A Trajectory Released Scheme for the Internet of Vehicles Based on Differential Privacy

Sujin Cai, Xin Lyu, Xin Li, *Member, IEEE*, Duohan Ban, and Tao Zeng

***Abstract*— The locations and users' information can be shared and interacted in the IoV (Internet of Vehicles), which provides sufficient data for traffic deployment and behavior pattern analysis. However, privacy issues had become more severe since personal or sensitive information is inclined to be revealed in a big data environment. In this work, a novel differential privacy-based algorithm named DPTD (Differentially Private Trajectory Database) is proposed for trajectory database releasing. Firstly, a 3-dimensional generalized trajectory dataset is established by considering the time factor. Then, the trajectory space is divided into several planes through the timestamps, and the set of the locations on each plane is further processed by clustering and generalizing to re-form new trajectories, that is, the trajectories to be released. This method is quite favorable to prefix-tree releasing because the spatiotemporal characteristics of the trajectories can be captured and spareness problem is fixed. Besides, a Markov assumption-based prediction method is suggested in order to reduce the cost of adding noise. Unlike the traditional method that the noise is added layer by layer, the noise is only added to the odd layers based on the prediction through spatio-temporal correlation, saving approximately 50% of the privacy budget. Theoretical analysis and experimental results show that the proposed algorithm has better data availability than the compared algorithms while guaranteeing the expected privacy level.**

***Index Terms*— Trajectory releasing, differential privacy, prefix tree, Markov based prediction.**

I. INTRODUCTION

WITH the increase of vehicles, it is challenging to guarantee road transport efficiency and traffic safety [1], [2]. Traditional transport scheduling systems acquire information monotonously, while the information in different systems is independent of each other, making it unable to meet the current complex traffic conditions. Then, the emergence of IoV (Internet of Vehicles) solves the problems mentioned above. The vehicles can easily exchange information with others and infrastructures, and the road traffic conditions in addition to the vehicles' driving information can be perceived in it [3]. Hence, there exists abundant location data in the IoV, which can be mined and analyzed to make more reasonable planning for road network construction and effectively alleviate urban traffic congestion, thus realizing intelligent traffic management [4], [5]. However, location data contains sensitive information, and a user can be easily discerned through a feature location in the trajectory. Moreover, the correlations between locations or the location distribution can also disclose the user's privacy. If the location data is released without sanitizing, it will raise serious privacy concerns. De Montjoye *et al.* [6] found that four spatio-temporal locations are enough to uniquely identify 95% of the individuals in a large dataset, even if some conventional sensitive information, such as name, gender, address, has been removed. Therefore, how to safely share and release location data remains challenging for target-specific applications in various fields.

Recently, researchers have carried out numerous studies on location protections and achieved substantial progress [7]–[10]. These methods can be summarized into four categories: 1) Encryption [11]–[13]. The users' queries are completely invisible to the server; thus, it provides a high level of privacy guarantee while suffering from expensive computation and limited expansibility. 2) Generalization [14], [15]. The actual location is generalized to a region to avoid the attackers detecting it. However, it leads to the loss of service quality, and privacy cannot be guaranteed in sparse areas. 3) Anonymity [16]–[20]. The k-anonymity makes the actual location is indistinguishable from other $k-1$ locations. However, the attacker can still infer the sensitive attribute range based on the equivalence class where the quasi-identifier is located. 4) Suppression [21], [22]. The released locations are selectively suppressed or reduced frequency according to the region's sensitivity. Whereas the techniques, such as generalization, anonymity, and suppression, all require to assume the attackers' background knowledge, which cannot be accurately identified in the big data environment. Since the data can be obtained in various ways, and the attackers can obtain additional information through data mining and fusion.

Manuscript received 2 June 2021; revised 28 August 2021 and 7 November 2021; accepted 17 November 2021. Date of publication 10 December 2021; date of current version 12 September 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61902110, in part by the National Key Research and Development Program of China under Grant 2018YFC0407105, and in part by the Technology Project of China Huaneng Group under Grant MW 2017/P28. The Associate Editor for this article was S. H. A. Shah. *(Corresponding author: Xin Lyu.)*

Sujin Cai is with the College of Computer and Information, Hohai University, Nanjing 211106, China, also with the Key Laboratory of Water Big Data Technology of Ministry of Water Resources, Hohai University, Nanjing 211106, China, and also with the School of Information Engineering, Yancheng Teachers University, Yancheng 224002, China (e-mail: sujin_cai@hhu.edu.cn).

Xin Lyu, Xin Li, Duohan Ban, and Tao Zeng are with the College of Computer and Information, Hohai University, Nanjing 211106, China, and also with the Key Laboratory of Water Big Data Technology of Ministry of Water Resources, Hohai University, Nanjing 211106, China (e-mail: lvxin@hhu.edu.cn; li-xin@hhu.edu.cn; dh_bjxy@163.com; tzeng.nj@hhu.edu.cn).

Digital Object Identifier 10.1109/TITS.2021.3130978

Differential privacy [23]–[25] is currently considered a de-facto standard for privacy protection, and its emergence can effectively address the problems mentioned above. It is supported by a robust mathematical theory and provides a rigorous definition of privacy protection, making the degree of privacy guarantee quantifiable and provable. Moreover, it is independent of the attackers' background knowledge and maximizes their ability, making whether a tuple is in the raw database has little impact on the query results. The mechanism enables to achieve privacy protection by adding random noise. However, it is challenging to apply differential privacy to the trajectory releasing when adding noise to ensure privacy. Because the trajectory data has multi-dimensional and spatio-temporal features; thus, we must add extensive noise to ensure the released trajectories satisfied differential privacy, resulting in a negative effect on data availability. Therefore, the differentially private trajectory releasing technologies focus on guarantee the trajectories' privacy while maintaining the spatio-temporal feature and keeping highly available.

In this study, we propose a novel releasing algorithm for trajectory datasets addressing the problems mentioned above. The specific contributions are summarized as follows:

1) In contrast to the most state-of-the-art methods, our proposed algorithm introduces the time dimension to released trajectories, then the 3D trajectory space is divided into several planes through the timestamps. Thus, it enables to reflect the location distribution at each timestamp and capture users' behavioral patterns. Furthermore, locations on each plane are clustered and then generalized to alleviate the sparseness caused by introducing the time dimension.
2) We propose a new method to release a cluster centroid without consuming any privacy budget. First, a quadtree is constructed for each location plane. Then, randomly select a location from the leaf node, where the actual centroid is located, as the releasing centroid.
3) According to the Markov assumption, a novel method is proposed to construct a noisy prefix tree for releasing the trajectory dataset. Unlike existing methods, the statistical value of nodes on even layers is predicted by the transition probability, saving nearly half of the privacy budget and significantly improving the data availability.
4) Extensive experiments are conducted on real-life data sets, and the results demonstrate that our algorithm outperforms the competitors concerning data usability.

The rest of our paper is organized as follows. Second II introduces the related work on trajectory privacy protection; Section III presents the preliminary knowledge; Section IV describes the proposed trajectory releasing algorithm in detail; Section V offers the experimental data set, experimental evaluation, and analyzes the experimental results; Section VI summarizes the paper and looks forward to future research work.

II. RELATED WORK

Currently, the existing trajectory releasing mechanisms can be roughly classified into two types. The first type aims to release only one trajectory [26], [27], and each location in the trajectory is regarded as one record. The algorithms usually predict the following location based on the current location's direction and velocity, then synthesize a trajectory. However, there are some limitations in implementation due to the accuracy of predictions cannot be guaranteed. The second type aims to release a trajectory dataset [28]–[30], which contains multiple trajectories, and each trajectory in the dataset is considered one record. In contrast to the first type, the second type has broader applications owing to the significance of mining trajectories. We can capture individuals' behavior patterns or traffic conditions by mining and analyzing trajectory data. This paper studies the privacy of trajectory datasets that our proposed algorithm belongs to the second type.

There have extensive works on the privacy protection of trajectory dataset releasing, and we discuss relevant works below.

A. Traditional Trajectory Protection

Nergiz *et al.* [31] first release trajectory datasets using k-anonymity so that each trajectory in the "whole" is indistinguishable from other $k-1$ trajectories. Hu *et al.* [32] protect the trajectory dataset based on k-anonymity, which contains sensitive items. They utilize local amplification to amend the trajectory dataset so that each sensitive item exists in at least k users. Terrovitis *et al.* [33] assume that the attacker owns some actual trajectory segments. If the user enters a sensitive area, his locations are suppressed, and the leakage risk of trajectory is kept within a privacy threshold. Nevertheless, the data is processed using global suppression; thus, the trajectory which contains problematic items will not be released. Due to the excessive suppression, the released data is lack of availability. Chen *et al.* [34] propose a local suppression method. The fake location replaces the actual one to release, which is generated by some transformations from the actual one. Furthermore, this method allows for customized privacy.

However, recent works on re-identification and de-anonymization attacks [35], [36] represent those traditional methods are insufficient to trajectory privacy protection, such as anonymization, suppression, and generalization. Moreover, the uniqueness of trajectory only decreases slightly even though spatial generalizing.

Unlike the traditional methods mentioned above, differential privacy is independent of the attackers' background knowledge and provides a quantifiable model of robust privacy protection.

