LDPTrace: Locally Differentially Private Trajectory Synthesis

Yuntao Du
Zhejiang University
ytdu@zju.edu.cn

Yujia Hu
Zhejiang University
charliehu@zju.edu.cn

Zhikun Zhang
Stanford University
zhikun@stanford.edu

Ziquan Fang
Lu Chen
Zhejiang University
{zqfang,luchen}@zju.edu.cn

Baihua Zheng
Singapore Management University
bhzheng@smu.edu.sg

Yunjun Gao
Zhejiang University
gaoyj@zju.edu.cn

ABSTRACT

Trajectory data has the potential to greatly benefit a wide-range of real-world applications, such as tracking the spread of the disease through people's movement patterns and providing personalized location-based services based on travel preference. However, privacy concerns and data protection regulations have limited the extent to which this data is shared and utilized. To overcome this challenge, local differential privacy provides a solution by allowing people to share a perturbed version of their data, ensuring privacy as only the data owners have access to the original information. Despite its potential, existing point-based perturbation mechanisms are not suitable for real-world scenarios due to poor utility, dependence on external knowledge, high computational overhead, and vulnerability to attacks. To address these limitations, we introduce LDPTrace, a novel locally differentially private trajectory synthesis framework. Our framework takes into account three crucial patterns inferred from users' trajectories in the local setting, allowing us to synthesize trajectories that closely resemble real ones with minimal computational cost. Additionally, we present a new method for selecting a proper grid granularity without compromising privacy. Our extensive experiments using real-world data, various utility metrics and attacks, demonstrate the efficacy and efficiency of LDPTrace.

PVLDB Reference Format:
Yuntao Du, Yujia Hu, Zhikun Zhang, Ziquan Fang, Lu Chen, Baihua Zheng, and Yunjun Gao. LDPTrace: Locally Differentially Private Trajectory Synthesis. PVLDB, 16(8): 1897 - 1909, 2023.
doi:10.14778/3594512.3594520

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/zealscott/LDPTrace.

1 INTRODUCTION

The widespread availability of location sensing technology, such as GPS, has revolutionized our ability to collect real-time data. As a result, there has been significant interest in studying human mobility patterns on a large scale for a variety of location-based applications, including traffic prediction, route planning, and recommendation. Despite the immense value of this data, privacy concerns surrounding the sensitive nature of trajectories have limited its use.

Differential privacy (DP) has become the *de facto* standard for protecting sensitive data while ensuring individual privacy. Despite the development of various DP algorithms for trajectory publishing and analysis, these methods rely on a *trustworthy aggregator* to collect users' raw trajectories. In contrast, local differential privacy (LDP) allows users to directly share a noisy version of their data, reducing the risk of data breaches from untrustworthy data curators.

The LDP provides a more practical setting and improved privacy properties, however, it imposes challenges in preserving the complex spatial patterns of trajectories due to its strict privacy requirements. Currently, the only solution that meets the rigorous privacy requirement of LDP is NGRAM [14]. This method uses the exponential mechanism to directly perturb individual trajectory in the local setting and leverages external knowledge (POIs, business opening hours, *etc.*) and overlapped n-grams to enhance the realism of the noisy trajectories. However, NGRAM has several major limitations that hinder its effectiveness:

- **Poor global utility.** As NGRAM only focuses on local trajectory proximity for utility optimization (similarity between original and perturbed trajectories), it results in poor global utilities, as reported in Section 6.3. Most location-based applications rely on population-level spatial statistics (*e.g.,* range query and spatial density) and moving patterns (*e.g.,* distribution of start/end points and frequent travel patterns), rather than individual behaviors (*e.g.,* routing preference), and the failure to preserve global utility significantly limits its applications.
- **Dependence on auxiliary knowledge.** The performance of NGRAM heavily relies on external knowledge (*e.g.,* POI categories and business opening hours), which may not always be accessible and can become outdated easily. This leads to a dramatic decrease in utility and authenticity of generated trajectories. In addition, users are required to store the external data on their own devices, which is highly impractical for wearable or low-cost GPS devices with limited storage.
- **High computational overhead.** To obtain accurate perturbed trajectories, NGRAM uses linear programming solvers, which are time-consuming and require pre-processing to deal with external knowledge (*e.g.,* POI processing and hierarchical decomposition).

Proceedings of the VLDB Endowment, Vol. 16, No. 8 ISSN 2150-8097.
doi:10.14778/3594512.3594520

This can result in significant delays for users, reducing their satisfaction with location-based services.

- **Vulnerable to attacks.** Point-based privacy mechanisms are vulnerable to location-based attacks, such as re-identification attack [34, 42] and outlier leakage [23, 28], due to the strong statistical correlation between the fake locations and user's true locations. This is even more concerning for NGRAM as it leverages external knowledge to maintain geographic and semantic similarities between real and perturbed points in the local setting (as to be detailed in Section 6.8). As a result, alternative attack-resilient approaches must be sought to address privacy concerns.

The existing shortcomings motivate us to develop a new local privacy-preserving paradigm that is both utility-aware and efficient. Instead of perturbing each trajectory individually, we aim to extract the key movement patterns of each user and use them to synthesize privacy-preserving and realistic trajectories. However, synthesizing trajectories in the local setting is *challenging* due to the following two reasons. First, previous trajectory synthesis methods either rely on global statistical metrics [4, 23] or spatial-aware data structures (*e.g.,* prefix tree [9, 25]) to model the spatial distributions. However, these are not feasible in local settings as there is no trusted data curator to collect these information and it is infeasible to directly collect the statistics from individual users, who typically only have a few trajectory footprints, leading to severely biased estimations. Second, existing LDP methods assume that each individual holds a single data record (*e.g.,* a single value) [11, 39], but this assumption is no longer valid for trajectory data, which is a sequence of spatial points with timestamps and has complex spatial context that cannot be simply modeled by the methods.

Therefore, we present LDPTrace, a simple yet effective framework for synthesizing locally differentially private and attack-resistant trajectories. LDPTrace achieves the following four objectives: (i) robust, rigorous statistical privacy, (ii) flexibility and low computational cost, (iii) strong preservation of global spatial utilities and authenticity, and (iv) deterministic resilience against trajectory privacy attacks. Specifically, LDPTrace approaches trajectory synthesis as a generative process, constructing a probabilistic model based on users' transition records to estimate the global moving patterns. The transition records capture the spatial relationship between adjacent points, while remaining low computational complexity. To enhance the authenticity of synthesized trajectories, LDPTrace includes *virtual* start and end points to each trajectory in the generation process to indicate the beginning and terminated transition states. Additionally, the framework estimates the trajectory length distribution for optimal transition budget allocation and deterministic generation constraints. We also provide a theoretical guideline for selecting grid granularity without consuming privacy budget. Last but not the least, an adaptive synthesis algorithm is employed to generate realistic trajectories without access to users' real trajectories.

Our synthetic trajectory generation process is locally differentially private, meaning that the global moving patterns are not strongly dependent on any specific user and the generation of synthetic trajectory is not bias towards any specific trajectory. We have conducted an extensive experimental evaluation to compare LDPTrace with the state-of-the-art method NGRAM. Experimental results indicate that LDPTrace significantly outperforms the competitor in terms of data utility, efficiency, and scalability, and it is equipped with robust resistance against various location-based attacks for superior privacy protection.

In summary, the key contributions of our work are below.

- We propose LDPTrace, the first trajectory synthesis solution with local differential privacy guarantee that is able to generate realistic trajectories without any external knowledge.
- We introduce a neat and effective framework that collects key moving patterns from users' trajectories with little computational cost, and devise an adaptive synthesis algorithm to generate authentic trajectories.
- We perform comprehensive analysis on the errors and complexity of the proposed framework, and present a guideline for selecting the grid granularity without consuming privacy budget.
- We conduct extensive experiments to demonstrate the superiority of LDPTrace in terms of utility, efficiency, and scalability. Moreover, we show that LDPTrace is able to resist various location-based attacks.

The rest of this paper is organized as follows. We review related work in Section 2, and elaborate our motivation in Section 3. Then, we present preliminaries in Section 4. Section 5 details the proposed synthesis framework. The experimental results are reported in Section 6. Finally, we conclude the paper, and offer directions for future work in Section 7.

2 RELATED WORK

Differential privacy (DP) [18] has become the *de facto* privacy standard. While centralized DP assumes data aggregators are reliable, local differential privacy (LDP) [17] assumes that aggregators cannot be trusted and relies on data providers to perturb their own data. Early studies [3, 12, 21, 29, 31–33, 38] on DP and LDP mostly focus on designing tailored algorithms for specific data analysis tasks, which suffers from poor flexibility, inefficiency, and scalability problems. One promising solution [6, 20, 46, 48, 49] to address this problem is generating a synthetic dataset that is similar to the private dataset while satisfying (local) differential privacy. However, these methods mainly aim at structured data like tables, which cannot be applied to trajectory data due to its high dimensionality and complex spatial dependence.

The privacy of trajectory data (surveyed in [26, 27]) has been a significant concern for over a decade and various solutions have been proposed to address the issue. Many existing solutions [1, 2, 5, 7, 28, 41, 47] employ spatial point perturbation techniques under the constraint of DP that add noises to the point locations of trajectory before it is published or used to answer predefined queries (*e.g.,* range query). For example, GL [28] aims to preserve both privacy and high utility by perturbing the local/global frequency distributions of important locations in a trajectory. SNH [47] introduces a neural database for spatial range queries and adds DP-compliant noise to the input queries to maintain the density features of location data. More recently, NGRAM [14] has been proposed to address the privacy concerns of trajectory sharing in a local setting. The method includes three phases: (1) hierarchical decomposition phase, where POIs are divided into spatial-temporal-category regions; (2) perturbation phase, where trajectories are converted to sequences of overlapping n-grams and perturbed using exponential mechanism; and (3) reconstruction phase, where NGRAM solves an optimization problem to reconstruct the continuous trajectory based on the perturbed n-grams. Despite its efforts, NGRAM still has several limitations, such as low global utility, high computational overhead, and vulnerability to attacks.

To protect privacy, some researchers have investigated alternative methods to point perturbation, such as generating synthetic trajectories [4, 9, 13, 22–25, 44]. The challenge is to create synthetic data that resembles genuine user traces while providing practical privacy protection. One approach, DPT [25], uses a hierarchical reference system to model trajectory movements at different speeds and encodes transitions between grid cells using prefix trees. By injecting Laplace noise to the prefix trees, the transition probabilities are distorted while still maintaining the movement patterns of the original traces. Further studies [4, 22] have extended DPT by incorporating trajectory semantics and temporal information. Another method, AdaTrace [23], extracts four statistical and spatial features and incorporates them into its private synopsis, including a density-aware grid, Markov mobility model, pickup/destination and length distribution. The authors also design a synthesizer with attack resilience constraints to balance both statistical privacy (differential privacy) and syntactic privacy (attack resilience). PrivTrace [37] applies an adaptive strategy to choose crucial first- and second-order Markov transitions for trajectory synthesis, resulting in better utilities than previous methods. Although current synthetic methods are effective, they all rely on a trustworthy data curator to aggregate useful statistics. Our work is the first to introduce an utility-aware and efficient trajectory synthesis framework without accessing users' real traces.

