Differentially Private and Utility Preserving Publication of Trajectory Data

Mehmet Emre Gursoy, Ling Liu, *Fellow, IEEE,* Stacey Truex, and Lei Yu

Abstract—The universal popularity of GPS-enabled mobile devices and traffic navigation services has fueled the growth of trajectory data, as evidenced by Uber Movement and NYC taxi data release. Although trajectory data can generate valuable insights and value-added services for many, publishing this data while respecting mobile users' privacy has been a long-standing challenge. In this paper we present DP-Star, a methodical framework for publishing trajectory data with differential privacy guarantee as well as high utility preservation. DP-Star relies on a novel combination of several components. First, DP-Star's normalization algorithm uses the Minimum Description Length metric to summarize raw trajectories using their representative points, thereby achieving a desirable trade-off between the preciseness and conciseness of their information content. Second, DP-Star constructs a density-aware grid which ensures spatial densities can be preserved despite the noise added to satisfy differential privacy. Third, DP-Star preserves the correlations between trajectories' end points through a private trip distribution, and intermediate points through a private Markov mobility model. Finally, DP-Star estimates users' trip lengths using a median length estimation method, and generates synthetic trajectories that preserve both differential privacy and high utility. Our experimental comparison shows that DP-Star significantly outperforms existing approaches in terms of trajectory utility and accuracy.

Index Terms—Trajectory data mining, differential privacy, privacy-preserving data publishing, spatio-temporal databases.

1 Introduction

The ubiquity of GPS-equipped mobile devices and smartphones, along with the popularity of location based social networks and traffic navigation services such as Waze and Google Maps, have greatly eased the collection of users' location traces. On one hand, there is great benefit in employing third party services for mining this data for purposes such as city planning, travel pattern analysis, route recommendation, and transportation management. Companies and governments are indeed interested in third party services that can mine cross-county, cross-city and cross-state trajectory data to understand business development potentials and economic trends, as recently evidenced by NYC Taxi & Limousine Commission's release of NYC taxi trips and Uber's release of anonymized trip statistics through Uber Movement[1].

On the other hand, releasing trajectory data raises legitimate privacy concerns since it can intrude users' privacy by revealing their sensitive locations, such as home and work locations, frequent stops, and frequent visits to hospitals or clinics. Traditional privacy protection methods rely on the likes of pseudo-anonymization, k-anonymity and spatial location cloaking [1], [2], [3]; however these methods nowadays face heavy criticism. First, recent studies show that human mobility is unique and identifiable, and its identifiability decreases only slightly despite spatial coarsening [4], [5]. Second, several works found that even with perfect pseudo-anonymization, an external dataset or additional background knowledge is often sufficient for re-identification of individuals with high accuracy [6], [7], [8], [9]. Thus, traditional methods may cause privacy breaches when releasing full trajectory datasets, and there is a need to explore new avenues for privacy-preserving trajectory release.

Differential privacy (DP) has recently emerged as the leading paradigm for privacy protection. It states that the output of a private algorithm should not reveal much about the presence, absence or content of a single record in a dataset (e.g., a single user's trajectory) [10]. Motivated by this privacy definition and the failure of traditional anonymization methods, we advocate that instead of publishing a dataset containing users' original trajectories, one should leverage DP to generate and publish a synthetic trajectory dataset. This preserves privacy from two aspects: First, released data is synthetic without one-to-one correspondence between actual and synthetic trajectories. Thus, re-identification and de-anonymization attacks (e.g., record linkage) are no longer applicable. Second, by definition of DP, an adversary observing the synthetic trajectories will not be able to infer with strong confidence the presence, absence or content of a user's trajectory in the original data. Therefore, if a mobile user contributes their location trace to a private dataset D, a synthetic dataset S_D is generated using D, and finally S_D is released; the release of S_D has limited privacy risk for the user.

Even though there has been previous work on differentially private data publishing, these works cannot sufficiently capture the spatial utility and quality aspects of trajectory data. This is because they either: (i) address the release of individual location points instead of trajectories which are correlated sequences of locations [11], [12], [13]; or (ii) assume locations come from a small, discrete domain such as a few hundred subway and bus stations [14], [15], whereas in today's GPS-powered mobile systems, locations are collected as *(lat,lon)* pairs at arbitrary positions. To illustrate our claim, in Figure 1 we compare our proposed approach (DP-Star) with two most relevant works: ngram [15] and DPT [16]. On the left, we illustrate the spatial densities of the original trajectories' locations. On the right, we illustrate the

- *M.E. Gursoy, L. Liu, S. Truex and L. Yu are with the School of Computer Science, Georgia Institute of Technology, Atlanta, GA, 30332. E-mail: {memregursoy, staceytruex, leiyu}@gatech.edu, ling.liu@cc.gatech.edu*

For obtaining reprints of this article, please send e-mail to: reprints@ieee.org

[1] See http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml and https://movement.uber.com/cities for more information.

densities of the synthetic trajectory datasets published using DPT, ngram and DP-Star. Clearly, DP-Star has better utility visually. In Section 6, we demonstrate its superiority also quantitatively.

This paper presents DP-Star (**d**ifferentially **p**rivate **s**ynthetic **tra**jectory publishe**r**) with strong utility. The novelty and utility of DP-Star are fueled by a combination of features. First, DP-Star normalizes input trajectories via the Minimum Description Length (MDL) principle, which reduces each trajectory into a sequence of representative points that is sufficient to summarize the trajectory. The privacy budget ε is then spent on preserving the representative points instead of all points, thus eliminating budget expenditure on redundant points. Second, DP-Star discretizes the 2D location space using an adaptive grid structure. This density-aware data structure allows DP-Star to better capture location densities and preserve them despite the noise added to satisfy differential privacy. Third, DP-Star's synthetic trajectory generation relies on a novel algorithmic combination of private statistics: The joint distribution of trajectories' start and end points is preserved using a privately extracted trip distribution, and their intermediate points are preserved using a private Markov mobility model. Finally, trajectories' lengths are approximated using a median length estimation mechanism. We experimentally compare DP-Star with the state of the art approaches in differentially private trajectory publishing [15], [16]. Our results on real and simulated trajectory datasets using multiple metrics show that DP-Star significantly outperforms existing approaches in terms of accuracy and utility.

2 Related Work

We classify related work into three broad categories below and discuss relevant works under each category.

Location anonymization. Traditional trajectory privacy methods are based on anonymizaton through k-anonymity and its extensions. The (k, δ)-anonymity notion introduced in [17] ensures that at least k different trajectories exist in cylinders of radius δ. (k, δ)-anonymity is enforced via clustering and space translation. [3] enforces trajectory k-anonymity using location generalization, and reduces the anonymization problem to time and space shifted alignment of location sequences. [18] introduces attack graphs to break traditional k-anonymity on trajectories using timestamps as quasi-identifying information, and proposes a Hilbert indexing method to circumvent such attacks. [19] applies $(K, C)_L$-privacy to thwart identity linkage and sensitive attribute disclosure attacks when trajectories contain sensitive information.

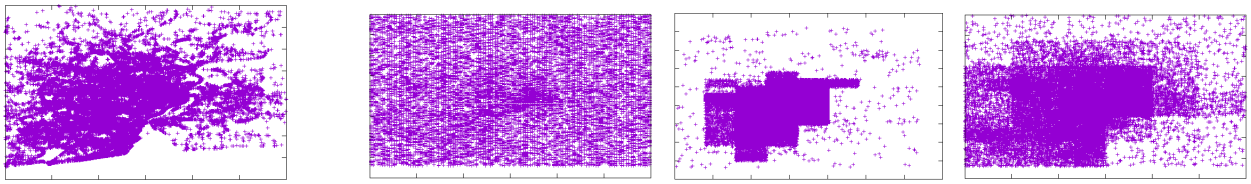
Recent studies on human mobility patterns [4], [5] show that individuals' trajectories are highly unique and predictable, and trajectory uniqueness decreases only slightly despite spatial coarsening (e.g., generalization). Recent reports on re-identification and de-anonymization attacks [6], [8], [9] further render the traditional methods such as anonymization, generalization and spatial cloaking insufficient to provide adequate privacy protection.

Differential location privacy. Research in location privacy started a decade ago with the notion of location k-anonymity and two landmark results: uniform location k-anonymity [2] and user-defined, personalized location k-anonymity [7]. After the proliferation of differential privacy (DP), two stronger notions of location privacy were proposed based on statistical quantification of attack resilience. The first one is geo-indistinguishability [20], which employs DP by measuring the posterior information gain of an adversary based on the release of pseudo-locations of mobile users. The second one is the expected inference error [21], [22], which proposes a privacy notion powered by attack resilience to the prior knowledge of an adversary. A number of location obfuscation mechanisms have been developed to satisfy these privacy notions [20], [22], [23], [24], [25].

However, location privacy differs from trajectory release in several ways. First, location privacy is concerned with perturbing a single instantaneous location of a mobile user; whereas in trajectory release, a trajectory is treated as a time-ordered sequence of locations. If the sequentiality aspect is not properly considered, perturbed locations will still be vulnerable to tracking and inference attacks [26]. Second, existing trajectory mining applications expect their inputs to consist of raw location traces. However, many location privacy works transform raw locations into location sets or generalized (cloaked) regions. Thus, outputs of location privacy algorithms may not be readily used by trajectory mining applications. Third, location privacy does not aim to hide the fact that a user participates in a trajectory database, whereas DP data release algorithms such as ours provide provable bounds on membership leakage.

Differentially private data release. DP guarantees that the output of an algorithm does not enable an adversary to distinguish, beyond a probability controlled by parameter ε, between two datasets D_1 and D_2 that differ in one user's record [10]. Research efforts on differentially private spatio-temporal data release can be divided into two subcategories: (i) DP release of density statistics, and (ii) DP release of synthetic datasets. In terms of the prior, most works focus on answering count queries and generating histograms for spatial data. [11] develops private versions of popular spatial indexing methods such as quadtrees, kd-trees and Hilbert R-trees. To improve the accuracy of these indexes, authors introduce techniques for optimizing the budgeting of ε. [12] claims that the main weakness of these indexes is the over-division of ε into an excess number of tree levels, and proposes one-level and two-level structures to address this problem. [13] presents PrivTree, a hierarchical method for releasing spatial distributions, allowing variable node depth in the indexing tree based on noise estimation prior to splitting a node. [27] provides a benchmark of DP query answering and histogram release algorithms. [28], [29] study the release of spatio-temporal densities of urban areas using call data records. None of these techniques are directly applicable to trajectory publication, since they only publish density and count statistics, and constructing synthetic trajectories from aggregate statistics remains an open problem.

Several recent works consider the problem of releasing sequential datasets or trajectories with DP. [30] publishes trajectories with a weaker notion of DP by injecting noise to their locations. [26] publishes trajectories with an orthogonal privacy metric called plausible deniability, which relies on a privacy test that rejects a candidate synthetic trajectory from being published as long as there are not enough trajectories that are semantically similar to it in the original data. It was later shown that a randomized form of this privacy test can yield a relaxation of ε-DP, namely (ε, δ)-DP, but only for a rigid set of parameters ε, δ [31]. This makes their approach inappropriate for ε-DP trajectory publication, which enforces a stronger privacy guarantee and allows flexibility to choose the desired privacy level ε. [14] publishes synthetic sequential datasets utilizing the *prefix tree* data structure, which stores the private noisy counts of each sequence. [15] extends the work in [14] with the ngram model in which sequences stored in the tree can be of varying length. [16] develops the DPT system for private trajectory synthesis, with the novelty of *hierarchical reference systems*, i.e., location discretization using hierarchically organized



(a) Original distribution (b) DPT [16] (c) ngram [15] (d) DP-Star (our solution)

Fig. 1: Spatial density of the Taxi dataset (on the left), densities of the same region in synthetic data generated by DPT, ngram and DP-Star with $\varepsilon = 1.0$ (on the right).

grids. ngram and DPT are regarded as the current state of art in DP trajectory release. In Section 6, we experimentally compare DP-Star with both, and show that DP-Star provides superior utility.

3 Design Overview

In this section, we give a preliminary overview of the DP-Star framework. We first review the reference trajectory data model and basic terminology of differential privacy. Then we formulate the problem statement that motivates the design of DP-Star, and highlight our solution approach.