B. Differential Privacy Trajectory Protection

Several recent studies consider the ways of releasing trajectories with differential privacy. Chen *et al.* [28] first release trajectory dataset with differential privacy and propose a method called STM-Full by constructing a prefix tree. However, the node count decreases drastically with the growth of the prefix tree, resulting in leaf node sparseness. Thus, the count cannot withstand the injected noise, making the released trajectory data low availability. Due to the drawbacks of the STM-Full, Chen *et al.* [29] extend the work [28] in the subsequent study and propose a variable n-gram model, which constructs an exploration tree based on the Markov assumption, preserving the spatial correlation of trajectory locations while reducing the added noise. Both methods mentioned above [28], [29] implicitly assume that the trajectories in the dataset share amounts of identical prefixes or n-grams. However, locations are represented by latitude and longitude in real-life datasets and almost non-existent identical prefixes. Therefore, Hua *et al.* [30] remove the above assumption and propose the first differentially private generalization algorithm for releasing trajectories, which cluster trajectories close to each other using the exponential mechanism. It does not need to explicitly consider every possible entry over the output universe, effectively reducing the output scope. However, the variety of location distribution is ignored, and locations at each timestamp are clustered into the same number of groups. He *et al.* [37] discrete the trajectories using hierarchical reference systems called DPT, which are based on the speed of individuals. Then private prefix trees are constructed by these systems for releasing trajectories. The algorithm significantly reduces the number of nodes in the prefix tree and speeds up the indexing. However, Gursoy *et al.* [38] point out that DPT changes the data distribution, making sparse regions become dense. Meanwhile, its output trajectories contain diamond-shaped movement and dead-ends, which are seldom observed in actual trajectories. Khalil *et al.* [39] improve the traditional prefix tree by dividing each layer into two sub-layers which are classified by locations and timestamps; meanwhile, the branches are pruned whose count is less than the threshold. This method ensures that the nodes count on each layer can withstand the added amount of noise, thereby increasing the data availability. Zhao *et al.* [40] devise three attack models for trajectory clustering analysis, then provide corresponding resistance methods and noise design. Concretely, the standard Laplacian noise is added to the count of locations within the cluster to resist the continuous query attack. Then, the planar Laplacian noise with a limited radius is added to the location within the cluster to avoid too much noise affecting the clustering effect. Cao *et al.* [41] propose a continuous location releasing method. The privacy leakage of the current location is measured by backward and forward transition probability under the Markov model. Then, the privacy budget partition is transformed to the optimal solution of linear programming, which makes the partition more reasonable. Unfortunately, the computation is expensive, and it is hard to protect location privacy in real-time. Ghane *et al.* [42] present a trajectory generative mechanism (TGM), which is the first mechanism that captures the stay locations in trajectories. It encodes the trajectories as a graphical generation and generates the trajectories of arbitrary length. Inherently, this mechanism boosts computation efficiency. Gursory *et al.* [38] propose a differentially private framework DP-Star, which preserves the correlations between the end points of trajectories through a private trip distribution, and intermediate points are predicted through the Markov model. Besides, DP-Star utilizes feature locations and statistics learned from the raw data to generate and release synthetic trajectories.

III. PRELIMINARIES

In this section, we introduce differential privacy, followed by an introduction to the trajectory data model and l-order Markov assumption; lastly, we present the prefix tree for releasing trajectories.

A. Differential Privacy

Definition 1 (ε-Differential Privacy): Any neighboring datasets D and D' that have the same data structure and only have one record difference between them, that is, $\left|D' \Delta D\right| \leq 1$. Given a randomized algorithm A, we define the algorithm A satisfies ε-differential privacy if for two neighboring datasets D and D', and all the possible outputs $O\,(O \in Range\,(A))$, $Range\,(A)$ represents the output range of A, we have:

$$\Pr\left[A\left(D\right) = O\right] \leq e^{\varepsilon} \times \Pr\left[A\left(D'\right) = O\right] \tag{1}$$

where the $\Pr\left[\cdot\right]$ denotes the probability of a user's privacy leakage. The parameter ε is the private budget that controls the degree of privacy protection. A smaller ε corresponds to stronger privacy protection, and vice versa.

Definition 2 (Global Sensitivity): For a query function $f : D \rightarrow R^d$ and any neighboring datasets D and D', the global sensitivity of the function f is:

$$\Delta f = \max_{D,\,D'} \left\| f\left(D\right) - f\left(D'\right)\right\|_p \tag{2}$$

where R is the real number field mapped by dataset D, d denotes the query dimension of function f, and p is used to measure the norm distances of Δf, and generally, $p = 1$.

Definition 3 (Laplace Mechanism): Let $f : D \rightarrow R^d$ denote a query function over a dataset D, then a random algorithm A satisfies ε-differential private if its output is

$$A\left(D\right) = f\left(D\right) + Lap\left(\Delta f/\varepsilon\right) \tag{3}$$

where Δf is the global sensitivity of the function, $Lap\left(\Delta f/\varepsilon\right)$ is a random variable sampled from the Laplace distribution, and the density function of the Laplace distribution is as follows:

$$p(x) = \frac{\varepsilon}{2\Delta f} e^{-|x|\varepsilon/\Delta f} \tag{4}$$

The Laplace distribution has a mean of 0 and $2\left(\Delta f/\varepsilon\right)^2$ variance. The amount of noise is proportional to the Δf and inversely proportional to the private budget ε; that is, if Δf is fixed, the smaller the ε, the more the noise injected and the higher the degree of privacy, and vice versa.

Differential privacy has two essential properties: sequential composition and parallel composition. Sequential composition prescribes that if a sequence of computations is performed on the same data, each part provides differential privacy independently, then the privacy guarantee of the entire sequence is accumulated. Parallel composition stipulates that if a sequence of computations is carried out on disjoint subsets of data, the entire sequence provides the worst privacy guarantee. The two theorems are formally described as follows:

Theorem 1: Sequential Composition. Let $A_i\,(1 \leq i \leq n)$ be a set of random algorithms, each providing

$\varepsilon_i\ (1 \le i \le n)$- differential privacy. Then, the sequence of all algorithms satisfies $\sum_{i=1}^{n} \varepsilon_i$-differential privacy.

Theorem 2: Parallel Composition. If $D_i\ (1 \le i \le n)$ are the disjointed subsets of the original dataset D and $A_i\ (1 \le i \le n)$ is a set of random algorithms, each provides $\varepsilon_i\ (1 \le i \le n)$-differential privacy for each D_i. Then the sequence of these algorithms satisfies Max (ε_i)-differential privacy.

B. Trajectory Data Model

Definition 4 (trajectory): Trajectory T is a sequence of time-stamped locations, formally represented as:

$$(t_1, L_1) \to (t_2, L_2) \to \ldots \to (t_{|T-1|}, L_{|T-1|}) \to (t_{|T|}, L_{|T|}) \tag{5}$$

where $|T|$ is the length of trajectory T. The position node C_i in trajectory T is a time-location pair: (t_i, L_i), and L_i is represented by the latitude-longitude coordinate. The location at time t_i in T is denoted by $T(t_i) = L_i, \forall i (1 \le i \le |T|)$, and the set of timestamps in T are denoted by $Time(T) = \{t_1, t_2, \cdots, t_{|T|}\}$.

Definition 5 (Trajectory Database): The trajectory database D of size $|D|$ contains multiple trajectories: $D = \{T_1, T_2, \cdots, T_{|D|}\}$, where $T_m\ (1 \le m \le |D|)$ denotes a trajectory in the database D. Specifically, the position node in the trajectory T_m at time t_i can be represented by $C_i^m = (t_i, L_i^m)$, where the $L_i^m \in \Gamma_i$ is a location of T_m at time t_i, and Γ_i is the location universe at time t_i, namely, $\Gamma_i = \{L_i^k \,|k = 1, 2, \cdots, |T|\}$.

For convenience, we assume that all the trajectories in the database D with the same length, namely,

$$Time(T_i) = Time(T_j) \quad \forall 1 \le i, j \le |D|,\ i \ne j \tag{6}$$

where $Time(T_i)$ represents the set of timestamps of object i's trajectory.

C. l-Order Markov Assumption

The l-order Markov assumption is a classic probabilistic prediction model to deal with the sequential data. It helps to estimate the following location based on the k previous items.

Definition 6 (l-Order Markov Assumption): A time-series sequence $(L_1 \to L_2 \to \ldots \to L_n) \in \Sigma^n$ is followed a l-order Markov process if for every $l \le i \le n$, $L \in \Sigma$,

$$\begin{aligned} \Pr(L_{i+1} &= L \,|L_1 \to \ldots \to L_i) \\ &:\approx \Pr(L_{i+1} = L \,|L_{i-l+1} \to \ldots \to L_i) \end{aligned} \tag{7}$$

We refer to the probability $\Pr(L_{i+1} = L \,|L_{i-l+1} \to \ldots \to L_i)$ is a transition probability of the Markov process, and the probability can be estimated through:

$$\begin{aligned} \Pr(L_{i+1} &= L \,|L_{i-l+1} \to \ldots \to L_i) \\ &= \frac{C\,(L_{i-l+1} \to \ldots \to L_i \to L)}{C\,(L_{i-l+1} \to \ldots \to L_i)} \end{aligned} \tag{8}$$

where $C\,(\cdot)$ denotes the count of the sequence.

D. Prefix Tree

A trajectory database can be represented by constructing a prefix tree. The prefix tree groups sequences with the same prefix into the same branch. We formally define the prefix tree as follow:

Definition 7 (Prefix Tree): A prefix tree $PT = (V,\ E,\ Root)$ constructed by trajectory database D is a triplet, where V is a collection of nodes labeled with locations. E is the set of edges connecting nodes of the PT. $Root$ is a virtual node which presents the location universe of the database D. The unique prefix of a node $\upsilon \in V$ represents by the path of $Root$ to υ, and denotes by $prefix\,(\upsilon,\ PT)$.

Each node $\upsilon \in V$ is denoted by the form of $\langle tr\,(\upsilon)\,, c\,(\upsilon)\rangle$, and $c\,(\upsilon) = |tr\,(\upsilon)| + Lap\,(\Delta f/\varepsilon)$ represents the noisy count of $|tr\,(\upsilon)|$.

IV. TRAJECTORY RELEASING ALGORITHM

A. The Basic Idea

Generally speaking, this study aims to design an ε-differentially private trajectory releasing algorithm, synthesizing a database S_D corresponds to raw database D. Consequently, S_D can adapt to multiple tasks with high accuracy and availability.

The noisy prefix tree is always used to synthesize and release trajectory datasets since it merges trajectories with the same prefixes into an identical branch, capturing the time-series characteristics. However, most algorithms only focus on the spatial features of trajectories and ignore the temporal characteristics. Actually, it is necessary to introduce the time dimension since the locations in the trajectories are spatio-temporal correlated; meanwhile, it is easy to acquire location distribution at different times and probe into individual behavior patterns. Unfortunately, the time dimension introduction will make the data sparser, which is hard to withstand the injected noise, decreasing the worth of the released data. To address this problem, we divide the 3D trajectory space into several planes through the timestamps. Then, locations on each plane are clustered followed by generalized to alleviate the sparseness caused by introducing the time dimension. Furthermore, we propose a novel method to allocate the privacy budget using the Markov assumption, which reduces injected noise and saves nearly half of the budget consumption.

