CIT. Journal of Computing and Information Technology, Vol. 29, No. 1, March 2021, 1–12
doi: 10.20532/cit.2021.1005276

A Clustering-Anonymity Approach for Trajectory Data Publishing Considering both Distance and Direction

Huo-wen Jiang[1] and Ke-kun Hu[2]

[1]College of Mathematics and Computer Science, Jiangxi Science and Technology Normal University, Nanchang, China
[2]State Key Laboratory of High-end Server and Storage Technology, Inspur Group Co., Ltd., Jinan, China

Trajectory data contains rich spatio-temporal information of moving objects. Directly publishing it for mining and analysis will result in severe privacy disclosure problems. Most existing clustering-anonymity methods cluster trajectories according to either distance- or direction-based similarities, leading to a high information loss. To bridge this gap, in this paper, we present a clustering-anonymity approach considering both these two types of similarities. As trajectories may not be synchronized, we first design a trajectory synchronization algorithm to synchronize them. Then, two similarity metrics between trajectories are quantitatively defined, followed by a comprehensive one. Furthermore, a clustering-anonymity algorithm for trajectory data publishing with privacy-preserving is proposed. It groups trajectories into clusters according to the comprehensive similarity metric. These clusters are finally anonymized. Experimental results show that our algorithm is effective in preserving privacy with low information loss.

ACM CCS (2012) Classification: Security and privacy → Database and storage security → Data anonymization and sanitization

Security and privacy → Human and societal aspects of security and privacy → Privacy protections

Keywords: trajectory data, privacy-preserving, clustering-anonymity, distance, direction

1. Introduction

With the widespread use of location-aware devices such as GPS-enabled phones, PDAs, and cars, location-based services have been developing very fast in recent years. Users acquire these services through the devices listed above to orient themselves, generating a large amount of spatio-temporal data. The data of a single moving object for a continuous period form a **trajectory**. Analyzing and mining trajectory data is useful and helpful. For example, analyzing the trajectory data of urban traffic can provide reliable data support for optimizing traffic lines [2] and urban planning [3]. However, without protection mechanisms, mining these data can cause harm to people. For instance, mining trajectory data of a person can easily lead to leakage of his private information such as home address, workplace, hobbies, and even behavior patterns [4]. For example, according to the trajectory of a person who usually travels between locations A and B on working days, one can infer that A and B are the locations of his home and organization, respectively. According to a person's medical records, one may deduce the disease he or she has got. Thus, it is necessary to do some privacy-preserving processing over trajectory data before publishing to prevent such privacy disclosure.

Gruteser *et al.* [5] first apply the *k*-anonymity technique to the field of privacy-preserving of location data. Based on this work, researchers propose a variety of trajectory *k*-anonymity methods for trajectory data publishing. The key to these methods lies in the construction of *k*-anonymity clusters of trajectory data [6]. Some of the existing *k*-anonymity methods divide trajectory data into *k*-anonymity sets according to the distance between trajectories, while oth-

ers are divided according to the differences in the direction of different trajectories. Although these two kinds of methods achieve the goal of preserving privacy in different ways, they both have the problem of high information loss.

To alleviate this, in this paper, we present a clustering-anonymity approach for trajectory data publishing considering both trajectory distance and direction. The approach strikes a balance between the privacy-preserving and data availability. We first design a trajectory synchronization algorithm to synchronize trajectories. Then, we define the distance- and direction-based similarity metrics between synchronized trajectories. These two similarity metrics are combined to form a comprehensive one, according to which synchronized trajectory data are clustered into k-anonymity clusters. Finally, all these clusters are anonymized. The main contributions of this work are, as follows:

1. We present a trajectory synchronization algorithm. It synchronizes trajectories to prepare for k-anonymity clustering.
2. We propose a comprehensive similarity metric to quantify the proximity between trajectories. It takes into consideration both the trajectory distance and direction.
3. We put forward a k-anonymity clustering algorithm. It divides trajectories into k-anonymity clusters according to the comprehensive similarity metric.

The remaining of this work is organized, as follows: Section 2 reviews the related work on privacy-preserving methods for trajectory data publishing; Section 3 introduces basic concepts regarding trajectory k-anonymity. Section 4 quantitatively defines three similarity metrics that measure the proximity between trajectories. Section 5 presents our trajectory k-anonymity algorithm. Section 6 describes conducted experiments that demonstrate the method's performance. Section 7 concludes this work with future directions.

2. Related Work

Privacy-preserving methods for trajectory data publishing include generalization, false data, and suppression techniques [4, 6]. Generalization is a technique for expanding locations into regions to prevent privacy disclosure [7]. Komishani *et al.* [8] apply it to the personalized trajectory data and propose a personalized privacy-preserving method. Wang *et al.* [9] use this method to solve the problem of preserving the privacy of uncertain trajectories. These trajectories are first generalized into more realistic trajectory regions by using probability statistics techniques, where trajectories of high similarity are aggregated into equivalent classes to hide the information. Then, these classes are anonymized and published. Although generalization techniques can maintain the authenticity of data better, their computational cost is very high [6, 7]. False data methods reduce the risk of privacy disclosure by adding false trajectory data to interfere with the real ones. Their key is how to generate false trajectories. Lei *et al.* [10] enhance the trajectory privacy-preserving level by increasing the number of crossover points on rotated trajectories. Lei *et al.* [11] reduce the discrimination probability by increasing the similarities between true and false trajectories. They propose a privacy-preserving scheme for trajectory data publishing based on the spatio-temporal correlation. In the process of generation of false trajectories, many factors are considered, such as the distance between location samples and the corresponding time reachability, the overall moving direction, and the same in- and out-degree of one location sample. False data methods are simple and inexpensive, but they have the problem of low data availability [6]. Suppression is the privacy-preserving technique for trajectory data publishing that hides certain sensitive or frequently accessed locations. The key is how to determine the suppression degree that strikes a balance between the privacy-preserving effect and the data availability. Terrovitis *et al.* [12] measure the sensitivity of a region based on the ratio of the number of users in this region to the total number of users. Then, all locations of the sensitive region are suppressed to ensure that the risk of privacy disclosure is no higher than the threshold set by users. Zhao *et al.* [13] propose two privacy-preserving methods for trajectory data publishing based on trajectory frequency suppression. With the same privacy-preserving strength as other similar schemes, the data availability of their methods is significantly improved. Suppression is a simple, yet effective privacy-preserving method based

on attackers' background knowledge. However, when the attackers' background knowledge is not mastered, this method cannot effectively preserve the privacy of trajectory data [7].