3.1 Trajectory Data Model and Notation

A trajectory, denoted by $T = p_1p_2...p_{jTj}$, is a time-ordered sequence of location points. jTj denotes the number of points in T. A dataset D is a collection of jDj trajectories. We allow trajectories of varying length and varying sampling rate. We denote by $\quad (D)$ the geographical location space of D, which is a 2D spatial domain. We assume that $\quad (D)$ by itself is not sensitive. For instance, if D consists of NYC taxi trajectories, $\quad (D)$ is the geographic boundaries of NYC.

For a trajectory $T = p_1p_2...p_{jTj}$, we call p_1 and p_{jTj} the start and end points of T respectively. We refer to (p_i, p_{i+1}) as a line segment of trajectory T $(1 \quad i < jTj)$ with p_i as the start point and p_{i+1} as the end point of this line segment. Then, T can be represented by a sequence of $jTj \quad 1$ line segments. We call the collection of consecutive segments between any two points p_i and p_j $(1 \quad i < j \quad jTj)$ of T a subsequence (subtrajectory) of T. We use $len(p_ip_j)$ to denote the length between points p_i and p_j, measured by the Euclidean distance between p_i and p_j.

3.2 Differential Privacy

Our goal is to publish trajectories while preserving ε-differential privacy (ε-DP). Differential privacy ensures that the output of a private algorithm is not strongly dependent on any one trajectory in the input dataset.

Definition 1 (Neighboring datasets)**.** *Two datasets D_1, D_2 are called neighboring datasets if either $D_1 = D_2 \ [\ fTg$ or $D_2 = D_1 \ [\ fTg$, where T denotes a single trajectory.*

Definition 2 (ε-differential privacy [10])**.** *A randomized algorithm A is ε-differentially private if for all neighboring datasets D_1, D_2 and for all possible outcomes of the algorithm $S \quad Range(A)$,*

$$Pr[A(D_1) \ 2 \ S] \quad e^{\varepsilon} \quad Pr[A(D_2) \ 2 \ S]$$

where the probabilities are over the randomness of A.

The privacy parameter ε controls how much an adversary can distinguish between D_1 and D_2. Smaller ε yields higher privacy. There are well-known, general purpose building blocks that help in designing ε-DP algorithms. We start with the definition of *sensitivity*.

Definition 3 (Sensitivity [32])**.** *Let $f : D_1 \ ! \ \mathsf{R}^m$ be a function that maps a dataset D_1 into a fixed-size vector of m real numbers. The sensitivity of f is defined as:*

$$\quad f := \max_{D_1,D_2} jjf(D_1) \quad f(D_2)jj$$

for all neighboring D_1, D_2, where $jj.jj$ denotes the L_1 norm.

A popular ε-DP algorithm for answering numeric queries is the Laplace mechanism. When answering a set of numeric queries (e.g., a set of count queries), the Laplace mechanism retrieves the true answers of these queries, and then perturbs each answer by adding random noise scaled according to their sensitivity.

Definition 4 (Laplace mechanism [32])**.** *Let $Lap(\sigma)$ denote a random variable sampled from the Laplace distribution with mean 0 and scale parameter σ. For a numeric-valued function $f : D \ ! \ \mathsf{R}^d$, the algorithm A that answers f by:*

$$A(f, D) = f(D) + < Y_1, ..., Y_d >$$

satisfies ε-DP if Y_i are i.i.d. random variables drawn from $Lap(\sigma)$ where $\sigma \qquad f/\varepsilon$.

When answering categorical queries, the exponential mechanism will be used instead. The exponential mechanism is useful for selecting a discrete output r from a domain R in a differentially private manner. It employs a scoring function (i.e., quality criterion) q that associates each output $r \ 2 \ R$ with a non-zero probability of being selected. The sensitivity of q is:

$$\quad q := \max_{8r2R,\ \text{neighboring } D_1,\ D_2} jjq(D_1, r) \quad q(D_2, r)jj$$

Definition 5 (Exponential mechanism [33])**.** *Let $q : (D \quad R) \ ! \ \mathsf{R}$ be a scoring function for a dataset D and a domain of discrete outputs R. The algorithm A that returns the output $r \ 2 \ R$ with probability proportional to $e^{\frac{\varepsilon q(D,r)}{2 \quad q}}$ satisfies ε-DP.*

Differentially private algorithms have the following composition properties, which allow us to build more complex algorithms by combining the aforementioned building blocks.

Definition 6 (Composition properties [10], [34])**.** *Let A_1, A_2, ..., A_n be n algorithms, such that for each $i \ 2 \ [1, n]$, A_i satisfies ε_i-DP. Then, the following properties hold:*

Sequential Composition: Releasing the outputs $A_1(D)$, $A_2(D)$, ..., $A_n(D)$ satisfies $(\sum_{i=1}^{n} \varepsilon_i)$-DP.

Parallel Composition: Executing each algorithm on a disjoint subset of D satisfies $max_i(\varepsilon_i)$-DP.

Immunity to Post-processing: Computing a function of the output of a differentially private algorithm does not deteriorate its privacy, e.g., publicly releasing the output of $A_i(D)$ or using it as an input to another algorithm does not violate ε_i-DP.

ε is also referred to as the *privacy budget*. Given a total budget ε, the goal is to create an ε-DP algorithm A by cleverly combining the building blocks according to the composition properties.

3.3 Solution Approach

The problem we study in this paper can be stated as follows: Given a private, sensitive trajectory dataset D, design an ε-DP algorithm A that outputs a synthetic dataset S_D, such that S_D has high accuracy and utility in various data mining tasks. We aim to create a general purpose framework while remaining agnostic to the types of tasks S_D will be used in. We capture and quantify the utility-driven resemblance between D and S_D via multiple metrics, such as accuracy in answering count queries, preserving frequent patterns, and distribution of trajectory diameters.

Design Objectives. In DP-Star, we prioritized the following design objectives to inject increased utility while satisfying the same level of privacy as in the state of the art works (ε-DP). Below we analyze the pros and cons of two representative approaches (ngram and DPT) on ε-DP trajectory publishing [15], [16] with respect to our design objectives.

Our first objective is effective discretization of (D). Given D and (D), we have an unbounded number of possibilities to simulate the transition from one point to another in a trajectory T 2 D. To develop an efficient and accurate algorithm, we need to bound such transition possibilities to high utility choices. A popular solution is to employ a *grid* as an index structure to discretize (D) into a collection of grid cells. However, choosing an appropriate grid size and structure is not trivial. If the grid is too coarse (e.g., 2x2), then each cell covers a large spatial area, and knowing that T visited a certain cell is uninformative. On the other hand, if the grid is too fine-grained (e.g., 20x20), we end up with many empty cells in which there are few or no visits by trajectories in D. In this case, more queries and additions of Laplace noise to perturb the answers of count and transition queries will lead to high inaccuracy, in addition to inefficiency.

Our second objective is effective preservation of the spatial utility of D in S_D. We break this into two sub-objectives: (i) preservation of trajectories' start and end points and the correlations therein, and (ii) preservation of intra-trajectory mobility. The former improves S_D for the likes of passenger demand analysis, taxi service prediction, and residential and workplace annotation. The latter improves S_D for mining frequent patterns and moving together patterns, subtrajectory clustering, etc. However, achieving both goals simultaneously is non-trivial and often overlooked in the literature. Among the previous works, ngram preserves the density of frequently occurring subtrajectories using prefix trees. Yet, trajectories' start and end points often fall in residential areas, which are sparser regions. Since ngram prunes sparse entries in its prefix tree, its trajectories have an uneven spatial distribution as shown in Figure 1c: Frequent intermediate regions become excessively dense, and the remaining regions become excessively sparse. On the other hand, DPT synthesizes trajectories using a weighted random walk (i.e., Monte Carlo sampling) and a novel data structure called hierarchical reference systems. Although this approach preserves intra-trajectory mobility, no correlation exists between trajectories' start and end points due to the random walk. This results in the poor spatial utility illustrated in Figure 1b.

DP-Star Design. The above motivated us to develop DP-Star with several novel features. First, we use a density-aware adaptive grid structure to effectively discretize (D), and achieve a better trade-off between errors due to coarse grid structure and Laplace noise. Second, we use a three-stage approach for spatial utility preservation: We identify utility-critical representative points in a normalization step. We preserve correlations of trajectories' start and end points using a trip distribution. Finally, we preserve intra-trajectory mobility using a Markov mobility model.

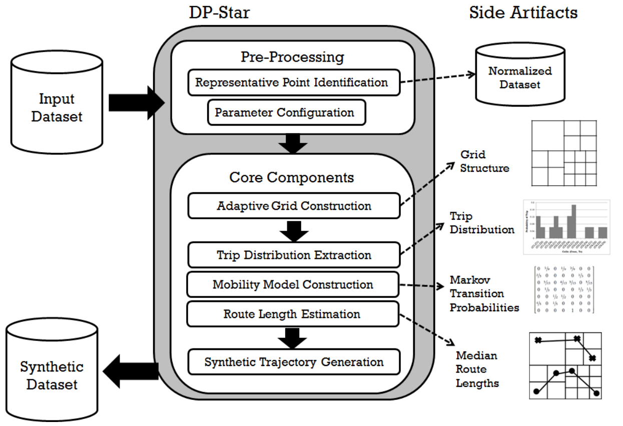


Fig. 2: DP-Star System architecture

An architectural overview of DP-Star is given in Figure 2. DP-Star starts with a pre-processing phase to normalize raw trajectories through representative point identification, and to initialize and configure its parameters. Then, DP-Star follows its five core components. Descriptions of the pre-processing phase is given in Section 4, and the core components are presented in Section 5. The main output of DP-Star is a synthetic trajectory dataset S_D. In addition, DP-Star also produces some functional side artifacts, as shown on the right of Figure 2, which are metadata used by DP-Star in generating S_D. These artifacts satisfy DP by themselves, therefore they can also be published safely to be used by external applications for other purposes.

4 DP-Star Pre-Processing

4.1 Representative Point Identification

DP-Star pre-processes the input dataset D by *normalizing* each trajectory T 2 D into a time ordered sequence of its *representative point*s. Representative points are points that summarize and characterize a trajectory's overall movement. We have two main motivations for normalization. First, D may contain trajectories with different sampling rates, movements, velocities and so forth. Normalization allows us to focus on the utility-critical location samples and improves efficiency by removing insignificant intermediate points. For instance, a sequence of points collected along one road segment can be represented by the two junction points of the road segment. Second, consider the privacy budget ε: If the budget is spent to preserve the distribution of representative points (that are sufficient to accurately represent a trajectory) instead of preserving the distribution of all points (which may contain redundancy), data synthesis will be faster and more accurate.

Figure 3 shows a sample normalization. For $T = p_1 ... p_9$, we identify its representative points as $p_1 p_4 p_6 p_9$. Consequently, we build the trajectory $\tilde{T} = p_1 p_4 p_6 p_9$. We illustrate T using a straight black line and $\tilde{T}$ using a dashed red line. We call $\tilde{T}$ a normalized trajectory. We denote every point that appears in a normalized trajectory with a tilde, i.e., $\tilde{p}_i$ 2 $\tilde{T}$. A dataset consisting of normalized trajectories is denoted by $\tilde{D}$.

Finding an optimal $\tilde{T}$ from T depends on a trade-off between preciseness and conciseness. Preciseness requires $\tilde{T}$ to closely resemble T, which is better achieved when $\tilde{T}$ conserves many points in T (the extreme case $\tilde{T} = T$ is most precise). Conciseness requires $\tilde{T}$ to represent T with as few points as possible, which is better achieved when $\tilde{T}$ discards many points in T.