The main steps are summarized as follows:

1) The location universe Γ_i at each timestamp t_i is clustered using DBSCAN (Density-Based Spatial Clustering of Applications with Noise), which can capture the location distribution. Then, the generalization is performed to reduce the data sparse; namely, all the locations in each cluster are replaced by the cluster centroid. Moreover, it solves the problem that few trajectories with the identical prefix when locations are represented by latitude and longitude in real life. However, releasing the actual centroids will leak the users' privacy. This study proposes a novel approach to release the cluster centroids privately. First, each location universe Γ_i is partitioned with a quadtree. Secondly, randomly select a location from the leaf node where the actual centroid is located, finally releasing the location as the centroid.

Algorithm 1 DPTD

```
Input: Raw Trajectory Dataset $D$, timeStampCnt, minCluster,
       maxDistance, $\varepsilon$;
Output: Differential-Private Trajectory Dataset $\tilde{D}$;
1.   Preprocess the Raw Trajectory Dataset $D$;
2.   for (int i = 0; i < timeStampCnt; i++) {
3.     clusterResult = DBSCANClusterer
       (i, minCluster, maxDistance);
4.     QuadTree partition;
5.   }
6.     for (int j = 0; j< clusterResult; j++) {
7.       centrCluster = centroid (j);
8.         for (int k=0; k<centrCluster; k++) {
9.           Search QuadTree node (k);
10.          centrGen = Randomly select a location from the node;
11.        }
12.        Generalize (clusterResult, centrGen);
13.  }
14.  Trie=Construce PrefixTree (Root, timestamp, numberCnt);
15.  $\tilde{D}$ = Constrain Reference Trie;
16.  Return $\tilde{D}$;
```

2) A generalized dataset is synthesized according to the results of step 1 mentioned above; then, the Markov transition probability from each cluster at time t_{i-1} to each cluster at time t_i are calculated.

3) Traditionally, noisy prefix-tree is constructed by adding Laplace noise to each layer for the guarantee of differential privacy, in other words, each layer is allocated ε/h budget, where h is the height of the prefix-tree. Nevertheless, If the tree is tall, the privacy budget would be over divided, making the noise excessive. This study presents a novel method to construct a noisy prefix-tree using the Markov assumption. Unlike traditional methods, we only add noise to the nodes in odd layers with the privacy budget of $\varepsilon/\lceil h/2 \rceil$. In contrast, the node count of even layers is predicted using the Markov transition probability. This method can save nearly half of the privacy budget and dramatically decrease the amount of noise.

4) The data availability is improved by adopting the constraint conditions and post-processing.

The overview of our proposed DPTD is presented in Algorithm 1. The locations are clustered using DBSCAN and construct a quadtree at each timestamp (Lines 2-4). The actual centroid of each cluster is saved (Lines 6-7). Randomly select a location from the leaf node where the actual centroid is located as the releasing centroid. (Lines 8-10). Generalize the cluster using the releasing centroid (Line 12). Construct a noisy prefix-tree (line 14). Finally, constraint inference is applied to the noisy prefix-tree (Line 15).

B. Generalization

The differentially private generalization algorithm was first proposed by Hua [30] for trajectory releasing, which merges locations according to the distance between trajectories at each timestamp. Trajectories that are close together will be clustered into a group using the exponential mechanism. However, they ignore the variety of location distribution at different times. By contrast, DPTD cluster the location universe Γ_i at each timestamp t_i using DBSCAN, without predetermining the number of clusters and forming clusters with arbitrary shapes and numbers based on the data distribution. Then, the cluster centroid replaces other locations within the cluster for generalization. This method can reduce the data sparsity caused by introducing the time dimension and remove the assumptions that the trajectories contain many identical prefixes. However, unlike K-MEANS, there are multiple core points within each cluster by DBSCAN; thus, it is necessary to select an optimal core point as the centroid for generalization. We are motivated by the idea of K-MEANS to find the optimal core point of each cluster. DPTD calculates the sum of Euclidean distances from all locations to each core point within a cluster and selects the core point that has the shortest distance as the centroid.

Procedure 1 Generalized-Cluster

```
Input: centroidList, quadtreeArrayList, clusterResult
Output: setGenLoc
1.  for (int j = 0; j < centroidList.size(); j++) {
2.    cent = centroidList.get(j);
3.    QuadTree = quadTreeArrayList.get(cent);
4.    centroGenSet = quadTree.retrieve(cent);
5.    centrGen = Random select (Quadnode)
6.    GenList.add (centrGen);
7.  }
8.  setGenLoc = replaceLocation(clusterResult, gen_centroidList);
9.  Return setGenclus;
```

Nevertheless, it violates privacy if the centroids are released without protection. Therefore, existing methods always allocate parts of the privacy budget to protect the centroid for guaranteeing differential privacy. For example, suppose the trajectory length is 10, and the privacy budget ε is 0.1. The privacy budget allocated to each timestamp is $\varepsilon/10 = 0.01$ if ε is evenly allocated to each timestamp. Besides, this part of the budget still needs to be divided into two parts, one half is used to protect the centroids, and the other is used to construct the noisy prefix tree. As a result, the budget assigned to protect centroids at each timestamp is $\varepsilon/20 = 0.005$. According to Definition 3, it is clear that the amount of noise is inversely proportional to the privacy budget. Excessive noise will be added if the budget is 0.005 and affect the availability of releasing data.

We present a new method to release cluster centroids without allocating additional privacy budget to protect them. A quadtree partition the location universe Γ_i at each timestamp. Then, we randomly select a location from the leaf node where the actual centroid is located as the releasing centroid. Finally, the other locations are replaced for generalization within the cluster. The other clusters in Γ_i are released in the same way.

Due to the releasing centroids being randomly selected from the leaf node; thus, the privacy budget allocated to protect the actual centroid is saved.

Procedure1 describes how to generalize the locations within a cluster. Input the cluster centroids list and the quadtree. Find the quadtree where the centroid of the current cluster is located (Lines 1-3). Retrieve the leaf node of the centroid is located (Line 4). Randomly select a location from the leaf node as the releasing centroid (Lines 5-6). Replace

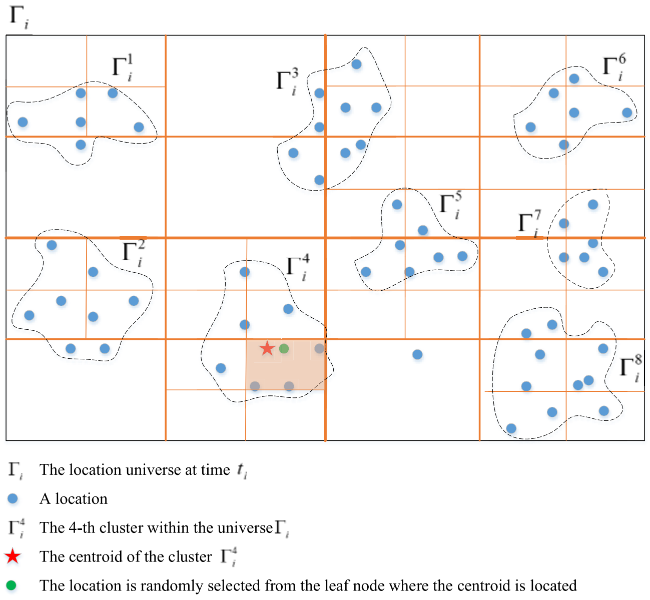

Γ_i
Γ_i^1
Γ_i^2
Γ_i^3
Γ_i^4
Γ_i^5
Γ_i^6
Γ_i^7
Γ_i^8
Γ_i The location universe at time t_i
● A location
Γ_i^4 The 4-th cluster within the universe Γ_i
★ The centroid of the cluster Γ_i^4
● The location is randomly selected from the leaf node where the centroid is located

Fig. 1. Γ_i is the location universe at time t_i. All the locations within the location universe Γ_i are clustered into 8 groups by DBSCAN, denoted by $\Gamma_i^k (k = 1, 2, \ldots, 8)$, respectively. The locations, which belong to the same group, are delineated by the dashed gray line. The solid orange lines are the dividing lines of the quadtree, and the location universe Γ_i is the root of the quadtree. The region will be partitioned into four equal sub-regions if the number of locations within it exceeds the threshold θ, and partition iteratively until the number of locations in all sub-regions is less than θ.

other locations with the releasing centroid for generalization (Line 8).

Example: Figure 1 shows a generalization example. The blue dots are the locations within the location universe Γ_i (i.e., all the locations at the time t_i). Firstly, all the locations within the universe Γ_i are clustered into eight groups by DBSCAN, denoted by $\Gamma_i^k (k = 1, 2, \ldots, 8)$, respectively. Then, we search for the centroid of each cluster. For example, the red pentacle in Fig. 1 is the actual centroid of the cluster Γ_i^4. Thirdly, the location universe Γ_i is partitioned using a quadtree, and we index to the leaf nodes where the actual centroids located, such as the translucent orange area in Fig 1. The green dot is the releasing centroid which is randomly selected from the leaf node, and subsequently the locations in the cluster Γ_i^4 are all replaced by it. Finally, other clusters within the location universe Γ_i are generalized in the same way.

The locations at each timestamp are processed by clustering and generalizing, and then the trajectory dataset is re-formed. The generalized dataset is synthesized, as shown in Table I.

The Γ_i denotes the location universe at time t_i, and the Γ_i^k denotes the k-th cluster within the Γ_i. As shown in Table I, through the timestamps t_i, the individuals' location-to-location movements are generalized to cluster-to-cluster movements.

C. Markov Probability Prediction Model

The work [43] shows that low-order Markov models can capture the spatio-temporal correlation as accurately as other complex methods; by contrast, high-order models are computationally expensive. Thus, we utilize the first-order Markov model for prediction in this study. According to the generalized trajectory dataset (TABLE I), the transition probabilities between clusters can be calculated, as shown in TABLE II.

TABLE I

GENERALIZED TRAJECTORY DATA SET

ID	$\Gamma_1 \to \Gamma_2 \to \Gamma_3 \to \Gamma_4$
1	$\Gamma_1^1 \to \Gamma_2^2 \to \Gamma_3^2 \to \Gamma_4^3$
2	$\Gamma_1^2 \to \Gamma_2^1 \to \Gamma_3^3 \to \Gamma_4^2$
3	$\Gamma_1^2 \to \Gamma_2^2 \to \Gamma_3^1 \to \Gamma_4^1$
4	$\Gamma_1^1 \to \Gamma_2^1 \to \Gamma_3^2 \to \Gamma_4^2$
5	$\Gamma_1^3 \to \Gamma_2^2 \to \Gamma_3^4 \to \Gamma_4^3$
6	$\Gamma_1^2 \to \Gamma_2^1 \to \Gamma_3^3 \to \Gamma_4^3$
7	$\Gamma_1^3 \to \Gamma_2^2 \to \Gamma_3^2 \to \Gamma_4^2$
8	$\Gamma_1^1 \to \Gamma_2^1 \to \Gamma_3^1 \to \Gamma_4^1$
9	$\Gamma_1^2 \to \Gamma_2^2 \to \Gamma_3^4 \to \Gamma_4^1$

The Γ_i denotes the location universe at time t_i, and the Γ_i^k denotes the k-th cluster within the Γ_i. As shown in Table I, through the timestamps t_i, the individuals' location-to-location movements are generalized to cluster-to-cluster movements.

