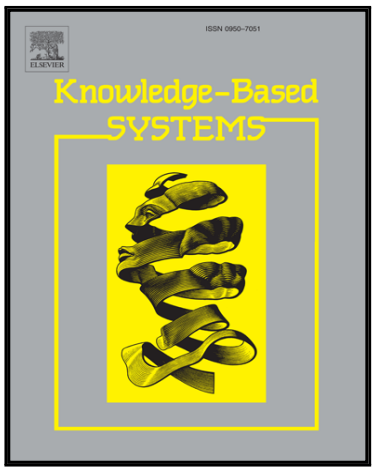


Novel Privacy-preserving Algorithm Based on Frequent Path for Trajectory Data Publishing

Yulan Dong , Dechang Pi

PII: S0950-7051(18)30007-8
DOI: 10.1016/j.knosys.2018.01.007
Reference: KNOSYS 4180

To appear in: *Knowledge-Based Systems*

Received date: 9 August 2017
Revised date: 2 January 2018
Accepted date: 3 January 2018

Please cite this article as: Yulan Dong , Dechang Pi , Novel Privacy-preserving Algorithm Based on Frequent Path for Trajectory Data Publishing, *Knowledge-Based Systems* (2018), doi: 10.1016/j.knosys.2018.01.007

Novel Privacy-preserving Algorithm Based on Frequent Path for Trajectory Data Publishing

Yulan Dong, Dechang Pi

College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

Corresponding author address: nuaacs@126.com. (D. Pi)

Abstract- Existing location-based services have collected a large amount of location data, which contain users' personal information and has serious personal privacy leakage threats. Therefore, the preservation of individual privacy when publishing data is receiving increasing attention. Most existing methods of preserving user privacy suffer a serious loss in data usability, resulting in low usability of data. In this paper, we address this problem and present TOPF, a novel approach for preserving privacy in trajectory data publishing based on frequent path. TOPF aims to achieve better quality of trajectory data for publishing and strike a balance between the conflicting goals of data usability and data privacy. To the best of our knowledge, this is the first paper that uses frequent path to preserve data privacy. First, infrequent roads in each trajectory are removed, and a new way is adopted to divide trajectories into candidate groups. A new method for finding the most frequent path is then proposed, and then, the representative trajectory is selected to represent all trajectories within a group. Experimental results show that our algorithm not only effectively guarantees the privacy of the user but also ensures the high usability of the data.

KEYWORDS

Information publication; Location-based services; Trajectory privacy; Frequent path

1 INTRODUCTION

With the rapid development of location-based services, many mobile positioning devices have emerged, such as car navigation, GPS-enabled mobile phones, tablet PCs and position

sensors. As a result, major manufacturers have launched their own location-based service applications, with which users can send their location and query content to the location server, and then, the location server will return the corresponding query results—for example, Google Maps for navigating services when travelling, Baidu glutinous rice for geo-location search services for nearby restaurants, and WeChat for social services that can share geo-labelling.

These applications can be divided into two types. One is online applications based on the real-time location provided by the user, which require the corresponding services—e.g., location-based services (LBS), push services based on geo-real-time information and real-time monitoring of moving objects—to be provided. The other is offline applications, in which location service providers or other agencies collect and analyse mobile data or publish the trajectory data to third parties. For example, through the excavation and analysis of trajectory data, it is possible to optimize traffic network and traffic management strategies and analyse user behaviour to support business decisions. Although these two types of applications have brought great convenience to people's lives [1], disclosure of their private locations to potentially untrusted LBS service providers poses privacy concerns. Two surveys reported in July 2010 showed that more than half of users who use LBS services are concerned about the disclosure of location privacy [2], and 50 percent of U.S. residents who have a profile on a social networking site are concerned about their privacy [3]. The research results confirmed that location privacy is one of the key obstacles to the success of location-dependent services [4]. Privacy in offline applications is more challenging than online applications because an attacker can infer the user's location information by using the spatial and temporal correlations in the user's location samples. However, the trajectory formation is important for many applications in real life, such as business analysis, city planning, or transportation planning. Therefore, privacy protection in offline applications and trajectory data publication has increasingly drawn attention from the industry and academia.

Many approaches have been proposed for preserving privacy in trajectory data publishing, but most do not consider the usability of data for publishing. The result is that our privacy may be preserved well while the trajectory data are of no value to the applications (e.g., city planning), leading to a significant loss of information. Since most publishing information is used for data mining and analysis, it is necessary to focus on the hot region or frequent path. In this paper, we address this problem by using frequent path to preserve data privacy, which has not been studied in previous work, so that not only can data privacy be preserved but also the usability of data can be increased.

The main contributions of this paper are summarized as follows:

- We present TOPF, a novel approach for preserving data based on frequent path. TOPF considers not only data privacy but also the usability of data. To the best of our knowledge, this is the first study that uses frequent path to preserve data privacy.

- TOPF removes infrequent roads and adopts a new method to divide all trajectories into candidate groups. Trajectories in each group are similar and construct k-anonymity, which leads to a low average error rate.

- TOPF also employs a new approach to find the most frequent path. Concerned with the average error rate in each group, TOPF first uses the new approach to select the *f* most frequent trajectories and then chooses the one with the highest similarity to all other trajectories within the group as the representative trajectory. Hence, the frequent patterns are well preserved.

- We use average error rate and standard deviation as metrics to evaluate the quality of the anonymized dataset of trajectories. F-measure is also employed to evaluate the frequent pattern discovered by TOPF. The extensive experimental results show that our method can effectively preserve the user's trajectory privacy and better retain the usability of the data.

The rest of this paper is organized as follows: Section 2 reviews related work. Section 3 provides the problem statement. Section 4 presents our proposed approach. Section 5 reports experimental results. Finally, Section 6 concludes the article. For clarity, the main notations used in the rest of this paper are summarized in table 1.

Table 1 Summary of Notations

Notation	Description
k	the anonymity threshold
G	road network
P	path
T	trajectory
n_j	j^{th} node
V, E	vertex, edge
$F(P)$, f_j	sequence of frequent of path P, *frequency of* j^{th} *road*
θ	threshold to compare the frequencies of roads in paths
δ	minimal number of trajectories to construct a group

2 RELATED WORK

The most widely used privacy-preserving method for data publishing is k-anonymity, which was first proposed by Sweeney et al. in [5]. The method requires each record to be indistinguishable with at least k−1 other records with respect to the quasi-identifier, i.e., each equivalence class contains at least k records. In [6], Gruteseter et al. first applied the k-anonymity method to the location service and proposed the concept of location k-anonymity, which requires that the space-time location point sent by a user be indistinguishable from the space-time location sent by k-1 other users. In [7], Abul et al. proposed a model named (k, δ)-anonymity, based on k-anonymity, and designed an algorithm named Never Walk Alone (NWA) to achieve (k, δ)-anonymity. They clustered the trajectory with the same period of time according to Euclidean distance and obtained the cluster group composed of the trajectories with similar distances. However, that method does not generate anonymized trajectories that follow the road-network constraints, which increases the difficulty of future trajectory mining. Moreover, although the trajectories are similar in the European space theory, they may not be similar in the actual road network space, which will result in anonymous failure. Subsequently, Abul et al. [8] put forward an improved algorithm called Wait For Me (W4M), using EDR [9] instead of the Euclidean distance as the similarity function in the trajectory clustering process. However, the anonymized trajectories generated by W4M still do not follow road-network constraints. In [10], Nergiz et al. proposed a condensation-based grouping algorithm for trajectory k-anonymity. The method enforces k-anonymity by clustering trajectories based on log cost distance and then reconstructing trajectories by randomly selecting location samples from anonymized regions. Although the privacy-preserving degree of this kind of algorithm is high, it can support only simple aggregation analysis and cannot be applied to other applications such as behaviour pattern discovery and the mining of association rules. In[11], Wang et al. proposed a novel continuous query privacy-preserving framework in road networks based on the concepts of k-anonymity and l-diversity, assuming that conventional preserving solutions designed in Euclidean space cannot easily be applied to the road network environment. They also designed two types of cloaking algorithms, including one for a single user and one for a batch of users; the algorithms can resist typical attacks and effectively preserve users' query privacy in road networks.