k-anonymity is one of the most important generalization methods. Its basic idea is to divide the trajectory data to be published into some *k*-anonymity clusters, with each set having *k* trajectories. Then, location samples of *k* trajectories in each set with the same sampling time are generalized to an anonymous region to preserve privacy. Trajectory clustering is an important step of trajectory *k*-anonymity and defining a reasonable trajectory similarity metric is a prerequisite. Trajectory *k*-anonymity to minimize information loss is a class of NP-hard problems. Existing trajectory *k*-anonymity methods try to make all *k* trajectories within the same *k*-anonymity set as concentrated in time and as close in space as possible. That is, when designing a trajectory *k*-anonymity algorithm, one should focus on how to reasonably define trajectory similarity metric and choose appropriate partitioning method to ensure that trajectories in each *k*-anonymity cluster are of high similarity. For example, Huo *et al.* [14] measure the trajectory similarity between trajectories according to the Euclidean distance metric. They establish a graph model to represent relationships between trajectories and obtain an anonymity set of trajectories by using graph partitioning techniques. Gao *et al.* [15] define the trajectory similarity metric as a function of the angle between trajectories' direction and construct an anonymity region according to the trajectory direction. However, this trajectory similarity metric neglects the influence of trajectory shapes and results in low data availability of anonymity set. To solve this problem, Wang *et al.* [16] propose a quick and accurate trajectory similarity metric taking the trajectory shape into account. As an important trajectory privacy-preserving method, trajectory *k*-anonymity has received extensive attention from both industry and academia.

3. Concepts of Trajectory Clustering-Anonymity

Definition 1 (Trajectory). A trajectory of a moving object is a sequence of location samples denoted as

$$T = \{(t_1, x_1, y_1), (t_2, x_2, y_2), ..., (t_n, x_n, y_n)\},$$

where (t_i, x_i, y_i) represents the coordinate of this moving object at time t_i for all $i \in [1, n]$.

Definition 2 (Trajectory segment). A trajectory segment is a path between two adjacent location samples of a single trajectory. Suppose the trajectory of a moving object p is denoted as

$$T_p = \{({t_p}^1, {x_p}^1, {y_p}^1), ({t_p}^2, {x_p}^2, {y_p}^2), ..., ({t_p}^n, {x_p}^n, {y_p}^n)\}.$$

Then the path ${L_p}^i$ that connects $({t_p}^i, {x_p}^i, {y_p}^i)$ and $({t_p}^j, {x_p}^j, {y_p}^j)$ is a trajectory segment of T_p.

Definition 3 (Synchronized Trajectories). Given two trajectories T_p and T_q, they are synchronized trajectories if both have the same sampling sequence, *i.e.*, the number of location samples and the corresponding sampling time of these two trajectories are the same. If any pairs of trajectories in a set of trajectories are synchronized, then this set is said to be synchronized.

Suppose that

$$T_p = \{({t_p}^1, {x_p}^1, {y_p}^1), ({t_p}^2, {x_p}^2, {y_p}^2), ..., ({t_p}^{n_p}, {x_p}^{n_p}, {y_p}^{n_p})\},$$

$$T_q = \{({t_q}^1, {x_q}^1, {y_q}^1), ({t_q}^2, {x_q}^2, {y_q}^2), ..., ({t_q}^{n_q}, {x_q}^{n_q}, {y_q}^{n_q})\},$$

then T_p and T_q are synchronized trajectories if $n_p = n_q$ and ${t_p}^i = {t_q}^i$ for all $i \in [1, n_p]$. $\mathcal{T} = \{T_1, T_2, \ldots, T_N\}$ is a trajectory set to be published. It is synchronized if $\forall i, j \in [1, N]$, $i \neq j$: T_i and T_j are synchronized. Different trajectories are often not synchronized, because their sampling times and sampling intervals are not the same. To apply the clustering-anonymity algorithm on them, they should be first synchronized. Without a loss of generality, we assume that:

1. all trajectories in $\mathcal{T}$ are consisted of location samples that are sampled from the same spatio-temporal region; and
2. the moving object of each trajectory maintains a uniform linear motion between adjacent location samples.

According to assumption (1), all tracks have the same starting and ending times. For any two adjacent sampling times t_i^j and t_i^{j+1} on any track T_i, the coordinate of a moving object at time t be-

tween t_i^j and t_i^{j+1} can be easily calculated based on the assumption (2). Then all trajectories in $\mathcal{T}$ can be synchronized by inserting missing locations samples. The trajectory synchronization algorithm will be discussed in Section 5.

Definition 4 (Trajectory clustering). It is the process of dividing all trajectories in the synchronized trajectory set into clusters with each having k trajectories according to the predefined trajectory similarity metric, where k is a constant.

Trajectory clustering can be represented as a mapping

$$\varphi: \mathcal{T} = \{T_1, T_2, ..., T_N\} \rightarrow \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_m\},$$

where φ satisfies the following two requirements:

1. $\bigcup_{i=1}^{m} \mathcal{T} = \mathcal{T}$ and $k \le |\mathcal{T}_i| < 2k$. m is the number of trajectory clusters and $|\mathcal{T}_i|$ the number of trajectories in cluster $\mathcal{T}_i$;
2. $\forall i, j \in \{1, 2, ..., m\}$ and $i \ne j$: $\mathcal{T}_i \cap \mathcal{T}_j = \varnothing$.

