allows us to recover a larger amount of frequent sub-trajectories. Suppose for example that a sub-trajectory *S* always appears as a suffix of *k* different trajectories that are cut during the phase of pruning. *BF-P2kA* loses these sub-trajectories, whereas we are able to recover them. Another difference is that our method inserts less noise in the anonymized dataset. In fact, when we find a frequent sub-trajectory we append it to the *Root*, while *BF-P2kA* reattaches it to the path in the tree closest to the

```
Algorithm 6: TrjRecovery(\mathcal{PT}, \mathcal{L}_{cut}, p)
```

```
Input: A prefix tree \mathcal{PT}, a list of sequences \mathcal{L}_{cut}, a percentage p

Output: An anonymized reconstructed prefix tree \mathcal{PT}'

foreach distinct T \in \mathcal{L}_{cut} do

// Compute LCS between T and the trajectories in the prefix tree

LCS_{PT} = ClosestLCS(T, \mathcal{PT});

// Compute LCS between T and the trajectories in list \mathcal{L}_{cut}

LCS_{\mathcal{L}_{cut}} = ClosestLCS(T, \mathcal{L}_{cut});

S = max(LCS_{PT}, LCS_{\mathcal{L}_{cut}});

if length of the sub-trajectory S is at least p\% of T then

| Append S to the Root of \mathcal{PT};

end

end

return \mathcal{PT}
```

sub-trajectory, thus introducing noise.

#### 6.3 A running example

We will now present an example which shows how our two *k*-anonymization approaches work, highlighting the main difference between them. We consider the dataset of generalized trajectories in Figure 4(a) and a *k*-anonymity threshold equal to 2. Note that, each letter in a trajectory represents a centroid of a generalized area.

ſ	$t_1$	ABCDEFG			$t'_1$	ABCDEFG
	$t_2$	ABCDEFG	$t'_1$	ABCDEFG	$t'_2$	ABCDEFG
	$t_3$	ABCDEFG	$t'_2$	ABCDEFG	$t'_3$	ABCDEFG
	$t_4$	ADEF	$t'_3$	ABCDEFG	$t'_4$	ADEF
	$t_5$	ADEF	$t'_4$	ADEF	$t'_5$	ADEF
	$t_6$	ADEF	$t'_5$	ADEF	$t'_6$	ADEF
	$t_7$	CHL	$t'_6$	ADEF	$t'_7$	CHL
	$t_8$	DECHL	$t'_7$	DE	$t'_8$	CHL
	$t_9$	DEJFG	$t'_8$	DE	$t'_9$	DEFG

(a) Generalized Trajectories (b) Trajectories by  $KAM\_CUT$  (c) Trajectories by  $KAM\_REC$ 

Figure 4: A toy example

During the first step our two methods build the prefix tree in Fig. 5(a).

During the anonymization step, as explained in the previous sections, the methods anonymize the tree in different ways.

First we show how KAM\_CUT works. The prefix tree is modified by the *TreePruning*<sub>cut</sub> function pruning the tree with respect to the *k*-anonymity threshold. This procedure searches the tree for all the highest nodes in the tree hierarchy with a support of less than 2. When it finds the three nodes  $\langle 11, C, 1 \rangle$ ,  $\langle 16, J, 1 \rangle$  and  $\langle 19, C, 1 \rangle$  the sub-trees that have these nodes as root are eliminated, thus obtaining the anonymized tree in Figure 5(b). Finally, the *Anony-mousDataGeneration* procedure provides the anonymized dataset shown in Figure 4(b).

In the Algorithm KAM\_REC we need to consider another input parameter, i.e. a percentage p that for example we assume to be equal to 40%. This percentage allows us to recover frequent sub-trajectories only if they preserve at least 40% of the points of the original trajectories. The *TreePruning* function searches the tree for all the highest nodes in the



Figure 5: Prefix tree processing in KAM\_CUT

tree hierarchy with a support of less than 2 and finds the nodes  $\langle 11, C, 1 \rangle$ ,  $\langle 16, J, 1 \rangle$  and  $\langle 19, C, 1 \rangle$ . Next, it identifies the paths that contain these nodes and which start from the root and reach each leaf belonging to the subtrees of these nodes. Then, it generates all the sequences represented by these paths and inserts them into the list  $\mathcal{L}_{cut}$ . Now the list contains the sub-trajectories (*CHL*, 1), (*DEJFG*, 1) and (*DECHL*, 1). Finally, the *TreePruning* procedure eliminates the paths representing the sub-trajectories listed above from the tree, and updates the support of the remaining nodes. The prefix tree obtained after the pruning step is shown in Fig. 6(a).