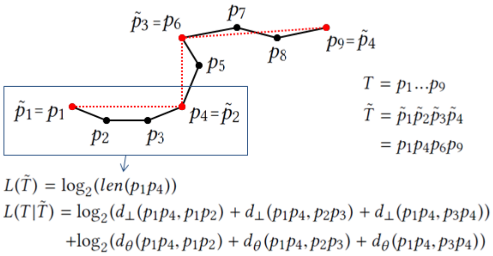

$\tilde{p}_3 = p_6$
p_7
p_8
$p_9 = \tilde{p}_4$
p_5
$\tilde{p}_1 = p_1$
p_2
p_3
$p_4 = \tilde{p}_2$
$T = p_1 \ldots p_9$
$\tilde{T} = \tilde{p}_1\tilde{p}_2\tilde{p}_3\tilde{p}_4$
$= p_1 p_4 p_6 p_9$
$L(\tilde{T}) = \log_2(len(p_1p_4))$
$L(T|\tilde{T}) = \log_2(d_\perp(p_1p_4, p_1p_2) + d_\perp(p_1p_4, p_2p_3) + d_\perp(p_1p_4, p_3p_4))$
$+\log_2(d_\theta(p_1p_4, p_1p_2) + d_\theta(p_1p_4, p_2p_3) + d_\theta(p_1p_4, p_3p_4))$

Fig. 3: Costs for representative point identification

We choose to use the Minimum Description Length (MDL) metric [35], [36] for making the trade-off decision, since MDL is non-parametric, applicable to diverse types of data, and allows for varying $j\tilde{T}j$ instead of imposing a fixed $j\tilde{T}j$ or $j\tilde{T}j/jTj$ as in other approaches [37], [38]. MDL consists of two components: H is the hypothesis and A is the data. $L(H)$ denotes the length of the hypothesis, and $L(AjH)$ denotes the length of the description of the data with the help of the hypothesis. The best hypothesis H is the one that minimizes $L(H) + L(AjH)$.

In trajectory normalization, H corresponds to the normalized $\tilde{T}$, and A corresponds to the actual T. Then, AjH measures the discrepancy between $\tilde{T}$ and T. Via this formulation, $L(H) = L(\tilde{T})$ is the measure of conciseness, and $L(AjH) = L(Tj\tilde{T})$ is the measure of preciseness. We wish to find $\tilde{T}$ that minimizes $L(\tilde{T}) + L(Tj\tilde{T})$. This is mathematically quantified as:

$$L(\tilde{T}) = \sum_{i=1}^{j\tilde{T}j \quad 1} \log_2(len(\tilde{p}_i\tilde{p}_{i+1}))$$

$$L(Tj\tilde{T}) = \sum_{i=1}^{j\tilde{T}j \quad 1} f\log_2(\sum_{\substack{j=j_{min} \\ p_{j_{min}}=\tilde{p}_i,\ p_{j_{max}}=\tilde{p}_{i+1}}}^{j_{max} \quad 1} d_?(\tilde{p}_i\tilde{p}_{i+1}, p_jp_{j+1})) + \log_2(\sum_{\substack{j=j_{min} \\ p_{j_{min}}=\tilde{p}_i,\ p_{j_{max}}=\tilde{p}_{i+1}}}^{j_{max} \quad 1} d_\theta(\tilde{p}_i\tilde{p}_{i+1}, p_jp_{j+1}))g$$

where $d_?$ and d_θ denote the *perpendicular* and *angular* distance between two line segments, respectively. We illustrate the two distance functions in Figure 4, where $L_i = s_ie_i$ is the first line segment (subtrajectory) with start point s_i and end point e_i, and $L_j = s_je_j$ is the second line segment. Without loss of generality we assume L_i is the longer line segment, L_j is the shorter segment, and α is the smaller intersecting angle between the two segments. For the example in Figure 3, we compute the $L(\tilde{T})$ and $L(Tj\tilde{T})$ costs for the portion highlighted by the box. This corresponds to one iteration of the distance functions.

With the above cost formulation, computing the $\tilde{T}$ that minimizes $L(\tilde{T}) + L(Tj\tilde{T})$ becomes an optimization problem, which can be solved optimally or heuristically. The optimal solution needs to consider all possible subsets of points in T (ordered by timestamps) as candidate $\tilde{T}$. Since this is neither efficient nor scalable, we employ a heuristic solution instead. Our linear-time greedy heuristic initializes $\tilde{T} = p_1$ and iterates through 2 $i < jTj$. In each iteration, it checks whether appending p_i to $\tilde{T}$ decreases the cost $L(\tilde{T}) + L(Tj\tilde{T})$ based on the $\tilde{T}$ constructed thus far. p_i is appended to $\tilde{T}$ if it causes a decrease in expected cost. The intuition behind why this is a heuristic solution is because it assumes the $\tilde{T}$ constructed in the previous iterations is optimal, which is not guaranteed. In the final step, p_{jTj} is appended to $\tilde{T}$ for trip conservation. We refer the reader to [35], [36] for alternative solutions to MDL optimization problems.

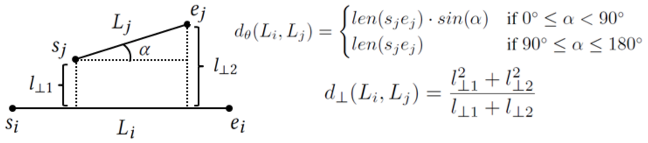

L_j
e_j
s_j
α
$l_{\perp 1}$
$l_{\perp 2}$
s_i
L_i
e_i
$d_\theta(L_i, L_j) = \begin{cases} len(s_je_j) \cdot sin(\alpha) & \text{if } 0° \le \alpha < 90° \\ len(s_je_j) & \text{if } 90° \le \alpha \le 180° \end{cases}$
$d_\perp(L_i, L_j) = \frac{l_{\perp 1}^2 + l_{\perp 2}^2}{l_{\perp 1} + l_{\perp 2}}$

Fig. 4: Illustration of perpendicular and angular distance

4.2 Parameter Configuration

The second component in DP-Star's pre-processing phase is for configuring the parameters that will be used in upcoming components. A critical issue in DP-Star is the distribution of the privacy budget ε. Given a total budget ε, DP-Star divides ε into four sub-budgets: ε_1, ε_2, ε_3 and ε_4, such that $\sum_{i=1}^{4} \varepsilon_i = \varepsilon$. Each ε_i is consumed in one of the first four core components of DP-Star (as shown in Figure 2), which prepare for the final step of synthetic trajectory generation. DP-Star satisfies ε-DP according to the sequential composition property as defined in Definition 6.

The distribution of ε to ε_i is determined by two criteria: utility and privacy. From the utility perspective, we empirically found that the following distribution maximizes trajectory utility across multiple datasets: $\varepsilon_1 = \varepsilon/9$ for adaptive grid construction, $\varepsilon_2 = 3\varepsilon/9$ for trip distribution extraction, $\varepsilon_3 = 4\varepsilon/9$ for mobility model construction and $\varepsilon_4 = \varepsilon/9$ for route length estimation. This distribution summarizes our empirical analysis (detailed in Section 6.5), and reflects our observation that the grid construction and route length estimation components can withstand lower budgets compared to remaining components. From the privacy perspective, the budget distribution also reflects the privacy implications of the four functional artifacts of DP-Star shown in Figure 2. A higher ε_i implies that the corresponding artifact is better preserved under DP, whereas a lower ε_i implies that it is more perturbed. Since budget distribution is parametric, this enables the data owner to modify the budget distribution to reflect which artifact he perceives as more sensitive, and should therefore be more perturbed to protect its privacy. For example, if the data owner feels that spatial densities/regions of his raw trajectories are more sensitive, then a lower ε_1 can be assigned to grid construction, which results in a less accurate grid. If trip distributions are more sensitive, then a lower ε_2 can be assigned. This provides flexibility with respect to different perceptions regarding what needs to be protected.

Seeking a better balance between utility and privacy is a common desire for maximizing overall utility under a given privacy level ε [15], [16]. Determining such balance by trial-and-error can be tedious and inefficient. This raises a natural question: *Can DP-Star determine its optimal parameter configuration in automated fashion?* Having the ability to do so is beneficial – it can guide those users who are not intimately familiar with DP to generate privacy-preserving trajectories using DP-Star without having to try multiple parameter configurations. Furthermore, in case we have a previously unseen dataset which invalidates our recommended budget allocation (e.g., because it is significantly skewed compared to the datasets we experiment with), an automated parameter configuration will enable DP-Star to determine a better configuration on the fly.

We pose the automated parameter configuration problem for budget distribution as an optimization problem in which the goal is maximizing utility given the total privacy budget. Maximizing utility is equivalent to minimizing total error. We measure error using 4 error metrics: average relative error in answering spatial density queries (Query AvRE), Kendall-tau coefficient for frequent patterns, trip error, and diameter error. We provide formal definitions of these metrics in Section 6.2. We employ a *hill climbing with random restart* technique to solve the optimization problem. In hill climbing, one starts with a random set of parameters and iteratively tries to reduce error by incrementally changing one parameter at a time. For budget configuration, we use increments of 0.02ε in each iteration. Since the main shortcoming of hill climbing is that it can get stuck at a local optima, we employ the popular technique of *random restart*, which runs several instances of hill climbing starting with different configurations each time. We enforce that in at least one of the restarts, our recommended budget distribution given above will be used. We keep track of the best parameter configuration found across all restarts and iterations, and finally assign this configuration to produce the end results and side artifacts that will be released to the user.

5 DP-Star Core Components

In this section, we describe the 5 core components of DP-Star one by one. The first four components are designed to preserve different types of spatial utility in trajectory data with DP guarantee, and the last component generates complete synthetic trajectories that can be published while preserving both privacy and utility.

5.1 Adaptive Grid Construction

Recall from Section 3.3 that one of the main design objectives in DP-Star is effective discretization of $\quad(D)$. The first core component is dedicated to fulfill this objective. We design a density-aware *adaptive grid* structure to discretize $\quad(D)$. Our adaptive grid (AG) is a two-level grid with density adaptive cell granularity. For low density regions in $\quad(D)$, AG places large, coarse cells; whereas for high density regions it divides the region into smaller cells with finer granularity. We denote an adaptive grid by A, and use the grid size parameter N to denote the total number of cells at the coarse level. One can conceptually view A as a hierarchical grid with the top level being an $N \quad N$ grid with uniform cells, and the bottom level grid with cells of varying size to reflect the density distribution of D.

We denote a cell in A by $C_i \; 2$ A, and a location p_j spatially residing within that cell by $p_j \; 2 \; C_i$. The location count query $\eta(D, C_i)$ is defined as:

$$\eta(D, C_i) = \sum_{T 2 D} \frac{\text{\# of } p_j \text{ in } T \text{ such that } p_j \; 2 \; C_i}{jTj}$$

Division by jTj is necessary in order to place an upper bound on the sensitivity of a collection of η queries, as will be shown in Theorem 1 and discussed in detail thereafter.

AG construction takes four inputs: Database D, privacy budget ε_1, initial grid size N and grid constant β. It outputs A, a collection of non-overlapping cells overlaid on $\quad(D)$. To construct the AG, the following steps are executed:

1) Lay a top level $N \quad N$ uniform grid on $\quad(D)$. Denote the cells in this grid by $(C_1, C_2, ..., C_{N^2})$.
2) For each C_i, issue a query $\eta(D, C_i)$ and perturb the true answers with Laplace noise, i.e., $\hat{\eta}_{C_i} = \eta(D, C_i) + Lap(1/\varepsilon_1)$.
3) Divide each cell C_i into $M \quad M$ smaller cells, where $M = \sqrt{\beta \quad \hat{\eta}_{C_i}}$.

Values of the two parameters N and β can be determined by DP-Star's parameter configuration phase, but here we give some guidelines based on our empirical analysis in Section 6.4. First, our analysis shows that AG construction is not extremely sensitive to the choice of N, hence setting it close to our suggested values will suffice, e.g., $N = 7$ typically performs well across multiple datasets. Second, note that M is positively correlated with the noisy count $\hat{\eta}_{C_i}$ and grid constant β. For the noisy count the intuition is that when $\hat{\eta}_{C_i}$ is small, M should also be small so that we do not over-partition a sparse region. If we over-partition, Laplace noise dominates true counts. When $\hat{\eta}_{C_i}$ is large, we can have larger M to improve granularity, and since C_i is dense, we are less likely to be dominated by noise even after we zoom in.