Definition 5 (Trajectory generalization). It is the process of replacing coordinates of location samples of all trajectories at each sampling time in a synchronized trajectory set with minimum regions that contain all the corresponding location samples.

Given a synchronized trajectory set $\mathcal{T}_p = \{T_1, ..., T_k\}$, the location region at time t_j is denoted as $([\underline{x}_{\tau_p}^{\ j}, \overline{x}_{\tau_p}^{\ j}], [\underline{y}_{\tau_p}^{\ j}, \overline{y}_{\tau_p}^{\ j}])$. It contains all the corresponding location samples of $T_1, ..., T_k$ at t_j. According to Definition 5, we know that

$$\underline{x}_{\tau_p}^{\ j} = \min\{x_i^j \mid i = (1, ..., k)\},$$

$$\overline{x}_{\tau_p}^{\ j} = \max\{x_i^j \mid i = (1, ..., k)\},$$

$$\underline{y}_{\tau_p}^{\ j} = \min\{y_i^j \mid i = (1, ..., k)\},$$

$$\overline{y}_{\tau_p}^{\ j} = \max\{y_i^j \mid i = (1, ..., k)\}.$$

That is, after trajectory generalization, all trajectories in the synchronized trajectory set $\mathcal{T}_p$ will be generalized into an anonymous trajectory that includes only n location regions. Let this anonymous trajectory be denoted as $\hat{\tau}_p$, then

$$\hat{\tau}_p = \{(t_j, [\underline{x}_{\tau_p}^{\ j}, \overline{x}_{\tau_p}^{\ j}], [\underline{y}_{\tau_p}^{\ j}, \overline{y}_{\tau_p}^{\ j}])\}, \text{where } j = (1, ..., n).$$

Trajectory generalization leads to information loss. We quantitatively define the information loss IL as follows:

$$IL = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{Area\left(\left[\underline{x}_{\tau_i}^{\ j}, \overline{x}_{\tau_i}^{\ j}\right], \left[\underline{y}_{\tau_i}^{\ j}, \overline{y}_{\tau_i}^{\ j}\right]\right)}{m \times n \times Area\left(\left[\underline{x}_{\tau}^{\ j}, \overline{x}_{\tau}^{\ j}\right], \left[\underline{y}_{\tau}^{\ j}, \overline{y}_{\tau}^{\ j}\right]\right)} \quad (1)$$

Trajectory generalization transforms k concrete trajectories into a relatively obscure anonymous trajectory by properly amplifying the location region. It can effectively reduce the privacy disclosure probability by hiding the original trajectories. At the same time, this method cannot significantly change the statistical properties of original data when k is small (typically k is 3 to 5). That is, an anonymous trajectory after generalization is still of high data availability.

Definition 6 (Trajectory *k*-anonymity). It is a 3-step privacy-preserving approach for trajectory data publishing. First, all trajectories in the set to be published are synchronized. Second, the synchronized set is divided into trajectory clusters with a size of k. Finally, all clusters are anonymized.

Trajectory k-anonymity is an important trajectory privacy-preserving method, and its key is the trajectory k-clustering. This work presents a novel and effective trajectory k-anonymity algorithm, which converts the to-be-published trajectory dataset $\mathcal{T} = \{T_1, T_2, ..., T_N\}$ into $\mathcal{T} = \{\hat{\tau}_1, \hat{\tau}_2, ..., \hat{\tau}_m\}$. The key is to define a reasonable trajectory similarity metric to guide trajectory clustering. The quality of this metric has a significant impact on the performance of trajectory k-anonymity algorithms. It is the main content of Section 4.

4. Measuring Similarity between Trajectories

Definition 7 (Proximity of location samples). It is a real number used to measure the degree of closeness between any two synchronized trajectories at any sampling time.

Suppose that $SimD(T_p^{\ i}, T_q^{\ i})$ denotes the location sample proximity between any two syn-

chronized trajectories T_p and T_q at sampling time i. It can be calculated by the equation below:

$$SimD\left(T_p^i, T_q^i\right) = 1 - \frac{Dist\left(T_p^i, T_q^i\right) - \underline{Dist}\left(\mathcal{T}_s^i\right)}{\overline{Dist}\left(\mathcal{T}_s^i\right) - \underline{Dist}\left(\mathcal{T}_s^i\right)} \tag{2}$$

where

$$Dist\left(T_p^i, T_q^i\right) = \sqrt{\left(x_p^i - x_q^i\right)^2 + \left(y_p^i - y_q^i\right)^2}$$

is the distance between location samples (t_i, x_p^i, y_p^i) and (t_i, y_q^i, y_q^i);

$$\underline{Dist}(\mathcal{T}_s^i) = \min\{Dist(\mathcal{T}_p^i, \mathcal{T}_q^i) \mid \mathcal{T}_p, \mathcal{T}_q \in \mathcal{T}_s\}$$

$$\overline{Dist}(\mathcal{T}_s^i) = \max\{Dist(\mathcal{T}_p^i, \mathcal{T}_q^i) \mid \mathcal{T}_p, \mathcal{T}_q \in \mathcal{T}_s\}.$$

This metric is a normalized measurement that represents the relative distance between any two synchronized trajectories at any sampling time concerning the maximum distance at that time. Thus, we know that $SimD(T_p^i, T_q^i) \in [0, 1]$. The larger the $SimD(T_p^i, T_q^i)$ value is, the closer the distance between the two location samples is. $SimD(T_p^i, T_q^i) = 0$ when their distance equals the maximum distance between these two trajectories. $SimD(T_p^i, T_q^i) = 1$ when T_p^i and T_q^i are the same location sample. Furthermore, we define the proximity between two synchronized trajectories as the sum of all their location samples' proximities.

$$SimD\left(T_p, T_q\right) = \frac{\sum_{i=1}^{n} SimD\left(T_p^i, T_q^i\right)}{n}, \tag{3}$$

where n is the number of location samples of a trajectory. It can be seen that $SimD(T_p, T_q)$ is a normalized metric representing the relative distance between any pair of trajectories in $\mathcal{T}$ concerning the maximum distance. Thus, we have that $SimD(T_p, T_q) \in [0, 1]$. The larger the trajectory similarity, the smaller the distance between trajectories.