The infrequent sub-trajectories within  $\mathcal{L}_{cut}$  are then processed, using the notion of the *longest common subsequence*, in this way: (CHL, 1) occurs twice in  $\mathcal{L}_{cut}$  and has at least 40% of the points of the original trajectory. Thus, it is appended to the root obtaining a branch with nodes  $\langle 11, C, 1 \rangle$ ,  $\langle 12, H, 1 \rangle$  and  $\langle 13, L, 1 \rangle$ ; (DEJFG, 1) contains the frequent sub-trajectory DEFG that has at least 40% of the points of the original trajectory. Thus, it is appended to the root producing the branch with the nodes  $\langle 14, D, 1 \rangle$ ,  $\langle 15, E, 1 \rangle$ ,  $\langle 17, F, 1 \rangle$  and  $\langle 18, G, 3 \rangle$ ; finally, (DECHL, 1) contains the frequent sub-trajectory CHL that has at least 40% of the points of the original trajectory. Thus, it is appended to the root increasing the support of the branch with nodes  $\langle 11, C, 1 \rangle$ ,  $\langle 12, H, 1 \rangle$  and  $\langle 13, L, 1 \rangle$ . The prefix tree obtained after the anonymization step is shown in Fig. 6(b). Finally, the *AnonymousData-Generation* procedure provides the anonymized trajectories shown in Fig. 4(c).



Figure 6: Prefix tree processing in KAM\_REC

TRANSACTIONS ON DATA PRIVACY 3 (2010)

#### 6.4 Complexity Analysis

In this section, we discuss the time complexity of our approach, analyzing the complexity of each step of Algorithm 1. In the following we denote by n the total number of points in the original trajectory dataset.

The *SpatialGeneralization* function has a time complexity of O(n) as each step of this procedure is linear w.r.t. the number of points n. In fact, the extraction of characteristic points from the trajectories and the transformation of each trajectory into a generalized version require to visit all the points of the original dataset. Also, the spatial grouping of the characteristics points costs O(n) as at the worst case we have n points of this kind.

Each iteration of *ProgressiveGeneralization* procedure analyzes the centroids of the cells. The number of centroids at the worst case is equal to the number of points n, thus the time cost is  $O(n \times I)$ , where I is the number of iterations.

Regarding the time cost of the two methods for the *k*-Anonymization of the generalized trajectories, note that at the worst case the total number of points of the generalized dataset is equal to *n*. Thus, KAM\_CUT algorithm has time cost O(n) and KAM\_REC algorithm costs  $O(n^2)$ . The cost of KAM\_CUT is linear w.r.t. *n* as it only requires the construction of the prefix tree and the visit of the tree for the pruning. Instead, KAM\_REC also tries to recover portions of trajectories which are frequent at least *k* times computing the longest common subsequence and this requires  $O(n^2)$  time.

## 7 Privacy Analysis

In this section we discuss the privacy guarantees obtained applying our anonymization approaches. We will formally show that our Algorithm 1, using both KAM\_CUT and KAM\_REC, always guarantees that the disclosed dataset  $D^*$  is a *k*-anonymous version of D.

**Theorem 12.** Given a moving object dataset D and an anonymity threshold k > 1, Algorithm 1 using Algorithm 4 produces a dataset  $D^*$  that is a k-anonymous version of D.

*Proof.* First, we show that  $\mathcal{D}^{\mathcal{G}}$  is a generalized version of  $\mathcal{D}$ . Secondly, we show that the dataset  $\mathcal{D}^*$  generated by Algorithm 4 is a *k*-anonymous version of  $\mathcal{D}^{\mathcal{G}}$  and finally, that  $\mathcal{D}^*$  also is *k*-anonymous version of  $\mathcal{D}$ .