We employ the grid constant β as a balancing factor to determine the range of values for M, so that M remains sensitive to changes in spatial density but the total number of cells resulting from this procedure is not excessively large. The square root in the definition of M ensures sublinear growth with respect to $\hat{\eta}$ so that the grid does not become extremely fine-grained for large datasets. In practice, a value of $\beta = \frac{\varepsilon \quad \varepsilon_1}{80}$ works well. Since the upcoming components will use the remaining budget $\varepsilon \quad \varepsilon_1$, when this is high, the expected noise in upcoming components is small, thus higher granularity becomes desirable. Division by constant 80 is an empirical choice we found to be suitable for our datasets differing in cardinality and trajectory length, but similar to N, choosing a reasonably similar value suffices to remain accurate.

Figure 5 illustrates the AG construction process with initial grid size $N = 2$. First, a $2 \quad 2$ uniform grid is laid. Then, one η is issued per the four initial cells, e.g., $\eta(D, C_1) = 2/4 + 1/3 + 1/2 = 4/3$. Since $\eta(D, C_4) = 13/4 \quad \eta(D, C_3) = 17/12 > \eta(D, C_1) = 4/3 \quad \eta(D, C_2) = 0$, C_4 is divided into $3 \quad 3$ cells, and C_1 and C_3 are divided into $2 \quad 2$ cells. C_2 is not divided any further.

Theorem 1. *Adaptive grid construction satisfies* ε_1*-DP.*

Proof. The AG construction process accesses D via the count queries η. A total of N^2 count queries are issued, denoted by:

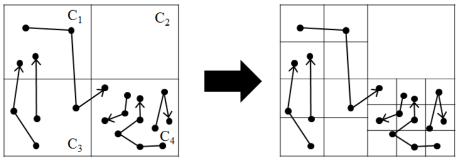
$$W(D) =< \eta(D, C_1), \eta(D, C_2), ..., \eta(D, C_{N^2}) >$$

The answer to each query is perturbed by $Lap(1/\varepsilon_1)$. Per the Laplace mechanism, AG construction satisfies ε_1-DP if $\quad W = 1$. To show $\quad W = 1$, we turn to Definitions 1 and 3. Let D_1 and D_2 be arbitrary neighboring datasets, and wlog $D_2 = D_1 \; [\; f\tau g$ where τ denotes a single trajectory. We need to show $\quad W = \max_{D_1, D_2} jjW(D_2) \quad W(D_1)jj = 1$. We have:

$$\begin{aligned} jjW(D_2) \quad W(D_1)jj &= \sum_{C_i} (\eta(D_2, C_i) \quad \eta(D_1, C_i)) \\ &= \sum_{C_i} ((\sum_{T 2 D_1} + \sum_{T 2 f\tau g}) \quad \sum_{T 2 D_1}) \\ &= \sum_{C_i} \sum_{T 2 f\tau g} \frac{\text{\# of } p_j \text{ in } T \text{ s.t. } p_j \; 2 \; C_i}{jTj} \\ &= \sum_{T 2 f\tau g} \sum_{C_i} \frac{\text{\# of } p_j \text{ in } T \text{ s.t. } p_j \; 2 \; C_i}{jTj} \\ &= \sum_{T 2 f\tau g} \frac{\text{\# of } p_j \text{ in } T}{jTj} = 1 \end{aligned}$$

□

Note that in the definition of η, location counts are divided by the sizes of trajectories jTj. The reason behind this becomes clear from the proof: Without division, the last step of the proof reduces to $\sum_{T 2 f\tau g}(\text{\# of } p_j \text{ in } T)$. However, in the general case this quantity cannot be bounded, leading to unbounded sensitivity



(a) Initial 2x2 uniform grid (b) Resulting adaptive grid

Fig. 5: Adaptive grid construction

$\boldsymbol{W}$. Previous works facing this problem resort to: (i) truncating trajectories, i.e., keep only the first T_{max} points of trajectories and remove the remaining points (where T_{max} is an input parameter), or (ii) random sampling, i.e., randomly choose T_{max} points from each trajectory and remove the rest [39], [40]. Both approaches yield $\quad \boldsymbol{W} = T_{max}$. However, we argue that our approach of representative point identification and division by $\mathcal{T}$ is more appropriate for our goal. The problem with truncating is that by removing all points after T_{max}, we lose significant portions of trajectories that are much longer than T_{max}. (This can be circumvented by having a high T_{max}, but doing so increases sensitivity, beating the purpose of bounding $\quad \boldsymbol{W}$ in the first place.) The problem with sampling is that a random sample might consist of many insignificant, repetitive points with little movement, and totally miss out on larger movements. For example, if we randomly sample an Uber driver's trajectory during rush hour, we could get many nearby locations stuck in traffic and very few locations in areas with light traffic. In contrast, our approach ensures that all representative points are properly considered with equal weight.

5.2 Trip Distribution Extraction

The correlation between trajectories' start and end points is valuable for many purposes such as passenger demand analysis, taxi service prediction, city planning, etc. DP-Star explicitly preserves this correlation by extracting the *trip distribution*. A trip is defined using the start and end points of a trajectory, e.g., for trajectory $\mathcal{T} = p_1 p_2 ... p_{j\mathcal{T}j}$, we denote its trip by $p_1 \quad p_{j\mathcal{T}j}$. Assume that we discretize $\quad (\mathcal{D})$ using grid A. Let $\mathcal{D}_{C_i \quad C_j}$ denote the subset of $\mathcal{D}$ that consists of trajectories in $\mathcal{D}$ that have trip $C_i \quad C_j$. That is, for a trajectory $\mathcal{T}$, we have $\mathcal{T}\ 2\ \mathcal{D}_{C_i \quad C_j}$ if and only if $p_1\ 2\ C_i$ and $p_{j\mathcal{T}j}\ 2\ C_j$. Then, the empirical trip distribution $R_{\mathcal{D},\mathbb{A}}$ is defined as follows:

$$R_{\mathcal{D},\mathbb{A}}((C_i, C_j)) := \begin{cases} \frac{j\mathcal{D}_{C_i \quad C_j}j}{j\mathcal{D}j} & \text{for } (C_i, C_j)\ 2\ \mathsf{A} \quad \mathsf{A} \\ 0 & \text{otherwise} \end{cases}$$

Division by $j\mathcal{D}j$ ensures that R remains a probability mass function. We omit the subscripts in R when they are clear in the context. In DP-Star, we need to extract a differentially private trip distribution $\hat{R}$, which is a noisy version of R. To extract $\hat{R}$, we retrieve $j\mathcal{D}_{C_i \quad C_j}j$ for each pair of cells $(C_i, C_j)\ 2\ \mathsf{A} \quad \mathsf{A}$, and perturb these values by adding $Lap(1/\varepsilon_2)$ to each. We ensure no negative probabilities exist in $\hat{R}$ by post-processing each negative value and making them equal to 0. We give an example of this procedure in Figure 6, in which a 2x2 uniform grid and a dataset of 16 trajectories (as drawn on the left) is assumed.

Theorem 2. *Trip distribution extraction satisfies ε_2-DP.*

Proof. First observe that by construction A does not contain overlapping cells. Hence, the subsets $\mathcal{D}_{C_i \quad C_j}$ are disjoint subsets of $\mathcal{D}$. As a result, parallel composition is applicable and we can fetch each $j\mathcal{D}_{C_i \quad C_j}j$ using budget ε_2 instead of dividing the budget. Second, observe that $j\mathcal{D}_{C_i \quad C_j}j$ can change by at most one if a trajectory is added to or removed from $\mathcal{D}$, hence sensitivity is 1. Combining the two observations, perturbing each $j\mathcal{D}_{C_i \quad C_j}j$ by $Lap(1/\varepsilon_2)$ satisfies ε_2-DP per the Laplace mechanism. □

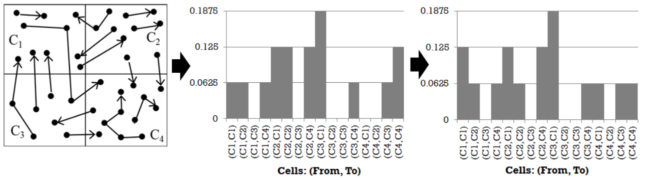


Fig. 6: Trip distribution extraction with a 2x2 uniform grid (Trajectories ! actual trip distribution ! private noisy trip distribution)

5.3 Mobility Model Construction

A vital requirement in synthesizing utility-preserving trajectories is being able to mimic the mobility patterns of actual trajectories. The most widely accepted trajectory mobility model is the Markov model. A Markov model of order r asserts that the next location in a trajectory, say p_{n+1}, depends on $p_{(n \quad r+1)} p_{(n \quad r+2)} ... p_n$ instead of $p_1 p_2 p_3 ... p_n$. From a differential privacy perspective, a low r is preferable, which is explained as follows. Let $r_1 < r_2$ be two Markov orders, and $c(p_i ... p_j)$ denote the number of occurrences of the location sequence $(p_i ... p_j)$ in a dataset. When using the Markov model, we calculate: $Pr[p_{n+1} j p_{(n \quad r+1)} ... p_n]$, which is directly proportional to $c(p_{(n \quad r+1)} ... p_n p_{n+1})$. Since $r_1 < r_2$, we always have $c(p_{(n \quad r_1+1)} ... p_n p_{n+1}) \quad c(p_{(n \quad r_2+1)} ... p_n p_{n+1})$. However, in either case these counts are perturbed with the *same magnitude* of Laplace noise when enforcing DP. Since the counts with r_1 are always larger, noise is less likely to dominate, and their signal-to-noise ratio will be better. However, in the case of large r_2, the true counts of long subsequences are already small, and the added noise dominates.

Motivated by the above, in addition to works showing that low-order Markov models are often as accurate as more complex methods to capture urban mobility [41], [42], and the fact that training high-order models is computationally less efficient; we employ a first-order Markov model in DP-Star. The model consists of: (i) A set of states: a state corresponds to a cell C_i in grid A. (ii) A set of transitions, which represent the probabilities of moving from one state to another. We encode the transition probabilities using an $j\mathsf{A}j \quad j\mathsf{A}j$ transition matrix $\boldsymbol{X}$. Each entry $\boldsymbol{X}_{i,j}$ denotes $Pr[C_j j C_i]$, that is, the probability of the next cell being C_j given the current cell is C_i. By definition, $\sum_j \boldsymbol{X}_{i,j} = 1$ for all i.

We now study the problem of constructing an ε_3-DP Markov model. Since the states are determined by the adaptive grid A, and A is already built privately in a previous component, we focus on obtaining the private transition matrix $\hat{\boldsymbol{X}}$. For a normalized trajectory $\mathcal{T}$ and grid cells C_i, C_j in A, the transition count is:

$$\phi(\mathcal{T}, C_i, C_j) = \frac{\begin{matrix}\#\text{ of consecutive points } (p_a p_{a+1}) \text{ in } \mathcal{T} \\ \text{such that } p_a\ 2\ C_i \text{ and } p_{a+1}\ 2\ C_j\end{matrix}}{j\mathcal{T}j \quad 1}$$

The process of obtaining $\hat{\boldsymbol{X}}$ requires the following steps:

1) Build $j\mathsf{A}j \quad j\mathsf{A}j$ matrix $\boldsymbol{O}_{i,j} = \sum_{\mathcal{T} 2 \mathcal{D}} \phi(\mathcal{T}, C_i, C_j)$.
2) Build $j\mathsf{A}j \quad j\mathsf{A}j$ noise matrix $\boldsymbol{I}_{i,j} = Lap(1/\varepsilon_3)$.
3) Let $\hat{\boldsymbol{O}} = \boldsymbol{O} + \boldsymbol{I}$. If $\hat{\boldsymbol{O}}_{i,j} < 0$, set $\hat{\boldsymbol{O}}_{i,j} = 0$.
4) Compute $\hat{\boldsymbol{X}} = \frac{\hat{\boldsymbol{O}}_{i,j}}{\sum_j \hat{\boldsymbol{O}}_{i,j}}$.

For example, let D and A consist of the trajectories and grid in Figure 5a. We have: $O_{3,4} = 1/3$ and $O_{3,1} = 1/1 + 1/2 = 3/2$.