Definition 9 (Trajectory segment-segment angle). It is the angle between two trajectory segments. These segments come from two different trajectories in a synchronized trajectory set with the same starting and ending sampling times.

For any two synchronized trajectories

$$T_p = \{(t_1, x_p^1, y_p^1), (t_2, x_p^2, y_p^2), ..., (t_n, x_p^n, y_p^n)\} \text{ and}$$

$$T_q = \{(t_1, x_q^1, y_q^1), (t_2, x_q^2, y_q^2), ..., (t_n, x_q^n, y_q^n)\},$$

θ^i is denoted as the segment-segment angle between $\overrightarrow{L_p^i}$ and $\overrightarrow{L_q^i}$, where

$$\overrightarrow{L_p^i} = (x_p^{i+1} - x_p^i, y_p^{i+1} - y_p^i),$$

$$\overrightarrow{L_q^i} = (x_q^{i+1} - x_q^i, y_q^{i+1} - y_q^i).$$

Then the angle can be calculated by the following equation.

$$\cos\theta^i = \frac{\overrightarrow{L_p^i}\,\overrightarrow{L_q^i}}{\left|\overrightarrow{L_p^i}\right|\left|\overrightarrow{L_q^i}\right|} = \frac{\left(x_p^{i+1} - x_p^i\right)\left(x_q^{i+1} - x_q^i\right) + \left(y_p^{i+1} - y_p^i\right)\left(y_q^{i+1} - y_q^i\right)}{\sqrt{\left(x_p^{i+1} - x_p^i\right)^2 + \left(y_p^{i+1} - y_p^i\right)^2}\sqrt{\left(x_q^{i+1} - x_q^i\right)^2 + \left(y_q^{i+1} - y_q^i\right)^2}} \tag{4}$$

Definition 8 (Distance similarity between trajectories). It is a real number used to measure the degree of proximity between any two trajectories in a trajectory synchronized set.

Suppose that $SimD(T_p, T_q)$ denotes the distance similarity between any two trajectories T_p and T_q in a trajectory synchronized set $\mathcal{T}$. It can be calculated as follows:

$\cos\theta^i$ accurately reflects the difference in directions of $\overrightarrow{L_p^i}$ and $\overrightarrow{L_q^i}$. The larger $\cos\theta^i$ is, the smaller θ^i is and the closer they are. The mean values of $\theta^1, \theta^2, ..., \theta^n$ can generally reflect the differences in the overall directions of T_p and T_q. The greater the mean value, the closer they are.

Definition 10 (Direction similarity between trajectories). It is a real number used to mea-

sure the differences in the overall directions of any two synchronized trajectories.

Let $SimO(T_p, T_q)$ denote the trajectory direction similarity between any two synchronized trajectories T_p and T_q. It can be calculated by the following equation:

$$SimO\left(T_p,\ T_q\right) = \frac{1}{n-1}\sum_{i=1}^{n-1}\cos\theta^i, \qquad (5)$$

where n is the number of location samples of a trajectory. From Equation 3, we know that $SimO(T_p, T_q) \in [0, 1]$. The larger the $SimO(T_p, T_q)$ value, the smaller the difference in the overall directions of any two synchronized trajectories.

Definition 11 (Comprehensive similarity between trajectories). It is a real number used to measure the similarity between any two trajectories considering both trajectory distance and direction in a trajectory synchronized set.

Suppose that $Sim(T_p, T_q)$ denotes the comprehensive similarity between any two trajectories T_p and T_q in a trajectory synchronized set $\mathcal{T}$. It can be calculated as follows:

$$\begin{aligned} Sim(T_p, T_q) = \lambda\ SimO(T_p, T_q) + \\ + (1 - \lambda)\ SimD(T_p, T_q), \end{aligned} \qquad (6)$$

where $\lambda \in (0, 1)$ is a tuning parameter and $Sim(T_p, T_q) \in [0, 1]$. We can see that the comprehensive similarity metric accounts for both direction similarity and distance similarity between trajectories. The value of λ is a trade-off between these two similarities and has a great impact on the trajectory clustering effect. In general, the larger this value, the more the comprehensive similarity is based on the direction similarity. This is more likely to cluster trajectories whose overall shapes are similar into clusters. On the contrary, the smaller the value, the more the comprehensive similarity is based on the distance similarity. This is more likely to cluster trajectories that are closer to each other into clusters.

5. Trajectory Clustering-Anonymity Algorithm

Based on the comprehensive similarity metric and other related concepts discussed above, this section puts forward a trajectory k-anonymity algorithm, *i.e.*, trajectory clustering-anonymity algorithm. According to Definition 6, it consists of 3 steps, each of which is an algorithm to perform the corresponding task. They are algorithms of trajectory synchronization, trajectory clustering, and trajectory generalization, respectively. Their details are given as follows, followed by the complete trajectory k-anonymity algorithm.

Trajectory synchronization algorithm. Suppose the trajectory set to be synchronized is denoted as $\mathcal{T} = \{T_1, T_2, ..., T_N\}$, where

$$\begin{aligned} T_i = \{(t_i^1, x_i^1, y_i^1), (t_i^2, x_i^2, y_i^2), ..., \\ (t_i^{ni}, x_i^{ni}, y_i^{ni})\} \end{aligned}$$

and n_i is the number of location samples of T_i ($i \in [1, N]$). According to Definition 3, we know that

$$t_1^1 = t_2^1 = ... = t_N^1,$$
$$t_1^{n_1} = t_2^{n_2} = ... = t_N^{n_N}.$$

Based on the assumption (2) stated in Section 3, we can summarize the synchronization process of $\mathcal{T}$ as follows:

1. for each T_i, calculate the number and coordinates of location samples needed to be interpolated. These samples are with timestamp tp that is not on T_i but the other trajectories.
2. Interpolate these location samples into T_i.