3 MOTIVATION

Trajectory is a time-order sequence of location points generated from human mobility, which is highly sensitive since it can reveal people's home/work place, travel patterns, and other preferences. The population level aggregate spatial information on where/when do residents commute could be useful for the authority to gain better understanding of residents' commuting patterns, but many people are unwilling to share their own trajectories because of privacy concerns. Therefore, we aim to propose a method for various parties (e.g., authority/service providers) to collect useful mobility patterns from crowd without accessing individual's real trajectories. In the following, we present four critical design principles that motivate and guide our solution, including *privacy protection*, *global utility*, *practicability and efficiency*, and *attack resilience*.

Privacy protection. The primary goal of our work is to protect each individual's privacy so that the untrusted data curator cannot access people's real traces. We achieve this by utilizing LDP mechanism to perturb user's trajectories before sharing the data. We detail the privacy implications of our method in Section 5.7.

Global utility. Based on the rigorous privacy guarantee of LDP, our solution is expected to be designed in such a way that it can preserve the high global utility for synthetic trajectories. We argue that a feasible way to boost global utility is to extract the key moving patterns from user's trajectories, and leverage these information to guide the synthesis process. Since the synthesizer is designed to capture the intrinsic features of user's movements, the synthetic trajectories can be used for various spatial analysis tasks like range query and frequent pattern mining, instead of tailor-made for specific utility like [14].

Table 1: Symbols and Description

Notation	Description
$T, \mathcal{T}$	Trajectory, and a set of trajectories
$C, \mathcal{C}$	Grid cell, and a set of grid cells
$L, \mathcal{L}$	Trajectory length, length distribution
$s, \mathcal{S}$	Intra-trajectory transition, mobility model
$M, \mathcal{M}$	Aggregated transition, aggregated mobility model
C_a, C_b	Virtual start/end point
$\mathcal{N}, \mathcal{N}^*$	Neighborhood cells without/with virtual end point
$\hat{g}(\cdot), \tilde{g}(\cdot)$	Report times, unbiased estimation of frequency
ϵ	Privacy budget

Practicability and efficiency. Considering the wide usage of trajectory data, our solution is preferred to be as simple as possible so that it can be easily deployed to any local devices without heavy computation. Besides, it is equally critical to ensure the efficiency, thus privacy and utility do not come at the cost of user experience.

Attack resilience. Although LDP is a strong data sharing technique with provable privacy guarantees, it still suffers from various syntactic attacks. This is especially true for trajectory privacy protection, as blindly forcing the perturbed trajectories to resemble original ones would make them vulnerable to various location-based attacks like re-identification attack and outlier leakage. Hence, the synthetic trajectories should be robust, and they are expected to be able to resist common attacks.

Applications. Our work focuses on synthesizing trajectories such that the global aggregate statistics is preserved as much as possible, which is essential to many important location-based applications. A notable one is trajectory monitoring that identifies people's movement patterns, which can be used for policy making decisions like traffic control or disease spread forecasting (like Covid-19). Other applications include location-based services and advertising, *e.g.*, a tourism recommendation system can utilize common trajectories taken by people to recommend popular trips/destinations, and an outdoor advertising company can use people's movement patterns to better estimate the traffic flows at different locations.

4 PRELIMINARIES

In this section, we first introduce the definition of local differential privacy (LDP) and then formulate our problem. Table 1 lists the notations used in this paper.

4.1 Differential Privacy in the Local Setting

4.1.1 ***ϵ-Local Differential Privacy.*** In the local setting of DP, there are many *users* and one untrusted *aggregator*, and each user perturbs the input value x using an algorithm Ψ and sends $\Psi(x)$ to the aggregator. The formal privacy requirement is that the algorithm $\Psi(\cdot)$ satisfies the following property:

Definition 1 (ϵ-Local Differential Privacy). *An algorithm $\Psi(\cdot)$ satisfies ϵ-local differential privacy (ϵ-LDP), where $\epsilon \geq 0$, if and only if for any input x_1, x_2 and output y:*

$$Pr[\Psi(x_1) = y] \leq e^{\epsilon} Pr[\Psi(x_2) = y]. \tag{1}$$

The privacy budget ϵ is a metric used to measure the level of privacy protection in local differential privacy (LDP). It represents the probability that an attacker can determine the true value of the input based on the output. The higher the privacy budget, the higher the probability that the attacker can infer the true value, and the lower the privacy protection. In practice, the privacy budget can be set according to the privacy requirements of the application. For example, a smaller privacy budget may be chosen when collecting highly sensitive data such as health information, while a larger privacy budget may be used for less sensitive data such as typing patterns. A privacy budget of less than 2 is typically considered acceptable [14–16, 19, 28, 39, 40, 47]. LDP provides privacy protection by allowing the user to report a perturbed version of the input $\Psi(x)$ instead of the true value x to the aggregator. This ensures that even if the aggregator is malicious, the user's privacy is still protected. LDP possesses two fundamental properties used in our mechanism [11]: the composition theorem, which states that k ϵ_i-LDP mechanisms can be combined to achieve ϵ-LDP protection, where $\epsilon = \sum_i \epsilon_i$; and the ability to perform post-processing on private outputs without affecting the privacy guarantee. For more information on LDP, please refer to recent surveys [11, 43, 45].

4.1.2 *Optimized Unary Encoding.*

A *frequency oracle* (FO) protocol enables the estimation of the frequency of any value x, which serves as a building block of many LDP tasks. In this paper, we opt for Optimized Unary Encoding (OUE) as the FO protocol to achieve frequency estimation under LDP, which consists of three stages: encoding, perturbing, and aggregation [39].

- **Encoding.** The original value x is first encoded as a length-d binary vector V, where only the x-th bit is set to 1, *i.e.,* $V[i] = \mathbb{1}(i == x)$.
- **Perturbing.** When reporting the encoded vector V, it is perturbed as follows:

$$Pr[\hat{V}[i] = 1] = \begin{cases} \frac{1}{2}, & \text{if } V[i] = 1 \\ \frac{1}{e^{\epsilon}+1}, & \text{if } V[i] = 0, \end{cases} \tag{2}$$

  where ϵ is the privacy budget and $\hat{V}$ is the reported noise vector.
- **Aggregation.** In order to obtain the unbiased estimation of the real value from noise vectors, the data curator needs to aggregate and adjust the received data as follows:

$$\tilde{g}(x) = \frac{\hat{g}(x) - nq}{\frac{1}{2} - q}, \; q = \frac{1}{e^{\epsilon} + 1}, \tag{3}$$

  where n is the total number of reported noise vectors, and $\hat{g}(x)$ is the total number of the reported vectors $\hat{V}$ whose x-th bit is 1, *i.e.,* $\hat{g}(x) = |\{\hat{V}|\hat{V}[x] = 1\}|$. Notice that this adjustment requires the budget ϵ to remain the same across all the reported data.

It can be theoretically proved [39] that the adjusted estimation $\tilde{g}(x)$ is unbiased. The mean and variance of OUE are listed below.

$$\mathrm{E}[\tilde{g}(x)] = f_x, \quad \mathrm{Var}[\tilde{g}(x)] = n\frac{4e^{\epsilon}}{(e^{\epsilon} - 1)^2}, \tag{4}$$

where f_x is the frequency of value x, and ϵ is the privacy budget.

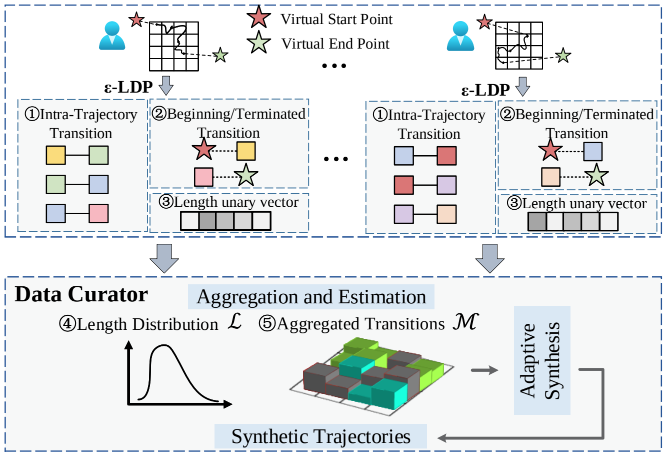
Virtual Start Point
Virtual End Point
ε-LDP
①Intra-Trajectory Transition
②Beginning/Terminated Transition
③Length unary vector
ε-LDP
①Intra-Trajectory Transition
②Beginning/Terminated Transition
③Length unary vector
Data Curator
Aggregation and Estimation
④Length Distribution $\mathcal{L}$
⑤Aggregated Transitions $\mathcal{M}$
Adaptive Synthesis
Synthetic Trajectories

Figure 1: The LDPTrace **framework.**

4.2 Problem Statement

Consider there is a collection of trajectories generated by mobile travelers on the roads, denoted by $\mathcal{T}$. People are unwilling to report their own trajectories to the untrusted data curator due to privacy concerns. Thus, we want to build a generative model over $\mathcal{T}$ by extracting key moving patterns from $\mathcal{T}$ with provable guarantees on individual privacy. We then employ the generative model to output a set of synthesized trajectories, denoted by $\mathcal{T}_{syn}$. The synthesized trajectories $\mathcal{T}_{syn}$ should collectively retain a high resemblance to the real trajectories $\mathcal{T}$, so that $\mathcal{T}_{syn}$ has many useful statistical and spatial features in common with $\mathcal{T}$. Finally, the synthetic trajectories $\mathcal{T}_{syn}$ should be robust against various location-based attacks, in order to strengthen privacy by maximizing attackers' probability of errors in identifying the true traces of users.

5 OUR METHODOLOGY

In this section, we first give an overview of our proposed method LDPTrace, and discuss the necessity and method for trajectory discretization. Then, we detail the three crucial components of LDPTrace, which are estimated from collaborative mobility patterns and satisfy the strict privacy guarantee of LDP. Next, we detail the adaptive generation process by building a probabilistic model with learned spatial patterns. Finally, we conduct thorough discussion to analyze the privacy and computational cost of LDPTrace, and present an approach to select a proper grid granularity.

5.1 Solution Overview

As illustrated in Figure 1, both user and data curator participate in the process of LDPTrace. On the user side, LDPTrace first discretizes the trajectory into a sequence of adjacent cells, and exacts three key features of the trajectory by perturbing the trajectory length, intra-trajectory mobility, and beginning/terminated transitions. On the data curator side, LDPTrace collects the perturbed information (*i.e.,* three features mentioned earlier) from all users, and builds a probabilistic model by estimating the key mobility features (*i.e.,* length distribution and aggregated transitions). Finally, LDPTrace performs the synthesis to sample start point and subsequent transitions from the learned patterns, with its termination controlled by the estimated trajectory length and sampled end point.

Since the synopsis of LDPTrace consists of three features learned from user's trajectory, the security budget ϵ is divided into three sub-budgets ϵ_1, ϵ_2, and ϵ_3, such that $\sum_{i=1}^{3} \epsilon_i = \epsilon$. Detailed analysis on the privacy is presented in Section 5.7.

5.2 Geospatial Discretization

The representation of a trajectory as a sequence of points in a continuous two-dimensional domain, such as latitude-longitude coordinates, can be challenging to model. To overcome this, one common approach is to discretize the geographic space into grid cells. This is achieved by partitioning the entire space into equal-sized cells using a grid granularity of N.

Each trajectory is then transformed into a sequence of enumerable cells, $T = \{C_1, C_2, \cdots, C_{|T|}\}$, where $T[i]$ refers to the i-th cell visited by the trajectory T and $|T|$ is the length of the trajectory in grid cells. The choice of N affects the size of each grid cell, and thus the granularity of the discretized trajectory.