In [12], Chen et al. adopted a model named (K, C) L-privacy for trajectory data anonymization, which considers not only identity linkage attacks on trajectory data but also attribute linkage attacks via trajectory data. They also proposed an anonymization framework that can remove all privacy threats from a trajectory database by both local and global suppression. In [13], Gao et al. Proposed a personalized anonymization model to balance the trajectory privacy and data utility. Existing methods ignore the trajectory similarity and direction, which they think has a large impact on privacy. As a result, they proposed using the trajectory angle to evaluate trajectory similarity and direction and construct an anonymity region based on trajectory distance. Huo et al. [14] put forward a new idea that the background information obtained by attackers is more relevant to where the moving objects really visit rather than where they merely pass by. Thus, they proposed an approach called You Can Walk Alone (YCWA), which divides the location samples into two categories: pass-by points and stay points. The stay points are then

replaced by corresponding zones based on split map, and pass-by points are either deleted or unprocessed, depending on whether they are inside a zone. That method protects trajectory privacy only through generalization of stay points on trajectories, so the attacker can still analyse the movement mode of the trajectory to obtain background knowledge. Wu et al. [15] proposed the (k, δ, Δ)-anonymity model to avoid re-cluster attacks and presented a clustering hybrid-based algorithm, CH-TDP, for privacy-preserving trajectory data publishing. The method first hybridizes between clustering groups generated by the (k, δ, Δ)-anonymity model and the related algorithm, and then it adopts perturbation within each clustering group. Although it avoids suffering re-cluster attacks effectively, the data quality of the released trajectory data has been reduced. A segment clustering-based privacy-preserving algorithm was proposed by Li et al. [16], which first divides the original database into blocks and then partitions trajectories in each block into segments, which are finally anonymized with a cluster-constraint strategy. The anonymization results of that method are diverse but not beneficial for data analysis. Zhang et al. [17] proposed a UGC scheme that utilizes the uniform grid, order-preserving symmetric encryption (OPSE) and a k-anonymity mechanism to preserve users' location privacy. Thus, the anonymizer knows nothing about a user's real location, and it can implement only simple matching and comparison operations. However, if a user always uses the same key to encrypt the coordinates in continuous queries, the security of the key will not be guaranteed. Most current practices fall into the k-anonymity model, which suffers from many constraints according to Ni et al. [18]. As a result, they proposed a novel location privacy model, (s, ε)-anonymity, which features location protection strength and scaling of intermediate results. Users need only to set parameters s and ε to meet their preferential query requirements on privacy protection strength and query efficiency. Furthermore, they also developed a thin server solution to realize the model instead of using any trusted third parties' intervention.

Most existing works on privacy-preserving location publishing consider trajectories represented as sequences of coordinates and output anonymization results in the form of cloaking regions or centres of clusters. However, these approaches do not generate anonymized trajectories that follow the road network constraints. Their anonymization results preserve user privacy but are not beneficial for trajectory analysis on individual roads. There are few works consider trajectories represented by roads. Pensa et al. [19] proposed a prefix-tree based anonymization algorithm that guarantees k-anonymity of the published trajectories such that no trajectories with support less than k will be published. However, their anonymization result will be an empty set since the prefix tree treats trajectories with different starting points independently. This result obviously loses too much useful information. Gurung et al. [20] adopted a method which generates anonymized trajectories that follow the road-network constraints. Their method groups the similar trajectories into clusters and anonymizes them by using a representative trajectory. Although that method follows the road-network constraints, it divides the trajectories into clusters by comparing the similarity of roads in the trajectory, which leads to the large differences between trajectories in the same cluster and reduces the final data usability.

The key challenge for trajectory data publication is how to wisely use the data without violating each user's location privacy concerns. To be more effective for trajectory analysis, we use sequences of roads to represent trajectories. Moreover, considering that current trajectory

analysis is based on frequent trajectory patterns, we believe that frequent patterns should be retained well in trajectory data publication. From this point of view, we present a k-anonymity privacy protection method based on frequent path without destroying data usability.

3 PROBLEM STATEMENT

3.1 Trajectory Database

In general, trajectories of moving objects are collected and stored in moving object databases. For a moving object u_i, its trajectory T_i is a set of discrete locations at sampling times, represented as $T_i = \{ID_i, (x_1, y_1, t_1), (x_2, y_2, t_2), \ldots (x_n, y_n, t_n)\}$, where ID_i is the identifier of the trajectory, (x_j, y_j, t_j) represents the moving object's position at sampling time t_j, and (x_j, y_j, t_j) is a location sample on trajectories. To ensure the usability of data, this paper adopts data representation based on road networks. Each of the sampling points in the trajectory is matched to a road and represented by the road ID. This not only ensures the authenticity of the trajectory but also effectively protects the specific location of a user's visit. Thus, the anonymized dataset is in the form of $\{AID_i, (r_1, t_1), (r_2, t_2) \ldots (r_n, t_n)\}$, where AID_i is an anonymized object ID, r_j is a road ID, and t_j is the sampling time. In the following, we provide some definitions in our study.

Definition 1 (Trajectory k- anonymity). Let D^* be a k-anonymized trajectory dataset to publish, and let T_{us} be the trajectory of user u where $T_{us} \in D^*$; this should be indistinguishable from no less than $k-1$ other trajectories by its own AIDs for all time stamps.

Definition 2 (Road network). A road network is a directed graph $G=(V,E)$ where V is a set of vertices representing road intersections, and E is a set of edges representing road segments.

Definition 3 (Frequent road). Let k be a threshold; we say a road is a frequent road if the number of moving objects along one direction on this road is no less than k. We call the number of moving objects the frequency of the road.

Definition 4 (Path). Let G be a road network; an n_j - n_k path is a non-empty graph $P=(V_p, E_p)$ of the form $V_p = \{n_j, n_{j+1}, n_{j+2}, \ldots, n_k\}$ and $E_p = \{(n_j, n_{j+1}), \ldots (n_{k-1}, n_k)\}$. In addition, n_j and n_k represent the start and end points of a path, respectively.

Definition 5 (Path support and Path frequency). Let P be a path; we call the number of trajectories going through P the support of path P. In addition, we use a sequence

$F(P)=(f_1,...,f_n)$ that represents the frequency of path P, where f_j is the frequency of the road and $f_1 \le f_2 \le ... \le f_n$.

Because it is nontrivial to provide a satisfactory definition to well reflect people's common sense notions, we will illustrate Definition 5 by a concrete example. Fig. 1 shows a road network along with 46 trajectories, which are divided into groups according to whether they traverse the same path. As depicted by dashed curves, there are 7 trajectory groups (G_1 through G_7), containing 1, 4, 15, 7, 8, 6, and 5 trajectories, respectively. For ease of presentation, we use the notation v_j-v_k to denote a path from v_j to v_k. Hence, the paths traversed by G_1, G_2, G_3, G_4, G_5, G_6 and G_7 are denoted as $V_1 \to V_4 \to V_5 \to V_{13}$, $V_1 \to V_4$, $V_4 \to V_5 \to V_{13}$, $V_1 \to V_2 \to V_3 \to V_{13}$, $V_1 \to V_2 \to V_6 \to V_{13}$, $V_3 \to V_{13}$ and $V_1 \to V_7 \to V_8 \to V_9 \to V_{10} \to V_{11} \to V_{12} \to V_{13}$, respectively.

According to Definition 5, in other words, $F(P)$ is a non-decreasing sequence of the frequencies of all roads in P. For example, the frequencies of all roads in path $V_1 \to V_2 \to V_3 \to V_{13}$ in Fig. 1 are 15(7+8=15), 7, 13(7+6=13) respectively, so the frequency of $V_1 \to V_2 \to V_3 \to V_{13}$ is (7, 13, 15).