This algorithm's pseudo-code is shown in Algorithm 1.

For the TSynchzing algorithm, its main work is done in a two-layer nested loop, of which the innermost operation is step 4 with a cost of $O(1)$. The number of iterations in the outer loop is N and that of the inner loop is n. Thus, the time complexity of TSynchzing algorithm is $O(n^2)$.

Trajectory clustering algorithm (TClustering). It clusters all N trajectories of $\mathcal{T}$ into clusters with each having k trajectories according to the principle of maximizing the comprehensive similarity. To be more specific, suppose m is the smallest integer that is equal to or greater than N/k. Our proposed TClustering algorithm first constructs a seed set $\mathcal{T}_r$ by identifying m (*if* $N = m*k$) or $m - 1$ (otherwise) trajectories from

$\mathcal{T}$ according to [17] and then constructs trajectory clusters of size k one-by-one. For each new cluster, the TClustering algorithm first initializes it with a single trajectory randomly selected from $\mathcal{T}_r$. Then the algorithm inserts one unclustered trajectory into this cluster every time that it has the largest sum of comprehensive similarities between it and trajectories within the cluster until the cluster size is k. Note that the size of the last cluster may be smaller than k. The pseudo-code of the TClustering algorithm is shown in Agorithm 2.

Algorithm 1. TSynchzing.

Input: $\mathcal{T} = \{T_1, T_2, ..., T_N\}$; // $T_i = \{(t_i^1, x_i^1, y_i^1), (t_i^2, x_i^2, y_i^2), ..., (t_i^{ni}, x_i^{ni}, y_i^{ni})\}$, $i \in [1, N]$

Output: $\mathcal{T} = \{T_1, T_2, ..., T_N\}$; // $T_i = \{(t_1, x_i^1, y_i^1), (t_2, x_i^2, y_i^2), ..., (t_n, x_i^n, y_i^n)\}$ and $SetT = \bigcup_{i=1}^{n} t_i$

1. **for** $i = 1$ to N **do**
2. **for** $j = 1$ to $|SetT|$ **do** // $SetT = \bigcup_{i=1}^{n} \bigcup_{j=1}^{n_i} t_i^j$
3. **if** $t \in (t_i^p, t_i^{p+1})$ **then** // $p \in \{1, ..., n_i - 1\}$
4. Calculating (x_i^t, y_i^t) according to $\dfrac{t - t_i^p}{t_i^{p+1} - t_i^p} = \dfrac{\sqrt{\left(x_i^t - x_i^p\right)^2 + \left(y_i^t - y_i^p\right)^2}}{\sqrt{\left(x_i^{p+1} - x_i^p\right)^2 + \left(y_i^{p+1} - y_i^p\right)^2}}$;
5. $T_i = T_i \cup (t, x_i^t, y_i^t)$;
6. **end if**
7. **end for**
8. **end for**

Algorithm 2. TClustering.

Input: $\mathcal{T} = \{T_1, T_2, ..., T_N\}$, where $T_i = \{(t_1, x_i^1, y_i^1), (t_2, x_i^2, y_i^2), ..., (t_n, x_i^n, y_i^n)\}$;

Output: $\mathcal{T}^* = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_m\}$, where $\bigcup_{i=1}^{m} \mathcal{T}_i = \mathcal{T}$ and $\mathcal{T}_i \cap \mathcal{T}_j = \varnothing$, $i, j \in \{1, ..., m\}$;

1. $\mathcal{T}_r$ = initial_seed_ selection($\mathcal{T}$, α) // $\alpha = 0.8$;
2. $\mathcal{T}^* = \varnothing$, $i = 1$; $\mathcal{T} = \mathcal{T} - \mathcal{T}_r$;
3. **while** ($|\mathcal{T}| \geq k$) **do** // $|\mathcal{T}|$ denotes the number of trajectories left in $\mathcal{T}$
4. $\forall\ T_p \in \mathcal{T}_r$; $\mathcal{T}_i = \{T_p\}$; $\mathcal{T}_r = \mathcal{T}_r - \{T_p\}$;
5. **while** ($|\mathcal{T}_i| < k$) **do**
6. $\mathcal{T}_i = \mathcal{T}_i \cup \{T_p\}$ where $\max_{T_p \in \mathcal{T}} \left(\sum_{T_q \in \mathcal{T}_i} Sim\left(T_p, T_q\right) \right)$;
7. $\mathcal{T} = \mathcal{T} - \{T_p\}$;
8. **end while**
9. $\mathcal{T}^* = \mathcal{T}^* \cup T_i$, $i = i + 1$;
10. **end while**
11. **while** ($\mathcal{T} \neq \varnothing$) **do**
12. $\forall\ T_p \in \mathcal{T}$, $\mathcal{T}_s = \mathcal{T}_s \cup \{T_p\}$ where $\max_{T_s \in \mathcal{T}^*} \left(\sum_{T_q \in \mathcal{T}_s} Sim\left(T_p, T_q\right) \right)$;
13. $\mathcal{T} = \mathcal{T} - \{T_p\}$;
14. **end while**

Like Algorithm 1, the main work of the TClustering algorithm is done in a two-layer nested loop, of which the innermost operation is step 5 with a cost of $O(n)$. The number of iterations in the outer loop is m and that of the inner loop is k. Thus, the time complexity of the TClustering algorithm is $O(n*m*k)$. Because $m*k$ is approximately equal to n, the time complexity of the TClustering algorithm is $O(n^2)$.

Trajectory generalization algorithm (TGeneralizing). It anonymizes each cluster of trajectories into an anonymized trajectory consisting of location samples with their coordinates denoting regions rather than specific locations samples. Each region is the one with the minimum area that covers all the location samples having the same timestamp. Suppose that $\hat{\mathcal{T}}_s$ is the anonymized trajectory of $\mathcal{T}_s$ ($\mathcal{T}_s \in \mathcal{T}*$), then the trajectory generalization algorithm can be described, as shown in Algorithm 3.