- 1. The dataset  $\mathcal{D}^{\mathcal{G}}$  is obtained by applying the generalization (Algorithms 2 & 3) to all the trajectories in  $\mathcal{D}$ , thus it is the generalized version of  $\mathcal{D}$ .
- 2. By construction, the pruning step of Algorithm 4 prunes all the subtrees with support less than k. The pruned prefix tree  $\mathcal{PT}'$  then only contains generalized trajectories that are frequent at least k times in  $\mathcal{D}^{\mathcal{G}}$ . Therefore, after this step, if a generalized trajectory  $T_g$  occurs less than k times in  $\mathcal{D}^{\mathcal{G}}$  then there is no occurrence of it in  $\mathcal{PT}'$ . If  $T_g$  occurs at least k times in  $\mathcal{D}^{\mathcal{G}}$  then it can occur either 0 or at least k times in  $\mathcal{PT}'$ .
- 3. Next, we must prove that the dataset  $\mathcal{D}^*$ , containing only the trajectories represented by  $\mathcal{PT}'$ , is a *k*-anonymous version of  $\mathcal{D}$ . Two cases arise:
  - a) if a trajectory *T* is *k*-harmful in  $\mathcal{D}$  then after the generalization step we can have either  $supp_{\mathcal{D}^{\mathcal{G}}}(g(T)) < k$  and through point 2 of this proof we have  $supp_{\mathcal{D}^{*}}(g(T)) = 0$ , or  $supp_{\mathcal{D}^{\mathcal{G}}}(g(T)) \geq k$  and by point 2 of this proof we have either  $supp_{\mathcal{D}^{*}}(g(T)) = 0$  or  $supp_{\mathcal{D}^{*}}(g(T)) \geq k$ .

b) if a trajectory T is not k-harmful in  $\mathcal{D}$  then after the generalization step we have  $supp_{\mathcal{D}^{\mathcal{G}}}(g(T)) \geq k$  and by point 2 of this proof we have either  $supp_{\mathcal{D}^*}(g(T)) = 0$  or  $supp_{\mathcal{D}^*}(g(T)) \geq k$ . This concludes the proof.

**Theorem 13.** Given a moving object dataset D and an anonymity threshold k > 1, Algorithm 1 by using the Algorithm 5 produces a dataset  $D^*$  that is a k-anonymous version of D.

*Proof.* The proof is similar to Theorem 13. First, we show that  $\mathcal{D}^{\mathcal{G}}$  is a generalized version of  $\mathcal{D}$ . Secondly, we show that the dataset  $\mathcal{D}^*$  generated by Algorithm 5 is a *k*-anonymous version of  $\mathcal{D}^{\mathcal{G}}$  and finally, that  $\mathcal{D}^*$  is also a *k*-anonymous version of  $\mathcal{D}$ .

- 1. The dataset  $\mathcal{D}^{\mathcal{G}}$  is obtained by applying the generalization (Algorithms 2 & 3) to all the trajectories in  $\mathcal{D}$ , thus it is the generalized version of  $\mathcal{D}$ .
- 2. By construction, the pruning step of Algorithm 5 prunes all the subtrees with support less than k, then the pruned prefix tree  $\mathcal{PT}$  only contains generalized trajectories that are frequent at least k times in  $\mathcal{D}^{\mathcal{G}}$ . Thus, after this step, if a generalized trajectory  $T_g$ occurs less than k times in  $\mathcal{D}^{\mathcal{G}}$  then there is no occurrence of it in  $\mathcal{PT}$ . If  $T_g$  occurs at least k times in  $\mathcal{D}^{\mathcal{G}}$  then it can occur either 0 or more times in  $\mathcal{PT}$ . Nevertheless, the reconstruction step does not change the tree structure of  $\mathcal{PT}$ , it only increases the support of existing generalized trajectories which are already frequent at least k times in  $\mathcal{D}^{\mathcal{G}}$ . In conclusion, at the end of the second step of Algorithm 5, all the sub-trajectories which are represented in  $\mathcal{PT}'$  are frequent at least k times in  $\mathcal{D}^{\mathcal{G}}$ .
- 3. Now, we need to prove that the dataset  $D^*$ , containing only the trajectories represented by PT', is a *k*-anonymous version of D. Two cases arise:
  - a) if a trajectory T is k-harmful in  $\mathcal{D}$  then after the generalization step we can have either  $supp_{\mathcal{D}^{\mathcal{G}}}(g(T)) < k$  and through point 2 of this proof we have  $supp_{\mathcal{D}^{\mathcal{G}}}(g(T)) = 0$ , or  $supp_{\mathcal{D}^{\mathcal{G}}}(g(T)) \geq k$  and through point 2 of this proof we have  $supp_{\mathcal{D}^{\mathcal{G}}}(g(T)) \geq 0$ .
  - b) if a trajectory T is not k-harmful in  $\mathcal{D}$  then after the generalization step we have  $supp_{\mathcal{D}^{\mathcal{G}}}(g(T)) \geq k$  and through point 2 of this proof we have  $supp_{\mathcal{D}^{*}}(g(T)) \geq 0$ . This concludes the proof.