TABLE II

THE TRANSITION PROBABILITIES BETWEEN CLUSTERS

$P(\Gamma_1^i \to \Gamma_2^j)$, $(i = 1,2,3 \quad j = 1,2)$		
$P(\Gamma_2^1 \mid \Gamma_1^1) = 2/3$	$P(\Gamma_2^1 \mid \Gamma_1^2) = 1/2$	$P(\Gamma_2^1 \mid \Gamma_1^3) = 0$
$P(\Gamma_2^2 \mid \Gamma_1^1) = 1/3$	$P(\Gamma_2^2 \mid \Gamma_1^2) = 1/2$	$P(\Gamma_2^2 \mid \Gamma_1^3) = 1$

$P(\Gamma_3^h \to \Gamma_4^k)$, $(h = 1,2,3,4 \quad k = 1,2,3)$			
$P(\Gamma_4^1 \mid \Gamma_3^1) = 1$	$P(\Gamma_4^1 \mid \Gamma_3^2) = 0$	$P(\Gamma_4^1 \mid \Gamma_3^3) = 0$	$P(\Gamma_4^1 \mid \Gamma_3^4) = 1/2$
$P(\Gamma_4^2 \mid \Gamma_3^1) = 0$	$P(\Gamma_4^2 \mid \Gamma_3^2) = 2/3$	$P(\Gamma_4^2 \mid \Gamma_3^3) = 1/2$	$P(\Gamma_4^2 \mid \Gamma_3^4) = 0$
$P(\Gamma_4^3 \mid \Gamma_3^1) = 0$	$P(\Gamma_4^3 \mid \Gamma_3^2) = 1/3$	$P(\Gamma_4^3 \mid \Gamma_3^3) = 1/2$	$P(\Gamma_4^3 \mid \Gamma_3^4) = 1/2$

For example, when $t_i = 1$, the locations within universe Γ_1 are grouped into three clusters, i.e., $\Gamma_1^1, \Gamma_1^2, \Gamma_1^3$. It can be seen that, there are three generalized trajectories (i.e., clusters) start from Γ_1^1; then, the three clusters move apart when $t_i = 2$, two clusters get to Γ_2^1, and one get to cluster. Thus, the Markov transition probability can be calculated as: $P(\Gamma_2^1 | \Gamma_1^1) = 2/3$, $P(\Gamma_2^2 | \Gamma_1^1) = 1/3$. As shown in TABLE II, we only calculate the transition probabilities of $\Gamma_1^i \to \Gamma_2^j$ and $\Gamma_3^h \to \Gamma_4^k$, and the node count of even layers in the prefix tree can be predicted by

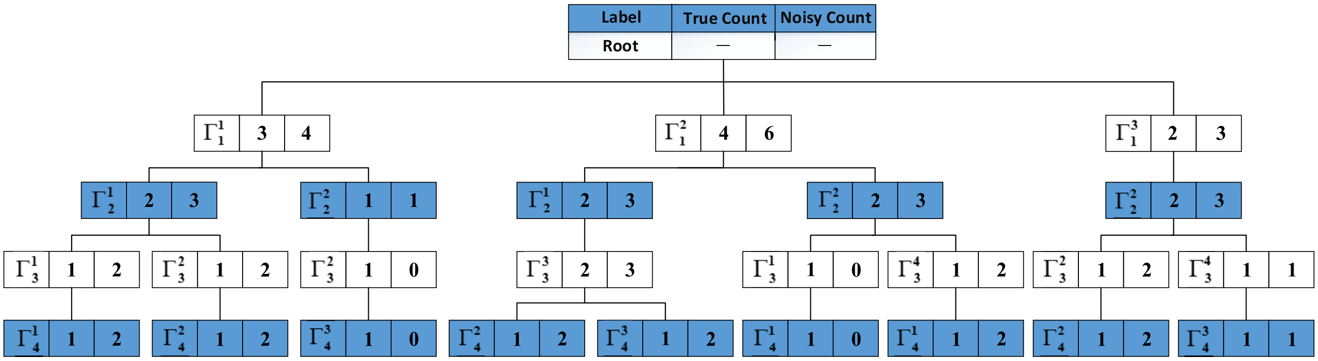


Fig. 2. Taking the data in Table 1 as an example, the real count of the cluster Γ_1^1 on the first layer is 3, and we add the Laplace noise with the privacy budget of $\varepsilon/\lceil h/2\rceil = \varepsilon/2$ to it, then its noise count is 4. The transition probabilities of moving from Γ_1^1 to Γ_2^1 and Γ_2^2 are $P(\Gamma_2^1 | \Gamma_1^1) = 2/3$ and $P(\Gamma_2^2 | \Gamma_1^1) = 1/3$, respectively. Then we calculate the noisy counts of the cluster Γ_2^1 and the cluster Γ_2^2 are $\Gamma_2^1 = 4 \times 2/3 \approx 3$ and $\Gamma_2^2 = 4 \times 1/3 \approx 1$. The noisy counts of other nodes on the second layer can be calculated in the same way. It should be noted that we calculate the Markov transition probability based on TABLE II, and we only need to calculate the transition probabilities of $\Gamma_i \rightarrow \Gamma_j$ $(i \mod 2 = 1,\ j = i + 1)$.

transition probabilities. Details will be stated and analyzed in Section IV-D. Then we construct a noisy prefix tree to release the trajectory dataset. Nearly half of the privacy budget can be saved by using the Markov probability prediction model.

D. Construct a Noisy Prefix Tree

The prefix tree is suitable for capturing the spatio-temporal features of trajectory data, and it groups trajectories with the same prefix into the same branch [28], [29]. However, most location data in real life is represented in latitude and longitude coordinates, and it is difficult to have many identical prefixes in the trajectory dataset, forming lots of vacant nodes. Meanwhile, sparse data is hard to withstand the added Laplace noise, resulting in significant perturbation errors. Furthermore, besides alleviating node data sparsity, we also should improve the privacy budget utilization to reduce the number of added noises. In this study, the location universe is clustered and generalized at each timestamp, dramatically alleviate data sparsity. Moreover, we also need to improve the privacy budget utilization. The traditional method is evenly allocated the privacy budget to each layer of the prefix tree. Thus, the privacy budget allocated to each layer is ε/h, where h is the height of the prefix tree. Visibly, if the trajectories are long, namely, the value of h is relatively large, the privacy budget ε is still over-partitioned, resulting in excessive added noise.

This paper proposes a new method for constructing a noisy prefix tree based on the Markov probabilistic prediction model. When the height of the layer is $l \bmod 2 = 1$, i.e., odd layer, the Laplace noise with the privacy budget of $\varepsilon/\lceil h/2\rceil$ is added to the nodes of layer l, where h is the height of the noisy prefix tree. Then, the count of nodes in layer $l+1$ is predicted using the Markov assumption. In other words, a node in layer $l + 1$ is obtained by its father node's noisy count multiplies the Markov transition probability of transfer from the father node to it. So, we only need to assign the privacy budget to odd layers, saving nearly half of the privacy budget compared to traditional methods that assign the budget evenly to each layer.

Procedure 2 Noisy Prefix Tree

```
Input: Privacy budget epsilon, Prefix Tree
Output: Noisy Prefix Tree
1.  map=PrefixTreenode.next;
2.  while (map.values()) {
3.    if (node.isleaf) {
4.    return;
5.    }
6.    if (node.deep % 2 = 1) {
7.    node.noisyCount = node.count+Lap(1/$\varepsilon$);
8.    Add (NoisyPrefixTree, node, noisyCount);
9.    childMap = node.next; }
10.   while (childMap.values())
11.     childnode.noisyCount = childnode.count/
        node.count $*$ node.noisyCount;
12.     Add (NoisyPrefixTree, childnode, noisyCount);
13.     end while
14.   end while
15.   Return NoisyPrefixTree;
```

Fig.2 depicts the example of constructing a noisy prefix tree based on TABLE I, and the process of constructing a noisy prefix tree is described in Procedure 2.

Procedure 2 inputs the privacy budget ε and the constructed prefix tree, then output a noisy prefix tree. If the node's height is odd, the allocated Laplace noise is added, then the node and its noisy count are inserted into the prefix tree (Lines 6-8). The count of its children is obtained by its noisy count multiplies the Markov transition probability, then the child nodes and their noisy count are inserted into the noisy prefix tree (Lines 9-12).

In this study, the Markov transition probability is calculated using the generalized dataset, but this will not reveal the users' privacy, based on the assumption that an attacker cannot accurately get the transition probability among all the clusters in the generalized dataset. The reason is that unless the attacker already has obtained the entire raw dataset, or it is impossible to obtain the transition probability among the generalized clusters. Moreover, if the attacker has obtained the raw dataset, no one can protect its privacy.

E. Post-Processing

The Laplace noise is random and independent of the underlying dataset; thus, the noisy prefix tree constructed by DPTD is prone to inconsistencies. For instance, the count of a node is not equal to the sum of its child nodes counts. This paper adopts the post-processing method proposed in [44] based on the least square estimation. The post-processing is in two steps:

Firstly, it scans from leaf nodes to the root node, and if the node is a leaf node, assigns the noisy count $\tilde{h}[v]$ to $z[v]$; otherwise, $z[v]$ is the weighted average of the current node's noisy count and its child nodes' counts. Formally expressed as:

$$z[v] = \begin{cases} \tilde{h}[v], \quad if\ v\ is\ a\ leaf\ node \\ \dfrac{k^l + k^{l-1}}{k^l - 1}\tilde{h}[v] + \dfrac{k^{l-1} - 1}{k^l - 1} \displaystyle\sum_{u \in child(v)} z[u], \quad o.w. \end{cases} \tag{9}$$

where $z[v]$ is an intermediate value, $\tilde{h}[v]$ is the noisy count of node v, k is the number of v's siblings, l is the height of the node v, u is the children of the node v.