Next, we present how to find the most frequent path. A straightforward method is to count the trajectories going through the path and select the one with the highest support. For example, there are four V_1-V_{13} paths with non-zero support in Fig. 1; *i.e.*, the paths traversed by G_1, G_4, G_5, G_7 whose supports are 1, 7, 8, 5, respectively. Hence, the frequency relationship among the four paths is $G_1 < G_7 < G_4 < G_5$. However, this method is only a comparison of the frequency of each path and cannot reflect the frequency of each road in the paths. For example, the support of path G_1 is 1, but the moving objects on the path $V_1 \to V_4 \to V_5 \to V_{13}$ are 5(1+4=5), 16(1+15=16), and 16(1+15=16), which are more than path G_7, whose frequency is 5 for each road.

Another approach in [21] is to adopt a scalar-valued score function to calculate the path frequency. One possible score function is the sum of all weights of the edges along the path where the edge weight describes the road frequency. However, this approach suffers from two major drawbacks. The first drawback is that the number of path edges can significantly affect the overall path score. For example, it is intuitive that path $V_1 \to V_2 \to V_6 \to V_{13}$ in Fig. 1 is more frequent than path $V_1 \to V_7 ... \to V_{12} \to V_{13}$. However, if we adopt the above sum-of-edge-frequency definition, the frequencies of all roads in $V_1 \to V_2 \to V_6 \to V_{13}$ are 15(7+8=15), 8, 8 respectively. There are 7 roads in $V_1 \to V_7 ... \to V_{12} \to V_{13}$, and the frequency of each road is 5. Hence, the score of the former (15+8+8=31) is lower than that of the latter (5*7=35), which contradicts the intuition. The second drawback is that the resulting frequent path may contain very infrequent

edges because the weight of the infrequent edges can be easily offset by the weight of the frequent ones. For example, the frequencies of all roads in $V_1 \to V_4 \to V_5 \to V_{13}$ are 5(4+1), 16(1+15), 16(1+15), and the score is 5+16+16=37, which is the highest among all V_1 - V_{13} paths. However, it contains road $V_1 \to V_4$, whose support is smallest in Fig. 1.

A new approach is proposed in [22], which adopts an ascending frequency sequence to define the path frequency and considers that the size of the smallest element in the sequence determines the frequency of the path. For example, the frequencies of all roads in path $V_1 \to V_2 \to V_6 \to V_{13}$ are 15(7+8), 8, 8 respectively, so the frequency of the path is (8, 8, 15). Similarly, the frequency of path $V_1 \to V_2 \to V_3 \to V_{13}$ is (7, 13, 15). As a result, if we adopt this method, path $V_1 \to V_2 \to V_6 \to V_{13}$ (8, 8, 15) in Fig. 1 is more frequent than path $V_1 \to V_2 \to V_3 \to V_{13}$ (7, 13, 15), but the latter seems to be more frequent than the former.

In this article, we use a threshold θ to make the method that finds the most frequent path more reasonable. The value of θ will be discussed in section 4.4.

Definition 6 (More-frequent-than). Given two path frequencies $\mathrm{F}(P) = (f_1, \ldots, f_m)$ and $\mathrm{F}(P') = (f_1', \ldots, f_n')$, $F(P')$ is more frequent than $F(P)$, denoted as $F(P') \geq F(P)$, if one of the following statements holds:

1) There exists a $q \in \{1, \ldots min(m,n)\}$ such that $|f_j' - f_j| \leq \theta$, $f_q - f_q' > \theta$ for all $j \in \{1, \ldots, q-1\}$, if $q \geq 1$.

2) There exists a $q \in \{1, \ldots min(m,n)\}$ such that $|f_j' - f_j| \leq \theta$, $\sum_{1}^{n} f'_n \geq \sum_{1}^{m} f_m$ for all $j \in \{1, \ldots, q\}$, if $q \geq 1$.

We set θ = 2; according to the definition, the path $V_1 \to V_2 \to V_6 \to V_{13}$ (8, 8, 15) is more frequent than the path $V_1 \to V_2 \to V_3 \to V_{13}$ (7, 13, 15). Specifically, we first compare the first element of each sequence, which is |8-7|=1< θ; we continue comparing the second element and find that |8-13|=5> θ and 8<13. Hence, the former is more frequent than the latter.

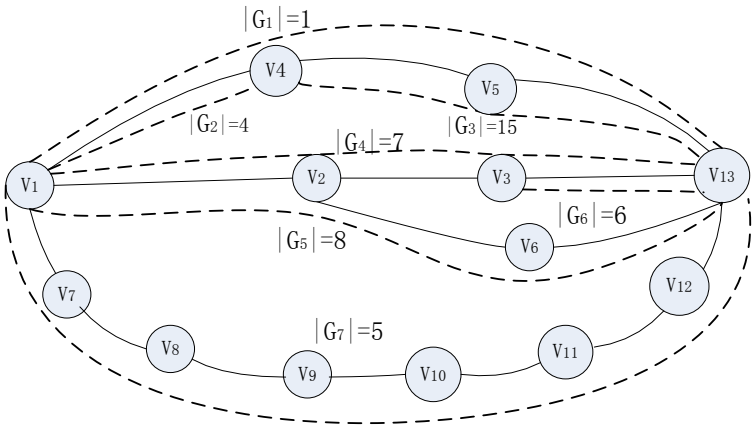


Fig. 1. An illustrative example

3.2 Privacy Attack

Definition 7 (Identity-linked attack). If a trajectory in the trajectory database is very specific, such that few moving objects can match it, the adversary using some background knowledge may uniquely identify the data record of the target victim and, therefore, its sensitive attribute values[12].

For example, consider four users who leave their homes (E, F, G, H) and head for work (D) in Fig. 2. The trajectories of u_1, u_2, u_3 and u_4 are (F, A, B, D), (G, A, B, D), (H, A, B, D) and (E, A, C, D), respectively. Since the road map can be found everywhere in the domain of privacy-preserving location publication, it is reasonable to assume that road network information is available to any adversary. For example, Fig. 2. is accessible to adversary Tom. If Tom observes that Mary is passed by (C, D), then Tom can infer that u_4 is Mary, who is the only one with trajectory entering (C, D). Upon knowing the anonymous ID of Mary, Bob can track Mary's remaining trajectories. From this, we can see that the infrequent trajectories with high probability combined with certain external knowledge can be used to identify a particular individual's trajectory information in the published dataset.

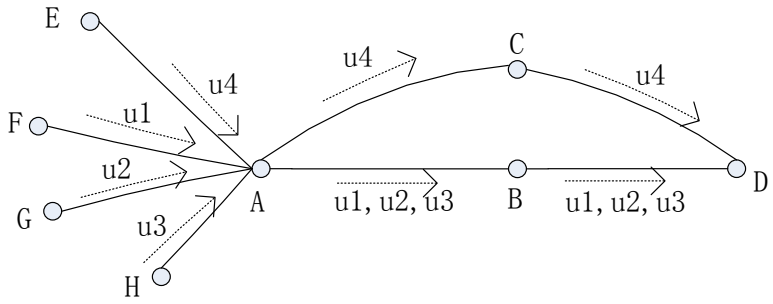


Fig. 2. An example of an identity-linked attack

With adequate background knowledge, an adversary can easily perform an identify linkage attack and identify the user's trajectory information. Let S_i be the background knowledge used by adversary $Å$ to attack user u_i, where $S_i = \{l_1, l_2, \ldots, l_n\}$ and $l_j = (x_j, y_j, t_j)$. Given a trajectory data record T_j, according to Bayes' theorem, the adversary may identify T_j with confidence $Pr(T_j \mid S_i)$:

$$Pr(T_j \mid S_i) = \frac{Pr(S_i, T_j)}{Pr(S_i)} = \frac{Pr(S_i \mid T_j)\,Pr(T_j)}{Pr(S_i)} \propto Pr(S_i \mid T_j) \tag{1}$$

The adversary need only select the one with the highest confidence as the object of attack. Note that the denominator is a constant. In addition, without any knowledge about how the victim is chosen, we set the a priori distribution of the victim to be uniform (i.e., $Pr(T_j) = 1/n,\ \text{for}\ i = 1, 2 \ldots N$, where *N* is the number of trajectories in the database). Assuming that all terms of each track are independent, the probability of $Pr(S_i \mid T_j)$ can be calculated as

$$Pr(S_i \mid T_j) = \prod_{Z=1}^{m} Pr(S_i * l_z - T_j * l_z) \tag{2}$$

where *m* is the number of items in a trajectory, and $S_i * l_z - T_j * l_z$ is the difference between items of two trajectories. Since the Gaussian distribution is one of most common distributions for identity linkage attacks, we assume that the noise obeys a Gaussian distribution $N(0, \sigma^2)$. Thus, expression (2) can be simplified as

$$\Pr(S_i \mid T_j) = C * \exp\{-\frac{1}{2\sigma^2}\sum_{Z=1}^{m} | S_i * I_Z - T_j * I_Z |^2\} \quad (3)$$

for some constant C. Hence, the goal of the adversary is to find the minimal value of $\sum_{Z=1}^{m} | S_i * I_Z - T_j * I_Z |^2$.