When N is small, the space is partitioned into a limited number of grid cells, and each cell covers a large spatial region. Many points in the original trajectory will be represented by one single cell, and thus, the mobility patterns captured by the discretized trajectory would be very general and uninformative. On the other hand, if N is big, each grid cell covers a very small region, resulting in the risk of having many empty cells that have not been passed by any trajectories. Perturbing these empty cells leads to high noise and inefficiency. While some literature [23, 35] has provided guidelines for selecting a proper grid granularity, they all rely on global statistics like spatial density, which are typically not available in the local setting where a user only has access to his/her own trajectories. Instead of estimating the global statistics which costs extra privacy budget, we theoretically analyze the trade-off between estimation errors and the granularity of grid cells, and propose a novel method to choose N in the local setting without consuming any budget, as to be detailed in Section 5.9.

5.3 Trajectory Length Distribution

The first component in LDPTrace is the estimated length distribution of trajectories, which is an indispensable character of trajectory synthesis since it indicates the probabilistic travel distance of users' trace, and serves as the deterministic constraint to terminate the synthesis process. However, estimating the length distribution is a non-trivial task in the local setting since there is no aggregated statistics available. Instead, we aim to collect the length information from each user and approximate the distribution with frequency. Specifically, we first define the domain of lengths as $|C|$, under the assumption that the maximum travel distance of any trajectory is $|C|$. Note that the techniques proposed in this paper can take any maximum distance as an input.

On the user side (③ in Figure 1), for trajectory T with length m (*i.e.,* $|T| = m$), we encode it into a $|C|$-bit binary vector V, which sets only the m-th bit to 1 but all other bits to zero. Next, the binary vector V is perturbed individually and locally with budget ϵ_1 according to Equation (2), and the user only reports the noisy vector $\hat{V}$ to the untrusted data curator.

On the data curator side (④ in Figure 1), after collecting noisy vector $\hat{V}$ from different users, the curator estimates the frequency of each length value m by counting the non-zero entry of each vector and adjusting the total count of each length via the unbiased statistic using Equation (3). Finally, we view the length distribution as the categorical distribution $\mathcal{L}$, and the probability of length m is $Pr(m) = \tilde{g}(m)/\sum_{i=1}^{|C|} \tilde{g}(i)$, where $\tilde{g}(m)$ is the unbiased OUE estimator of the true frequency of length m.

Error analysis. We analyze the error of estimated length distribution $\mathcal{L}$ by using the Theorem A.1 in Appendix A. $Pr(m)$ is the unbiased estimator of length probabilities:

$$\mathrm{E}^*[Pr(m)] = f_m / \sum_{i=1}^{|C|} f_i, \tag{5}$$

where f_i is the frequency of trajectory length i in the whole trajectory set $\mathcal{T}$. Thus, the estimated length distribution $\mathcal{L}$ approximates the true distribution with the error:

$$\mathrm{Error}(\mathcal{L}) = \sum_{i=1}^{|C|} (\frac{f_i}{|\mathcal{T}|})^2 [\frac{\sigma^2}{(f_i)^2} - \frac{2\sigma^2}{f_i|\mathcal{T}|} + \frac{8\sigma^2}{|\mathcal{T}|^2}], \tag{6}$$

where $\sigma^2 = |\mathcal{T}| \frac{4e^{\epsilon_1}}{(e^{\epsilon_1}-1)^2}$ is the variance of OUE estimator in Equation (4), and $|\mathcal{T}|$ is the number of trajectories. Given the fixed statistics of trajectories (e.g., size and frequency of each length), the error of estimated length distribution $\mathcal{L}$ is on the order of $e^{-\epsilon}$, *i.e.,* a higher budget could help reduce the length error.

5.4 Intra-Trajectory Mobility Model

To generate high utility and realistic synthetic trajectories, it is necessary to mimic actual intra-trajectory mobility (*i.e.,* the transition from $T[i]$ to $T[i+1]$). We achieve this by building a Markov chain for mobility modelling. A first-order Markov chain asserts that a location $T[l+1]$ in a trajectory depends only on its previous location $T[l]$ instead of all previous locations:

$$Pr(T[l+1] = C \mid T[1]...T[l]) = Pr(T[l+1] = C \mid T[l]), \tag{7}$$

which simplifies the complex sequential dependency $T[1]...T[l]$ with closest grid $T[l]$ for $T[l+1]$. We assume that the main mobility patterns of trajectories can be captured by the Markov chain, which is a collection of such probability $Pr(T[l+1] = C_{next}|T[l])$. Considering the continuity of trajectories, we interpolate each trajectory to make sure that any two consecutive points in a trajectory corresponding to two adjacent grid cells, *i.e.,* $T[i]$ is adjacent to $T[i+1]$. Accordingly, we can define the transition state s_{ij} from grid cell C_i to another grid cell C_j as follows:

$$Pr(s_{ij}) = \begin{cases} Pr(T[l+1] = C_j \mid T[l] = C_i), & \text{if } C_j \in \mathcal{N}_{C_i} \\ 0, & \text{otherwise,} \end{cases} \tag{8}$$

where $Pr(s_{ij})$ is the transition probability from C_i to C_j, $\mathcal{N}_{C_i}$ captures the set of adjacent cells of C_i, and the transition model $\mathcal{S}$ is a set of all transition probabilities aggregated from trajectories.

Similar to the estimation of length distribution, we aim to collect the transition information from each user (① in Figure 1) and then aggregate them on the data curator side to estimate the overall transition states. Thus, each user's trajectory T can be represented as a sequence of transition states S_T with length $|T| - 1$. On the user side, for each state in S_T, we also opt for OUE to encode it into a $|\mathcal{S}|$-bit binary vector and then report the perturbed noisy version, where $|\mathcal{S}|$ is the domain of possible states. Since we only consider transition between two adjacent grids and each grid cell has up to 8 adjacent cells, we have $|\mathcal{S}| \approx 8|C|$.

However, if we directly perturb each trajectory's transition by averaging the privacy budget ϵ_2 with its own length $|S_T|$, it is impossible to estimate the unbiased frequency of each transition state s from all trajectories since OUE protocol demands the budget used to be same across different users/trajectories. A straightforward solution to this problem is to equally divide budget ϵ_2 across $|C|$, the maximum length of T, meaning that each trajectory is allowed to have up to $|C|$ transitions, and each transition is assigned a budget of $\epsilon_2/|C|$. Nevertheless, this approach suffers from huge waste of budget as the length of most trajectories could be far shorter than $|C|$. Therefore, we set the number of transitions as the k-th quantile of estimated length distribution L_k, so that we only upload L_k transition states for all trajectories, and omit the remaining if the number of transitions $|S_T|$ is longer than L_k. Here, k is a hyper-parameter to balance the noise error and bias error. On one hand, we would like k to be large, *i.e.,* every transition in trajectory data $\mathcal{T}$ can be captured for modelling mobility patterns, and the amount of noise added to state s can be measured by:

$$N(s, \epsilon_2, L_k) = \mathrm{Var}^*[Pr(s), \epsilon_2/L_k], \tag{9}$$

where $\mathrm{Var}^*(Pr(s), \epsilon_2/L_k)$ is the approximated variance of transition probability $Pr(s)$ with budget ϵ_2/L_k, which can be calculated via Theorem A.1. On the other hand, omitting the transitions for trajectories with length longer than L_k will introduce bias into the model (*i.e.,* some of the transitions are not accurately captured), and the bias of transition s is expressed as $(1-k)^2 \cdot f_s^2$, where f_s is the frequency of transition s. The sum of the noise and bias terms defines the total error of $\mathcal{S}$:

$$\mathrm{Error}(\mathcal{S}, \epsilon_2, L_k) = \sum\nolimits_{s\in\mathcal{S}} [N(s, \epsilon_2, L_k) + (1-k)^2 \cdot f_s^2]. \tag{10}$$

If k is large, more transition information would be reported, which can effectively reduce the bias error. However, the budget of each transition will be small, which results in large noise for estimated frequency and hence more noise error. On the other hand, if k is small, many useful transition states are omitted, leading to insufficiency to capture the global moving patterns from crowds. Therefore, the optimal k is chosen to optimize the total error of $\mathcal{S}$:

$$k^* = \mathrm{argmin}_{0<k\le 1}\ \mathrm{Error}(\mathcal{S}, \epsilon_2, L_k). \tag{11}$$

However, due to the unavailability of true frequency of transitions in the local setting, it is impossible to directly derive the optimal k. We analyze the impact of k on utility in Section 6.5.3.

Discussion. In LDPTrace, we model the intra-trajectory moving patterns with the first-order Markov chain (*aka.* transition states), while we can naturally apply a higher-order Markov chain. However, since the size of Markov chain grows quickly *w.r.t.* the order, the noise introduced by the randomness mechanism in LDP can drown out the real signal due to the limited budget. Empirical results in Section 6.3 show that the first-order Markov chain is sufficient to model the intra-mobility, and generates authentic trajectories.

5.5 Beginning/Terminated Transitions

Real-world trajectory databases often consist of various trips, such as taxi trips, home-work commutes, *etc.* These trips usually exhibit special start point and end point, which reveal the important spatial semantic of trajectories: pickups, home/work places, destinations, *etc.* Besides, the start/end points are also useful to guide the random walk during synthesis. Although we have a mobility model $\mathcal{S}$ for intra-trajectory movement, we still need a start state and an end state to specify the two endpoints of generated trajectory. We can naively assume a uniform distribution to randomly choose a cell from C as a start point to perform random walk and to terminate the synthesis process when the trajectory reaches the assigned length, but it contradicts with a well-known fact that the distribution of start/end points of real trajectories is often heavily skewed [23]. For example, many trips might start from or end at homes, while residential areas are not uniformly distributed in a city. Thus, naive solution mentioned above, even though simple, will incur bogus synthetic trajectories that jeopardize utility and authenticity.

To model the distribution of start/end points in trajectories, we add two special cells, namely, *virtual* start point C_a and *virtual* end point C_b, which are connected to all the geographic cells in C. C_a/C_b serves as start/end point of any trajectories T to record T's beginning/terminated state. To be more specific, the beginning transition state A_i denotes that trajectory T begins with C_i (from virtual point C_a to cell C_i), and the terminated transition state B_j means that trajectory T stops with C_j (from C_j to virtual end point C_b). On the user side, we also utilize the OUE protocol to report a noisy version of the beginning/terminated transition states with budget ϵ_3 on the user side, which is separated from the intra-trajectory transitions during reporting to reduce their noise since they only occur once per trajectory (② in Figure 1). On the data curator side, we combine the intra-transitions with beginning/terminated transitions to form the aggregated mobility model $\mathcal{M}$ (⑤ in Figure 1). Specifically, the transition probability from C_i to C_j can be calculated as:

$$Pr(M_{ij}) = \frac{\tilde{g}(M_{ij})}{\sum_{r\in\mathcal{N}^*_{C_i}} \tilde{g}(M_{ir})}, \tag{12}$$

where M_{ij} represents the transition state from C_i to C_j. If C_i is the virtual start point C_a, then $M_{ij} = A_j$; if C_j is the virtual end point C_b, then $M_{ij} = B_i$; otherwise $M_{ij} = s_{ij}$. The aggregated neighbor $\mathcal{N}^*$ is defined as follows:

$$\mathcal{N}^*_{C_i} = \begin{cases} \mathcal{N}_{C_i} \cup \{C_b\}, & \text{if } C_i \in C \\ C, & \text{otherwise } (\textit{i.e.,}\ C_i \in \{C_a, C_b\}). \end{cases} \tag{13}$$

Hence, the intra-transitions and the beginning/terminated transitions are seamlessly integrated into the aggregated mobility model $\mathcal{M}$, which can denote the overall moving patterns of trajectories.