Secondly, it reversely scans from the root node to leaf nodes, if v is the root node, assigns $z[v]$ to $\bar{h}[v]$; otherwise, calculates the difference between its parent node's noisy count and the sum of its siblings' noisy count, and divide the difference equally to k siblings. The formula indicates as:

$$\bar{h}[v] = \begin{cases} z[v], \quad if\ v\ is\ the\ root \\ z[v] + \dfrac{1}{k}\left(\bar{h}[u] - \displaystyle\sum_{w \in child(u)} z[w]\right), \quad o.w. \end{cases} \tag{10}$$

where $\bar{h}[v]$ is the releasing value after consistent constraint, and u is the parent of node v.

V. Analysis

This section proves that DPTD satisfies ε-differential privacy and analyzes the computational cost.

A. Privacy Analysis

Theorem 3: DPTD satisfies ε-differential privacy.

Proof: DPTD consists of three parts: generalization, construct a noisy prefix tree, and post-processing. According to the Sequential Composition in Theorem 1, the privacy budget of DPTD is the sum of these three parts.

(1) In the process of generalization, a quadtree partitions the location universe at each timestamp, and a location is randomly selected as the releasing centroid from the leaf node where the actual centroid is located. Then, the releasing centroid replaces other locations within the cluster for a generalization. Since the releasing centroid is selected randomly, the process does not consume the privacy budget.

(2) The privacy budget is consumed in constructing a noisy prefix tree, and the following shows that the process satisfies ε-differential privacy.

From Definition 1, to prove releasing mechanism A satisfies ε-differential privacy, it is necessary to certify that for two adjacent datasets D and D', the output O of mechanism A satisfies:

$$\Pr[A(D) = O] \le e^{\varepsilon} \times \Pr[A(D') = O]$$

Therefore, we set $A : D \to Z^{|h|}$ represent prefix-tree releasing function, and $f(x) : D \to Z$ is the function that output the releasing centroid at each timestamp. Assuming that, D and D' are adjacent datasets, for an arbitrary output $z \in Z^{|h|}$, the following holds:

$$\begin{aligned} &\frac{\Pr[A(D) = z]}{\Pr[A(D') = z]} \\ &= \prod_{i=1}^{\lceil h/2 \rceil} \frac{\exp\left(-\frac{\frac{\varepsilon}{\lceil h/2 \rceil} \times |f(D)_i - z_i|}{\Delta f}\right)}{\exp\left(-\frac{\frac{\varepsilon}{\lceil h/2 \rceil} \times |f(D')_i - z_i|}{\Delta f}\right)} \\ &= \prod_{i=1}^{\lceil h/2 \rceil} \exp\left(\frac{\frac{\varepsilon}{\lceil h/2 \rceil} \times \left(\left|f\left(D'\right)_i - z_i\right| - \left|f(D)_i - z_i\right|\right)}{\Delta f}\right) \\ &\le \prod_{i=1}^{\lceil h/2 \rceil} \exp\left(\frac{\frac{\varepsilon}{\lceil h/2 \rceil} \times \left|f\left(D'\right)_i - f(D)_i\right|}{\Delta f}\right) \\ &= \exp\left(\frac{\varepsilon \times \left\|f(D)_i - f\left(D'\right)_i\right\|_1}{\Delta f}\right) \\ &= \exp(\varepsilon) \end{aligned}$$

where $\Delta f = \left\|f(D)_i - f\left(D'\right)_i\right\|_1 = 1$.

That is, $\Pr[A(D) = z] \le e^{\varepsilon} \Pr\left[A\left(D'\right) = z\right]$. Therefore, the process of constructing a noisy prefix tree satisfies ε-differential privacy.

(3) Hay *et al.* [44] have proved that post-processing does not cause privacy disclosure, and there is no extra privacy budget allocated to post-processing.

(4) Therefore, according to the sequential composition in Theorem 1, DPTD satisfies ε-differential privacy.

B. Complexity Analysis

The computational cost of DPTD mainly relates to the calculation of the generalization, Markov prediction, and prefix tree construction.

Firstly, we analyze the computational complexity brought by the generalization, which is mainly composed of DBSCAN clustering and quadtree partition. The DBSCAN clustering is performed at each timestamp, and its computational complexity is $O\left(|D|^2 \times h\right)$, where $|D|$ is the number of trajectories in the original database D and h is the trajectory length. The complexity of constructing a quadtrees with height h is $O(h \times |D|)$. Since h is a small constant, the total computational complexity of generalization is $O\left(|D|^2\right)$.

Secondly, we predict the node counts on even layers, which is transition probability matrix multiplies the prior node counts. Since the transition probability matrix's dimension is $m \times m$ and the number of nodes on the previous layer approximately equal to m/h, where m is the number of total clusters, the computational complexity of Markov prediction is $O\left(m^2\right)$.

TABLE III
DATASETS USED IN OUR EXPERIMENTS

	$\|D\|$	$\Omega(D)$	Sampling $\|D\|$	Sampling interval	$\|T\|$	$\|\tilde{T}\|$
T-drive	14,650	Beijing, China	1000	177seconds	12.11	6 □ 10
Geolife	4,800,000	Beijing, China	1000	2.5seconds	19.75	6 □ 10

Thirdly, we construct a noisy prefix tree, the complexity of constructing a prefix tree's layer is $O\left(|S_D| \times m\right)$, where $|S_D|$ is the number of the generalized trajectory dataset.

Therefore, the total computational complexity of building a noisy prefix tree of height h is $O\left(|S_D| \times m \times h\right)$. Since $m \times h < |S_D|$, and $|S_D| < |D|$, the computational complexity of our proposed DPTD is $O\left(|D|^2\right)$.

VI. EXPERIMENT EVALUATION

This section conducts experiments on real-life datasets to verify DPTD's privacy and the validity of released dataset. The experiment results show that the privacy budget utilization and our releasing synthetic dataset's effectiveness are better than other algorithms under the same privacy guarantee.

A. Experiment Setup

We implement DPTD in JAVA, and perform experiments on a laptop with Intel Core i7-10510U CPU (2.3GHz) and 8G RAM.

Competitors. For verifying the effectiveness of our proposed algorithm, this paper mainly analyzes data availability and privacy promotion. We compare DPTD with three relative algorithms Prefix, n-gram, and DPNG. Chen *et al.* [28] propose a trajectory releasing algorithm (Prefix) by constructing a noisy prefix tree. Chen *et al.* [29] put forward a variable n-gram model (n-gram), which employs an exploration tree based on the Markov assumption. Li *et al.* [45] present a differentially private noise generation algorithm (DPNG), utilizing the exponential mechanism to group the near trajectories. Basically, the comparative methods strive to release trajectory datasets using differential privacy. Moreover, the Prefix [28] and n-gram [29] employ prefix trees to store, perturb, and release trajectories while satisfying differential privacy. Besides, DPNG [45] enhanced a differentially private generalization algorithm to narrow down the output domain. Our proposed DPTD flexibly integrate prefix tree and generalization, relieving the sparsity of data distribution and maintaining individuals' mobility pattern with differential privacy guarantee.

Dataset. We use two real-life trajectory datasets in our experiments to evaluate the privacy level and utility of DPTD: T-drive[1] and Geolife.[2] The information regarding the datasets is summarized in Table III. The T-drive records the GPS trajectories of 10,357 taxis in Beijing. The total number of locations in the dataset is about 15 million, and the total distance of trajectories is 900 kilometers, the average sampling interval is 177 seconds. The interval distance is about 623 meters. The Geolife data set collected 17,621 trajectories of 182 users, with a total distance of about 1.2 million kilometers. These trajectories were recorded by different GPS recorders and GPS phones with different sampling rates. 91% of the trajectories were sampled in a relatively dense manner, such as one sampling is conducted at an interval of 1-5 seconds or 5-10 meters per sampling point. The data set records a wide range of users' outdoor activities, including not only daily life like going home and going to work, but also some recreational and sports activities like shopping, sightseeing, dining, hiking, and biking.

Geolife was collected during the five years from 2007 to 2012, whereas most T-drive's data is collected within a week. These trajectories still had almost no identical timestamps. The trajectories are long or short, which are unsuitable for direct use, so we did some preprocessing on these trajectories. We selected the trajectories with relatively uniform time intervals and set the sampling interval to 6-8 minutes and at least 6 timestamps. In the preprocessing of the experiment, we needed to truncate the trajectory to the same length and compare the availability of trajectories with different lengths. For example, the trajectory lengths are 6, 8, and 10, respectively. Therefore, some shorter or fragmented trajectories will be filtered out, and we randomly select 1000 of remaining trajectories for the experimental data.

B. Availability Metric

The purpose of privacy protection is not to hide secrets entirely completely but to expose useful information to the clients while hiding sensitive information. That is, to protect user privacy while maxing the availability of data. We use three popular availability metrics to quantify the difference between the original dataset and releasing dataset. The low difference represents high utility.