3.3 Trajectory Privacy Metric

To evaluate user u_i 's degree of privacy, we use information entropy as a metric. When the trajectory of u_i is not protected, the degree of privacy of u_i is defined as

$$H_i^0(T) = -\sum_{j=1}^{n} \Pr(T_j \mid S_i) \log_2 \Pr(T_j \mid S_i) \quad (4)$$

The most widely used privacy-preserving method for data publishing is k-anonymity, and we must cluster at least k trajectories to achieve k-anonymity. Given a set of k-1 trajectories $T_i = \{T_{i_1}, T_{i_2}, \ldots, T_{i_{k-1}}\}$ used to preserve the privacy of u_i , the privacy level is raised to

$$H_i(T, T_i) = H_i^0(T) + H_i^X(T_i)$$

$$= -\sum_{j=1}^{n} \Pr(T_j \mid S_i) \log_2 \Pr(T_j \mid S_i) - \sum_{Z=1}^{k-1} \Pr(T_{iZ} \mid S_i) \log_2 \Pr(T_{iZ} \mid S_i) \quad (5)$$

where $H_i^X(T_i)$ is the degree of privacy increased by k-anonymity, and T_{iZ} is one of the trajectories in the k-anonymity group.

3.4 Privacy Model

As mentioned before, we must cluster at least k trajectories to achieve k-anonymity, which will inevitably produce some costs that we assume are mainly information loss (i.e., remove infrequent roads, translate into representative trajectory, and add dummy trajectories). Each trajectory must spend some efforts to join a group. K-anonymity not only enhances privacy protection for a user who joins the group but also benefits other users within the same group. When publishing trajectory data, we want our privacy to be well preserved and have high data quality. That is, the cost must be low, and the degree of privacy must be high. As a result, the ultimate goal of this article is to minimize the cost-benefit ratio (CBR); that is,

$$\min_{V_i, T_i} \quad \frac{\sum_{i=1}^{N} V_i}{\sum_{i=1}^{N} H_i(T, T_i)} \tag{6}$$

$$\text{s.t.} \quad V_i \in R^+ \tag{7}$$

$$T_i \in \{T_1, T_2, \ldots T_n\} \tag{8}$$

where V_i is the cost of user u_i to join a group.

TOPF adopts k-anonymity to preserve trajectory privacy, which ensures that each group contains at least k trajectories; the greater the number of trajectories, the greater the degree of privacy and the greater the denominator of formula (6). Now let us assume a minimal denominator; that is, the number of trajectories is k. We then need only prove that the molecule is small – that is, the quality of data after preserving is high. The extensive experimental results show that our method retains high usability of the data.

4. OUR APPROACH

Most existing privacy-preserving approaches for trajectory data publishing methods mainly transform the trajectory anonymous problem into a trajectory clustering constraint problem to protect user privacy. First, the trajectory data are clustered according to the similarity in those methods. The generated clustering groups are then transformed into corresponding anonymous groups by using the constraint operations. The most related work was proposed by Gurung et al. [20]. They first compared the similarity between the trajectory and representative trajectory of each cluster and then added the trajectory to the cluster with the highest similarity. Finally, they added the dummy trajectories to the cluster to guarantee k-anonymity. However, if only some roads in the trajectory are similar, there is no way to tell whether the users' trajectories are associated in time and space. Moreover, that approach leads to large differences between trajectories in the same cluster and reduces the final data quality.

Since the most publishing information is used for data mining and analysis, it is necessary to focus on the hot region or frequent path. In this paper, we propose a novel method based on

frequent paths that not only presents a new way of trajectory division but also protects frequent patterns in trajectories. The method will be described in the following subsections.

4.1 An Overview of TOPF Anonymization

First, we must select a proper way to represent trajectories. Our method anonymizes the trajectories that follow road-network constraints, which generalizes the request point to the road where the request point is located. To represent the user's moving direction, we represent the trajectory by road IDs.

Next, we will present our TOPF algorithm. TOPF consists of three main steps: (1) data preprocessing (*i.e.,* remove infrequent roads), (2) finding candidate groups, and (3) selecting the representative trajectory.

An attacker can easily acquire all trajectory information by observing and analysing some infrequent roads in the trajectory and performing an identity linkage attack on infrequent trajectories. Our algorithm first processes all trajectories by removing records associated with infrequent roads (i.e., roads with less than k objects) in the obtained database (line 1). The first step is relatively straightforward; therefore, the following discussion will focus on step 2 and step 3.

We then construct partial trajectories for the remaining objects. This paper argues that if the users request the same destination at the same location, their trajectories must be similar or relevant in time and space. We use the hash table to determine whether there exist trajectories with the same start point and end points and, if so, place them into the same entry (lines 2-10). The entry consists of two parts. One stores the start and end points of the trajectory, and the other holds the trajectories corresponding to these start and end points.

If the number of trajectories in the entry is greater than the anonymization threshold k, these trajectories themselves form a group (lines 13-14). For the remaining trajectories, we place them in a list (lines 15-16) sorted in ascending order of their length (line 20) and compare them with existing groups. If a trajectory contains a sub-trajectory whose start and end points already exist in a group, the FindGroup (Algorithm 2) algorithm adds this trajectory to a suitable group (lines 22-23). Otherwise, a new cluster is created for this trajectory (lines 24-25).

Finally, there is a grouping adjustment phase that handles groups containing less than k trajectories. We check whether the number of trajectories in a group is less than δ; if the group contains less than δ ($\delta < k$) trajectories, we directly remove it (lines 29-30). Otherwise, we use SelectRep (Algorithm 3) to select a representative trajectory for each group, and then, we add dummy trajectories to the group by increasing the support of the representative trajectory to k (lines 32-33). Finally, we translate representative trajectories into output format (line 36).

The algorithms for finding candidate groups and selecting representative trajectories will be elaborated in the following subsections.

ALGORITHM 1: TOPF Anonymization (Trj, k)

Input: *Trj is a set of trajectories that consist of road IDs, and k is the threshold of anonymization*

Output: *Trajectories after the anonymization*

01. *Delete roads that are infrequent*

02. *Create a hashtable Mtrj*

03. ***for*** *each trj in Trjs* ***do***

04. ***if*** *the start and end points of trj is one of Mtrj's key* k_1 ***then***

05. *list = Mtrj.get*(k_1)

06. *list.add(trj)*

07. ***else***

08. *Put(start and end points, trj) into Mtrj*

09. ***end if***

10. ***end for***

11. ***for*** *entryset in Mtrj* ***do***

12. *trjs = entryset.getValue*

13. ***if*** *trjs.size* $\geq k$ ***then***

14. *create a new group for trjs*

15. ***else***

16. *add trjs into Ltrjs*

17. ***end if***

18. ***end for***

19. *Add each group into groups*

20. *Sort the trjs in Ltrjs by length in ascending order*

21. ***for*** *trj in Ltrjs* ***do***

22. ***if*** *sub-trjs are in group which in groups* ***then***

23. *FinfGroup(trj, groups)*

24. ***else***

25. *Create a new group*

26. ***end if***

27. ***end for***

28. ***for*** *each group in groups* ***do***

29. ***if*** *group.size* $\leq \delta$ ***then***

30. *remove group*

31. ***else***

32. *SelectRep(group)*

33. *set group.size = k*

34. ***end if***

35. ***end for***

36. *Translate representative trajectories into output format*

4.2 Finding Candidate Groups

The essential idea of Algorithm 2 is to find a candidate group for a new trajectory that cannot be added to a certain group directly. First, we resolve this new trajectory into several nodes. The algorithm compares these nodes with start and end points in the groups to see whether the node set converts both the start and end points of a trajectory cluster (*i.e.*, this new trajectory contains a sub-trajectory with the same start and end points as other trajectories in the group) (lines 2-3). If only one sub-trajectory is found, it will be simply added to the corresponding group (lines 6-7).