## 8 **Experiments**

The two anonymization approaches were applied to a dataset of trajectories, which is a subset of a dataset donated by *Octotelematics S.p.A.* for the research of the GeoPKDD project. This dataset contains a set of trajectories obtained by GPS-equipped cars moving during one week in April 2007 in the city of Milan (Italy). When two consecutive observations where too far in time, or in space, we applied a pre-processing to the whole dataset to cut trajectories. Note that we did not reconstruct the missing points by interpolation: if a point was missing then we assumed that the object stays still in the last position observed. After the pre-processing, the whole dataset contained more than 45k trajectories. In our experiments we used 5707 trajectories collected from 06.00 AM to 10.00 AM on 4 April 2007.

#### 8.1 Statistics on the dataset before and after anonymization

The anonymization process influences the geometrical features of the trajectories. In Figures 7(a) and 7(b) the distribution of length of the anonymized trajectories obtained with both methods is shown in meters. It is evident how the generalization removes the shortest trajectories of the original data: this depends on the parameter r used for the tesselation. The generalized datasets used for the experiments were computed starting with parameter r = 500m and then by executing several steps of progressive generalization.



Figure 7: Distribution of trajectory lengths

Figure 8 depicts the summarization of the trajectories after the anonymization process. A comparison with the original dataset shows how for k = 8, for example, the shape is preserved for the denser regions of the geographical area. As expected, KAM\_REC preserves the dataset best.



Figure 8: Visual summarization of original dataset and anonymized version with k = 8

#### 8.2 Visual comparison of clusters

In our experiments we used the density-based clustering algorithm OPTICS [6] and the distance function for trajectory "route similarity" [24]. It is well known that results of clustering algorithms are sensitive to the choice of *MaxDistance* and *MinNeighbours* values.

Suitable values are typically chosen in an empirical way by trying several different combinations and evaluating the clustering results from the perspective of an internal cohesion of the clusters and their interpretability as well as their number and sizes (thus, a large number of very small clusters is not desirable). For the clustering of the original dataset we found the combination *MaxDistance* = 450m and *MinNeighbours* = 5 to be appropriate in terms of producing a reasonable number of sufficiently coherent clusters. For the clustering of the generalized and anonymized trajectories, we used the same value as the minimum population threshold *MinNeighbours* = 5. The main criterion for choosing the values for the distance threshold was the similarity of the resulting clusters to the clusters of the original trajectories. The distance thresholds for the generalized and anonymized trajectories need to be smaller than for the original trajectories. This is because the distances between similar trajectories decrease in the process of generalization due to the replacement of close points by identical points and further decrease in the process of anonymization due to the removal of infrequent segments.

Figure 9 shows the 10 largest clusters of the original trajectories. The clusters are presented on a map of Milan in a summarized form using the flow map technique, where arrows indicate the direction of the movement and have a thickness proportional to the number of trajectories. Cluster 12, which consists of very short trajectories, is represented by a special circular symbol with the width proportional to the number of trajectories. Above the image of each cluster, its numerical index and size (in parentheses) are shown. Figure 10 shows the 10 largest clusters of the generalized trajectories resulting from Algorithms 2 and 3. The clusters were obtained with the distance threshold of 200m. Figure 11 shows the 10 largest clusters of the anonymized trajectories resulting from Algorithm 5 with k = 5 and p = 40%. The distance threshold for the clustering was 50m.

Above the images of the clusters in Figures 10 and 11, the numbers in bold are the indexes of the corresponding clusters of original trajectories. The symbols "??" denote the cases when no corresponding clusters were found, in particular, cluster 9 in Figure 10, and clusters 37 and 132 in Figure 11. The absence of a cluster of original trajectories corresponding to a cluster of generalized or anonymized trajectories means that the original trajectories do not satisfy the necessary conditions for making a cluster, i.e. each has less than MinNeigh*bours* (= 5) trajectories lying within the distance MaxDistance (= 450m) from it. This does not mean, however, that the shapes (routes) of the original trajectories differ much from the shapes of the corresponding generalized or anonymized trajectories. As an example, Figure 12 demonstrates the original trajectories corresponding to cluster 9 of the generalized trajectories (bottom right corner of Figure 10). It can be seen that the shapes of the original trajectories match the shape of cluster 9 very well. Similarly, clusters 37 and 132 of the anonymized trajectories have corresponding subsets of original trajectories with similar shapes, although they could not be united into clusters according to the formal conditions. Hence, we can say that the generalization and anonymization may reveal some frequent patterns which are difficult to find among the original trajectories due to high variability.