For the TGeneralizing algorithm, its main work is done in a three-layer nested loop, of which the innermost operations are steps 6–9 with each step having a cost of $O(1)$. The number of iterations in the three-layer nested loop are m, n, and $k - 1$, respectively. Thus, the time complexity of the TGeneralizing algorithm is $O(n^2)$.

Trajectory Clustering-Anonymization Algorithm (TCAA). It includes three functional steps: trajectory synchronization, trajectory clustering and trajectory generalization. Based on the above three algorithms, the pseudo-code of the clustering-anonymity algorithm is given in Algorithm 4.

The TCAA algorithm is a k-anonymity scheme with the same privacy-preserving strength as other trajectory k-anonymity methods. However, these methods' time complexities are higher than or equal to ours. In addition, the TCAA algorithm has lower information loss caused by trajectory anonymization. This is because the highest similarity clustering principle ensures that each cluster has the highest cohesiveness. We will validate the effectiveness of the TCAA algorithm in Section 6.

6. Evaluation Results

This section designs two experiments to validate the effectiveness of the proposed trajectory clustering anonymization algorithm TCAA. As for the dataset, we use the OLEDN dataset generated by the Brinkhoff generator [18]. This dataset includes 100,000 trajectories, which simulate one-day movements of moving ob-

Algorithm 3. TGeneralizing.

Input: $\mathcal{T}* = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_m\}$, where $\forall\ T_p \in \mathcal{T}_i\ (i = 1, ..., m)$, $T_p = (t_i^p, x_i^p, y_i^p)$;

Output: $\mathcal{T}* = \{\hat{\mathcal{T}}_1, \hat{\mathcal{T}}_2, ..., \hat{\mathcal{T}}_m\}$, where $\forall\ \hat{\mathcal{T}}_p \in \mathcal{T}*$, $\hat{\mathcal{T}}_p = \{(t_i, [\underline{x}_{\mathcal{T}_p}{}^i, \bar{x}_{\mathcal{T}_p}{}^i], [\underline{y}_{\mathcal{T}_p}{}^i, \bar{y}_{\mathcal{T}_p}{}^i]) \mid i = (1, ..., n)\}$;

1. **for** $s = 1$ to m **do**
2. $\hat{\mathcal{T}}_s = \varnothing$, $k = |\mathcal{T}_s|$; // $\mathcal{T}_s = \{T_1, ..., T_k\}$
3. **for** $j = 1$ to n **do**
4. $X_{min} = x_1^j$, $X_{max} = x_1^j$, $Y_{min} = y_1^j$, $Y_{max} = y_1^j$; // (x_1^j, y_1^j) denotes the coordinate of T_1 at t_j
5. **for** $i = 2$ to k **do**
6. **if** $X_{min} > x_i^j$ **then** $X_{min} = x_i^j$;
7. **if** $X_{max} < x_i^j$ **then** $X_{max} = x_i^j$;
8. **if** $Y_{min} > y_i^j$ **then** $Y_{min} = y_i^j$;
9. **if** $Y_{max} < y_i^j$ **then** $Y_{max} = y_i^j$;
10. **end for**
11. $\hat{\mathcal{T}}_s = \hat{\mathcal{T}}_s \cup \{(t_j, [X_{min}, X_{max}], [Y_{min}, Y_{max}])\}$;
12. **end for**
13. $\mathcal{T}* = \mathcal{T}* - \mathcal{T}_s$, $\mathcal{T}* = \mathcal{T}* \cup \hat{\mathcal{T}}_s$;
14. **end for**

Algorithm 4. TCAA.

Input: $\mathcal{T} = \{T_1, T_2, ..., T_N\}$, where $T_i = \{(t_i^1, x_i^1, y_i^1), (t_i^2, x_i^2, y_i^2), ..., (t_i^{ni}, x_i^{ni}, y_i^{ni})\}$;
Output: $\mathcal{T}^* = \{\hat{\mathcal{T}}_1, \hat{\mathcal{T}}_2, ..., \hat{\mathcal{T}}_m\}$, where $\forall\ \hat{\mathcal{T}}_p \in \mathcal{T}^*$, $\hat{\mathcal{T}}_p = \{(t_i, [\underline{x}_{\mathcal{T}_p}^i, \overline{x}_{\mathcal{T}_p}^i], [\underline{y}_{\mathcal{T}_p}^i, \overline{y}_{\mathcal{T}_p}^i]) \mid i = (1, ..., n)\}$;

begin
1. TSynchzing($\mathcal{T}, \mathcal{T}'$);
2. TClustering($\mathcal{T}', \mathcal{T}''$);
3. TGeneralizing($\mathcal{T}'', \mathcal{T}^*$);

end

jects on the road network of Oldenburg city of Germany. From it, we select 3,000 trajectories that are consistent with the assumptions (1) and (2). For comparisons, we select the HTP-GP algorithm [14] and the PAM-AD algorithm [15]. They are current, excellent privacy-preserving algorithms for trajectory data publishing. All these algorithms, including ours, are implemented in C++. All experiments are run on a PC with Intel(R) Core (TM) i5-4210U CPU @ 1.70GHz (2394 MHz), 4.00 GB RAM, and Microsoft Windows 8.1 operating system. All experiments are conducted five times and the average results are presented.

Experiment 1 (Information loss analysis). It is designed to evaluate the performance of our algorithm in terms of information loss *IL*, which is calculated according to Definition 1. This experiment includes two parts. The first is to test the *IL* of our algorithm concerning k under different values of λ; the second is to test the *IL* of different algorithms concerning k while fixing λ to 0.6. Experimental results are shown in Figure 1 and Figure 2, respectively.