Theorem 3. *Mobility model construction satisfies ε_3-DP.*

Proof. Due to I, each entry in O is perturbed with $Lap(1/\varepsilon_3)$. Thus it suffices to show that $O = 1$, i.e., adding/removing a trajectory to D causes a total weight change of at most 1 in O. Similar to the proof of Theorem 1, wlog we define neighboring datasets D_1 and D_2 as $D_2 = D_1 [f\tau g$ where τ denotes a single trajectory. We have:

$$O = \sum_i \sum_j \sum_{T 2 D_2} \phi(T, C_i, C_j) \quad \sum_i \sum_j \sum_{T^0 2 D_1} \phi(T^0, C_i, C_j)$$

Following the intermediate steps in Theorem 1 and the fact that $D_2 \quad D_1 = f\tau g$ it is easy to show that:

$$O = \sum_i \sum_j \phi(\tau, C_i, C_j)$$

By definition of ϕ, we have $\sum_i \sum_j \phi(\tau, C_i, C_j) = 1$ for any trajectory τ. Hence $O = 1$, which completes our proof. □

5.4 Route Length Estimation

When generating trajectories, DP-Star needs to determine their route length (distance travelled). We heuristically approximate route length using the number of representative points contained in a trajectory. In order to give a data-aware decision on route lengths, for each pair of cells $(C_i, C_j)\ 2\ A \quad A$, DP-Star retrieves and stores the median length of the trajectories that start in C_i and end in C_j. We denote this by $\hat{\ell}_{C_i \quad C_j}$. We adapt the following private median retrieval procedure proposed in [11].

Definition 7 (Median mechanism [11]). *Let $K = fx_1, x_2, .., x_{jKj}g$ be a multiset of values in non-decreasing order in some range $x_i\ 2\ [lo, hi]$, and m be the median value of K. For any $x\ 2\ [lo, hi]$, let $rank(x)$ denote the rank of x in K. The median mechanism M returns x as the noisy median with the following probability:*

$$Pr[M(K) = x] \ / \ e^{\frac{\varepsilon}{2} jrank(x) \quad rank(m)j}$$

where ε is the privacy budget allocated to M.

We observe that M is an invocation of the exponential mechanism with scoring function $q := \quad jrank(x) \quad rank(m)j$. The intuition is that if x is close to m, then its rank will be similar to the rank of m, thus the score of each candidate x is negatively proportional to how much its rank deviates from the rank of m. Clearly we have $q = 1$. We also note that the domain $[lo, hi]$ in our case is determined by the number of representative points in trajectories. Since this quantity is always a non-negative integer, our domain is naturally discrete, and the exponential mechanism is applicable. We use $lo = 0$, and $hi = 1.25 \quad jT_{max}j$ as a safe upper bound, where $jT_{max}j$ is the number of representative points in the longest trajectory in D.

For each $(C_i, C_j)\ 2\ A \quad A$, to compute $\hat{\ell}_{C_i \quad C_j}$ we first build K as $K = \bigcup_{T 2 D_{C_i \quad C_j}} jTj$. Then, we sort K in non-decreasing order. Finally, we invoke M with inputs K and ε_4. The returned result is assigned to $\hat{\ell}_{C_i \quad C_j}$. For example, let D and A consist of the data in Figure 5a, and let us compute $\hat{\ell}_{C_4 \quad C_4}$. Then, we have $K = f3, 3, 5g$, and the actual median is 3. The noisy median is returned via M probabilistically.

Theorem 4. *Route length estimation satisfies ε_4-DP.*

Algorithm 1: Generating a synthetic trajectory dataset

```
Input : Adaptive grid $A$, trip distribution $\hat{R}$, transition
        matrix $\hat{X}$, set of median lengths $\hat{L}$, number of
        synthetic trajectories desired $nSyn$
Output: Synthetic trajectory dataset $S_D$
Initialize $S_D$ as empty set
for $i = 1$ to $nSyn$ do
    Pick a sample $S = (C_{start}, C_{end})$ from $\hat{R}$
    Retrieve $\hat{\ell}_{C_{start} \quad C_{end}}$ from $\hat{L}$ and assign to $\hat{\ell}$
    Pick a random sample $s$ from $exp(\frac{ln2}{\hat{\ell}})$
    Build trajectory $T^0 : C^0_1\ !\ C^0_2\ !\ ...\ !\ C^0_s$, where
      $C^0_1 = C_{start}$ and $C^0_s = C_{end}$
    for $j = 2$ to $s \quad 1$ do
        Let $C_{prev}\ 2\ A$ denote $C^0_{j \quad 1}$ of $T^0$
        Sample $C_{samp}$, where $8C_k\ 2\ A$ the probability of
          being sampled as $C_{samp}$ is $/\ \hat{X}^{s \quad j}_{k,end} \quad \hat{X}_{prev,k}$
        Set $C^0_j$ of $T^0$ as $C_{samp}$
    end
    for $j = 1$ to $s$ do
        Convert $C^0_j$ in $T^0$ to point $p^0_j$ by uniformly
          randomly emitting a location in $C^0_j$
    end
    Add $T^0$ to $S_D$
end
return $S_D$
```

Proof. First observe that we invoke the median mechanism once for each unique $\hat{\ell}_{C_i \quad C_j}$, and each invocation relies only on $D_{C_i \quad C_j}$. In the proof of Theorem 2 we had established that $D_{C_i \quad C_j}$ are disjoint subsets, hence parallel composition is applicable here. It remains to show that an invocation of the median mechanism with ε_4 satisfies ε_4-DP. Since the median mechanism is equivalent to the Exponential mechanism with a specific quality function, by Definition 5 our proof is complete. □

5.5 Synthetic Trajectory Generation

In this section we describe how DP-Star generates complete synthetic trajectories using the features it has learned so far. We give a technical description of trajectory generation in Algorithm 1 and explain it step-by-step below. Theorem 5 states the ε-DP guarantee offered by DP-Star's trajectory generation.

Step 1: Generate start and end cells. DP-Star draws a random sample from the trip distribution $\hat{R}$. Let the returned sample be denoted by the pair of cells (C_{start}, C_{end}). For example, sampling from Fig. 6 would return $(C_{start}, C_{end}) = (C_1, C_1)$ with probability 0.125, and $(C_{start}, C_{end}) = (C_1, C_2)$ with probability 0.0625. Since trips are determined in the very first step, we ensure that the trip distribution of the actual dataset D is accurately modeled in synthetic dataset S_D.

Step 2: Determine route length. When synthesizing a trajectory, we face two challenges: determining its length and determining the coordinates of its points. The current step is concerned with the prior problem, which can be stated formally as follows: Given a synthetic trajectory T^0 with trip $C_{start} \quad C_{end}$, how do we decide on jT^0j? To answer this question, we turn to relevant recent works [29], [43]. We find that trajectory lengths in urban trajectory datasets can be approximated using an exponential distribution. The exponential distribution is also desirable because it has a single parameter λ, which is directly related to the median of the

TABLE 1: Datasets used in our experiments

	$\lvert D\rvert$	$\Omega(D)$	$\lvert T\rvert$ (mean ± stdev)	GPS sampling rate	$\lvert\tilde{T}\rvert$ (mean ± stdev)
Geolife	14,650	Beijing, China	931.4 ± 1602.5	~ 2.5 seconds	38.8 ± 42.9
Taxi	30,000	Porto, Portugal	43.1 ± 33.5	~ 15 seconds	8.9 ± 4.7
Brinkhoff	50,000	Oldenburg, Germany	64.6 ± 35.9	~ 15.6 seconds	11.5 ± 5.6

distribution m, as: $\lambda = {}^{ln2}/_{m}$. Since we had already calculated median lengths, here we only build the exponential distribution with $\lambda = {}^{ln2}/_{\ell_{C_{start} \quad C_{end}}}$ and sample from it. The returned sample, denoted s, becomes the length of our synthetic trajectory T^0. Note that length is determined after trip generation. This allows DP-Star to take into account trajectories' start and end points when determining their length, and better serve cases where certain trip routes being longer than others.

Step 3: Synthesize trajectory as a sequence of cells. Now that the length $jT^0j = s$ and (C_{start}, C_{end}) are known, T^0 can be initialized as: $T^0 : C_1^0 \; ! \; C_2^0 \; ! \; ... \; ! \; C_s^0$, where $C_1^0 = C_{start}$ and $C_s^0 = C_{end}$. Then, the next challenge is to determine the unknown intermediate cells C_2^0 to $C_{s \quad 1}^0$. To solve this, we iterate from $2 \quad j \quad s \quad 1$, and calculate the probability of C_j^0 being a certain cell $C_k \; 2 \; A$ using the Markov mobility model. We denote the previous cell $C_{j \quad 1}^0$ as $C_{prev} \; 2 \; A$, and derive:

$$Pr[C_j^0 = C_k \, j \, C_{j \quad 1}^0, C_s^0] = \frac{Pr[C_k, C_{prev}, C_{end}]}{Pr[C_{prev}, C_{end}]}$$
$$/ \quad Pr[C_{end} \, j \, C_k, C_{prev}] \quad Pr[C_k \, j \, C_{prev}] \quad Pr[C_{prev}]$$
$$/ \quad Pr[C_{end} \, j \, C_k] \quad Pr[C_k \, j \, C_{prev}] = \hat{X}_{k,end}^{s \quad j} \quad \hat{X}_{prev,k}$$

Here, $\hat{X}$ denotes the 1-step transition matrix and $\hat{X}^{s \quad j}$ denotes the $(s \quad j)$-step transition matrix. Note that this is essentially a random walk on the Markov mobility model, where the initial and final cells are fixed. It is straightforward to obtain the $(s \quad j)$-step transition matrix from the 1-step transition matrix. However, for a fixed j observe that which transition matrix is used depends on $s \quad j$, thus the trajectory length s plays a critical role in determining the transition matrix and probabilities within, i.e., DP-Star directly considers trajectory lengths in its random walk.

Step 4: Convert from cells to locations. At this point a synthetic trajectory T^0 is available as a sequence of cells. In the last step we convert from cells to locations by randomly sampling a location within each corresponding cell. The resulting sequence of locations constitutes the final trajectory, which is added to S_D.

Theorem 5. *Synthetic trajectory generation satisfies* ε*-DP.*

Proof. Trajectory generation has 4 data-dependent inputs: A, $\hat{R}$, $\hat{X}$, $\hat{L}$. By Theorems 1-4, constructing all necessary inputs satisfies ε-DP for $\varepsilon = \sum_{i=1}^{4} \varepsilon_i$. Then, since trajectory generation only performs post-processing on private inputs, it satisfies ε-DP. □

6 Experimental Evaluation

6.1 Experiment Setup

Datasets. We used three datasets in our experiments: Geolife, Taxi and Brinkhoff. Geolife and Taxi are real datasets, whereas Brinkhoff is simulated. Information regarding the datasets is summarized in Table 1. Geolife was collected and released by Microsoft as part of the Geolife project [44]. It contains GPS traces of 182 users over 5 years. Majority of the data is from Beijing, China, with few outliers in other cities in China, Europe, etc. We pre-processed the data to remove these outliers and obtain a more even data distribution and well-defined (D). The resulting dataset contained 14,650 trajectories. Taxi contains GPS trajectories of taxis operating in the city of Porto, Portugal. The data was made available as part of the Taxi Service Prediction Challenge at ECML-PKDD 2015 [45]. Since the original data is large and scattered we extracted 30,000 trajectories from the denser areas. The spatial density of these trajectories is illustrated in Figure 1. Brinkhoff contains trajectories simulated using Brinkhoff's network generator for moving objects [46]. The map of Oldenburg, Germany was used to simulate 50,000 trajectories. Locations were sampled at equal time intervals.

Competitors. We implemented DP-Star in Java, and performed experiments on a laptop with Intel i7 CPU and 16 GB memory. We compare DP-Star with two most relevant works DPT and ngram. We obtained their implementations from the respective authors [15], [16]. When applicable, we used the parameters recommended by the authors. Otherwise, we experimented with different parameter values and report only the best results in our paper. Since Laplace and Exponential mechanisms are probabilistic, all experiments are repeated 15 times and results are averaged.

6.2 Evaluation Metrics

We generated synthetic datasets S_D with cardinality $jS_Dj = jDj$, and used the following metrics to quantify the resemblance (or difference) between D and S_D. High resemblance (low difference) is desired for improved utility, considering DP-Star's competitors also guarantee the same level of privacy (ε-DP) as DP-Star.