However, the operation becomes complex if two sub-trajectories are involved. This scenario can be divided into three different cases. First, one sub-trajectory includes the other one. The longer sub-trajectory is added to the group with a dummy ID, and the shorter one is ignored (lines 10-11). Second, these two sub-trajectories intersect each other. Both will be added to the corresponding group and assigned dummy IDs. Third, these two sub-trajectories do not intersect. In this situation, we also add the sub-trajectories to corresponding groups and assign them dummy IDs, the same operation as in the second case (line 14).

What is more, if more than two sub-trajectories are involved, we can also address this situation by using a similar scheme as in the above two sub-trajectory cases. This method protects the authenticity of frequent path data.

To illustrate this, we use the example in Fig. 3. Suppose t_1 and t_2 are two new trajectories that must be inserted into suitable groups; g_1, g_2 and g_3 are existing groups that each contain trajectories with the same start point and end point. For example, in group g_1, the start point and end point of $trj_1,\ldots,trj_5$ are all n_2 and n_4, respectively. These two trajectories cannot be added directly to any of the above three groups (g_1, g_2 and g_3). Trajectory t_1 contains nodes n_4 and n_6 (*i.e.*, g_3) and only g_3 is contained in this trajectory. Thus, we simply extract this sub-trajectory (with n_4 and n_6 as start and end points, respectively) and add it into g_3. For t_2, we can find three candidate groups—namely, g_1, g_2 and g_3. The paths traversed by g_1, g_2, g_3 are n_2-n_3-n_4, n_2-n_3, n_4-n_5-n_6, respectively. It is obvious that g_2 is included by g_3. Hence, we add sub-trajectory n_2-n_3-n_4 and sub-trajectory n_4-n_5-n_6 into g_1 and g_2 with different dummy IDs, respectively.

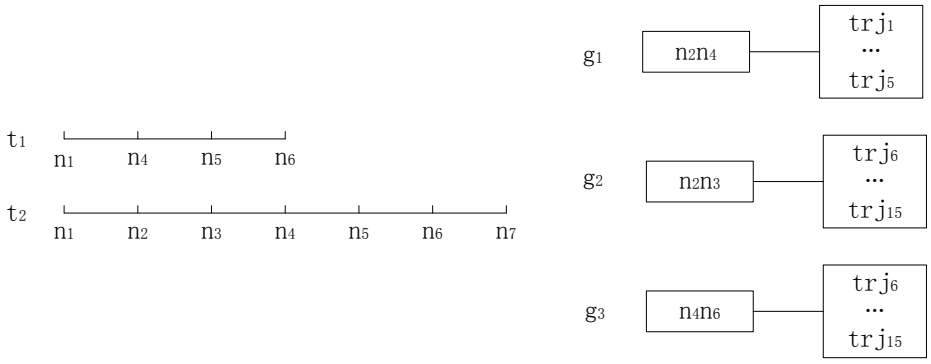


Fig. 3 An illustrative example

ALGORITHM 2: FindGroup (trj, groups)

Input: *trj is a trajectory that must be added into a candidate group, and groups is a set of groups*

Output: *candidateG is a set of candidate groups*

01. ***for*** *group in groups* ***do***

02. ***if*** *trj contains group* ***then***

03. *candidate.add(group)*

04. ***end if***

05. ***end for***

06. ***if*** *candidateG.size ==1* ***then***

07. *put trj into group in candidate*

08. ***else***

09. ***for*** *goup in candidateG* ***do***

10. ***if*** *group_i contains group_j* ***then***

11. *candidateG.remove(group_j)*

12. ***end if***

13. ***end for***

14. *Divide trj and put it into groups in candidateG respectively and give different IDs*

15. ***end if***

4.3 Selecting Representative Trajectory

There are two key requirements when selecting a representative trajectory. First, the frequency of the representative trajectory should be high. Second, the representative trajectory should have the highest average similarity (Equations (9) and (10)) with trajectories in the group, to ensure a low error rate.

In a group, we sort the trajectories in descending order of their frequent relationships (line 1) and select the top *f* most frequent trajectories (line 2); we then compare these m trajectories with other trajectories within the group and select the one with the highest average similarity as the group representation (lines 3-5). The average similarity function is as follows:

$$sim(trj_a, trj_b) = \frac{|S(trj_a) \cap S(trj_b)|}{|S(trj_{b)})|} \tag{9}$$

$$avgS = \frac{1}{N}\sum_{j=1}^{N} sim(trj_j, trj_{rep}) \tag{10}$$

Suppose there are N roads in a group, and trj_j represents trajectory j in the group. Let $S(trj_a)$ denote the set of IDs in trajectory trj_a. In Equation 1, *sim* then computes the percentage

of common roads included in trj_a and trj_b. $avgS$ is defined as the average similarity among the representative trajectory and other trajectories in the group. A high $avgS$ indicates that the representative trajectory is similar to the other trajectories in the group.

ALGORITHM 3: selectRep (group)

Input: *group consists of start and end points and a set of trajectories*

Output: *trj, which is the representative trajectory of group*

01. *Sort the trajectory by frequency*

02. *Select the top f trajectories*

03. ***for*** *trj in top* ***do***

04. *Find the trj with biggest similarity*

05. ***end for***

06. ***Return*** *trj*

4.4 Selection of Threshold

Threshold selection is a critical task that affects anonymization accuracy. In this subsection, we will discuss how to determine the threshold θ for selecting the most frequent path and the threshold δ for grouping the adjustment phase.

The threshold θ determines whether a path is the most frequent in a group. If a low threshold is used, the most frequent path may be determined by the size of a small element in the sequence. For example, let θ be 1; according to Definition 6, path A (10, 11, 12) is said to be more frequent than path B (8, 20, 25), while path B seems to be more frequent. Moreover, a low threshold also leads to a low diversity in the most *f* frequent trajectories, which will increase the average error rate. If a high threshold is chosen, even some trajectories containing low frequencies can rise to the top of the group. For example, let θ be 10; path C (2,20,25) is said to be more frequent than path D (10,11,12), while path C contains a road with the lowest frequency. However, the greater the value of θ, the greater the tolerance of the frequency relationship and

the higher the diversity of the trajectory. Thus, the average error rate will be reduced. To reach a balance, we define the threshold θ as shown in Equation (11).

$$\theta = \frac{k}{4} \tag{11}$$

This threshold is derived according to *k*; a larger *k* yields a higher threshold θ, because *k* determines the smallest element in the frequency sequence.

After grouping all trajectories, a group may contain less than *k* trajectories. For such groups, the threshold δ is used to determine whether to remove the groups or add dummy trajectories into them. To minimize error, we define the threshold δ as shown in Equation (12).

$$\delta = \frac{k}{2} \tag{12}$$

The basic idea of this threshold is to induce less error when inserting or deleting few trajectories. Specifically, if the total number of trajectories in a group is greater than *k*/2, adding less than *k*/2 trajectories will introduce less error than moving the whole group. In the other case, if a group has less than or equal to *k*/2 trajectories, removing the group will introduce less error by adding more than *k*/2 dummy trajectories.

4.5 Complexity Analysis

As mentioned before, TOPF consists of three main steps: (1) data preprocessing (*i.e.*, removing infrequent roads), (2) finding candidate groups, and (3) selecting a representative trajectory.