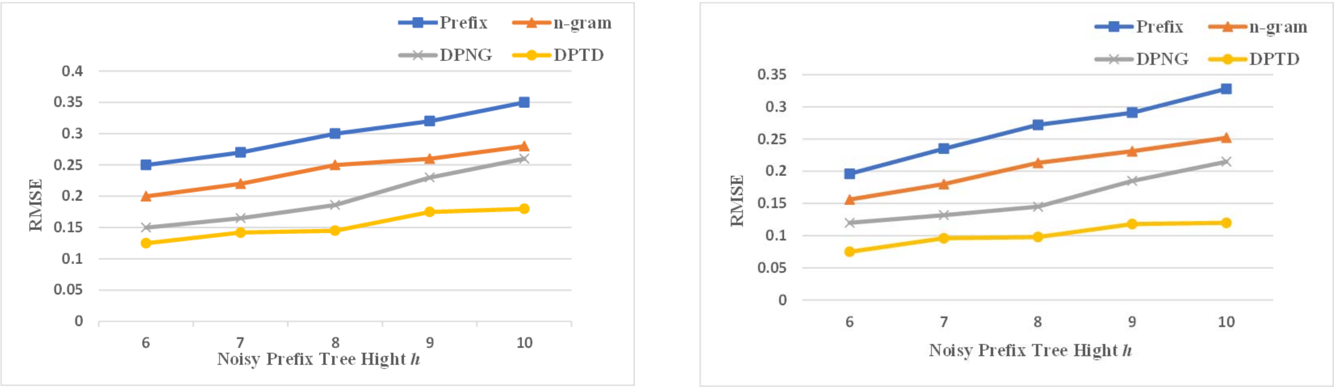
1) Root Mean Square Error (RMSE): It is a popular method that measures the difference between releasing data and raw data, meanwhile, it evaluates releasing data's accuracy. Let T be the original trajectory and R be the released trajectory, then the Root Mean Square Error between T and R is defined as:

$$RMSE_{(T,R)} = \frac{1}{|T|}\sum_{i=1}^{|T|} d_2(L_i, Z_i) \tag{11}$$

---

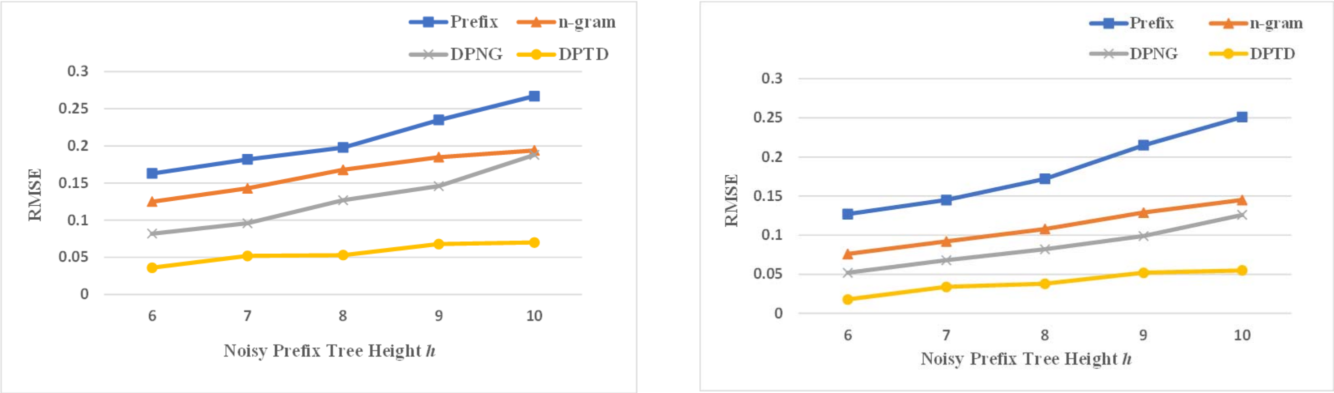
[1] https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/

[2] https://www.microsoft.com/en-us/download/details.aspx?id=52367



(a) Epsilon=0.1

(b) Epsilon=0.5



(c) Epsilon=1.0

(d) Epsilon=2.0

Fig. 3. RMSE is with T-Drive.

where L_i and Z_i are the i-th locations in T and R, respectively, $d_2(L_i, Z_i)$ is the Euclidean distance between L_i and Z_i, $|T|$ is the length of trajectory.

Similarly, the Root Mean Square Error of the trajectory data set is:

$$RMSE_{(D)} = \frac{1}{|D|}\sum_{j=1}^{|D|} RMSE_{(T,R)} \tag{12}$$

where the $|D|$ represents the number of trajectories in the data set $D = \left\{T_1, T_2, \cdots, T_{|D|}\right\}$.

2) Query Error (RE): This method is often used to evaluate the accuracy of data releasing algorithms. We consider spatial count queries of the form: "Return the number of trajectories passing through a certain region R". Let Q denote a query of this form, and $Q(D)$ denotes its answer when issued on raw dataset D, and $Q(S_D)$ denotes its answer when issued on synthetic dataset S_D. The relative error (RE) of Q is defined as:

$$RE = \frac{|Q(D) - Q(S_D)|}{\max\{Q(D),\ b\}} \tag{13}$$

where b is a threshold that mitigates the effort of extremely selective queries. In this paper, we set $b = 1\% \times |D|$. We randomly generate 500 queries and computer the average *RE* of all queries.

3) Frequent Pattern (FP) Mining: For each trajectory in a dataset, we map the trajectory on a uniform grid U, then obtain the sequence of cells it passes through, that is, $P:\ C_2 \rightarrow C_4 \rightarrow C_3$, where C_i is the cell identifier, and P is an order of cells. We define $supp(D,\ P)$ as the support of a pattern that represents the number of occurrences of P in D. The top-k pattern in dataset D is represented as $F_U^k(D)$. For each pattern $P \in F$, the frequent pattern relative error denotes the difference between $supp(D,\ P)$ and $supp(S_D,\ P)$, and its formal definition is:

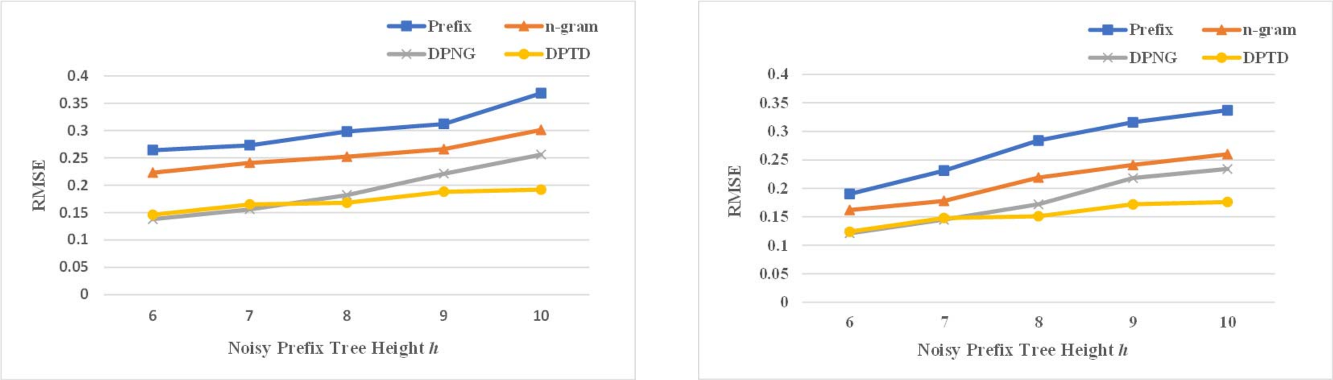
$$FP\ AvRE = \frac{\sum_{P \in F_U^k(D)} \frac{|supp(D,\ P) - supp(S_D,\ P)|}{supp(D,\ P)}}{k} \tag{14}$$

where S_D is the synthetic dataset.

In our experiment we use $k = 20$ to concern the most frequent patterns with significant support. Furthermore, we only consider patterns which are at most 4 cells long. The uniform grid U is 6×6 on T-drive and 10×10 on Geolife for FP AvRE metric.

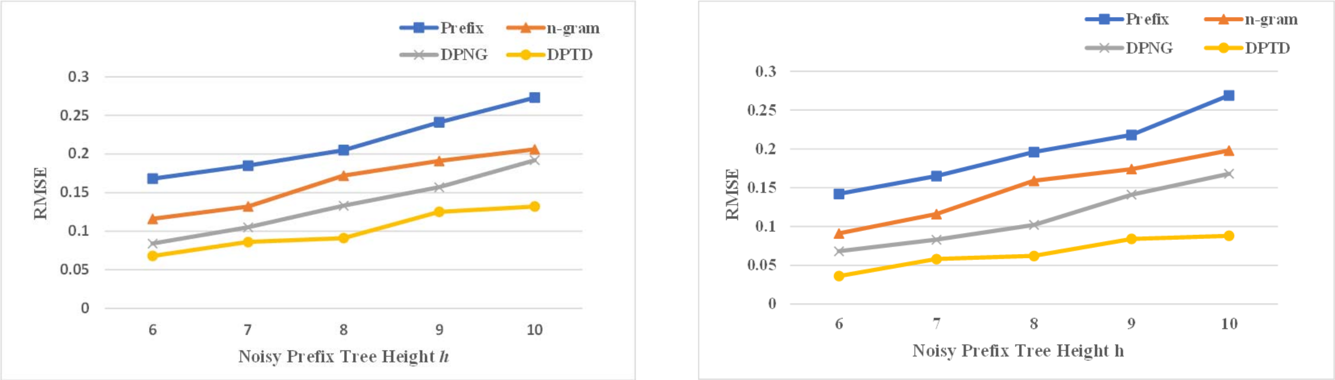
C. DPTD Performance Evaluation

As shown in Fig 3 and Fig 4, the height of prefix tree h (i.e., the length of the trajectory) is 6 to 10, and the privacy budget ε is 0.1 to 2.0. The X-axis represents the noisy prefix



(a) Epsilon=0.1

(b) Epsilon=0.5



(c) Epsilon=1.0

(d) Epsilon=2.0

Fig. 4. RMSE is with Geolife.

tree height, and Y-axis expresses the RMSE. Fig 3 and Fig 4 show the changes in the RMSE when the noisy prefix tree heights are 6, 7, 8, 9, and 10 respectively.

It can be seen from Fig 3 that the RMSE decreases with the increase of the privacy budget. Since the privacy budget increase, the noise injected into the prefix tree decrease, and the data availability increases so the RMSE gradually decrease.

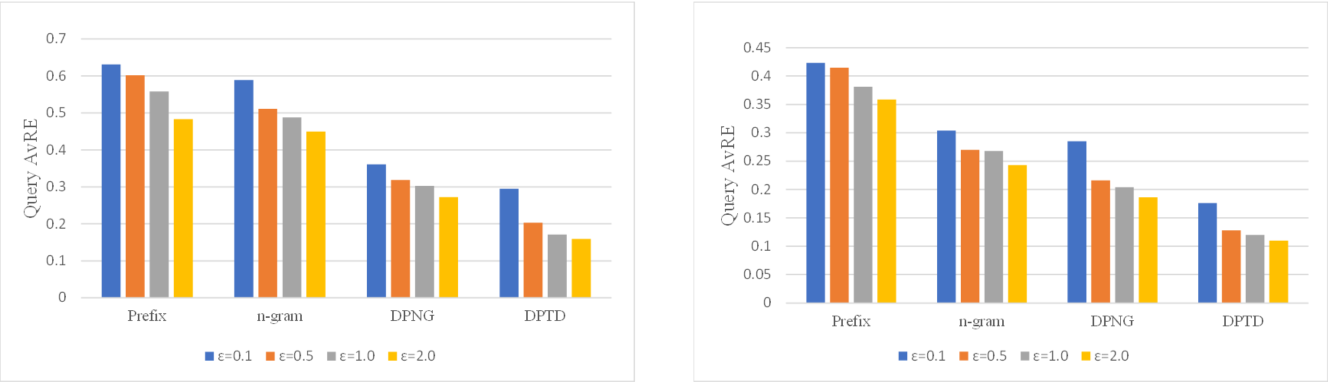
Meanwhile, RMSE shows an increasing trend as the trajectory gets longer. That is because the privacy budget is split more often, causing the noise added to each layer to increase, affecting data availability. But the root mean square error of DPTD is still lower than other competitors, which shows that our proposed algorithm has higher data availability.