Query Answering. A popular method for evaluating data publishing algorithms based on anonymization or ε-DP is measuring accuracy in answering counting queries. We consider spatial count queries of the form: "Return the number of trajectories passing through a circular region with center c and radius r". Let Q denote a query of this form, and $Q(D)$ denote its answer when issued on dataset D. The *relative error (RE)* of Q is:

$$RE = \frac{jQ(D) \quad Q(S_D)j}{max fQ(D), bg}$$

where b is a sanity bound that mitigates the effect of queries with extremely high selectivities [15]. We set $b = 1\% \quad jDj$ as this value is in parallel with what is often used in the literature [3], [13], [15] and yields meaningful results considering the cardinalities of our datasets. We generate 500 random queries and compute the average relative error (*AvRE*) by averaging the individual RE of all queries.

Frequent Pattern (FP) Mining. Trajectory FP mining is a well-studied problem with many applications. Using the following metrics, we measure how well S_D preserves the FPs in D. For each trajectory in a dataset, we first project its path on a uniform grid U, thus obtaining the sequence of cells it passes through. We define a pattern P as an ordered list of cells, e.g., $P : C_2 \; ! \; C_4 \; ! \; C_3$. We define the support of a pattern, $supp(D, P)$, as the number of occurrences of P in D. We denote the top-k patterns in D, i.e., the k patterns with highest support, by $F_U^k(D)$.

Our first metric measures differences in patterns' support. For each pattern $P \; 2 \; F$, we find the relative difference between $supp(D, P)$ and $supp(S_D, P)$. Then, the FP average relative error is calculated as:

$$FP \ AvRE = \frac{\sum_{P 2 F_U^k(D)} \frac{jsupp(D,P) \quad supp(S_D,P)j}{supp(D,P)}}{k}$$

Our second metric measures the similarities and discrepancies between the *ranking* of FPs in D and S_D. We say that the ranking of a pair of patterns P_i, P_j is concordant if either of the following is true, and discordant otherwise:

$$(supp(D,P_i) > supp(D,P_j)) \wedge (supp(S_D,P_i) > supp(S_D,P_j))$$
$$(supp(D,P_i) < supp(D,P_j)) \wedge (supp(S_D,P_i) < supp(S_D,P_j))$$

That is, they are concordant if their ranks in sorted order agree in D and S_D, and discordant otherwise. The Kendall-tau coefficient can then be applied as:

$$KT = \frac{\text{(\# of concordant pairs)} \quad \text{(\# of discordant pairs)}}{n(n \quad 1)/2}$$

In our experiments we used $k = 50$ to focus on the most prominent patterns with significant support. We only considered patterns that are at least 3 cells long. We assumed a 6x6 uniform grid U for FP metrics and the metric below.

Trip Error. Many trajectory datasets are made up of trips, e.g., taxi trips, Uber trips, home to work commutes. Preservation of the start and end regions of these trips is critical for applications such as passenger demand analysis, city planning, residential/workplace annotation, etc. To measure errors in trip preservation, given a grid U, the private dataset D and the synthetic dataset S_D, we first compute the empirical trip distributions $R_{D,U}$ and $R_{S_D,U}$. Then, trip error is defined as: $JSD(R_{D,U}, R_{S_D,U})$, where $JSD(,)$ denotes the Jensen-Shannon divergence.

Diameter Error. We adopt this metric from [16]. The diameter of a trajectory T is defined as the maximum distance between any pair of points in T, i.e., $max_{i,j} len(p_i p_j)$ for i, j 2 [1..jTj]. We quantize diameters into 20 equi-width buckets $f[0,x), [x,2x), ..., [19x, 20x]g$, where $20x$ is the longest diameter present in D. Let $E(D)$ be the empirical, bucketized diameter distribution of trajectories in D. Then, the diameter error is calculated as: $JSD(E(D), E(S_D))$.

6.3 Comparison with Previous Work

In Table 2 we compare DP-Star with its competitors, while varying ε between 0.1 and 2.0. We first observe that DP-Star often provides 2-3 fold better utility than DPT and ngram with respect to all metrics. In a few cases its competitors do outperform DP-Star in the strictest privacy setting of $\varepsilon = 0.1$. This can be attributed to DP-Star's division of the privacy budget – when DP-Star divides an already small $\varepsilon = 0.1$ budget into 4 sub-budgets to use in its components, each of the components incur significant error. Second, we note that in most cases, errors decrease when the privacy budget ε is increased. This is an expected result since higher ε yields lower privacy but better expected utility. However, we also see that in some cases this expectation does not hold, which is because some metrics are not explicitly preserved by any of the competitors. For example, none of the works explicitly try to minimize diameter error, therefore a decrease in ε does not necessarily have a direct consequence on diameter error.

Next, we analyze the results of each evaluation metric one by one, starting with query error. We observe that query error is heavily related to the spatial data distribution (Figure 1). This relation can be confirmed by observing that visual distributions in Figure 1 imply DP-Star > ngram > DPT, which is confirmed by the Taxi dataset numbers in Table 2. In terms of FP AvRE, the reasons why DP-Star performs better than DPT and ngram are different for each competitor. In DPT's case, the cause of error is that $supp(S_D, P) \quad supp(D, P)$, which is supported by Figure 1 - since sparse regions become dense, they steal from the support of the frequent patterns in D. In ngram's case, we obtain $supp(S_D, P) \quad supp(D, P)$, which again yields high error. This behavior is due to ngram's high-pass filtering: Since regions with low counts are deleted, synthetic data is generated only from the high-count regions. Hence, dense regions become excessively dense, as shown in Figure 1. Our second metric for measuring FP accuracy is Kendall-tau. In majority of cases, Kendall-tau results agree with FP AvRE results, showing a strong correlation. More notably, even in cases where FP AvRE is high, Kendall-tau values remain positive, demonstrating that many of the FP rankings are actually preserved despite having different support values.

In terms of trip error, DP-Star's superiority is not surprising, since it explicitly aims to preserve the trip distribution of D. In terms of diameter error, DP-Star's low diameter errors imply that our strategy of route length estimation and approximation via the exponential distribution performs well in practice. In addition, in our experiments we observed that both DPT and ngram are biased towards generating synthetic trajectories with low displacement. That is, even though their trajectories' lengths can be higher, since they contain loops and U-turns, their displacement (and hence their diameter) is often lower. In contrast, DP-Star's synthetic trajectories contain few U-turns or loops, and are more realistic.

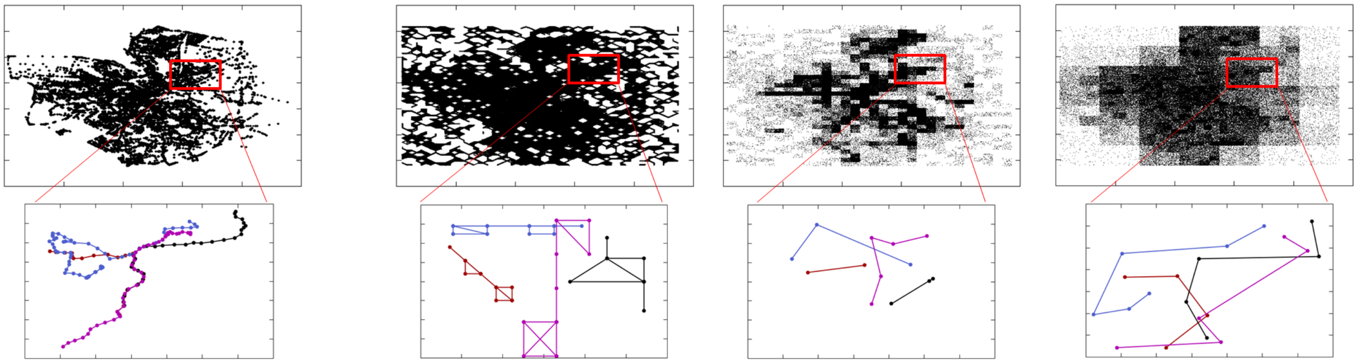
In Figure 7, we illustrate the last claim and perform a visual comparison. At the top, we illustrate the spatial density of Brinkhoff and the synthetic datasets published by DPT, ngram and DP-Star respectively. Then, we zoom into one specific region, and extract four sample trajectories from that region. Although no one-to-one correspondance exists between these trajectories, we color them individually based on their visual similarity, for ease of comparison. We observe that DPT's output trajectories contain diamond-shaped movements and dead-ends, which are caused by loops and U-turns that make a synthetic trajectory re-visit locations that it has previously visited. Such behavior is almost never observed in actual trajectories, making DPT's trajectories unrealistic. ngram's spatial distribution implies that its trajectories sit in small, dense cells. We confirm this observation by studying ngram's individual trajectories, and see that these trajectories are shorter and cover a smaller geographic region. DP-Star's spatial distribution exemplifies its cell-based nature, since spatial distributions change by rectangular regions. In general, DP-Star's trajectories are realistic. Note that while actual trajectories have detailed, fine granularity movements; DP-Star's trajectories appear less smooth, i.e., they have sharp edges and less curvature. Our use of MDL in DP-Star sacrifices finer granularity movements and smoothness, but in turn, it enables us to more accurately capture the coarser movements with high displacement.

6.4 Evaluation of Components and Design Objectives

In this section, we evaluate DP-Star's components that impact the two design objectives stated in Section 3.3. Our first design objective was effective discretization of $\quad(D)$, for which we designed the adaptive grid component. We analyze the behavior of this component with respect to parameter N, the initial grid size. We plot the results in Figure 8. Since results show similar characteristics across all metrics, we only include the results with the query error metric. We observe that on all three datasets, errors plateau and minimize between $N = 5$ to 9, hence our empirical choice of $N = 7$ is justified. Also, since many nearby values of N perform similarly, we can conclude that a rough choice of N is sufficient for high utility overall.

TABLE 2: Comparing DP-Star with its competitors

		Geolife				Taxi				Brinkhoff			
		ε=0.1	ε=0.5	ε=1.0	ε=2.0	ε=0.1	ε=0.5	ε=1.0	ε=2.0	ε=0.1	ε=0.5	ε=1.0	ε=2.0
Query AvRE	ngram	0.589	0.511	0.488	0.450	0.304	0.270	0.268	0.243	0.297	0.210	0.176	0.155
	DPT	0.361	0.319	0.303	0.272	0.433	0.415	0.381	0.359	0.405	0.388	0.358	0.289
	DP-Star	**0.295**	**0.203**	**0.171**	**0.159**	**0.176**	**0.128**	**0.120**	**0.110**	**0.202**	**0.154**	**0.145**	**0.126**
FP AvRE	ngram	0.458	0.416	0.413	0.395	**0.314**	0.309	0.295	0.284	0.568	0.535	0.499	0.421
	DPT	0.395	0.377	0.363	0.342	0.602	0.573	0.564	0.554	0.461	0.419	0.373	0.282
	DP-Star	**0.340**	**0.329**	**0.322**	**0.309**	0.318	**0.261**	**0.228**	**0.215**	**0.343**	**0.279**	**0.251**	**0.241**
FP Kendall-tau	ngram	0.24	0.32	0.41	0.47	**0.66**	0.69	0.71	0.74	0.28	0.43	0.47	0.55
	DPT	**0.44**	0.48	0.52	0.59	0.32	0.43	0.50	0.54	0.42	0.51	0.56	0.60
	DP-Star	0.42	**0.52**	**0.64**	**0.68**	0.64	**0.77**	**0.81**	**0.81**	**0.48**	**0.59**	**0.64**	**0.68**
Trip Error	ngram	0.455	0.453	0.441	0.410	0.266	0.231	0.225	0.218	0.189	0.156	0.149	0.139
	DPT	0.421	0.397	0.376	0.359	0.459	0.457	0.433	0.429	0.270	0.258	0.244	0.232
	DP-Star	**0.071**	**0.054**	**0.034**	**0.017**	**0.053**	**0.031**	**0.017**	**0.011**	**0.054**	**0.039**	**0.031**	**0.025**
Diameter Error	ngram	0.131	0.125	0.124	0.123	0.234	0.239	0.231	0.235	0.195	0.151	0.132	0.118
	DPT	0.261	0.253	0.250	0.249	0.385	0.371	0.362	0.358	0.208	0.205	0.179	0.180
	DP-Star	**0.103**	**0.089**	**0.078**	**0.076**	**0.054**	**0.026**	**0.022**	**0.022**	**0.078**	**0.034**	**0.030**	**0.027**



(a) Actual **(b) DPT** **(c) ngram** **(d) DP-Star**

Fig. 7: Visual comparison of DPT, ngram and DP-Star. Spatial densities of the Brinkhoff dataset and the synthetic (published) datasets are illustrated at the top. We zoom into one region and study four of the actual and synthetic trajectories in that region. For ease of comparison, trajectories are hand-colored according to visual similarity.