To remove infrequent roads from the original dataset, we must scan the roads that are contained in all trajectories just once. Suppose there are n trajectories in the original dataset and the maximal number of roads in the trajectory is l. Given a constant number l, the total number of such roads is $n \times l$. The complexity of the first step is $O(n)$.

For the second step, the major cost is the search for candidate groups. We must scan existing groups to find all suitable groups. Suppose there are l_c trajectories per group; then, the number of groups can be estimated as n/l_c. Therefore, the time to find the candidate group is $O(n/l_c)$—that is, $O(n)$.

The third step is to select the representative trajectory, and we must scan trajectories in each group. The average number of trajectories per group is l_c; therefore, the time complexity is

$O(l_c)$. Since l_c is proportional to n, the time complexity of selecting a representative trajectory is $O(n)$. In summary, the time complexity of TOPF is $O(n)+O(n)+O(n)$, which is $O(n)$

5. EXPERIMENT

To verify the accuracy of our method, we compared the TOPF with three methods: (1) ICBA described in [20]; (2) Prefix, proposed by [19]; and (3) NWA [7], which is a conventional algorithm whose result does not follow the road network constraints. To facilitate comparison with the other methods, we match the points in the result generated by the NWA to each road.

We use the generator by Brinkhoff [23] to generate datasets based on a map of Gutenburg in Germany. To ensure the accuracy of the experiment, we repeat each experiment five times. The experiment is evaluated based on three criteria:

(a) Average error rate as given by Equation (13). Suppose the number of roads is N and r_j represents road j. Let ori_{r_j} and ano_{r_j} denote r_j 's original frequency and frequency after the trajectories have been anonymized, respectively.

$$E = \frac{1}{N}\sum_{j=1}^{N} E_j = \frac{1}{N}\sum_{j=1}^{N} \frac{|ano_{r_j} - ori_{r_j}|}{ori_{r_j}} \tag{13}$$

(b) Standard deviation as given by Equation (14). The error function *E* is defined as the average difference between ano_{rj} and ori_{rj} in Equation (13) (i.e., E_j). A low standard deviation indicates that the anonymization quality of each road is similar and close to the average error rate.

$$\sigma = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(E_j - E)^2} \tag{14}$$

(c) Number of frequent patterns after anonymization. We use the widely adopted F-measure as defined by Equation (15), where P_r and P_a denote the sets of trajectories in the data mining results and anonymization results, respectively; N_m denotes the number of trajectories in the anonymization results that match those in the data mining results, and N_r and N_a denote the total number of trajectories in the data mining results and anonymization results, respectively. The equation is shown below:

$$F(P_r, P_a) = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (15)$$

$$Precision = \frac{N_m}{N_r}, \quad Recall = \frac{N_m}{N_a}$$

The algorithm is implemented in Java language, and all experiments were run on a PC with a 2.4 GHZ Intel Core i5-6200U processor, 8 GB of RAM, and the Windows 10 platform.

The parameters used are as follows:

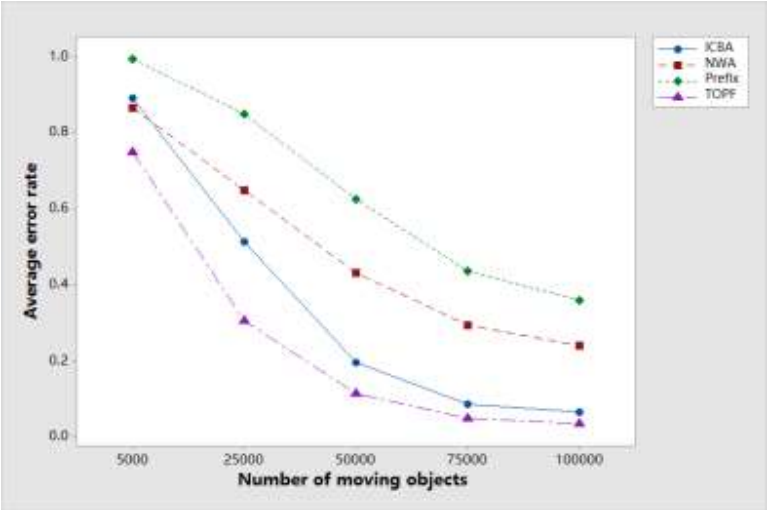
1) The anonymity threshold k is set to 2, 4, 6, 8 and 10.

2) The number of moving objects is set to 5K, 25K, 50K, 75K and 100K.

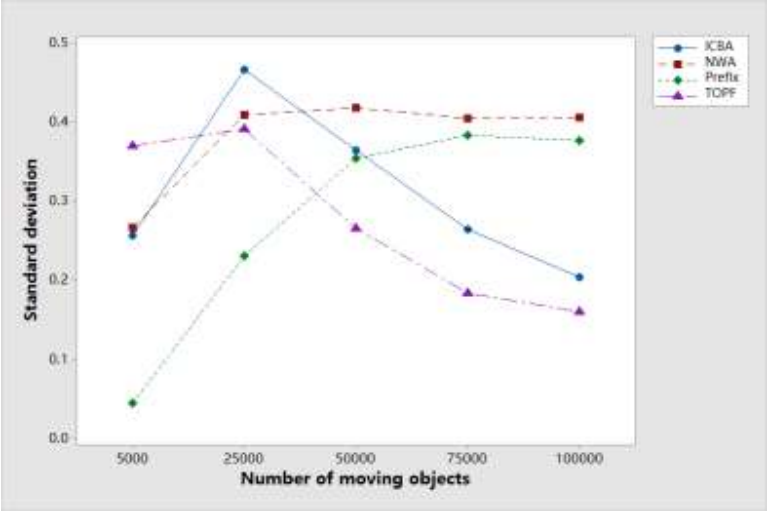
5.1 Effect of Data Sizes

We now compare the performance of our TOPF with ICBA, Prefix and NWA by varying the number of moving objects (*i.e.*, number of trajectories) from 5K to 100K. Fig. 4 (a) shows the average error rate of the overall anonymization results obtained from the four approaches. The x and y axes represent the number of objects and average error rate, respectively. It is unsurprising to see that TOPF yields less error than the other three algorithms in all cases. When the number of moving objects is small (*e.g.*, 5K), the anonymization results obtained from all algorithms have relatively high average error rates because the number of objects on each road is small, and even a small change to an object trajectory by the anonymization process will have a great impact on the average error rate. As the number of moving objects increases from 5K to 100K, the average error rate caused by TOPF continues decreasing and is far less than that of Prefix and NWA at all times. This is because Prefix and NWA do not group the similar trajectories effectively. ICBA is better than Prefix and NWA; however, the error rate of ICBA is still higher than that of TOPF. This is because TOPF protects frequent trajectories better, and the trajectories in a group are more similar.

Fig. 4 (b) shows the standard deviation of four algorithms. A low deviation indicates that the anonymization quality of each road is similar and close to the average error rare. TOPF generates much lower standard deviations in most cases. When the number of moving objects is small (i.e., 5K), the standard deviation generated by the other three algorithms is lower than that by TOPF. That is, the error rate of each road in the anonymization results generated by the other three algorithms is similar and very high because the average error rates are close to 90% when the number of objects is small. This is why the standard deviation generated by Prefix is lowest when the number of moving objects is small (i.e., 5K, 25K). With increasing data size, the standard deviation caused by TOPF decreases, which indicates not only that the anonymization result of TOPF has the lowest error rate but also that each road has similarly good quality.