Another interesting case is cluster 6 of the generalized trajectories (Figure 10, upper left corner), which combines clusters 5 and 16 of the original trajectories. Cluster 6 consists of trajectories that have a large common part going from east to west along the northern highway and then split into three branches. In the clustering of the original trajectories, two of the branches are together (cluster 5) and one is separated (cluster 16). However, putting these three branches together or separately depends on the distance threshold used for the clustering. Thus, with the distance threshold of 600m, they make up a single cluster with a size of 82, which is very close to the size of cluster 6 (79) in Figure 10. After the anonymization, the three branches are separated into three clusters, 4, 5, and 91 (the latter

112

two clusters are not visible in Figure 11) with the sizes of 41, 20, and 17, respectively. These variations in putting several slightly differing frequent routes together or separately still means that we can interpret the clustering results.



Figure 9: 10 largest clusters of the original trajectories obtained with the distance threshold 450m and minimum population threshold 5.



Figure 10: 10 largest clusters of the generalized trajectories (resulting from algorithms 2 & 3) obtained with the distance threshold 200m and minimum population threshold 5. The numbers in bold indicate the corresponding clusters of the original trajectories.

TRANSACTIONS ON DATA PRIVACY 3 (2010)

#### Anna Monreale, Gennady Andrienko, Natalia Andrienko, Fosca Giannotti, Dino Pedreschi, Salvatore Rinzivillo, Stefan Wrobel



Figure 11: 10 largest clusters of the anonymized trajectories (resulting from Algorithm 5 with k = 5 and p = 40%) obtained with the distance threshold 50m and minimum population threshold 5. The numbers in bold indicate the corresponding clusters of the original trajectories.



Figure 12: The original trajectories corresponding to cluster 9 of the generalized trajectories.

# 8.3 Measuring the quality of the clustering

To determine how the clusters of the original dataset are preserved after the anonymization phase, we designed *ad hoc* quality measures for the resulting clusterings of the densitybased approach. Since a direct effect of the anonymization process is an increase in the concentration of trajectories (i.e. several original trajectories are bundled on the same route), the clustering method will thus be influenced by the variation in the density distribution. The increase in the concentration of trajectories is mainly caused by the reduction of noisy data. In fact, the anonymization process tends to render each trajectory similar to the neighboring ones. This means that the original trajectories, initially classified as noise, can now be "promoted" as members of a cluster. This phenomenon may produce an enlarged version of the original clusters. To evaluate the actual increase in the density of the cluster, we computed the actual distribution of the trajectories after the anonymization steps. Information on the density of a clustering is implicitly present in the reachability plot obtained by the OPTICS algorithm, since the *reachability distance* is inversely correlated with the density of the data items: a high mean value of reachability distance stands for a sparse cluster, while a low value denotes a very dense group. Thus, we computed a clustering for the original dataset, the generalized dataset, and for each dataset from the two anonymization techniques, for k = 2, 4, 8, 16, 20, 25, 30. The results of the comparison are shown in Figure 13 on a log scale. Not surprisingly, the datasets produced by the KAM\_CUT algorithm are more dense than the dataset produced by KAM\_REC, since the recovery phase of KAM\_REC enables us to recover several trajectories that maintain a relative diversity in the original groups.



Figure 13: The density distribution of the original dataset (in red), the generalized dataset (in blu), and various level of anonymization for the two approaches

Our aim is to verify that these enlarged clusters are neither so large that they embrace all the original clusters (i.e. all the trajectories fall in a unique large cluster) nor so fragmented as to split each original cluster into several anonymized versions. We are interested in monitoring whether the objects of a cluster *C* are maintained in the same group after the clustering of the anonymized version of the dataset and, at the same time, we want to verify that the "shape" of the original cluster is preserved after the anonymization, i.e. the original clusters are not merged arbitrarily into the same large clusters. Moreover, since we adopted a density-based clustering method, we wanted to verify how the density varies during the anonymization process, in order to verify how well the clusters are separated in the anonymized version.