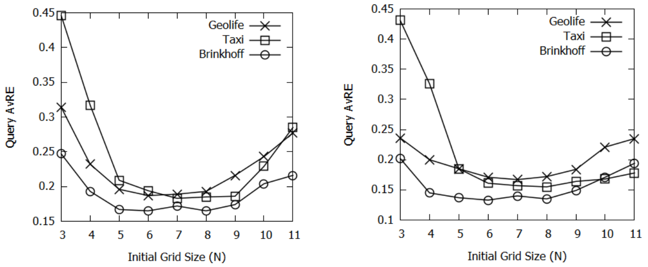


Fig. 8: Impact of N on query error (ε = 1.0, 2.0)

Our second design objective was effective preservation of spatial utility. Multiple components work towards this objective, e.g., trip distribution, Markov model, etc. Since previous literature showed that some components such as the Markov model perform well, our evaluation focuses on DP-Star's non-conventional feature: MDL-based representative point identification. For this analysis we implement two variants of DP-Star. In the first variant we do not use MDL and assign $\hat{T} = T$ after pre-processing. In the second variant we remove trajectories' intermediate points and keep only the start and end points, i.e., for $T = p_1 p_2 ... p_{jTj}$, we have $\hat{T} = p_1 p_{jTj}$. First variant represents maximal preciseness, second variant represents maximal conciseness, and the original MDL-enabled DP-Star represents the desirable trade-off.

In Figure 9 we show how DP-Star compares with the two variants. In summary, MDL is beneficial: It has roughly no negative impact on query error and trip error, significantly improves FP mining error, slightly improves diameter error, and greatly reduces execution time. In terms of query error, DP-Star performs similar with and without MDL, but Max-Concise performs significantly worse because it disregards trajectories' intermediate points. In terms of FP mining error, interestingly, DP-Star with MDL outperforms its variants. The poor performance of Max-Concise is because trajectories' intermediate regions, which often contain the FPs, are disregarded. To explain the poor performance of *No MDL*, we examine DP-Star's Markov mobility model: When constructing $\hat{\boldsymbol{X}}$, we use the transition count query ϕ which, by definition divides inter-cell transition occurrences by $j\hat{T}j$ 1. Without MDL, $\hat{T}$ is large and contains redundant intra-cell transitions. Thus, inter-cell transitions are represented with smaller weight in $\boldsymbol{O}$, and are affected more by the added Laplace noise $\boldsymbol{I}$. Whereas in DP-Star with MDL, redundant intra-cell moves are often removed and inter-cell moves are kept. As such, the absolute count of inter-cell moves is roughly the same with or without MDL, but since $j\hat{T}j$ is smaller under MDL, the weights of inter-cell moves are higher, thus the same noise matrix $\boldsymbol{I}$ does not distort the weights as much.

In terms of trip error, all three variants perform the same since trajectories' start and end points are not impacted by MDL. In terms of diameter error, DP-Star's superiority is because of its route length estimation component. In case of Max-Concise, trajectories' routes do not contain intermediate points, thus their median length is small. As such, synthetic trajectories end up having excessively short length. In case of *No MDL*, synthetic trajectories are longer and contain many intermediate points.

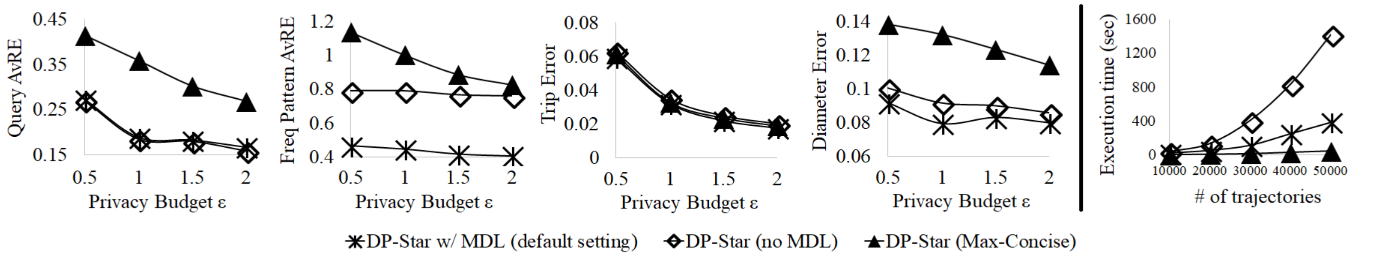
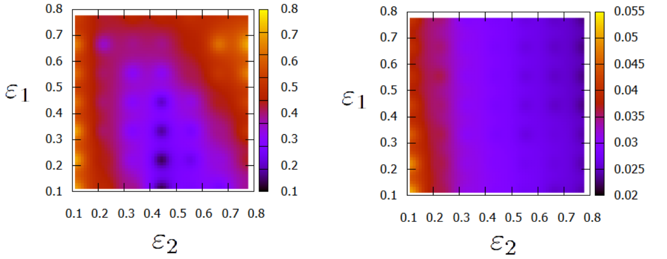


Fig. 9: Impact of representative point identification on utility and execution time. Utility metrics (first four graphs) are with Geolife, execution time experiment (rightmost graph) is with Brinkhoff since it is the largest dataset.



(a) Query AvRE (b) Trip Error

Fig. 10: Measuring how the budget distribution (ε_1, ε_2) impacts DP-Star's errors, where total $\varepsilon = 1.5$. Darker colors indicate lower error. (Brinkhoff dataset)

Having many intermediate points brings the risk of having an "off" location in an unrealistic, far away cell during the random walk on $\hat{X}$, which is caused by the noise in $\hat{X}$. In such cases, the diameter of a synthetic trajectory ends up being much longer than usual, thereby increasing diameter error.

Finally, we study the efficiency of DP-Star. We report execution time on the rightmost graph of Figure 9. We observe a clear performance improvement of using MDL over *No MDL*, thus MDL is beneficial for efficiency. We remark that publishing 50,000 trajectories takes less than 7 minutes with DP-Star. Since data publishing is an offline task with no performance impact on a real-time system, we conclude that DP-Star's execution time is reasonable, and it is feasible to publish large, real-world datasets.

6.5 Impact of Privacy Budget Distribution

Recall from Section 4.2 that a critical parameter in DP-Star's execution and performance is the distribution of the privacy budget ε into sub-budgets ε_i. In the experiments so far, we chose the budget distribution that yields the highest utility, using the strategy in Section 4.2. In this section, we study how manually changing the budget distribution affects DP-Star. We fix the total budget to $\varepsilon = 1.5$ and vary two of the sub-budgets: ε_1 for adaptive grid construction and ε_2 for trip distribution extraction. The remaining budget $1.5 \quad \varepsilon_1 \quad \varepsilon_2$ is distributed to the mobility model construction and route length estimation components in a 75%-25% allocation, in parallel with the original distribution. The results are illustrated in Figure 10.

Studying query error in Figure 10a, we observe that our choice of $\varepsilon_1 = \varepsilon/9 \quad 0.17$ and $\varepsilon_2 = 3\varepsilon/9 \quad 0.5$ roughly yields the lowest error. Interestingly, for small ε_2, query error is high no matter the value of ε_1, because trip distributions are too inaccurate. This indicates that only having an accurate grid for spatial distribution, or an accurate mobility model for intra-trajectory movement are not sufficient for accurate trajectory synthesis. Trajectories' start and end points should also be preserved. Another observation is that if ε_1 and ε_2 are significantly larger than our suggested values, this negatively impacts query error because they steal from the budgets of the remaining components of DP-Star. From the trip error in Figure 10b, we can confirm that DP-Star's components are responsive to changes in their allocated budgets. The primary component in DP-Star affecting trip error is the trip distribution extraction component, which uses sub-budget ε_2. Thus, an increase in ε_2 decreases trip error. In addition, we observe that ε_1 also has a small impact on trip error, which is because the trip distribution component relies on the adaptive grid A as input. As such, a more accurate grid can improve trip error.

7 Discussion

In this section we discuss various properties, strengths, and limitations of DP-Star in light of our experimental evaluation. We hope that this discussion provides research avenues for future work.

Privacy guarantees. DP-Star satisfies ε-DP as a whole, which guarantees that an adversary observing DP-Star's outputs will not be able to infer any user's participation or trajectory content in D with strong confidence. After Dwork showed the impossibility to achieve absolute disclosure prevention [47], this formal ε-DP guarantee has become a popular standard for privacy protection. However, we acknowledge that ε-DP is not necessarily a universal remedy. DP-Star naturally inherits some of DP's shortcomings, such as the inability to bound all prior and posterior knowledge distributions of an adversary, and the inability to exploit adversarial uncertainty to improve the privacy bounds. Some extensions of DP, such as coupled-worlds privacy [48] and Pufferfish privacy [49], address such shortcomings. Yet, these works are mostly theoretical in nature, and many of the practical building blocks and mechanisms of DP can no longer be used. Therefore their integration to DP-Star is non-trivial, but should nevertheless be considered in future work. Another direction for future work is to quantify and establish the resilience of DP-Star's output trajectories against specific location privacy-related attacks such as Bayesian attacks or threats stemming from outlier behavior.

Utility guarantees. It is possible to give formal utility bounds for some components of DP-Star. In particular, we use the notion of (α, γ)-usefulness to prove a utility bound for the trip distribution. We say that the noisy trip distribution $\hat{R}$ is (α, γ)-useful with respect to the actual distribution R, if with probability at least $1 \quad \gamma$, for each (C_i, C_j) in the grid, it holds that:

$$jj\hat{R}(C_i, C_j) \quad R(C_i, C_j)jj \quad \frac{\alpha}{jDj}$$

Given the privacy budget ε_2 dedicated to the trip distribution and the confidence bound γ, we can derive the matching α value as $\alpha = \quad {}^{ln(\gamma)}/_{\varepsilon_2}$ using the CDF of the Laplace distribution. To exemplify this bound: On the Taxi dataset, with a strong probability of 90% and a reasonable privacy budget of $\varepsilon_2 = 0.5$, the noisy trip distribution will deviate from the actual distribution at position (C_i, C_j) by only ${}^{\alpha}/_{jDj} = {}^{4.61}/_{30000} \quad 1.5x10 \quad {}^{4}$.

Furthermore, for DP-Star's route length estimation component, we can show that our use of the median mechanism is exponentially unlikely to return highly suboptimal results, i.e., probability that the noisy median $\hat{\ell}$ deviates from the actual median ℓ is exponentially disproportional to their difference $|\hat{\ell} - \ell|$ [11], [33]. As a result, DP-Star is unlikely to lose significant utility due to estimating actual medians by noisy medians.

Although formal utility guarantees can be given for individual components in the above fashion, the produced trajectories S_D rely on a complex algorithmic combination of multiple components. As such, it is difficult to give formal utility guarantees between D and S_D. Also, most utility guarantees intrinsically depend on the properties of D, such as dataset size, skewness, geographic coverage, etc. This is why we performed an empirical study to compare D and S_D instead of relying on theoretical properties. Our results showed that DP-Star outperforms its competitors that give comparable privacy guarantees (ε-DP).

Handling small ε. Our experiments unveil that for very small ε, e.g., ε=0.1, DP-Star can seldomly get outperformed by its competitors. There are two reasons for this. First, DP-Star is inversely impacted by small ε because it divides its total budget into 4 sub-budgets that are consumed by its core components. When an already small ε is divided into sub-budgets, noise increases super-linearly with respect to the decreasing budget, hence the negative utility impact is more significant. The second reason is regarding dataset characteristics. We observed that in Taxi and Geolife datasets, the distributions of frequent pattern support are more skewed compared to Brinkhoff, which explains why DP-Star is outperformed only on these datasets and under FP metrics. The skewness is difficult to preserve due to DP-Star's use of a low-order Markov chain, as well as DP's uniform nose addition behavior, e.g., we add a uniform amount of noise to all transition probabilities in the Markov chain which smooths the skewness.