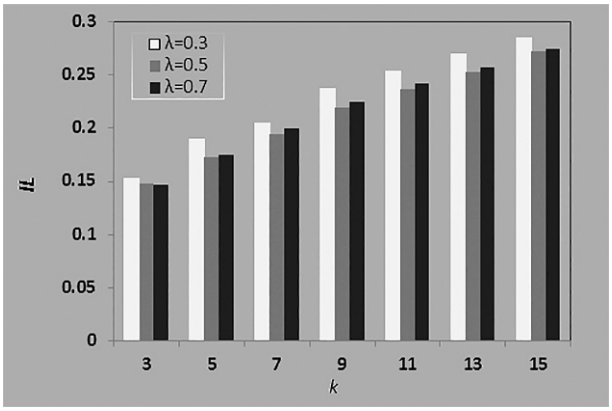

Figure 1. Variations of *IL*s of the TCAA concerning k under different values of λ.

From Figure 1, we can see that *IL* increases along with k. This is because the increase of k means that the number of trajectories included in each trajectory cluster becomes larger, leading to a larger anonymous region for each cluster. The larger the region, the higher the *IL*. Figure 1 also shows that there are slight differences in *IL* at different values of λ with k being the same. The difference reaches the maximum when $\lambda = 0.3$ while declines to the minimum when $\lambda = 0.5$ and $\lambda = 0.7$. This implies that the direction similarity is more effective than the distance similarity in reducing the information loss caused by the anonymization when using TCAA.

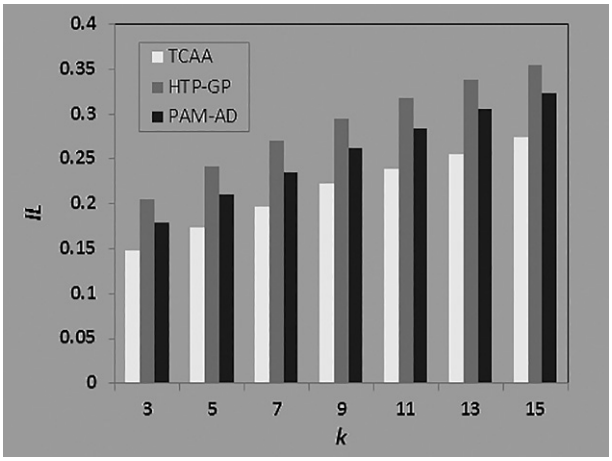

Figure 2. Variations of *ILs* of three algorithms with respect to k while fixing λ to 0.6.

We can see from Figure 2 that the *ILs* of all three algorithms TCAA, HTP-GP, and PAM-AD, increase quickly with k. The reason is the same as in the first part. Figure 2 also shows that among the three algorithms, HTP-GP has the largest *ILs* while TCAA has the smallest

ILs. This is because HTP-GP only takes into consideration the distance similarity without considering the direction similarity, leading to a poor comprehensive similarity. Thus, the anonymous regions of different clusters are large, the likelihood of confusing different trajectories is high, and the information loss is high. Although PAM-AD focuses on the direction similarity between trajectories, its minimum spanning tree-based similarity metric cannot ensure that k trajectories with the maximum similarity are grouped into the same cluster. This causes it to have a worse clustering quality and higher information loss than ours.

Experiment 2 (Algorithm efficiency analysis). It is designed to evaluate the time efficiency of our algorithm in terms of runtime. This experiment also includes two parts. The first part is to test the variations of the runtime of our algorithm concerning k under different values of λ. The second part is to evaluate the runtime of all three algorithms concerning k while fixing λ to 0.6. Experimental results are shown in Figure 3 and Figure 4, respectively.

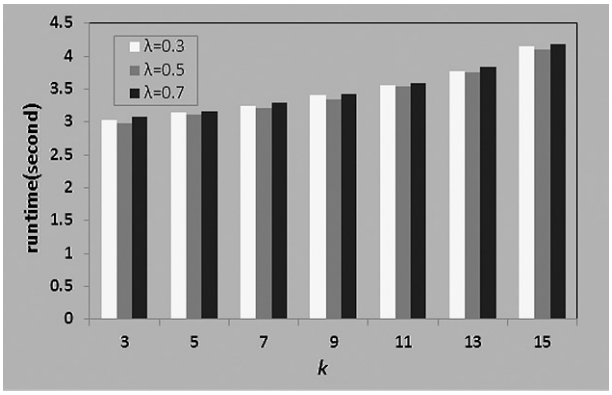

Figure 3. Variations of runtimes of the TCAA concerning k.

From Figure 3, we can see that the runtimes of TCAA increase with k under three different values of λ and they are approximately equal when fixing k. The former is because the overall workload increases with k even though the increase of k is beneficial in reducing the clustering time. The latter is because the workload is unchanged under different values of λ.

We can see from Figure 4 that the runtimes of the three algorithms all slightly increase with k. The reason is the same as it in the first part of this experiment. Figure 4 also shows that, when fixing k, the runtimes of HTP-GP, PAM-AD, and TCAA increase sequentially. The reason is that HTP-GP clusters trajectories only according to the distance similarity metric, which saves many similarity calculations. Compared to HTP-GP, TCAA and PAM-AD spend extra time in calculating the direction similarities between trajectories, leading to a longer runtime. The reason why the runtime of TCAA is longer than that of PAM-AD is that the cluster initialization method and clustering strategy of TCAA spend more calculations than PAM-AD.

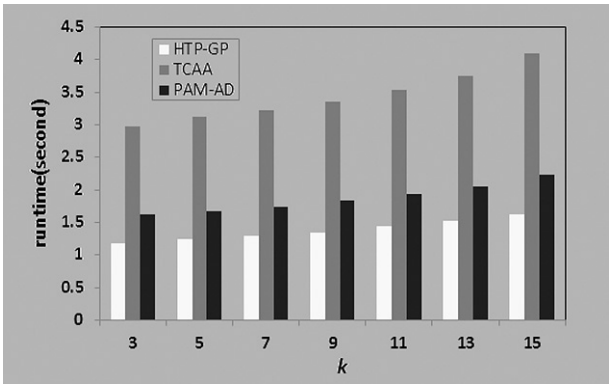

Figure 4. Variations of runtimes of three algorithms under different values of concerning k while fixing λ to 0.6.