Given a dataset of objects  $\mathcal{D}$ , the result of the clustering method can be expressed as  $\mathcal{C}_{\mathcal{D}} = \{C_1, C_2, \ldots, C_m, noise\}$ . We say also that all the elements in cluster  $C_i$  belong to class  $c_i$ . Given an anonymized version  $\mathcal{D}^*$  of  $\mathcal{D}$  and the clustering  $\mathcal{C}_{\mathcal{D}^*} = \{C_1^*, C_2^*, \ldots, C_l^*, noise^*\}$ , we denote with  $C_j^{*-1}$  the set of original objects whose anonymized versions form the cluster  $C_j^*$ . Given a cluster  $C_i \in \mathcal{C}_{\mathcal{D}}$  and the clustering  $\mathcal{C}_{\mathcal{D}^*}$  of an anonymized version of  $\mathcal{D}$  we define the *Precision* as

$$P(C_i, \mathcal{C}_{\mathcal{D}^*}) = \frac{|C_i \cap C_j^{*^{-1}}|}{|C_j^{*^{-1}}|}, \arg\max_j |C_i \cap C_j^{*^{-1}}| \land C_j^* \in \mathcal{C}_{\mathcal{D}^*}$$

and the Recall as

$$R(C_i, \mathcal{C}_{\mathcal{D}^*}) = \frac{|C_i \cap C_j^{*^{-1}}|}{|C_i|}, \arg\max_j |C_i \cap C_j^{*^{-1}}| \wedge C_j^* \in \mathcal{C}_{\mathcal{D}^*}.$$

The recall measures how the cohesion of a cluster is preserved: it is 1 if the whole original cluster is mapped into a single anonymized cluster, it tends to zero if the original elements are scattered among several anonymized clusters. The precision measures how the singularity of a cluster is mapped into the anonymized version: if the anonymized cluster contains only elements corresponding to the original cluster its value is 1, otherwise the value tends to zero if there are other elements corresponding to other clusters. The contamination of an anonymized cluster may depend on two factors: (*i*) there are elements corresponding to other original clusters or (*ii*) there are elements that were formerly noise and have been promoted to members of an anonymized cluster. Since we are interested on the variation of the original cluster members, we can refine the definition of the precision by defining a measure of *Precision*\* considering only non-noise objects and:

$$P^{*}(C_{i}, \mathcal{C}_{\mathcal{D}^{*}}) = \frac{|C_{i} \cap C_{j}^{*^{-1}}|}{|C_{i}^{*^{-1}} - noise|}, \arg\max_{j} |C_{i} \cap C_{j}^{*^{-1}}| \wedge C_{j}^{*} \in \mathcal{C}_{\mathcal{D}^{*}}$$

The F-measure is usually adopted to express the combined values of precision and recall and is defined as the harmonic mean of the two measures. In this context, given two clusterings  $C_D$  and  $C_{D^*}$  we define the F-measure as:

$$F(\mathcal{C}_{\mathcal{D}}, \mathcal{C}_{\mathcal{D}^*}) = \frac{\sum_{i=1,2,\dots,|\mathcal{C}_{\mathcal{D}}|} F(C_i, \mathcal{C}_{\mathcal{D}^*})}{|\mathcal{C}_{\mathcal{D}}|}, \text{ where } F(C_i, \mathcal{C}_{\mathcal{D}^*}) = 2\frac{P(C_i, \mathcal{C}_{\mathcal{D}^*}) \cdot R(C_i, \mathcal{C}_{\mathcal{D}^*})}{P(C_i, \mathcal{C}_{\mathcal{D}^*}) + R(C_i, \mathcal{C}_{\mathcal{D}^*})}$$

We denote with *F*-measure\* the F-measure computed using the *Precision*\* instead of *Precision*.

To compare the measure of precision and recall on different levels of anonymization, we performed a reference clustering on the original dataset, using the same parameters specified in Section 8.2 (MaxDistance=450m, MinNeighbors=5), and a clustering for each anonymized dataset for different levels of k. The results of the comparison are shown in Figure 14.

The values of recall for KAM\_REC are clearly better than the KAM\_CUT algorithm since the infrequent sub trajectories are recovered into frequent groups, hence guaranteeing that close sub-trajectories are kept together. The KAM\_CUT approach maintains a relative constant value for recall, since it preserves the most frequent core of each group of trajectories. In addition the level of fragmentation of the clusters is bounded by the density of population in each group: a significant loss in recall is visible when the value of *k* increases above the population of the identified clusters, since the trajectories in the smallest clusters are removed.

The values of precision were evaluated using the two definitions given above. Not surprisingly, the values of Precision\* are greater than the standard Precision computation. From a comparison of the two values of precision, we can estimate the number of trajectories that were formerly identified as noise in the original dataset and that are associated with a cluster in the anonymized versions. For example, for the KAM\_REC algorithm and for k = 2, each anonymized cluster contains 30% of the trajectories associated with noise in the original dataset.