5.6 Trajectory Synthesis

LDPTrace builds a probabilistic model for private synopsis, which consists of length distribution $\mathcal{L}$ and aggregated mobility model $\mathcal{M}$. Accordingly, the synthesis algorithm can be described in three steps, as depicted in Algorithm 1. First, it determines the length L of trajectory by sampling from the length distribution $\mathcal{L}$ (line 1). Second, it initializes T_{syn} by assigning its starting point to the cell sampled from $\mathcal{N}^*_{C_a}$ with probability proportional to $\mathcal{M}$ (lines 2-3).

Algorithm 1: Trajectory synthesis

```
Input: a grid $C$, a length distribution $\mathcal{L}$, an aggregated mobility
       model $\mathcal{M}$, virtual start point/end point $C_a/C_b$
Output: a candidate synthetic trajectory $T_{\text{syn}}$
1  trajectory length $L \leftarrow$ sample($\mathcal{L}$)
2  sample $C_{start} \leftarrow$ from $\mathcal{N}^*_{C_a}$ with probability proportional to $\mathcal{M}$
3  initialize $T_{syn}$: $T_{syn}[1] \leftarrow C_{start}$
4  for $l \leftarrow 2$ to $L$ do
5      reweight the terminated condition according to Equation (14)
6      sample $C_{next}$ from $\mathcal{N}^*_{T_{syn}[l-1]}$ with probability proportional
         to $\mathcal{M}$
7      if $C_{next} = C_b$ then
8          return $T_{syn}$
9      else
10         set $T_{syn}[l] \leftarrow C_{next}$
11 return $T_{syn}$
```

Third, it extends T_{syn} by including a new cell C_{next} based on its current location. It repeats the extension process until C_{next} is the virtual end point C_b or the length of T_{syn} reaches L, whichever is earlier (lines 4-10). Although the above synthesis scheme makes full use of the estimated patterns, a notable weakness is that the generated trajectory T_{syn} might be much shorter than L if it reaches C_b in the early part of its extension. Since the distribution of trajectory terminated points is heavily skewed (*i.e.,* some points are more likely to be the destination than others), directly sampling the end point according to the transition probability would lead to the inauthenticity and uninformativeness of synthetic trajectories. Thus, we re-weight the terminated probability $Pr(M_{ib})$ by taking the current length l into consideration:

$$\tilde{Pr}(M_{ib}) = (\alpha + \beta l) \times Pr(M_{ib}), \tag{14}$$

where $\tilde{Pr}(M_{ib})$ is the adjusted termination probability from current location C_i to virtual end point C_b, and α and β are two hyper-parameters to control the influence of length, *i.e.,* large α and β implies that the synthesis process tends to be stopped even when the length is small, and small α and β allows the model to synthesize trajectories with more points. The final adaptive synthesis algorithm is presented in Algorithm 1.

To obtain an authentic trajectory database with good utilities, we run the synthesis algorithm multiple times until the number of synthetic trajectories reaches the number of trajectories in $\mathcal{T}$. Then, this set of trajectories $\mathcal{T}_{syn} = \cup T_{syn}$ becomes the substitution of the real trajectory set $\mathcal{T}$ for various spatial analysis tasks without sacrificing users' privacy.

5.7 Privacy Analysis

We now analyze the privacy of the synthesis solution through a sketch proof, and discuss the budget allocation strategy.

Theorem 5.1. *The while process of* LDPTrace *satisfies ϵ-LDP.*

Proof of Theorem 5.1. LDPTrace treats ϵ as the total privacy budget, and distributes it to three sub-budgets (one for each key feature in the synopsis) such that $\sum_{i=1}^{3} \epsilon_i = \epsilon$. Perturbing and reporting a feature (length, intra-transition, and start/end transition) consumes the ϵ_i allocated to it, and thus, depleting the total ϵ after the reporting phase is complete, based on the sequential composition property. Besides, during the intra-mobility modelling, we divide the budget ϵ_2 for each transition state of trajectory, which also satisfies the sequential composition property. Then, any additional data-independent operations like aggregation and sampling on the perturbed statistics are viewed as post-processing. As a result, LDPTrace remains ϵ-locally differentially private. □

The budget allocation of ϵ can be configured by LDPTrace automatically or according to the demands of real applications. The current implementation of LDPTrace comes with a default budget distribution, which was empirically determined to yield high average utility: $\epsilon_1 = \epsilon/10$ for length distribution, $\epsilon_2 + \epsilon_3 = 9\epsilon/10$ for transitions, where we allocate equal budget to each transition (*i.e.,* intra-transitions and beginning/terminated transitions). We find that the estimation of length distribution only requires a small budget, while the transitions of trajectories consume the major budget since they determine the synthesis process. Although the above strategy is beneficial from the overall utility maximization perspective, the budget allocation is very flexible, and it can be easily assigned based on the specific needs of various applications (*e.g.,* pick-up identification may need more budget for beginning/terminated transition estimation).

5.8 Computational Cost

We also discuss the computational cost of the proposed LDPTrace, which highlights how it is far more practical than other alternatives. We analyze the computational cost from both user side and data curator side in the following.

Computation on users. We assume that each user keeps one trajectory of his/her own. Since each perturbation is done locally on individual device, we only analyze the computational cost for each user/trajectory. First, the cost of perturbing a binary vector of length is $O(|C|)$, where $|C|$ is the number of grids. Second, the computational complexity of intra-transition modelling is $O(|T||C|)$, where $|T|$ is the trajectory length. Finally, the cost of adding noise to the beginning/terminated transitions is also $O(|C|)$. Thus, the overall computation on the user side is $O(|T||C|)$. As all the perturbation can be implemented with bit operations, the computation is almost negligible and also affordable for any location-aware devices.

Computation on data curator. After collecting all the perturbed information from users, data curator first estimates the unbiased frequency with OUE, and calculates the quantile of length, at the cost of $O(|\mathcal{T}||C|)$, where $|\mathcal{T}|$ is the number of trajectories/users. Besides, the computational complexity of pattern estimations (*i.e.,* length distribution, mobility patterns, and beginning/terminated states) is $O(|C|)$. Moreover, the synthesis algorithm would cost $O(|\mathcal{T}|L)$ to synthesize the same number of trajectories as real trajectories $\mathcal{T}$, where L denotes the mean of length distribution $\mathcal{L}$. Empirical experiments in Section 6.4 suggest that the synthesis process dominates the running time of LDPTrace, and LDPTrace significantly outperforms the competitor by more than two orders of magnitude. Moreover, the data curator typically has much more computing power than the mobile devices typically used by end-users, indicating that LDPTrace is more flexible than locally point-based privacy mechanisms (*i.e.,* NGRAM).

5.9 Selecting the Grid Granularity N

Finally, we propose a guideline for selecting proper grid granularity N in the local setting. As mentioned before, grid granularity N is an important hyperparameter for trajectory representation. We follow [35] and analyze the effect of N by considering the range query on trajectories, which is the most common spatial task used by various real-world applications. The task of a range query is to retrieve all locations of trajectories that fall within the query region. Given a rectangular shape query region, let r be the portion of the entire space covered by the query region. There are mainly two sources of errors in LDPTrace when performing a range query. The first is *estimation error*. In our framework, noise is added to each transition locally, and the error of estimated transition probability is in the order of $\sqrt{ne^{\epsilon}/(e^{\epsilon}-1)^2}$. Since the query covers about rN^2 cells, the total error introduced by perturbation in this query is in the order of $N\sqrt{nre^{\epsilon}/(e^{\epsilon}-1)^2}$. The second is *non-uniformity error* that is proportional to the number of data points in the trajectories that fall on the boundary of the query region [35]. For a query region that covers r portion of the entire space (*i.e.,* grid), the length of each side is proportional to $\sqrt{r}$ of the domain length, and thus, the number of cells overlapped with the query's boundary is in the order of $N\sqrt{r}$, and the total number of points fallen on the boundary is in the order of $N_p/N^2 \times N\sqrt{r} = \sqrt{r}N_p/N$, where N_p represents the total number of points in all synthetic trajectories. The goal is to minimize the sum of two errors:

$$\text{minimize } N\sqrt{\frac{nre^{\epsilon'}}{(e^{\epsilon'}-1)^2}} + \frac{\sqrt{r}N_p}{N}, \tag{15}$$

where $\epsilon' = \epsilon_2/L$ is the budget for each transition. Since we cannot obtain trajectory length L before descretization, we replace it with the geographic distance, which is on the order of $L_{\mathbb{R}}/f$, where $L_{\mathbb{R}}$ is the average number of points in trajectory, and f is the sampling ratio of the device. Besides, we also use the number of points on real trajectories $|\mathcal{T}|L_{\mathbb{R}}$ to approximate N_p. Finally, by minimizing Equation (15), N should be set as follows:

$$N = \lambda \cdot \sqrt[4]{\frac{|\mathcal{T}|L_{\mathbb{R}}(e^{\epsilon f/L_{\mathbb{R}}}-1)^2}{e^{\epsilon f/L_{\mathbb{R}}}}}, \tag{16}$$

where λ is the hyperparameter which depends on the uniformity of the points distribution in the dataset. Note that all the parameters are easily obtained from the statistics of trajectory data (such as sampling ratio, average point count, and the data size), and we also evaluate the effectiveness of this guideline in Section 6.5.2.

6 EXPERIMENTAL EVALUATION

In this section, we first introduce the detailed experimental setup. Next, we conduct experiments on utility as well as efficiency to illustrate the superiority of LDPTrace. Then, we conduct insight studies to evaluate the impact of each component in LDPTrace. Finally, we evaluate the scalability of LDPTrace and its attack-resilient ability to various real-word location-based attacks.

Table 2: Statistics of the datasets used in our experiments.

Dataset	Size	Average Length	Sampling Interval
Oldenburg	500,000	69.75	15.6 sec
Porto	361,591	34.13	15 sec
Hangzhou	348,144	125.02	5 sec
Campus	1,000,000	35.98	25 sec

6.1 Experimental Setup

6.1.1 ***Datasets****.* We conduct our experiments on four benchmark trajectory datasets. Table 2 summarizes the overall statistics.

- **Oldenburg**[1] is a synthetic dataset simulated by Brinkhoff's network-based moving objects generator. We generate 500,000 trajectories based on the map of Oldenburg city.
- **Porto**[2] contains taxi traces over 8 months in the city of Porto. We extract 361,591 trajectories from the central areas.
- **Hangzhou** is a private trajectory database which consists of the trace of taxis in Hangzhou city.
- **Campus**[3] contains 434 buildings of British Columbia campus. We follow [14] to generate 1 million trajectories based on the campus buildings.

6.1.2 ***Baseline****.* We compare our method with NGRAM [14], which is the state-of-the-art (and the only) private trajectory publication method that satisfies rigorous LDP. Different from our approach, NGRAM is a point-based perturbation model which directly resembles user's trajectory in the local setting. For fair comparison, we discard the auxiliary temporal and POI information, and only leverage physical distance to ensure the closeness between original and sampled n-grams in geospatial space.