7. Conclusion

To solve the privacy disclosure problem caused by trajectory data publishing, we present a clustering-anonymity approach by combining both the trajectory distance and direction. A trajectory synchronization algorithm is designed to synchronize trajectories. Direction and distance similarity metrics and a comprehensive one combining them are defined. A clustering algorithm for dividing trajectories into clusters according to the comprehensive similarity metric is proposed. Compared with [14, 15], our algorithm preserves privacy better and has a lower information loss and a higher runtime efficiency. The drawbacks of our work are that we make relatively strong assumptions regarding the trajectory data and ignore individualized privacy-preserving requirements. We will focus on the drawbacks in our future work.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under grant No. 61762044; the Key Project of Science & Technology Plan by the Education Department of Jiangxi Province under grant No. GJJ170661; the University Humanity and Social Science Project by Education Department of Jiangxi Province under grant No. JC18109; and the Start-up Fund for Doctoral Research Project by Jiangxi Science & Technology Normal University under grant No. 2020BSQD014.

References

[1] B. Qin *et al.*, "Review on Location Privacy Protection Research", *Journal of East China Normal University (Nature Science)*, no. 5, pp. 14–27, 2015.

[2] J. Yuan *et al.*, "T-Drive: Enhancing Driving Directions with Taxi Drivers' Intelligence", *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 1, pp. 220–232, 2013.

[3] J. Yuan *et al.*, "Discovering Regions of Different Functions in a City Using Human Mobility and POIs", *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012, pp. 186–194.

[4] Z. Huo and X. F. Meng, "A Survey of Trajectory Privacy-Preserving Techniques", *Chinese Journal of Computers*, vol. 34, no. 10, pp. 1820–1830, 2011.

[5] M. Gruteser and D. Grunwald, "Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking", in *Proceedings of the 1st ACM International Conference on Mobile Systems, Applications, and Services*, 2003, pp. 31–42.

[6] Q. Gao *et al.*, "Trajectory Big Data: A Review of Key Technologies in Data Processing", *Chinese Journal of Software*, vol. 28, no. 4, pp. 959–992, 2017.

[7] Z. W. Hu and J. Yang, "Survey of Trajectory Privacy Preserving Techniques", *Computer Science*, vol. 43, no. 4, pp. 16–23, 2016.

[8] E. G. Komishani and M. Abadi, "A Generalization-Based Approach for Personalized Privacy Preservation in Trajectory Data Publishing", in *Proceedings of the 6th International Symposium on Telecommunications*, 2012, pp. 1129–1135.

[9] S. Wang *et al.*, "Uncertain Trajectory Privacy-Preserving Method of Moving Object", *Chinese Journal on Communications*, vol. 36, no. Z1, pp. 94–102, 2015.

[10] P. R. Lei *et al.*, "Dummy-Based Schemes for Protecting Movement Trajectories", *Journal of Information Science and Engineering*, vol. 28, no. 2, pp. 335–350, 2012.

[11] K. Y. Lei *et al.*, "Dummy Trajectory Privacy Protection Scheme for Trajectory Publishing Based on the Spatiotemporal Correlation", *Chinese Journal on Communications*, vol. 37, no. 12, pp. 156–164, 2016.

[12] M. Terrovitis and N. Mamoulis, "Privacy Preservation in the Publication of Trajectories", in *Proceedings of International Conference on Mobile Data Management*, 2008, pp. 65–72.

[13] Q. Zhao *et al.*, "A Trajectory Privacy Protection Approach via Trajectory Frequency Suppression", *Chinese Journal of Computers*, vol. 37, no. 10, pp. 2096–2106, 2014.

[14] Z. Huo *et al.*, "History Trajectory Privacy-Preserving Through Graph Partition", in *Proceedings of the 1st International Workshop on Mobile Location-Based Service*, 2011, pp. 71–78.

[15] S. Gao *et al.*, "Balancing Trajectory Privacy and Data Utility Using a Personalized Anonymization model", *Journal of Network and Computer Applications*, pp. 125–134, 2014.

[16] C. Wang *et al.*, "Privacy Preserving Algorithm Based on Trajectory Location and Shape Similarity", *Chinese Journal on Communications*, vol. 36, no. 2, pp. 144–157, 2015.

[17] S. S. Azimuddin and K. Desikan, "A Simple Density with Distance Based Initial Seed Selection Technique for K Means Algorithm", *Journal of Computing and Information Technology*, vol. 25, no. 4, pp. 291–300, 2017.

[18] T. Brinkhoff, "Generating Traffic Data", *IEEE Data Engineering Bulletin*, vol. 26, no. 2, pp. 19–25, 2003.

Received: June 2021
Revised: November 2021
Accepted: December 2021

Contact addresses:

Huo-wen Jiang
College of Mathematics and Computer Science
Jiangxi Science and Technology Normal University
Nanchang
China
e-mail: jhw_604@163.com

Ke-kun Hu
State Key Laboratory of High-end Server and
Storage Technology
Inspur Group Co., Ltd.
Jinan
China
e-mail: hookk@msn.com

Huo-wen Jiang received MSc and PhD degrees in computer science from the University of Electronic Science and Technology of China and Tongji University in 2006 and 2018, respectively. He is now a professor with the College of Mathematics & Computer Science, Jiangxi Science & Technology Normal University. His main research interests include privacy protection and computer education. He is a member of the China Computer Federation.

Ke-kun Hu received a PhD degree in computer science from Tongji University in 2019. He is now a researcher with the State Key Laboratory of High-end Server & Storage Technology, Inspur Group Co., Ltd. His research interests include parallel computing, big graph analytics, and deep learning on graphs.