The behavior of the two algorithms is also evident in the ratio of recall and precision they achieve: KAM\_REC has larger values of recall and lower values of precision w.r.t. KAM\_CUT, since infrequent branches of groups are recovered in possibly shorter groups that are, however, close to the original one. This ensures that original *k*-harmful trajectories are maintained in a form close to their original neighborhood, thus minimizing



Figure 14: Comparison of the clusterings of the anonymized dataset versus the clustering of the original trajectories. For both anaonymization startegies it is presented a comparison using the standard F-measure and the F-measure\*.

the fragmentation of the cluster and, hence, maintaining the recall. The recovery, however, can transform noise trajectories into anonymized versions that can be attached to an anonymized cluster, thus negatively affecting the level of precision.

#### 8.4 Measuring the Probability of Re-identification

We also analyze the probability of re-identification of a given user in the released dataset. In Section 7 we formally proved that our methods for the anonymization of movement data ensure that, for any sub-trajectory used by the attacker, the re-identification probability is always controlled below a given threshold  $\frac{1}{k}$ . Clearly, this threshold is only an upper bound. In fact, from an accurate investigation of our real-world data we discovered that given a sub-trajectory of a specific user, the probability of re-identification is often lower than  $\frac{1}{k}$ . It depends on the number of observations of the attacker: in general, a low number of observations reduces the probability of re-identification.

This phenomenon is highlighted by the plots in Figure 15, which show the cumulative distributions of the identification probability with varying numbers of observations. We calculated the cumulative distributions for a number of observations from 1 to 80 on the datasets anonymized by KAM\_REC with k = 2,4,8,16 and p = 40%. These are computed choosing randomly a total of 50000 sub-trajectories from the original dataset; they contain a number of points from 1 to 80. To make the plots easy to read we only show the dis-



tributions for some values of the number of observations. We also performed the same

Figure 15: Cumulative Distribution of Identification Probability on Milan Datasets anonymized by KAM\_REC

computation on the datasets anonymized by the algorithm KAM\_CUT, obtaining very similar behaviours.

Moreover, we computed the re-identification probability for each trajectory of the original dataset assuming an increasing number of observations known by the attacker. Given a trajectory we computed the probability when the attacker knows the first point, the first two points, and so on. Then, given a specific number of observations we calculated the average, the maximum and the minimum value of probability. The minimum value in each anonymous dataset and for each number of observations is always 0. The plots in Figure 16 depict the average and the maximum re-identification probability for the anonymization obtained by KAM\_CUT and KAM\_REC with different values of *k*. These plots confirm the theoretical upper bound  $\frac{1}{k}$  and show that given a dataset anonymized with an anonymity threshold *k*, in general, the protection guaranteed by our methods is much higher.

#### 8.5 Run Time Analysis

Our algorithms were implemented using Java 6.0. All experiments were performed on an Intel Core2 Duo processor with a 2.66GHz CPU and 6GB RAM over a Linux platform (ubuntu 8.10). We will now assess the total time needed to anonymize a database of movement data (generalization of trajectories and *k*-anonymization step). Figure 17 reports the timings for anonymizing a dataset of 5707 trajectories, according to the two methods for



Figure 16: Identification Probability by varying the number of observations

guaranteeing *k*-anonymity, KAM\_CUT and KAM\_REC. We show the runtime behavior of the two methods for different values of *k*. The approach using algorithm KAM\_CUT requires a lower execution time since this algorithm, after the pruning step, does not try to recover frequent sub-trajectories from the infrequent ones eliminated from the tree (see Algorithm 4). The use of KAM\_REC requires more execution time, however it enables us to recover more trajectories and to obtain better results in terms of a clustering analysis.



Figure 17: Execution Time of our methods

# 9 Conclusion

In this paper, we have studied the problem of publishing movement data while preserving the privacy. We have proposed a method that combine a well-known notion of *k*anonymity and a technique for the spatial generalization of trajectories. In particular, we introduced two *k*-anonymization strategies. The novelty of our approaches lies in finding a suitable tessellation of a geographical area into sub-areas depending directly of the input trajectory dataset. We have shown that our methods yield a formal theoretical protection guarantee against the re-identification attack. Through an extensive set of experiments on a real-life spatio-temporal dataset, we have showed that the anonymity protection achieved is considerably stronger than the theoretical worst case and the proposed techniques preserve the quality of the clustering analysis.