(a) Average error rate



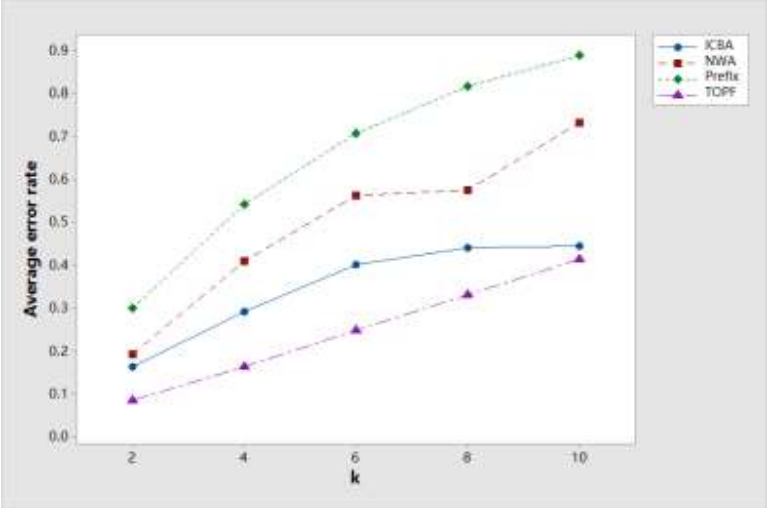
(b) Standard deviation

Fig. 4. Effect of data sizes

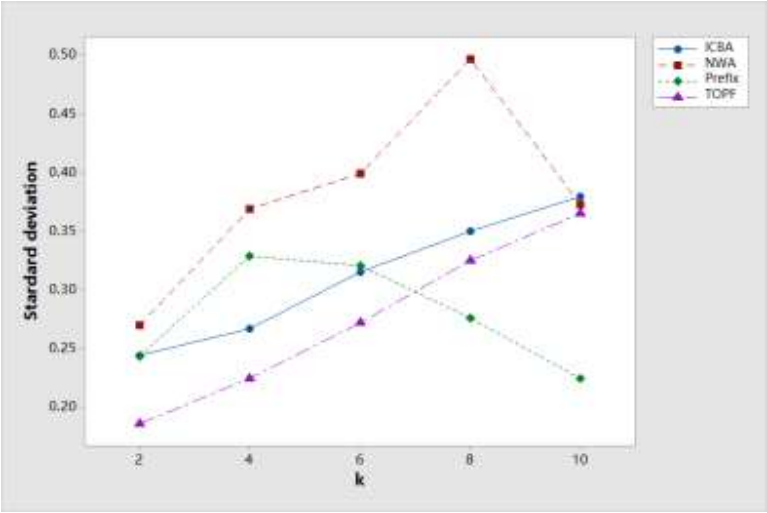
5.2 Effect of Parameter k

This set of experiments reflects the performance of four algorithms regarding different values of *k*. Fig. 5 (a) shows the average error rate of the anonymization results obtained from TOPF, ICBA, Prefix and NWA. The x and y axes represent the value of *k* and average error rate, respectively. We can observe that the average error rates of all approaches increase when *k* increases. The possible reason for this behaviour is that a larger *k* increases the number of deleted roads and dummy trajectories, which leads to an increase in the average error rate. It is obvious that TOPF has a much lower average error rate than that of the other three methods in all cases. This is because TOPF effectively groups similar trajectories and carefully selects representative trajectories, which minimizes the overall error rate.

We also measure the standard deviation of the anonymization results obtained from the four approaches. As shown in Fig. 5 (b), the anonymization result generated by TOPF has a much lower standard deviation than that by ICBA and NWA, which indicates that our anonymization result on each road has similarly good quality. It is interesting to note that Prefix has a lower standard deviation with large *k*. This is because Prefix must remove more infrequent trajectories for larger *k*, and its average error rate is close to 90%, where a low standard deviation indicates that each road has a very high error rate and is close to the average error rate.



(a) Average error rate



(b) Standard deviation

Fig. 5. Effect of parameter *k*

Moreover, we also apply the Wilcoxon signed-rank test and sign test to test all cases of TOPF, ICBA, Prefix and NWA. We vary the number of moving objects from 5K to 100K and vary the value of *k* from 2 to 10. The result is shown in Table 2, from which we can see that all P-values are less than 0.05; hence, our TOPF is much better than the other three methods.

Table 2 Wilcoxon signed-rank test and sign test

Comparison	Wilcoxon signed-rank test	Sign test
	P	P

TOPF vs ICBA	1.2290E-05	5.9605E-08
TOPF vs Prefix	1.2290E-05	5.9605E-08
TOPF vs NWA	4.5739E-05	1.5497E-06

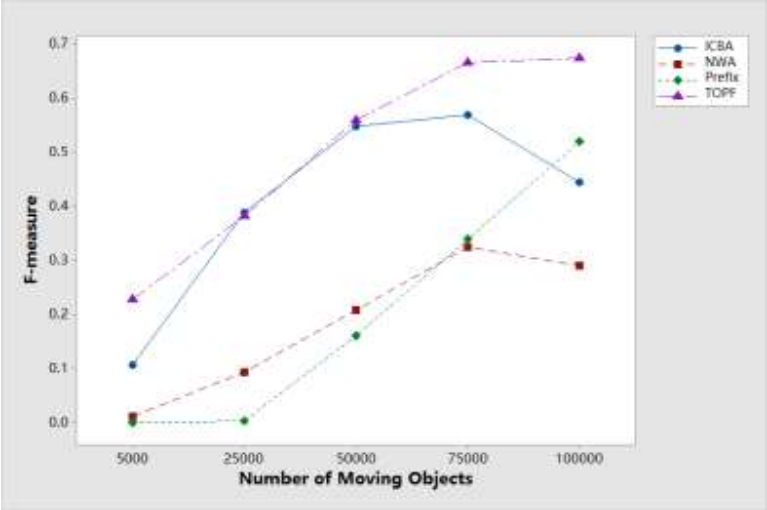
5.3 Preservation of Frequent Pattern

In general, the more frequent patterns are preserved, the better the anonymization result is. To measure this, we evaluate the quality of anonymization results by comparing the anonymized trajectories obtained from TOPF, ICBA, Prefix and NWA with frequent patterns discovered from original datasets using the PrefixSpan [24] data mining method. To make the experiment more obvious, we need a large *k* and set it as 10. When we use the PrefixSpan algorithm, each transaction corresponds to an original trajectory. Each item corresponds to a road ID in the trajectory. Moreover, the minimal support threshold corresponds to the anonymization parameter *k*. We use the widely adopted F-measure, which has been defined in Equation (7).

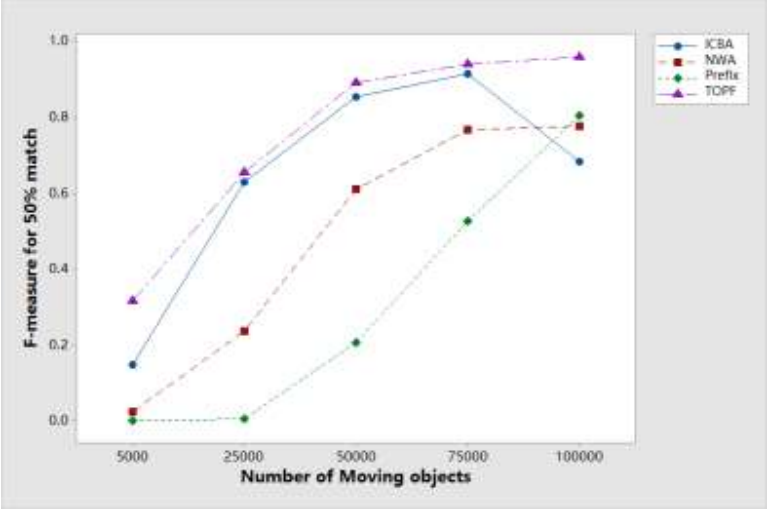
Fig. 6 (a) shows the F-measure values of the TOPF method and the other three methods; the x and y axes represent the number of moving objects and F-measure values, respectively. It is obvious that TOPF yields much higher F-measure values than Prefix and NWA in all cases, which indicates that TOPF preserves more frequent patterns. We can also observe that ICBA starts with a low F-measure value, then the F-measure values increase rapidly and are close to that of TOPF as the number of objects increases. However, as the number of objects continues to increase, the F-measure value of ICBA decreases rapidly and is much less than that of TOPF. This is because the ICBA method removes frequent trajectories when selecting a representative trajectory due to the difference of trajectories in a cluster, while our TOPF method selects the representative trajectory with high frequency and ensures the highest similarity within a group, which preserves the frequent pattern well.