6.1.3 ***Experimental Settings****.* Based on the guideline described in Section 5.9, the grid granularity parameter N is set to 6 for Oldenburg, Porto, and Campus dataset, and 8 for Hangzhou dataset. As for the k quantile of estimated length distribution, we set it to 0.9 for all the experiments. We generate synthetic database $\mathcal{T}_{syn}$ with cardinality $|\mathcal{T}_{syn}| = |\mathcal{T}|$ for utility comparison. We set $\alpha = 0.3$ and $\beta = 0.2$ for the reweighting function defined in Equation (14). We set $\lambda = 2.5$ for selecting the grid granularity. As for the query region, we set r as the $1/9$ proportion of the entire space. Our experiments are conducted on a computer with Intel Xeon 2.1GHz CPU and 32 GB main memory.

6.2 Utility Metrics

To comprehensively quantify the utility of the synthetic trajectories, we adopt various utility metrics from three categories, including global level, trajectory level, and semantic level.

Global level utility measures the spatial patterns of trajectories in a global view, which serves as a building block of various spatial applications like range query and traffic forecasting. We use the following metrics for evaluation:

[1]http://iapg.jade-hs.de/personen/brinkhoff/generator/
[2]http://www.geolink.pt/ecmlpkdd2015-challenge/dataset.html
[3]https://github.com/UBCGeodata/ubc-geospatial-opendata

Table 3: Utility performance comparison. The best result in each category is shown in bold. For Kendall-tau and FP F1 Similarity, higher values are better. For remaining metrics, lower values are better.

		Oldenburg			Porto			Hangzhou			Campus		
		$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 1.5$
Density Error	NGRAM	0.0323	0.0301	0.0274	0.3311	0.3218	0.3049	0.1930	0.1847	0.1831	0.1184	0.1172	0.1105
	LDPTrace	**0.0094**	**0.0077**	**0.0085**	**0.0090**	**0.0081**	**0.0069**	**0.0210**	**0.0194**	**0.0193**	**0.0042**	**0.0043**	**0.0037**
Query Error	NGRAM	0.3362	0.3321	0.3206	0.8171	0.8102	0.8055	0.6454	0.6411	0.6332	0.7605	0.7579	0.7611
	LDPTrace	**0.2691**	**0.2595**	**0.2522**	**0.3520**	**0.3312**	**0.3011**	**0.2933**	**0.2814**	**0.2792**	**0.2014**	**0.2011**	**0.1980**
Hotspot Query Error	NGRAM	0.2972	0.2972	0.2972	1.0000	1.0000	1.0000	0.1529	0.1529	0.1529	0.7001	0.7001	0.7001
	LDPTrace	**0.0593**	**0.0593**	**0.0530**	**0.0000**	**0.0000**	**0.0000**	**0.0131**	**0.0131**	**0.0131**	**0.0013**	**0.0013**	**0.0000**
Kendall-tau	NGRAM	0.7479	0.7512	0.7498	0.5442	0.5791	0.5934	0.6164	0.6242	0.6623	0.3794	0.3841	0.3852
	LDPTrace	**0.8874**	**0.8944**	**0.8942**	**0.6672**	**0.7114**	**0.7581**	**0.6782**	**0.7044**	**0.7174**	**0.8603**	**0.8574**	**0.8667**
Trip Error	NGRAM	0.1251	0.1231	0.1230	0.5122	0.5043	0.4979	0.4493	0.4490	0.4421	0.3052	0.3024	0.2995
	LDPTrace	**0.0697**	**0.0683**	**0.0672**	**0.0762**	**0.0778**	**0.0771**	**0.0531**	**0.0513**	**0.0504**	**0.0744**	**0.0740**	**0.0729**
Length Error	NGRAM	0.1142	0.1134	0.1112	0.1812	0.1823	0.1791	0.02922	0.02736	0.02607	0.1102	0.1041	0.1045
	LDPTrace	**0.0373**	**0.0370**	**0.0379**	**0.0410**	**0.0399**	**0.0392**	**0.0036**	**0.0037**	**0.0032**	**0.0984**	**0.0981**	**0.0987**
Diameter Error	NGRAM	0.1244	0.1242	0.1221	0.2201	0.2180	0.2174	0.2013	0.1989	0.1972	0.0682	0.0663	0.0658
	LDPTrace	**0.0568**	**0.0570**	**0.0562**	**0.0354**	**0.0340**	**0.0331**	**0.0572**	**0.0569**	**0.0563**	**0.0591**	**0.0589**	**0.0587**
Pattern F1	NGRAM	0.33	0.33	0.34	0.19	0.18	0.19	0.24	0.26	0.26	0.32	0.32	0.33
	LDPTrace	**0.69**	**0.69**	**0.69**	**0.67**	**0.63**	**0.65**	**0.80**	**0.80**	**0.80**	**0.71**	**0.71**	**0.72**
Pattern Error	NGRAM	0.8033	0.8004	0.7982	0.9206	0.9232	0.9181	0.8782	0.8725	0.8754	0.7992	0.7912	0.7789
	LDPTrace	**0.5693**	**0.5632**	**0.5594**	**0.6498**	**0.6687**	**0.6692**	**0.4593**	**0.4552**	**0.4554**	**0.5502**	**0.5508**	**0.5501**

- **Density error** evaluates the density difference between synthetic trajectory set $\mathcal{T}_{syn}$ and real trajectory set $\mathcal{T}$.

$$Density\ Error = JSD\big(\mathcal{D}(\mathcal{T}), \mathcal{D}(\mathcal{T}_{syn})\big), \tag{17}$$

  where $\mathcal{D}(\mathcal{P})$ denotes the grid density distribution in a given set $\mathcal{P}$, and $JSD(\cdot)$ represents the Jenson-Shannon divergence between two distributions.
- **Query error** is a popular measure for evaluating data synthesis algorithms ranging from tabular data to graph and location data [8, 10, 30]. We consider range queries of trajectories in a random spatial region R, *i.e.,* $Q(\mathcal{P})$ returns the number of points in any trajectory of a specified set $\mathcal{P}$ that are within the spatial region R.

$$Query\ Error = \frac{|Q(\mathcal{T}) - Q(\mathcal{T}_{syn})|}{max\{Q(\mathcal{T}), z)\}}, \tag{18}$$

  where z is the sanity bound to weaken the influence of queries that return very small counts. We set the sanity bound z to $\frac{\sum_{\mathcal{T}} |T|}{100}$, and report the average result of 200 random queries.
- **Hotspot query error** measures the ability of preserving spatial hotspots. Specifically, we choose top-n_h mostly visited cells in $\mathcal{T}$ and $\mathcal{T}_{syn}$ as the hotspots set H and H_{syn}, respectively.

$$HQE = 1 - \frac{\sum_{C_i \in H_{syn}} rel(C_i)/\log(rank_{H_{syn}}(C_i) + 1)}{\sum_{j=1}^{n_h} 1/(j \cdot \log(j+1))}, \tag{19}$$

  where $rank_{H_{syn}}(C_i)$ is the position of C_i in H_{syn}, $rel(C_i)$ is the relativity score of C_i: when $C_i \in H$, $rel(C_i) = 1/rank_H(C_i)$, else $rel(C_i) = 0$. We set $n_h = 5$.
- **Kendall's tau coefficient** is for modelling the discrepancies in locations' popularity ranking [23]. Let $\mathcal{D}(C_i)$ be the density of cell C_i, and (C_i, C_j) be a *concordant pair* if and only if $\mathcal{D}(C_i) \geq \mathcal{D}(C_j)$ or $\mathcal{D}(C_i) \leq \mathcal{D}(C_j)$ holds both on $\mathcal{T}$ and $\mathcal{T}_{syn}$. Otherwise, it's a *discordant pair.*

$$Kendall\text{-}tau = \frac{N_c - N_d}{|C|(|C|-1)/2}, \tag{20}$$

  where N_c and N_d represent the number of concordant pairs and the number of discordant pairs respectively, and $|C|$ captures the total number of grid cells.

Trajectory level utility denotes the spatial features within each trajectory, which also fascinates a wild range of applications like origin-destination analysis and commutes studies. We employ the following three evaluation metrics:

- **Trip error** measures how well the correlations between trips' starting points and ending points are preserved [23]. Specifically, we calculate the probability distribution of start/end points in $\mathcal{T}$ and that in $\mathcal{T}_{syn}$, and utilize Jensen-Shannon divergence to measure their difference.
- **Length error** focuses on the difference between real and synthetic datasets in terms of trajectory lengths (*i.e.,* distance travelled by each trajectory), which calculates the total distance in a trajectory by adding up the Euclidean distance between consecutive points.
- **Diameter error** is defined as the difference of maximum distance frequency between real and synthetic datasets, where the maximum distance refers to the maximum Euclidean distance between two points in a trajectory [25, 28].

Since travel distance and diameter are continuous, we follow [23] to separate them into 20 equi-width buckets, and calculate the distribution of the obtained histogram. We then use Jenson-Shannon divergence to quantify the above errors.

Semantic level metrics. Apart from the aforementioned statistic metrics of trajectories' attributes, we also adopt two semantic level metrics to mine the mobility patterns that are hidden behind those statistics. A pattern P is defined as an ordered sequence of consecutive cells, and we select top-n most occurred patterns in $\mathcal{T}$ and $\mathcal{T}_{syn}$, denoted as pattern sets FP and FP_{syn} respectively, and calculate the following metrics:

Table 4: **Average runtime in seconds. We report the average running time per 1,000 trajectories of each component.**

	LDPTrace: **User side**			LDPTrace: **Curator side**			**Total**	NGRAM **Total**
	Length	**Intra-traj tran.**	**Beg./Ter. tran.**	**Preparation**	**Pattern est.**	**Synthesis**		
Oldenburg	0.030	0.192	0.104	0.010	0.001	0.219	**0.556**	183
Porto	0.028	0.148	0.092	0.015	0.001	0.179	**0.463**	132
Hangzhou	0.028	0.302	0.098	0.012	0.002	0.304	**0.746**	418
Campus	0.028	0.138	0.083	0.009	0.001	0.189	**0.448**	197

- **Pattern F1** evaluates the similarity between the selected most frequent pattern sets FP and FP_{syn}:

$$Pattern\,F1 = 2 \times \frac{Precision(FP, FP_{syn}) \times Recall(FP, FP_{syn})}{Precision(FP, FP_{syn}) + Recall(FP, FP_{syn})}. \quad (21)$$

- **Pattern Error** measures the relative difference between the number of pattern occurrences in each dataset:

$$Pattern\ Error = \frac{1}{|FP|} \sum_{P \in FP} \frac{|n^P - n^P_{syn}|}{n^P}, \quad (22)$$

  where n^P / n^P_{syn} is the number of occurrence of pattern P in the dataset FP/FP_{syn}, and we use top 100 frequent patterns (*i.e.,* $|FP| = 100$) for evaluation.

6.3 Utility Evaluation

In our first set of experiments, we compare the utility performance of LDPTrace and NGRAM with various privacy budgets ϵ. For each experiment, we perform the synthesis 5 times, and report the average results in Table 3. Generally speaking, LDPTrace outperforms NGRAM in all utility metrics across all datasets, which well demonstrates the robustness and strong utility-preserving ability of LDPTrace. Based on the in-depth analysis of the results, we have made the following detailed observations.