In addition, from Fig 3, the RMSE increases more slowly than other competitors, which shows the stability of DPTD is relatively high. This is because, the number of privacy budget partitions in our algorithm is about half of the traditional methods. For example, the privacy budget ε allocated to layer 7 and layer 8 are both $\varepsilon/4$; thus, the perturbation errors are almost the same for the two layers. However, the utility is slightly degraded at $h = 8$ compared to $h = 7$, which is due to some errors in the Markov transition probability prediction. By the same token, the RMSE increases smoothly between $h = 9$ and $h = 10$.

As shown in Fig 4, we compare the RMSE of DPTD to competitors on the Geolife dataset. In this experiment, the RMSE of DPTD is almost smaller than other algorithms under each privacy budget. However, compared to Fig 3, the RMSE in Geolife is larger than in T-drive. This is expected since, in the preprocessing of the T-Drive dataset, we only selected the taxi trajectories in the four districts, including Haidian, Dongcheng, Xicheng, and Chaoyang, which are very close to each other, and the density of the selected trajectories is relatively large. Whereas in the preprocessing of Geolife, we randomly select all the vehicle trajectories in Beijing, and the trajectory distribution is more scattered.

The reason for sampling data in such a way is as follows. First, T-drive records the trajectories of Taxi, we observe the distribution and find that the data is density in these four districts. Second, we want to compare RMSE of dense and regular distribution using DPTD, and analyze which distribution is more suitable.

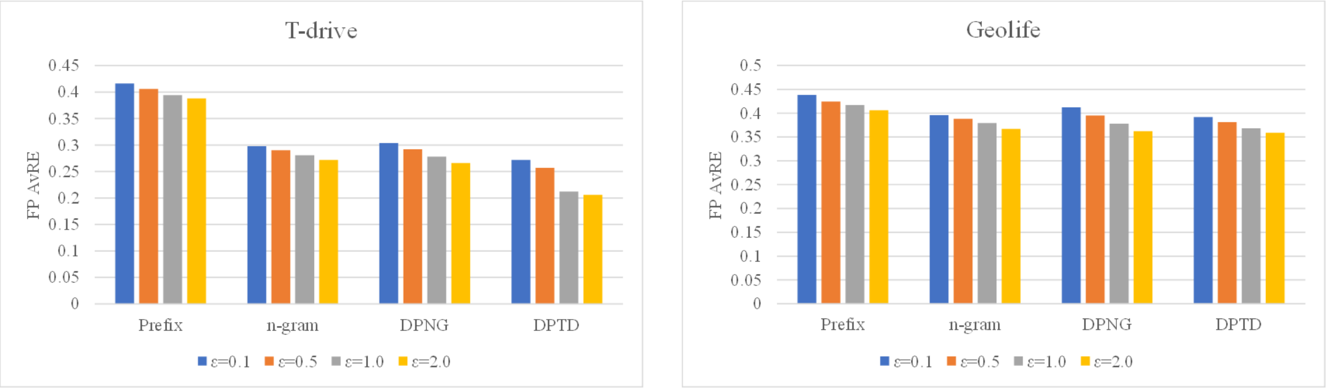
In the process of generalization, we cluster locations at each timestamp using DBSCAN, which performs superior on dense distribution than on sparse. Therefore, when the prefix tree heights are 6 and 7, the RMSE of DPTD is slightly higher than DPNG under $\varepsilon = 0.1,\ 0.5$. Nonetheless, the privacy budget utilization of DPTD is nearly twice that of the other algorithms, and the error does not change significantly. Thus, the Root Mean Square Error is still smaller than other algorithms when the heights are 8 to 10.



(a) Geolife

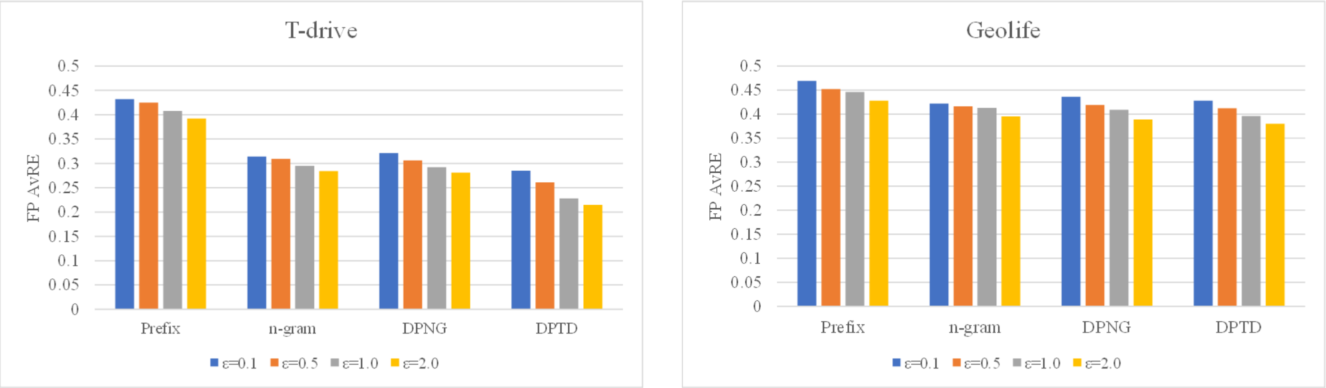
(b) T-drive

Fig. 5. Query AvRE on the Geolife and T-drive.



(a) h=8, T-drive

(b) h=8, Geolife



(c) h=10, T-drive

(d) h=10, Geolife

Fig. 6. FP AvRE on the Geolife and T-drive.

As shown in Fig 5, we compare the query average relative error of DPTD with its competitors while varying ε from 0.1 to 2.0. Since results show similar representation across different tree heights, we set the trajectory length of all algorithms as 10 in this experiment. The X-axis denotes compared algorithms, and the Y-axis represents the value of Query AvRE. As expected, all algorithms' Query AvRE decrease when ε increases, because less noise is added due to the privacy budget increases; thus, the releasing noisy prefix tree is more available. It can be seen that our proposed method DPTD maintains high utility under all the ε for Query AvRE on both T-drive and Geolife. DPTD's superiority is due to the reason that the Markov assumption can significantly reduce the privacy budget consumption, and the consistency constraints also promote the data availability. We can observe that the Query AvRE on T-drive also performs better than Geolife, and the reason is similar to the RMSE analyzed above. The preprocess of T-drive, making its distribution denser, thus, more effective in clustering.

Fig 6(a) to (b) is the FP AvRE comparison when $h = 8$, and (c) to (d) represents the situation of $h = 10$. For both Geolife and T-drive, the FP AvRE of each algorithm decreases when $h = 8$ compared to $h = 10$, and DPTD is superior to other algorithms. It is because the privacy budget consumes less when $h = 8$ than $h = 10$ for all algorithms. Specifically, DPTD predicts or perturbs prefix tree nodes alternatively by utilizing low-order Markov assumption with spatio-temporal correlations. As a consequence, nearly half of the privacy budget is saved, and the noise added to each node is reduced more significantly, thus improving the performance of FP AvRE. Although n-gram adaptively allocates the privacy budget, it intercepts the trajectories into small segments, affecting the sequential characteristic. However, DPTD is slightly inferior than n-grams when the privacy budget $\varepsilon = 0.1$, and $h = 10$ on the Geolife. That is because DPTD generalizes location sequences to cluster sequences, causing some errors when locations near the edge of cells. DPNG suffers from the same problem. It groups trajectories using the exponential mechanism, giving rise to the situation that locations within a cluster on the same universe are not near each other. Therefore, it leads to considerable errors when it projects the trajectories to a uniform grid. Thus, DPNG performs less effectively than n-grams in some cases.

VII. CONCLUSION

This paper proposes a trajectory dataset releasing mechanism based on differential privacy to address the privacy leakage when sharing location data on the Internet of Vehicles. Unlike most studies that only consider the trajectories' locations, we introduce the time dimension to better analyze users' behavior patterns. The algorithm adopts clustering followed by generalization to address the sparse data distribution and high global sensitivity caused by introducing the time dimension. The trajectory dataset is released by constructing a noisy prefix tree, using the Markov chain to model the Spatio-temporal correlation of locations in the trajectory. The count of nodes in the even layers can be predicted by the Markov transition probability, saving nearly half of the privacy budget and significantly reducing the total amount of added noise.

There are still some issues we need to discuss and improve in future work. In this study, we sample some trajectories and set the trajectory's length as a fixed value. In the next step, we will introduce trajectories of variable lengths, adaptively partition the privacy budget according to trajectory lengths and privacy requirements. Additionally, we will extend the Markov prediction framework to the dynamic trajectory data.

REFERENCES

[1] Z. Lu, G. Qu, and Z. Liu, "A survey on recent advances in vehicular network security, trust, and privacy," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 2, pp. 760–776, Feb. 2019.

[2] Z. Lv *et al.*, "Next-generation big data analytics: State of the art, challenges, and future research topics," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 2066–2076, Aug. 2017.

[3] W. Fang, X. Yao, X. Zhao, J. Yin, and N. Xiong, "A stochastic control approach to maximize profit on service provisioning for mobile cloudlet platforms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 4, pp. 522–534, Apr. 2018.

[4] Y. Jing, H. Hu, S. Guo, X. Wang, and F. Chen, "Short-term prediction of urban rail transit passenger flow in external passenger transport hub based on LSTM-LGB-DRS," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4611–4621, Jul. 2021.

[5] T. Zhou *et al.*, "Evaluation of urban bus service reliability on variable time horizons using a hybrid deep learning method," *Rel. Eng. Syst. Saf.*, vol. 217, Jan. 2022, Art. no. 108090.

[6] Y.-A. de Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel, "Unique in the crowd: The privacy bounds of human mobility," *Sci. Rep.*, vol. 3, no. 1, Mar. 2013, Art. no. 1376.