Since we remove several trajectories and add dummy trajectories during the anonymization, it is unrealistic to expect to receive a perfect F-measure value. Therefore, we add the trajectories with at least 50% road segments matching a frequent pattern in the original data mining results to N_m for computing the F-measure. As shown in Fig. 6 (b), we can see that the F-measure values are several times higher than those in Fig. 6 (b). Moreover, the F-measure values of TOPF increase drastically with increasing number of moving objects, which indicates that our method preserves partial frequent pattern information very well.

We also apply the Wilcoxon signed-rank test and sign test to test all results obtained by the four approaches using F-measure and F-measure for 50% match. The result is shown in Table 3, from which we can see all P-values are less than 0.05; hence, our TOPF is much better than the other three methods.



(a) Exact Match



(b) Partial Match

Fig. 6. F-measure

Table 3 Wilcoxon signed-rank test and sign test

Comparison	Wilcoxon signed-rank test	Sign test
	P	P
TOPF vs ICBA	0.0039	0.0215
TOPF vs Prefix	0.0020	0.0020
TOPF vs NWA	0.0020	0.0020

6. CONCLUSION

It is becoming increasingly important to preserve individual privacy when publishing trajectory data. How to widely use the data without violating user location privacy concerns has attracted the attention of scholars.

Most previous works examining the preservation of privacy in trajectory data publishing suffer a serious loss in data usability. We address this problem by focusing on the high usability of data and present TOPF, a novel approach for preserving privacy in trajectory data publishing that

uses frequent path to strike a balance between the conflicting goals of data usability and data privacy.

TOPF not only can provide data privacy protection in data publishing but also can ensure the high usability of the data. This is because TOPF first removes all infrequent roads, which avoids identity linkage and guarantees that each road has k-anonymity. TOPF then adopts a new way to divide trajectories into candidate groups and proposes a new method for finding the most frequent path, and then, we select a representative trajectory to represent all trajectories within the group. We evaluate TOPF in terms of average error rate, standard deviation and F-measure values, and we conduct an experiment comparing TOPF with ICBA, Prefix and NWA. The result demonstrates that our approach is superior to the others.

In future work, the proposed method can be extended in the following two aspects: (1) how to update the parameters θ, δ in this article to obtain a better result; (2) how to optimize the grouping method to reduce the error rate and preserve a more frequent pattern. Moreover, how to make this algorithm more efficient is also worth considering.

ACKNOWLEDGMENTS

This work was partially supported by the National Natural Science Foundation of China (U1433116) and Foundation of Graduate Innovation Center in NUAA (kfjj20171603).

REFERENCES

[1] J. G. Lee, J. Han, and X. Li, 2008. Trajectory Outlier Detection: A Partition-and-Detect Framework. In *IEEE International Conference on Data Engineering(ICDE'08)* IEEE, Cancun, Mexico, 140-149.

[2] I. Webroot Software. (2010). *Webroot survey finds geolocation apps prevalent amongst mobile device users, but 55% concerned about loss of privacy*. Available: http://pr.webroot.com/threat-research/cons/social-networks-mobile-security-071310.html

[3] M. I. f. P. O. (Mipo). (2010). *Half of Social Networkers Online Concerned about Privacy*. Available: Http://maristpoll.marist.edu/714-half-of-social-networkers-online%concerned-about-privacy/

[4] H. Zheng, 2011. A Survey of Trajectory Privacy-Preserving Techniques. *Chinese Journal of Computers 34*, 10, 1820-1830.

[5] L. Sweeney, 2012. k-ANONYMITY: A MODEL FOR PROTECTING PRIVACY. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10*, 05, 557-570.

[6] M. Gruteser and D. Grunwald, 2003. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In *International Conference on Mobile Systems, Applications, and Services (MobiSys '03)*, San Francisco, California, 31-42.

[7] O. Abul, F. Bonchi, and M. Nanni, 2008. Never Walk Alone: Uncertainty for Anonymity in Moving Objects Databases. In *2008 IEEE 24th International Conference on Data Engineering(ICDE'08)*, Cancun, Mexico, 376-385.

[8] O. Abul, F. Bonchi, and M. Nanni, 2010. Anonymization of moving objects databases by clustering and perturbation. *Information Systems 35*, 08, 884-910.

[9] S. Arunkumar, M. Srivatsa, and M. Rajarajan, 2015. A review paper on preserving privacy in mobile environments. *Journal of Network and Computer Applications 53*, 74-90.

[10] M. E. Nergiz, M. Atzori, and Y. Saygin, 2008. Towards trajectory anonymization: a generalization-based approach. In *Proceedings of the SIGSPATIAL ACM GIS 2008 International Workshop on Security and Privacy in GIS and LBS (SPRINGL '08)*. ACM, Irvine, California, 52-61.

[11] Y. Wang, Y. Xia, J. Hou, S.-m. Gao, X. Nie, and Q. Wang, 2015. A fast privacy-preserving framework for continuous location-based queries in road networks. *Journal of Network and Computer Applications 53*, 57-73.

[12] R. Chen, B. C. M. Fung, N. Mohammed, B. C. Desai, and K. Wang, 2013. Privacy-preserving trajectory data publishing by local suppression. *Information Sciences 231*, 01, 83-97.

[13] S. Gao, J. Ma, C. Sun, and X. Li, 2014. Balancing trajectory privacy and data utility using a personalized anonymization model. *Journal of Network and Computer Applications 38*, 125-134.

[14] Z. Huo, X. Meng, H. Hu, and Y. Huang, 2012. You can walk alone: trajectory privacy-preserving through significant stays protection. In *Proceedings of the 17th international conference on Database Systems for Advanced Applications- Volume Part I (DASFAA'12)*. Springer-Verlag, Busan, South Korea, 351-366.

[15] Y. Wu, Q. Tang, W. Ni, Z. Sun, and S. Liao, 2013. A clustering hybrid based algorithm for privacy preserving trajectory data publishing. *Journal of Computer Research & Development 50*, 03, 578-593.

[16] F. Li, F. Gao, L. Yao, and Y. Pan, 2016. Privacy Preserving in the Publication of Large-Scale Trajectory Databases. In *Big Data Computing and Communications: Second International Conference.* Springer International Publishing, Shenyang, China, 367-376.

[17] S. Zhang, K.-K. R. Choo, Q. Liu, and G. Wang, 2017. Enhancing privacy through uniform grid and caching in location-based services. *Future Generation Computer Systems*. DOI= http://dx.doi.org/http://dx.doi.org/10.1016/j.future.2017.06.022. (Available online 24 June 2017)

[18] W. Ni, M. Gu, and X. Chen, 2016. Location privacy-preserving k nearest neighbor query under user's preference. *Knowledge-Based Systems 103*, 19-27.

[19] R. G. Pensa, A. Monreale, F. Pinelli, and D. Pedreschi, 2009. Pattern-Preserving k-Anonymization of Sequences and its Application to Mobil- ity Data Mining. In *International Workshop on Privacy in Location-Based Applications*, DBLP, Malaga, Spain.

[20] S. Gurung, D. Lin, W. Jiang, A. Hurson, and R. Zhang, 2014. Traffic Information Publication with Privacy Preservation. *Acm Transactions on Intelligent Systems & Technology 5*, 03, 1-26.

[21] Z. Chen, H. T. Shen, and X. Zhou, 2011. Discovering popular routes from trajectories. In *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering (ICDE '11)*. IEEE Computer Society, Hannover, Germany, 900-911.

[22] W. Luo, H. Tan, L. Chen, and L. M. Ni, 2013. Finding time period-based most frequent path in big trajectory data. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (SIGMOD '13)*. ACM, New York, USA, 713-724.

[23] T. Brinkhoff, 2002. A Framework for Generating Network-Based Moving Objects. *Geoinformatica 6*, 02, 153-180.

[24] L. Wang, K. Hu, T. Ku, and X. Yan, 2013. Mining frequent trajectory pattern based on vague space partition. *Knowledge-Based Systems 50*, 100-111.