- We first analyze the performance *w.r.t.* different types of utility metrics. Since NGRAM fails to take the global features of trajectories into consideration, it suffers from severe performance degradation on the global level utility. Especially, the density error in LDPTrace is *three times* smaller than that in NGRAM, which implies that the trajectories generated by NGRAM fail to preserve the geospatial density due to the point-based perturbation. Also, our method shows strong performance in both the trajectory level and the semantic level utilities. We contribute the improvement to the mobility modelling and adaptive synthesis of LDPTrace: (1) By collecting transitions from both intra-trajectory and start/end points, LDPTrace is able to synthesize trajectories whose mobility patterns (*e.g.,* moving directions) assemble users' real traces. In contrast, NGRAM aims to preserve the local *n*-gram closeness when perturbing, but ignores the critical sequential features of trajectories in a global view. (2) Benefited from the adaptive synthesis algorithm, LDPTrace can generate authentic trajectories with proper length to maintain the relationship between start points and end points, while the point-based NGRAM fails to capture the long-term dependence between beginning and terminated transitions.
- Next, we evaluate the effectiveness *w.r.t.* different datasets. We find that the performance of our framework is robust, which keeps good utility on both synthetic and real-world datasets. However, the performance of NGRAM varies across datasets. Specifically, it achieves competitive results in terms of density error and trip error on Oldenburg, but the results of these metrics are much worse in the two real datasets. We argue that this is because the mobility patterns of real-world trajectories are much more complex than synthetic ones, and NGRAM is unable to capture these patterns with n-gram model.
- We also examine the effects *w.r.t.* different privacy budgets ϵ. It is worth mentioning that LDPTrace could achieve good performance even when the budget is small (*e.g.,* $\epsilon = 0.5$), which confirms the effectiveness of capturing trajectories' key patterns with only a few distributions. However, NGRAM relies on a large budget to achieve reasonable performance, meaning that it needs to relax the privacy guarantee for practical use. For privacy budget ϵ, the protection is acceptable as long as the budget is less than 2 [19, 36, 39, 47]. To align with other real-world deployments of LDP by Google [19] and Microsoft [15], we set $\epsilon = 1$ in the following experiments.

6.4 Efficiency Evaluation

Since efficiency is equally important as utility for real-world deployments, we conduct comprehensive experiments to evaluate the average running time of each component, which is detailed in Table 4. Generally speaking, LDPTrace is a very efficient privacy-preserving trajectory publication framework, which is more than *300 times* faster than NGRAM. The reason is that NGRAM suffers from time-consuming processes like solving linear programming problem and recursive reconstructions, not to mention the expensive cost of pre-processing for POIs and other external knowledge, which greatly hinders its use on mobile devices. We would like to highlight that this observation is consistent with the authors' claims in their paper [14], as they also report up to 4 seconds per generated trajectory.

As for the breakdown of time spent on each component, we find that the major computation at user side is the perturbation of intra-trajectory transitions because it has to perturb and upload multiple times to report the holistic transition states. Other one-time operations like trajectory length perturbation are much faster, consistent with the computational analysis presented in Section 5.8. On the data curator side, the main computation is incurred by the synthesis process, since it needs to generate location points sequentially according to the transition probability. Overall, LDPTrace is highly efficient and practical for real-world applications, and the time cost for privacy protection is nearly imperceptible for users.

6.5 Analysis of LDPTrace

As the pattern modelling is at the core of LDPTrace, we also conduct insight experiments to investigate its effectiveness, *i.e.,* how the

Table 5: Impact of beginning/terminated transitions. Best result is shown in bold. HQ Error denotes “Hotpot Query Error”.

Dataset	Model	Density Error	Query Error	HQ Error	Kendall-tau	Trip Error	Length Error	Diameter Error	Pattern F1	Pattern Error
Oldenburg	RandSyn	0.0569	0.7491	0.0724	0.6623	0.1779	0.0694	0.0611	0.58	**0.4313**
	CombTran	0.0168	0.2794	0.0593	0.8108	0.1052	0.0371	**0.0570**	0.67	0.5596
	NoAdapt	0.0081	0.3754	**0.0013**	0.8866	0.0971	0.0713	0.0932	**0.69**	0.6822
	LDPTrace	**0.0077**	**0.2595**	0.0593	**0.8944**	**0.0683**	**0.0370**	**0.0570**	**0.69**	0.5632
Porto	RandSyn	0.2687	7.4933	0.2583	0.2191	0.4372	0.1244	0.1460	0.39	0.6658
	CombTran	0.0243	0.6741	0.0464	0.5784	0.0932	0.0412	0.0351	0.59	0.6730
	NoAdapt	0.0098	0.3607	0.0464	0.6828	0.1052	0.0579	0.0631	**0.63**	0.7674
	LDPTrace	**0.0081**	**0.3312**	**0.0000**	**0.7114**	**0.0778**	**0.0399**	**0.0340**	**0.63**	0.6687
Hangzhou	RandSyn	0.1928	2.4842	0.0144	0.5252	0.4471	0.0549	0.1344	0.49	**0.3821**
	CombTran	0.0361	0.3426	0.0464	0.6492	0.0762	0.0040	0.0594	0.78	0.4890
	NoAdapt	0.0322	0.3641	**0.0131**	**0.7062**	0.0593	0.0041	0.0921	**0.80**	0.5866
	LDPTrace	**0.0194**	**0.2814**	**0.0131**	0.7044	**0.0513**	**0.0037**	**0.0569**	**0.80**	0.4552
Campus	RandSyn	0.0914	1.5203	0.3055	0.5587	0.2811	0.0982	0.0589	0.64	**0.4109**
	CombTran	0.0103	0.3025	0.0304	0.8539	0.0862	**0.0973**	0.0593	0.71	0.5780
	NoAdapt	0.0046	0.3766	0.0593	**0.8571**	**0.0502**	0.1496	0.0590	**0.71**	0.6892
	LDPTrace	**0.0043**	**0.2011**	**0.0013**	**0.8571**	0.0740	0.0981	**0.0587**	**0.71**	0.5508

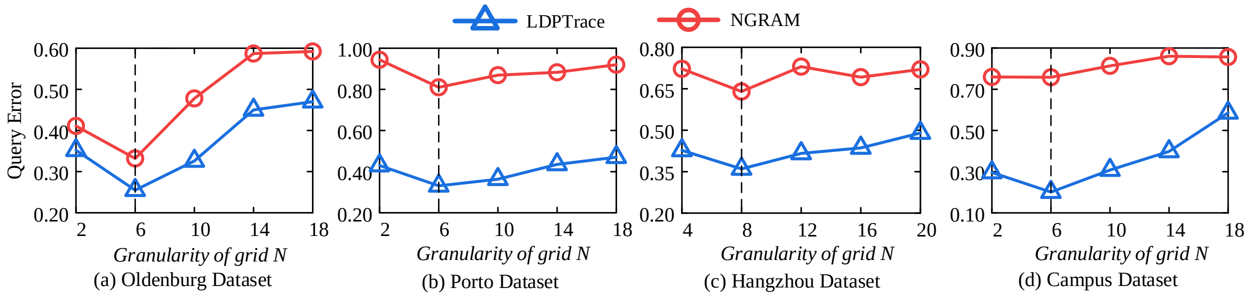

Figure 2: Impact of grid granularity N with optimal N values derived based on Equation (16) represented by dotted lines.

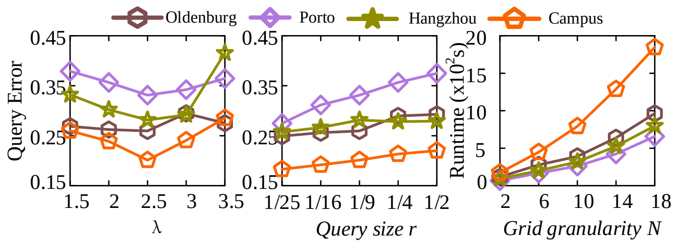

Figure 3: (a) Impact of λ. (b) Impact of query size r. (c) Impact of granularity N on total runtime (seconds).

presence of beginning/terminated transitions, grid granularity, and the adaptive synthesis algorithm affect our model.

6.5.1 ***Impact of beginning/terminated transitions.*** We first verify the effectiveness of the beginning/terminated transitions. To this end, we construct three variants of LDPTrace, including (1) RandSyn that discards the virtual start/end points from LDPTrace, and synthesizes trajectories according to intra-transition probabilities, (2) CombTran that combines the beginning/terminated transitions with intra-transitions, and perturbs/reports them as a whole, and (3) NoAdapt that removes the adaptive synthesis strategy (*i.e.,* Equation (14)). We summarize the results in Table 5.

Compared with the complete model LDPTrace, the absence of the beginning/terminated transitions (*i.e.,* RandSyn) dramatically degrades the utility, indicating the necessity of modelling the start/end point distribution. However, it is noticed that the pattern error of RandSyn remains small, since it measures the frequent intra-mobility patterns that are less irrelevant with endpoints. CombTran introduces unnecessary noise for estimating beginning/terminated transitions as they will be perturbed along the intra-transitions. Thus, it also results in some utility loss. Last but not the least, directly synthesizing trajectory without considering current length will make the synthetic trajectory too short to represent useful spatial patterns, incurring suboptimal performance.

6.5.2 ***Impact of grid granularity*** N. We analyze the influence of different grid granularity settings to empirically verify the effectiveness of our guideline for choosing N. Specifically, we utilize the query error metric to measure the impact of grid granularity. Figure 2 depicts the results, consistent with the analysis presented in Section 5.9. We can observe that the performance reaches near optimal at the estimated granularity which is derived from Equation (16), *i.e.,* 8 for Hangzhou dataset and 6 for the other three datasets. In addition, we see similar trend with respect to the performance of NGRAM. The reason behind is that our estimation only depends on the statistics of trajectory dataset, which is model-agnostic, and can be a good reference to choose the grid granularity for all locally private methods.

Besides, since the granularity will influence the trajectory length and the number of transition states, we also test the running time of LDPTrace under different N values in Figure 3(c). There is a clear increasing trend of runtime as N increases its value and each

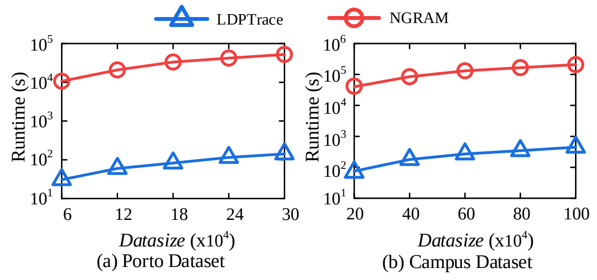

Figure 4: Scalability evaluation.

grid becomes more fine-grained. Consequently, a wrong choice of granularity will not only cause the utility degradation, but also result in larger computational complexity.

6.5.3 ***Impact of quantile*** *k.* For space reason, we only choose one representative utility metric each from global, trajectory, and semantic levels (*i.e.,* query error, diameter error, and pattern F1) to explore the influence of quantile k. In general, the selection of trajectory length has different impacts on different utilities. When the quantile k is small, more transitions will be truncated, and a large bias will be introduced to intra-transition modelling, leading to a large query error. On the other hand, when k becomes larger, the budget of each reported transition is smaller, and thus, the added noise increases and the synthetic trajectories may be unreliable, which results in a larger diameter error. In conclusion, we choose $k = 0.9$, since it achieves a good trade-off among different utilities.