The whole anonymization process is based on different steps using different transformation tools. At the moment there is no a strict integration of consecutive steps, although we believe that there several margins to improve the quality of the anonymization. For example, the initial generalization step, at the moment, is mainly independent from the successive anonymization phase. We plan to integrate the generalization phase into the anonymization step, in order to better estimate how to merge or separate the tessellation areas.

Further investigations could be directed to developing a generalization method that considers both spatial and temporal information in order to obtain generalized and anonymous spatio-temporal data.

**Acknowledgments:** The authors wish to acknowledge Octotelematics S.p.A. for providing the data. Dino Pedreschi acknowledges support by Google, under the Google Research Award program. Gennady Andrienko, Natalia Andrienko and Stefan Wrobel are supported by DFG priority research program on visual analytics (SPP 1335).

# References

- [1] O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *ICDE*, pages 376–385, 2008.
- [2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In SIGMOD, pages 439–450. ACM, 2000.
- [3] J. Alon, S. Sclaroff, G. Kollios, and V. Pavlovic. Discovering clusters in motion time-series data. In *CVPR* (1), pages 375–381, 2003.
- [4] G. Andrienko, N. Andrienko, and S. Wrobel. Visual analytics tools for analysis of movement data. *SIGKDD Explorations Newsletter*, 9(2):38–46, 2007.
- [5] N. Andrienko and G. Andrienko. Spatial generalization and aggregation of massive movement data. *IEEE Transactions on Visualization and Computer Graphics*, 2010, accepted.
- [6] M. Ankerst, M. M. Breunig, H.P. Kriegel, and J. Sander. Optics: ordering points to identify the clustering structure. *SIGMOD Rec.*, 28(2):49–60, 1999.
- [7] M. Atzori, F. Bonchi, F. Giannotti, and D. Pedreschi. Anonymity preserving pattern discovery. *The VLDB Journal*, 17(4):703–727, 2008.
- [8] J. Domingo-Ferrer and V. Torra. Ordinal, continuous and heterogeneous -anonymity through microaggregation. Data Min. Knowl. Discov., 11(2):195–212, 2005.

- [9] S. Gaffney and P. Smyth. Trajectory clustering with mixtures of regression models. In KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 63–72, New York, NY, USA, 1999. ACM.
- [10] M. Gruteser and D. Grunwald. A methodological assessment of location privacy risks in wireless hotspot networks. In SPC, pages 10–24, 2003.
- [11] R. J. Bayardo Jr. and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE*, 2005.
- [12] L. Kaufman and P. J. Rousseeuw. Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley, 1990.
- [13] J. Lee, J. Han, and K.Y. Whang. Trajectory clustering: a partition-and-group framework. In SIGMOD Conference, pages 593–604, 2007.
- [14] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In SIGMOD, pages 49–60. ACM, 2005.
- [15] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In ICDE, page 25, 2006.
- [16] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and ldiversity. In ICDE, pages 106–115. IEEE, 2007.
- [17] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. I-diversity: Privacy beyond k-anonymity. In ICDE. IEEE Computer Society, 2006.
- [18] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In PODS '04, pages 223–228. ACM, 2004.
- [19] M. F. Mokbel, C. Chow, and W. G. Aref. The new casper: Query processing for location services without compromising privacy. In VLDB, pages 763–774, 2006.
- [20] M. F. Mokbel, C. Chow, and W. G. Aref. The new casper: A privacy-aware location-based database server. In *ICDE*, pages 1499–1500, 2007.
- [21] M. Nanni and D. Pedreschi. Time-focused clustering of trajectories of moving objects. *Journal of Intelligent Information Systems*, 27(3):267–289, 2006.
- [22] M. E. Nergiz, M. Atzori, and Y. Saygin. Perturbation-driven anonymization of trajectories. Technical Report 2007-TR-017, ISTI-CNR, Pisa, 2007.
- [23] R. G. Pensa, A. Monreale, F. Pinelli, and D. Pedreschi. Pattern-preserving k-anonymization of sequences and its application to mobility data mining. In *PiLBA*, 2008.
- [24] S. Rinzivillo, D. Pedreschi, M. Nanni, F. Giannotti, N. Andrienko, and G. Andrienko. Visually-driven analysis of movement data by progressive clustering. *Information Visualization*, 7(3/4):225–239, 2007.
- [25] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: *k*-anonymity and its enforcement through generalization and suppression. Technical report, SRI International, 1998.
- [26] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In MDM, pages 65–72, 2008.
- [27] R. Yarovoy, F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Anonymizing moving objects: how to hide a mob in a crowd? In *EDBT*, pages 72–83, 2009.