There are several ways in which DP-Star can be improved in future work. We observe that when DP-Star performs less superior, it is only in terms of frequent pattern metrics (e.g., FP AvRE and Kendall-tau) but not others. In future work we should therefore investigate improving DP-Star's frequent pattern preservation ability for low ε, possibly using higher order Markov models or data structures such as prefix/suffix trees, lattice-based methods, and so forth. Another direction for future work could be to improve DP-Star's route length estimation component. DP-Star currently uses an Exponential distribution powered by median lengths, and although this was shown to be an effective strategy [29], [43], there could be datasets where other distribution shapes fit better. To find the best fit, we can initialize multiple distributions (e.g., Exponential, uniform, normal) and use a goodness of fit test.

Choice of utility notions. In DP-Star we chose to preserve and evaluate certain notions of trajectory utility, such as spatial densities, frequent travel patterns, and spatio-temporal travel metrics (e.g., trip correlation, trajectory diameter). We expect these notions to be useful in many kinds of applications. For example, spatial densities are useful for point-of-interest extraction, hotspot discovery, and location recommendation. Frequent travel patterns are useful for discovering associations between road segments, benefiting road network performance analysis. Spatio-temporal travel metrics have commercial benefits including taxi service prediction, passenger demand analysis, etc. However, since utility is often subjective and application-dependent, we cannot claim that our set of evaluation metrics is complete. In particular, both our utility notions and evaluation metrics are statistical in nature. This is expected, since ε-DP is a statistical privacy notion. There are strengths and weaknesses of using statistical notions and metrics, e.g., even when S_D preserves the aggregate traits in D, one-to-one similarity between trajectories in D and S_D can be small. This could decrease utility by making it difficult for a data analyst to learn the microscopic movements of a specific user. Conversely, one could also argue that this is beneficial for privacy protection, since the user's mobility is better hidden from a potential adversary.

8 Conclusion

As urban trajectory and mobility data becomes pervasive, privacy-preserving sharing of this data will attract increasing attention for deriving deep insights. In this paper we presented DP-Star, a framework for differentially private and utility preserving publishing of trajectory data. In a nutshell, DP-Star generates and publishes synthetic trajectories using key factors and statistics learned from the original input data. These statistics are chosen and processed with differential privacy in mind, such that the added privacy does not destroy utility. Our experiments show that DP-Star's output trajectory datasets have significantly higher utility than existing approaches that satisfy differential privacy.

Acknowledgments

This research was partially sponsored by NSF under grants SaTC 1564097 and an IBM Faculty Award. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies and companies mentioned above.

References

[1] B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity: Architecture and algorithms," *IEEE Transactions on Mobile Computing*, vol. 7, no. 1, pp. 1–18, 2008.

[2] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*. ACM, 2003, pp. 31–42.

[3] M. E. Nergiz, M. Atzori, B. Guc, and Y. Saygin, "Towards trajectory anonymization: a generalization-based approach," *Transactions on Data Privacy*, vol. 2, no. 1, pp. 47–75, 2009.

[4] Y.-A. De Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel, "Unique in the crowd: The privacy bounds of human mobility," *Scientific Reports*, vol. 3, p. 1376, 2013.

[5] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010.

[6] M. Douriez, H. Doraiswamy, J. Freire, and C. T. Silva, "Anonymizing nyc taxi data: Does it matter?" in *2016 IEEE International Conference on Data Science and Advanced Analytics*. IEEE, 2016, pp. 140–148.

[7] B. Gedik and L. Liu, "Location privacy in mobile systems: A personalized anonymization model," in *25th IEEE International Conference on Distributed Computing Systems*. IEEE, 2005, pp. 620–629.

[8] C. Y. Ma, D. K. Yau, N. K. Yip, and N. S. Rao, "Privacy vulnerability of published anonymous mobility traces," *IEEE/ACM Transactions on Networking (TON)*, vol. 21, no. 3, pp. 720–733, 2013.

[9] H. Zang and J. Bolot, "Anonymization of location data does not work: A large-scale measurement study," in *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*. ACM, 2011, pp. 145–156.

[10] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[11] G. Cormode, C. Procopiuc, D. Srivastava, E. Shen, and T. Yu, "Differentially private spatial decompositions," in *2012 IEEE 28th International Conference on Data Engineering (ICDE)*. IEEE, 2012, pp. 20–31.

[12] W. Qardaji, W. Yang, and N. Li, "Differentially private grids for geospatial data," in *2013 IEEE 29th International Conference on Data Engineering (ICDE)*. IEEE, 2013, pp. 757–768.

[13] J. Zhang, X. Xiao, and X. Xie, "Privtree: A differentially private algorithm for hierarchical decompositions," in *Proceedings of the 2016 International Conference on Management of Data*. ACM, pp. 155–170.

[14] R. Chen, B. Fung, B. C. Desai, and N. M. Sossou, "Differentially private transit data publication: a case study on the montreal transportation system," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2012, pp. 213–221.

[15] R. Chen, G. Acs, and C. Castelluccia, "Differentially private sequential data publication via variable-length n-grams," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. ACM, 2012, pp. 638–649.

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

[17] O. Abul, F. Bonchi, and M. Nanni, "Never walk alone: Uncertainty for anonymity in moving objects databases," in *2008 IEEE 24th International Conference on Data Engineering*. IEEE, 2008, pp. 376–385.

[18] R. Yarovoy, F. Bonchi, L. V. Lakshmanan, and W. H. Wang, "Anonymizing moving objects: How to hide a mob in a crowd?" in *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*. ACM, 2009, pp. 72–83.

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

[20] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2013, pp. 901–914.

[21] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux, "Quantifying location privacy," in *2011 IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2011, pp. 247–262.

[22] R. Shokri, G. Theodorakopoulos, C. Troncoso, J.-P. Hubaux, and J.-Y. Le Boudec, "Protecting location privacy: optimal strategy against localization attacks," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. ACM, 2012, pp. 617–627.

[23] N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Optimal geo-indistinguishable mechanisms for location privacy," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2014, pp. 251–262.

[24] R. Shokri, "Privacy games: Optimal user-centric data obfuscation," *Proceedings on Privacy Enhancing Technologies*, vol. 2015, no. 2, pp. 299–315, 2015.

[25] L. Yu, L. Liu, and C. Pu, "Dynamic differential location privacy with personalized error bounds," in *Network and Distributed System Security Symposium (NDSS)*, 2017.

[26] V. Bindschaedler and R. Shokri, "Synthesizing plausible privacy-preserving location traces," in *2016 IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2016, pp. 546–563.

[27] M. Hay, A. Machanavajjhala, G. Miklau, Y. Chen, and D. Zhang, "Principled evaluation of differentially private algorithms using dpbench," in *Proceedings of the 2016 ACM SIGMOD International Conference on Management of Data*. ACM, 2016, pp. 139–154.

[28] G. Acs and C. Castelluccia, "A case study: Privacy preserving release of spatio-temporal density in paris," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2014, pp. 1679–1688.

[29] D. J. Mir, S. Isaacman, R. Cáceres, M. Martonosi, and R. N. Wright, "Dp-where: Differentially private modeling of human mobility," in *2013 IEEE International Conference on Big Data*. IEEE, 2013, pp. 580–588.

[30] 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 *International Conference on Database and Expert Systems Applications*. Springer, 2013, pp. 357–365.

[31] V. Bindschaedler, R. Shokri, and C. A. Gunter, "Plausible deniability for privacy-preserving data synthesis," *Proceedings of the VLDB Endowment*, vol. 10, no. 5, pp. 481–492, 2017.

[32] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography Conference*. Springer, 2006, pp. 265–284.

[33] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *48th Annual IEEE Symposium on Foundations of Computer Science, (FOCS) 2007*. IEEE, 2007, pp. 94–103.

[34] F. D. McSherry, "Privacy integrated queries: an extensible platform for privacy-preserving data analysis," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, pp. 19–30.

[35] P. D. Grünwald, *The minimum description length principle*. MIT press, 2007.

[36] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: a partition-and-group framework," in *Proceedings of the 2007 ACM International Conference on Management of Data*. ACM, 2007, pp. 593–604.

[37] C. Long, R. C.-W. Wong, and H. Jagadish, "Trajectory simplification: on minimizing the direction-based error," *Proceedings of the VLDB Endowment*, vol. 8, no. 1, pp. 49–60, 2014.

[38] J. Muckell, P. W. Olsen Jr, J.-H. Hwang, C. T. Lawson, and S. Ravi, "Compression of trajectory data: a comprehensive evaluation and new approach," *GeoInformatica*, vol. 18, no. 3, pp. 435–460, 2014.

[39] H. To, K. Nguyen, and C. Shahabi, "Differentially private publication of location entropy," in *Proceedings of the 24th ACM International Conference on Advances in Geographic Information Systems*, 2016.

[40] W.-Y. Day and N. Li, "Differentially private publishing of high-dimensional data using sensitivity control," in *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security (AsiaCCS)*. ACM, 2015, pp. 451–462.

[41] L. Song, D. Kotz, R. Jain, and X. He, "Evaluating next-cell predictors with extensive wi-fi mobility data," *IEEE Transactions on Mobile Computing*, vol. 5, no. 12, pp. 1633–1649, 2006.

[42] S. Gambs, M.-O. Killijian, and M. N. del Prado Cortez, "Next place prediction using mobility markov chains," in *Proceedings of the First Workshop on Measurement, Privacy, and Mobility*. ACM, 2012, p. 3.

[43] L. Zhang, S. Luo, and H. Xia, "An investigation of intra-urban mobility pattern of taxi passengers," *arXiv preprint arXiv:1612.08378*, 2016.

[44] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th International Conference on World Wide Web*. ACM, pp. 791–800.

[45] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1393–1402, 2013.

[46] T. Brinkhoff, "A framework for generating network-based moving objects," *GeoInformatica*, vol. 6, no. 2, pp. 153–180, 2002.

[47] C. Dwork, "Differential privacy," in *International Colloquium on Automata, Languages, and Programming*. Springer, 2006, pp. 1–12.

[48] R. Bassily, A. Groce, J. Katz, and A. Smith, "Coupled-worlds privacy: Exploiting adversarial uncertainty in statistical data privacy," in *2013 IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2013, pp. 439–448.

[49] D. Kifer and A. Machanavajjhala, "Pufferfish: A framework for mathematical privacy definitions," *ACM Transactions on Database Systems (TODS)*, vol. 39, no. 1, p. 3, 2014.

Mehmet Emre Gursoy received his MS degree from University of California Los Angeles (UCLA) and his BS degree from Sabanci University, Turkey, both in Computer Science. He is currently pursuing his PhD in the School of Computer Science, Georgia Institute of Technology, where he is advised by Prof. Ling Liu. His research interests include privacy, security, machine learning, mobile computing, and big data analytics.

Ling Liu is a professor in the School of Computer Science, Georgia Institute of Technology. She directs the research programs in the Distributed Data Intensive Systems Lab (DiSL). She is an elected IEEE fellow, a recipient of the IEEE Computer Society Technical Achievement Award in 2012, and a recipient of the best paper award from a dozen of top venues, including ICDCS 2003, WWW 2004, 2005 Pat Goldberg Memorial Best Paper Award, IEEE Cloud 2012, IEEE ICWS 2013, ACM/IEEE CCGrid 2015, and IEEE Symposium on BigData 2016. She served as the general chair and PC chair of numerous IEEE and ACM conferences in data engineering, as well as on the editorial board of over a dozen international journals. Her current research is primarily sponsored by NSF, IBM, and Intel.

Stacey Truex is a PhD student in the School of Computer Science at the Georgia Institute of Technology. She is a graduate researcher in the Distributed Data Intensive Systems Lab directed by Prof. Ling Liu where her research focuses on data privacy, security, and privacy-preserving data mining. She received her M.S. in Computer Science and Systems from the University of Washington in 2016 and her B.S. in Computer Science and B.A. in Mathematics from Wake Forest University in 2012.

Lei Yu received his BS degree and MS degree in Computer Science from Harbin Institute of Technology, China. He is a PhD student in the School of Computer Science at Georgia Institute of Technology. His research interests include cloud computing, big data, privacy, sensor networks, wireless networks, and network security.