The choice of k also impacts the efficiency of LDPTrace. As illustrated in the last subfigure of Figure 5, the running time grows plainly with the growth of k. It is consistent with our expectation because a larger k requires a longer amount of time for perturbation and reporting. However, the magnitude of the time increase is insignificant, because our algorithm is super efficient. Hence, the influence of more perturbations is almost negligible.

6.5.4 Impact of λ and query size r. We conduct experiments on different λ and query size r to further analyze the effectiveness of the granularity selection method. Recall that λ is the hyperparameter which depends on the uniformity of the points distribution in the dataset. As shown in Figure 3(a), we find that $\lambda = 2.5$ achieves good performance across different datasets, since it reaches good balance between non-uniformity error and transition estimation error.

As shown in Figure 3(b), the query error increases plainly when the query size r becomes larger. There are two competing effects when increasing r: on the one hand, the error from more grid cells is aggregated; on the other hand, each query is less affected by noise, since actual counts are larger. The first effect is stronger, so the overall error increases with query size. The results are consistent with prior studies [47].

6.6 Scalability

We also study the scalability of LDPTrace by varying the cardinality of the trajectory datasets. We observe similar trends in all four datasets. Due to space limitation, We only report the total running time under Porto and Campus datasets in Figure 4. As observed, LDPTrace is consistently faster than NGRAM in all different dataset scales, and has *two orders of magnitude* improvement. LDPTrace also has stable performance, and the processing time will not grow sharply with the growth of the dataset size. For NGRAM, it takes more than two days in processing all the trajectories when the dataset is at the scale of millions, while LDPTrace can finish the whole process in less than 10 minutes. Therefore, LDPTrace is suitable for large-scale deployment with little computational cost.

6.7 Comparison with unmodified NGRAM

To further demonstrate the superiority of LDPTrace, we also compare it against the unmodified NGRAM with external knowledge. Specifically, we use the same Campus dataset described in [14], and carefully attach the auxiliary knowledge (POIs, temporal information, hierarchical category) to the original trajectories. The results of unmodified NGRAM are reported in Table 6. As observed, LDPTrace outperforms both NGRAM and unmodified NGRAM for all utility metrics by a large margin. Although unmodified NGRAM benefits from the deterministic constraints of external knowledge to ensure the semantic similarity between original and perturbed locations, it still fails to capture the moving patterns properly.

6.8 Attack Resilience

In our last set of experiments, we investigate LDPTrace's resistance to two common attacks, which are defined below.

Re-identification Attack. Suppose the attackers are able to obtain some spatial locations of a user, *e.g.,* users can be easily tracked when they enter some public sensitive zones like train stations and shopping malls. With this subtrajectory T_a *w.r.t.* a specific user u as external knowledge, the attackers aim to identify the individual u in the published dataset $\mathcal{T}_{syn}$, and acquire information about the attacked user u's whole trajectory. The defense goal is to ensure that there are at least κ traces in $\mathcal{T}_{syn}$ having their similarity distances to T_a below a given threshold ϑ, so that the attacker is unable to identify the true user u even with u's subtrajectory T_a. Formally, the re-identification attack is defined below:

Definition 2 (Re-Identification Attack). *Let M_{T_a} denote the set of trajectories in $\mathcal{T}_{syn}$ that are similar to a given subtrajectory T_a, $sim(T, T')$ measures the similarity between two trajectories T and T', and Z denote the selected sensitive zones.*

$$M_{T_a} = \{T | T \in \mathcal{T}_{syn}, sim(T_a, T \cap Z) \leq \vartheta\}. \tag{23}$$

If $|M_a| > \kappa$, the attack is successfully defensed.

In our experiments, we set 2×2 grids in the central area of the map as the sensitive zone, use DTW distance as the similarity distance, and set $\vartheta = 0.2 \times sim_{max}$, where sim_{max} denotes the maximum DTW distance of real trajectories in sensitive zones.

Outlier attack. In this scenario, the attackers focus on some records that have irregular attributes like outliers, which are relatively far from their neighbors, *e.g.,* trajectories with extremely long travel distance or unusual start/end locations that others seldom reach. Hence, it is easier to distinguish them from other normal trajectories, which may cause privacy leakage. We follow [23], and define this attack as follows:

Table 6: Utility performance comparison with unmodified NGRAM on Campus dataset.

Model	Density Error	Query Error	HQ Error	Kendall-tau	Trip Error	Length Error	Diameter Error	Pattern F1	Pattern Error
NGRAM	0.1172	0.7579	0.7001	0.3841	0.3024	0.1041	0.0663	0.32	0.7912
unmodified NGRAM	0.0536	0.4371	0.8108	0.6349	0.2006	0.1122	0.0638	0.55	0.8215
LDPTrace	**0.0043**	**0.2011**	**0.0013**	**0.8571**	**0.0740**	**0.0981**	**0.0587**	**0.71**	**0.5508**

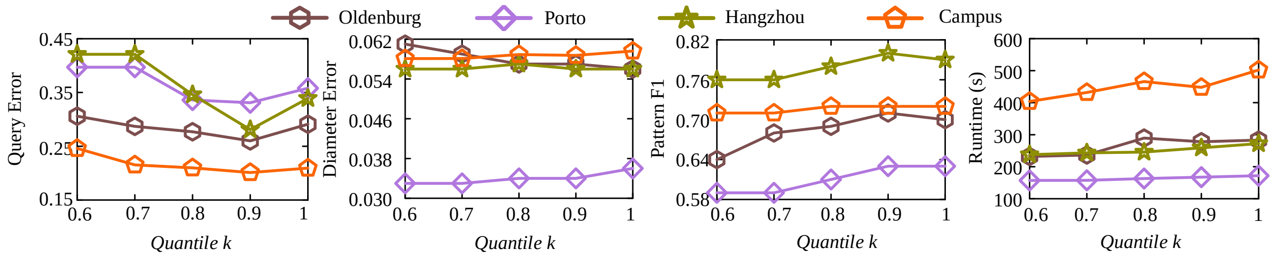

Figure 5: Impact of quantile k of length distribution.

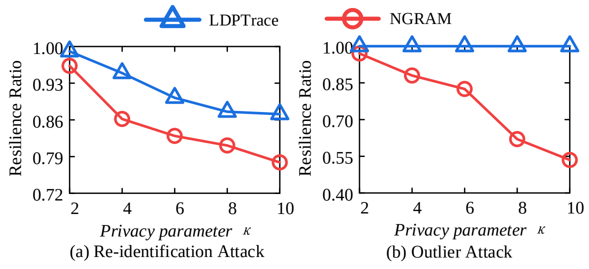

Figure 6: Attack resilience analysis on Oldenburg dataset.

Definition 3 (Outlier Attack). *For any outlier $T_o \in \mathcal{T}_{syn}$, let M_{T_o} denote the matching set that includes all the real trajectories in $\mathcal{T}$ that are similar to T_o, and similarity function $sim(\cdot)$ measures the travel distance difference between two trajectories with $\delta = 0.25 \times sim_{max}$. If $|M_{T_o}| > \kappa$, the attack is successfully defensed.*

$$M_{T_o} = \{T | T \in \mathcal{T}, sim(T_o, T) \leq \delta\} \tag{24}$$

To measure our model's defense ability to the aforementioned attacks, we define*resilience ratio* as follows:

$$Resilience\ Ratio = \frac{\sum_{T_a \in D_a} \mathbb{1}(|M_{T_a}| > \kappa)}{|D_a|} \tag{25}$$

where D_a is the set of trajectories to be attacked.

The results on Oldenburg dataset are shown in Figure 6, while the results of other datasets are omitted due to similar trend and limited space. The results indicate that LDPTrace has impressive ability to resist these two attacks: more than 88% trajectories can be successfully protected in re-identification attack, and all trajectories are hidden from outliers when the privacy parameter κ changes its value from 2 to 10. On the contrary, NGRAM cannot provide provable protections to these attacks, especially when the demand of protection is more strict (large privacy parameter κ). We contribute the superiority of LDPTrace to the synthesis design: since the published trajectories are synthesized from learned patterns, they do not resemble any real trajectory. Thus, it is much more difficult for attackers to identify traces that they are interested in.

7 CONCLUSIONS

In this paper, we develop a neat yet effective trajectory synthesis framework under the rigorous privacy of LDP, called LDPTrace, which achieves strong utility and efficiency simultaneously. Besides, LDPTrace can provide deterministic resilience against common location-based attacks. We also provide a theoretical guideline for selecting the grid granularity without consuming any privacy budgets. Extensive experiments conducted on three datasets demonstrate the superiority of LDPTrace. In the near future, we aim to extract more complex patterns from user's trajectory (like second-order Markov chain and average speed) to further enhance the authenticity of synthetic trajectories, and to investigate the LDP-based synthesis problem on streaming trajectories to empower real-time location-based applications.

A APPENDIX

Theorem A.1 (Mean and Variance of an OUE Ratio). *Given the unbiased frequency estimations of value x and value y (*i.e.*, $\tilde{g}(x)$ and $\tilde{g}(y)$) with OUE, Equation (26), and Equation (27) define the approximated mean and variance, respectively.*

$$\mathrm{E}^*[\frac{\tilde{g}(x)}{\tilde{g}(y)}] = \frac{f_x}{f_y}, \tag{26}$$

$$\mathrm{Var}^*[\frac{\tilde{g}(x)}{\tilde{g}(y)}] = \frac{(f_x)^2}{(f_y)^2}\left[\frac{\sigma_x^2}{(f_x)^2} - 2\frac{\mathrm{Cov}(x,y)}{f_x f_y} + \frac{\sigma_y^2}{(f_y)^2}\right], \tag{27}$$

where f_x and f_y are the true frequencies of x and y, respectively; σ_x^2 and σ_y^2 are the variance of OUE for estimators $\tilde{g}(x)$ and $\tilde{g}(y)$, respectively.

Proof of Theorem A.1. For any function $f(X,Y)$, we can choose the expansion point to be $\theta = (\mu_x, \mu_y)$, and the first order Taylor series approximation for $f(X,Y)$ is:

$$\begin{aligned} \mathrm{E}[f(X,Y)] &\approx \mathrm{E}[f(\theta)] + \mathrm{E}\left[f_x'(\theta)\,(X-\mu_x)\right] + \mathrm{E}\left[f_y'(\theta)\,(Y-\mu_y)\right] \\ &= \mathrm{E}[f(\theta)] + f_x'(\theta)\,\mathrm{E}\left[(X-\mu_x)\right] + f_y'(\theta)\,\mathrm{E}\left[(Y-\mu_y)\right] \\ &= f\left(\mu_x, \mu_y\right). \end{aligned}$$

Let $f(x,y) = x/y$, and the approximation holds $\mathrm{E}^*[f(X,Y)] = f(\mu_x,\mu_y) = \mu_x/\mu_y$. Therefore, the mean of an OUE ratio $\tilde{g}(x)/\tilde{g}(y)$ approximates f_x/f_y, where $\mathrm{E}[\tilde{g}(x)] = f_x$ and $\mathrm{E}[\tilde{g}(y)] = f_y$.