[7] W. Qardaji, W. Yang, and N. Li, "Differentially private grids for geospatial data," in *Proc. IEEE 29th Int. Conf. Data Eng. (ICDE)*, Brisbane, QLD, Australia, Apr. 2013, pp. 757–768.

[8] Y. Qu, S. Yu, W. Zhou, S. Chen, and J. Wu, "Customizable reliable privacy-preserving data sharing in cyber-physical social networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 1, pp. 269–281, Jan. 2021.

[9] J. Hua, W. Tong, F. Xu, and S. Zhong, "A geo-indistinguishable location perturbation mechanism for location-based services supporting frequent queries," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 5, pp. 1155–1168, May 2018.

[10] Y. Xiao and L. Xiong, "Protecting locations with differential privacy under temporal correlations," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, Denver, CO, USA, Oct. 2015, pp. 1298–1309.

[11] I. Memon, I. Hussain, R. Akhtar, and G. Chen, "Enhanced privacy and authentication: An efficient and secure anonymous communication for location based service using asymmetric cryptography scheme," *Wireless Pers. Commun.*, vol. 84, no. 2, pp. 1487–1508 2015.

[12] X.-Y. Li and T. Jung, "Search me if you can: Privacy-preserving location query service," in *Proc. IEEE INFOCOM*, Turin, Italy, Apr. 2013, pp. 2760–2768.

[13] J. Kang, Z. Xiong, D. Ye, D. I. Kim, J. Zhao, and D. Niyato, "Toward secure blockchain-enabled internet of vehicles: Optimizing consensus management using reputation and contract theory," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2906–2920, Mar. 2019.

[14] M. E. Andres, N. E. Bordenabe, and K. Chatzikokolakis, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proc. ACM Conf. Comput. Commun. Secur.*, Berlin, Germany, 2013, pp. 901–914.

[15] N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Optimal geo-indistinguishable mechanisms for location privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Scottsdale, AZ, USA, Nov. 2014, pp. 251–262.

[16] O. Abul, F. Bonchi, and M. Nanni, "Never walk alone: Uncertainty for anonymity in moving objects databases," in *Proc. IEEE 24th Int. Conf. Data Eng.*, Cancun, Mexico, Apr. 2008, pp. 376–385.

[17] C.-Y. Chow, M. F. Mokbel, and T. He, "A privacy-preserving location monitoring system for wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 10, no. 1, pp. 94–107, Jan. 2011.

[18] Z. Tu, K. Zhao, F. Xum Y. Li, Li Su, and D. Jin, "Protecting trajectory from semantic attack considering k-anonymity, l-diversity, and t-closeness," *IEEE Trans. Netw. Serv. Manag.*, vol. 16, no. 1, pp. 264–278, Mar. 2019.

[19] J. Wang and M.-P. Kwan, "Daily activity locations k-anonymity for the evaluation of disclosure risk of individual GPS datasets," *Int. J. Health Geograph.*, vol. 19, no. 1, pp. 1–4, Mar. 2020.

[20] L. Ni, F. Tian, Q. Ni, Y. Yan, and J. Zhang, "An anonymous entropy-based location privacy protection scheme in mobile social networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 1, pp. 1–9, Apr. 2019.

[21] D. Hemkumar, S. Ravichandra, and D. V. L. N. Somayajulu, "Impact of prior knowledge on privacy leakage in trajectory data publishing," *Eng. Sci. Technol., Int. J.*, vol. 23, no. 6, pp. 1291–1300, Dec. 2020.

[22] E. G. Komishani, M. Abadi, and F. Deldar, "PPTD: Preserving personalized privacy in trajectory data publishing by sensitive attribute generalization and trajectory local suppression," *Knowl.-Based Syst.*, vol. 94, pp. 43–59, Feb. 2016.

[23] C. Dwork, "Differential privacy," in *Proc. 33rd Int. Colloq. Automata Lang. Program.*, Venice, Italy, 2006, pp. 1–12.

[24] C. Dwork and J. Lei, "Differential privacy and robust statistics," in *Proc. 41st Annu. ACM Symp. Theory Comput.*, Bethesda, MD, USA, 2009, pp. 371–380.

[25] C. Dwork, M. Naor, and S. Vadhan, "The privacy of the analyst and the power of the state," in *Proc. IEEE 53rd Annu. Symp. Found. Comput. Sci.*, New Brunswick, NJ, USA, Oct. 2012, pp. 400–409.

[26] K. Jiang, D. Shao, S. Bressan, T. Kister, and K.-L. Tan, "Publishing trajectories with differential privacy guarantees," in *Proc. 25th Int. Conf. Sci. Stat. Database Manage. (SSDBM)*, Baltimore, MD, USA, 2013, p. 12.

[27] D. Shao, K. Jiang, T. Kister, S. Bressan, and K.-L. Tan, "Publishing trajectory with differential privacy: *A priori* vs. a posteriori sampling mechanisms," in *Proc. Int. Conf. Database Expert Syst. Appl.*, in Lecture Notes in Computer Science, Prague, Czech Republic, 2013, pp. 357–365.

[28] R. Chen, B. C. M. Fung, and B. C. Desai, "Differentially private trajectory data publication," *Comput. Sci.*, vol. 22, no. 1, pp. 11–22, 2011.

[29] R. Chen, G. Acs, and C. Castelluccia, "Differentially private sequential data publication via variable-length n-grams," in *Proc. ACM Conf. Comput. Commun. Secur. (CCS)*, Raleigh, NC, USA, 2012, pp. 638–649.

[30] J. Hua, Y. Gao, and S. Zhong, "Differentially private publication of general time-serial trajectory data," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Hong Kong, Apr. 2015, pp. 1298–1309.

[31] M. E. Nergiz, M. Atzori, and Y. Saygin, "Towards trajectory anonymization: A generalization-based approach," in *Proc. ACM Int. Workshop Secur. Privacy GIS LBS-SPRINGL*, Irvine, CA, USA, 2008, pp. 52–61.

[32] H. Hu, J. Xu, S. T. On, J. Du, and J. K.-Y. Ng, "Privacy-aware location data publishing," *ACM Trans. Database Syst.*, vol. 35, no. 3, pp. 1–42, Jul. 2010.

[33] M. Terrovitis and N. Mamoulis, "Privacy preservation in the publication of trajectories," in *Proc. 9th Int. Conf. Mobile Data Manage. (MDM)*, Beijing, China, Apr. 2008, pp. 65–72.

[34] R. Chen, B. C. M. Fung, N. Mohammed, B. C. Desai, and K. Wang, "Privacy-preserving trajectory data publishing by local suppression," *Inf. Sci.*, vol. 231, pp. 83–97, May 2013.

[35] M. Douriez, H. Doraiswamy, J. Freire, and C. T. Silva, "Anonymizing NYC taxi data: Does it matter?" in *Proc. IEEE Int. Conf. Data Sci. Adv. Anal.*, Montreal, QC, Canada, Oct. 2016, pp. 140–148.

[36] C. Y. T. Ma, D. K. Y. Yau, N. K. Yip, and N. S. V. Rao, "Privacy vulnerability of published anonymous mobility traces," *IEEE/ACM Trans. Netw.*, vol. 21, no. 3, pp. 720–733, Jun. 2013.

[37] X. He, G. Cormode, A. Machanavajjhala, C. M. Procopiuc, and D. Srivastava, "DPT: Differentially private trajectory synthesis using hierarchical reference systems," *Proc. VLDB Endowment*, vol. 8, no. 11, pp. 1154–1165, Jul. 2015.

[38] M. E. Gursoy, L. Liu, S. Truex, and L. Yu, "Differentially private and utility preserving publication of trajectory data," *IEEE Trans. Mobile Comput.*, vol. 18, no. 10, pp. 2315–2329, Oct. 2019.

[39] K. Al-Hussaeni, B. C. M. Fung, F. Iqbal, G. G. Dagher, and E. G. Park, "SafePath: Differentially-private publishing of passenger trajectories in transportation systems," *Comput. Netw.*, vol. 143, pp. 126–139, Oct. 2018.

[40] X. Zhao, D. Pi, and J. Chen, "Novel trajectory privacy-preserving method based on clustering using differential privacy," *Exp. Syst. Appl.*, vol. 149, Jul. 2020, Art. no. 113241.

[41] Y. Cao, M. Yoshikawa, Y. Xiao, and L. Xiong, "Quantifying differential privacy in continuous data release under temporal correlations," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 7, pp. 1281–1295, Jul. 2019.

[42] S. Ghane, L. Kulik, and K. Ramamohanarao, "TGM: A generative mechanism for publishing trajectories with differential privacy," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 2611–2621, Apr. 2020.

[43] S. Gambs, M.-O. Killijian, and M. N. del Prado Cortez, "Next place prediction using mobility Markov chains," in *Proc. 1st Workshop Meas., Privacy, Mobility (MPM)*, Bern, Switzerland, 2012, pp. 1–6.

[44] M. Hay, V. Rastogi, G. Miklau, and D. Suciu, "Boosting the accuracy of differentially private histograms," *Proc. VLDB Endowment*, vol. 3, no. 1, pp. 1021–1032, 2010.

[45] M. Li, L. Zhu, Z. Zhang, and R. Xu, "Achieving differential privacy of trajectory data publishing in participatory sensing," *Inf. Sci.*, vols. 400–401, pp. 1–13, Aug. 2017.

Sujin Cai received the B.E. degree in computer science and technology from Soochow University and the M.E. degree in computer science and technology from the Nanjing University of Science and Technology. She is currently pursuing the Ph.D. degree in computer science and technology with Hohai University, Nanjing, China. Her current research interests include security and privacy preserving.

Xin Lyu is currently an Associate Professor with the College of Computer and Information, Hohai University. He has published over 60 papers. His research interests include cryptography, network information security, and privacy-preserving theory and technology.

Xin Li (Member, IEEE) received the B.S. degree from Zhengzhou University and the M.S. degree from Hohai University, Nanjing, China, where he is currently pursuing the Ph.D. degree in computer science and technology with the College of Computer and Information. His research interests include computer vision and remote sensing image processing.

Duohan Ban received the B.E. and master's degrees in computer science and technology from Hohai University, Nanjing, China. Her research interests include image encryption and network security.

Tao Zeng received the B.E. degree in computer science and technology from Hohai University, Nanjing, China, where he is currently pursuing the Ph.D. degree with the College of Computer and Information. His research interests include computer vision, deep learning, and anomaly detection.