Besides, the variance of $f(X,Y)$ is:

$$\mathrm{Var}[f(X,Y)] = \mathrm{Var}\left\{[f(X,Y) - E(f(X,Y))]^2\right\} \approx \mathrm{Var}\left\{[f(X,Y) - f(\theta)]^2\right\}$$

Then using the first order Taylor expansion for $f(X,Y)$ around θ:

$$\begin{aligned}\mathrm{Var}[f(X,Y)] &\approx \mathrm{E}\left\{\left[f(\theta) + f'_x(\theta)\,(X-\theta_x) + f'_y(\theta)\,\left(Y-\theta_y\right) - f(\theta)\right]^2\right\} \\ &= f'^2_x(\theta)\,\mathrm{Var}(X) + 2f'_x(\theta)f'_y(\theta)\,\mathrm{Cov}(X,Y) + f'^2_y(\theta)\,\mathrm{Var}(Y)\end{aligned}$$

where $f(x,y) = x/y$, and the approximated variance is:

$$\mathrm{Var}^*[X/Y] = \frac{(\mu_x)^2}{(\mu_y)^2}\left[\frac{\sigma_x^2}{(\mu_x)^2} - 2\frac{\mathrm{Cov}(X,Y)}{\mu_x\mu_y} + \frac{\sigma_y^2}{(\mu_y)^2}\right]$$

□

ACKNOWLEDGMENTS

This work was supported by the NSFC under Grants No. (62025206, 61972338, and 62102351). Lu Chen is the corresponding author of the work.

REFERENCES

[1] Gergely Acs and Claude Castelluccia. 2014. A Case Study: Privacy Preserving Release of Spatio-Temporal Density in Paris. In *KDD*. 1679–1688.

[2] Miguel E Andrés, Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2013. Geo-indistinguishability: Differential privacy for location-based systems. In *CCS*. 901–914.

[3] Raef Bassily, Kobbi Nissim, Uri Stemmer, and Abhradeep Thakurta. 2017. Practical Locally Private Heavy Hitters. In *NeurIPS*. 2285–2293.

[4] Vincent Bindschaedler and Reza Shokri. 2016. Synthesizing Plausible Privacy-Preserving Location Traces. In *IEEE Symposium on Security and Privacy (SP)*. 546–563.

[5] Luca Bonomi and Li Xiong. 2013. A two-phase algorithm for mining sequential patterns with differential privacy. In *CIKM*. 269–278.

[6] Kuntai Cai, Xiaoyu Lei, Jianxin Wei, and Xiaokui Xiao. 2021. Data synthesis via differentially private markov random fields. In *VLDB*. 2190–2202.

[7] Yang Cao, Masatoshi Yoshikawa, Yonghui Xiao, and Li Xiong. 2017. Quantifying Differential Privacy under Temporal Correlations. In *ICDE*. 821–832.

[8] Rui Chen, Gergely Acs, and Claude Castelluccia. 2012. Differentially private sequential data publication via variable-length n-grams. In *CCS*. 638–649.

[9] Rui Chen, Benjamin CM Fung, Bipin C Desai, and Nériah M Sossou. 2012. Differentially private transit data publication: a case study on the montreal transportation system. In *KDD*. 213–221.

[10] Rui Chen, Benjamin C. Fung, Philip S. Yu, and Bipin C. Desai. 2014. Correlated Network Data Publication via Differential Privacy. *The VLDB Journal* 23, 4 (aug 2014), 653–676.

[11] Graham Cormode, Somesh Jha, Tejas Kulkarni, Ninghui Li, Divesh Srivastava, and Tianhao Wang. 2018. Privacy at scale: Local differential privacy in practice. In *SIGMOD*. 1655–1658.

[12] Graham Cormode, Samuel Maddock, and Carsten Maple. 2021. Frequency estimation under local differential privacy. In *VLDB*. 2046–2058.

[13] Teddy Cunningham, Graham Cormode, and Hakan Ferhatosmanoglu. 2021. Privacy-Preserving Synthetic Location Data in the Real World. In *SSTD*. 23–33.

[14] Teddy Cunningham, Graham Cormode, Hakan Ferhatosmanoglu, and Divesh Srivastava. 2021. Real-World Trajectory Sharing with Local Differential Privacy. In *PVLDB*. 2283–2295.

[15] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. 2017. Collecting telemetry data privately. In *NeurIPS*. 3574–3583.

[16] Linkang Du, Zhikun Zhang, Shaojie Bai, Changchang Liu, Shouling Ji, Peng Cheng, and Jiming Chen. 2021. AHEAD: Adaptive Hierarchical Decomposition for Range Query under Local Differential Privacy. In *CCS*. 1266–1288.

[17] John C Duchi, Michael I Jordan, and Martin J Wainwright. 2013. Local privacy and statistical minimax rates. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. 429–438.

[18] Cynthia Dwork. 2006. Differential Privacy. In *Automata, Languages and Programming*. 1–12.

[19] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In *CCS*. 1054–1067.

[20] Chang Ge, Shubhankar Mohapatra, Xi He, and Ihab F. Ilyas. 2021. Kamino: Constraint-Aware Differentially Private Data Synthesis. In *VLDB*. 1886–1899.

[21] Xiaolan Gu, Ming Li, Yang Cao, and Li Xiong. 2019. Supporting both range queries and frequency estimation with local differential privacy. In *2019 IEEE Conference on Communications and Network Security (CNS)*. 124–132.

[22] Mehmet Emre Gursoy, Ling Liu, Stacey Truex, and Lei Yu. 2018. Differentially private and utility preserving publication of trajectory data. *IEEE Transactions on Mobile Computing* 18, 10 (2018), 2315–2329.

[23] Mehmet Emre Gursoy, Ling Liu, Stacey Truex, Lei Yu, and Wenqi Wei. 2018. Utility-Aware Synthesis of Differentially Private and Attack-Resilient Location Traces. In *ACM SIGSAC Conference on Computer and Communications Security*. 196–211.

[24] M Emre Gursoy, Vivekanand Rajasekar, and Ling Liu. 2020. Utility-Optimized Synthesis of Differentially Private Location Traces. In *2020 Second IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*. 30–39.

[25] Xi He, Graham Cormode, Ashwin Machanavajjhala, Cecilia M. Procopiuc, and Divesh Srivastava. 2015. DPT: Differentially Private Trajectory Synthesis Using Hierarchical Reference Systems. In *PVLDB*. 2150–8097.

[26] Hongbo Jiang, Jie Li, Ping Zhao, Fanzi Zeng, Zhu Xiao, and Arun Iyengar. 2021. Location Privacy-Preserving Mechanisms in Location-Based Services: A Comprehensive Survey. *ACM Comput. Surv.* 54, 1, Article 4 (2021), 36 pages.

[27] Fengmei Jin, Wen Hua, Matteo Francia, Pingfu Chao, Maria Orlowska, and Xiaofang Zhou. 2022. A survey and experimental study on privacy-preserving trajectory data publishing. *TKDE* (2022).

[28] Fengmei Jin, Wen Hua, Boyu Ruan, and Xiaofang Zhou. 2022. Frequency-based Randomization for Guaranteeing Differential Privacy in Spatial Trajectories. In *ICDE*. 1727–1739.

[29] Noah Johnson, Joseph P Near, and Dawn Song. 2018. Towards practical differential privacy for SQL queries. In *VLDB*. 526–539.

[30] Haoran Li, Li Xiong, and Xiaoqian Jiang. 2014. Differentially private synthesization of multi-dimensional data using copula functions. In *EDBT*. 475.

[31] Ninghui Li, Wahbeh Qardaji, Dong Su, and Jianneng Cao. 2012. PrivBasis: Frequent Itemset Mining with Differential Privacy. In *VLDB*. 1340–1351.

[32] Ninghui Li, Wahbeh Qardaji, Dong Su, Yi Wu, and Weining Yang. 2013. Membership Privacy: A Unifying Framework for Privacy Definitions. In *CCS*. 889–900.

[33] Zitao Li, Tianhao Wang, Milan Lopuhaä-Zwakenberg, Ninghui Li, and Boris Škoric. 2020. Estimating numerical distributions under local differential privacy. In *SIGMOD*. 621–635.

[34] Yves-Alexandre Montjoye, Cesar Hidalgo, Michel Verleysen, and Vincent Blondel. 2013. Unique in the Crowd: The Privacy Bounds of Human Mobility. *Scientific reports* 3 (03 2013), 1376.

[35] Wahbeh Qardaji, Weining Yang, and Ninghui Li. 2013. Differentially private grids for geospatial data. In *ICDE*. 757–768.

[36] Apple Differential Privacy Team. 2017. Learning with Privacy at Scale. https://machinelearning.apple.com/research/learning-with-privacy-at-scale

[37] Haiming Wang, Zhikun Zhang, Tianhao Wang, Shibo He, Michael Backes, Jiming Chen, and Yang Zhang. 2023. PrivTrace: Differentially Private Trajectory Synthesis by Adaptive Markov Model. In *32th USENIX Security Symposium (USENIX Security 23)*.

[38] Ning Wang, Xiaokui Xiao, Yin Yang, Zhenjie Zhang, Yu Gu, and Ge Yu. 2017. PrivSuper: a superset-first approach to frequent itemset mining under differential privacy. In *ICDE*. 809–820.

[39] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. 2017. Locally differentially private protocols for frequency estimation. In *26th USENIX Security Symposium (USENIX Security 17)*. 729–745.

[40] Tianhao Wang, Joann Qiongna Chen, Zhikun Zhang, Dong Su, Yueqiang Cheng, Zhou Li, Ninghui Li, and Somesh Jha. 2021. Continuous Release of Data Streams under both Centralized and Local Differential Privacy. In *CCS*. 1237–1253.

[41] Benjamin Weggenmann and Florian Kerschbaum. 2021. Differential Privacy for Directional Data. In *CCS*. 1205–1222.

[42] Yonghui Xiao and Li Xiong. 2015. Protecting locations with differential privacy under temporal correlations. In *ACM SIGSAC Conference on Computer and Communications Security*. 1298–1309.

[43] Xingxing Xiong, Shubo Liu, Dan Li, Zhaohui Cai, Xiaoguang Niu, and Angel M. Del Rey. 2020. A Comprehensive Survey on Local Differential Privacy. *Sec. and Commun. Netw.* (jan 2020).

[44] Jianyu Yang, Xiang Cheng, Sen Su, Huizhong Sun, and Changju Chen. 2022. Collecting Individual Trajectories under Local Differential Privacy. In *MDM*. 99–108.

[45] Mengmeng Yang, Lingjuan Lyu, Jun Zhao, Tianqing Zhu, and Kwok-Yan Lam. 2020. Local differential privacy and its applications: A comprehensive survey. *arXiv preprint arXiv:2008.03686* (2020).

[46] Quan Yuan, Zhikun Zhang, Linkang Du, Min Chen, Peng Cheng, and Mingyang Sun. 2023. PrivGraph: Differentially Private Graph Data Publication by Exploiting Community Information. In *USENIX Security*.

[47] Sepanta Zeighami, Ritesh Ahuja, Gabriel Ghinita, and Cyrus Shahabi. 2022. A Neural Database for Differentially Private Spatial Range Queries. In *VLDB*. 1066–1078.

[48] Zhikun Zhang, Tianhao Wang, Ninghui Li, Shibo He, and Jiming Chen. 2018. CALM: Consistent adaptive local marginal for marginal release under local differential privacy. In *CCS*. 212–229.

[49] Zhikun Zhang, Tianhao Wang, Ninghui Li, Jean Honorio, Michael Backes, Shibo He, Jiming Chen, and Yang Zhang. 2021. PrivSyn: Differentially Private Data Synthesis. In *30th USENIX Security Symposium (USENIX Security 21)*. 929–946.