DPT: Differentially Private Trajectory Synthesis Using Hierarchical Reference Systems

Xi He
Duke University
hexi88@cs.duke.edu

Graham Cormode
University of Warwick
g.cormode@warwick.ac.uk

Ashwin Machanavajjhala
Duke University
ashwin@cs.duke.edu

Cecilia M. Procopiuc *
AT&T Labs-Research
mpro@google.com

Divesh Srivastava
AT&T Labs-Research
divesh@research.att.com

ABSTRACT

GPS-enabled devices are now ubiquitous, from airplanes and cars to smartphones and wearable technology. This has resulted in a wealth of data about the movements of individuals and populations, which can be analyzed for useful information to aid in city and traffic planning, disaster preparedness and so on. However, the places that people go can disclose extremely sensitive information about them, and thus their use needs to be filtered through privacy preserving mechanisms. This turns out to be a highly challenging task: raw trajectories are highly detailed, and typically no pair is alike. Previous attempts fail either to provide adequate privacy protection, or to remain sufficiently faithful to the original behavior.

This paper presents DPT, a system to synthesize mobility data based on raw GPS trajectories of individuals while ensuring strong privacy protection in the form of ε-differential privacy. DPT makes a number of novel modeling and algorithmic contributions including (i) discretization of raw trajectories using hierarchical reference systems (at multiple resolutions) to capture individual movements at differing speeds, (ii) adaptive mechanisms to select a small set of reference systems and construct prefix tree counts privately, and (iii) use of direction-weighted sampling for improved utility. While there have been prior attempts to solve the subproblems required to generate synthetic trajectories, to the best of our knowledge, ours is the first system that provides an end-to-end solution. We show the efficacy of our synthetic trajectory generation system using an extensive empirical evaluation.

1. INTRODUCTION

The ubiquity of sensor-enabled devices (like smartphones, wearable technology, etc.) has significantly advanced our capabilities in real-time data collection. In particular, the popularity of location sensing technology (like GPS) and the rising popularity of location-based applications has resulted in a wealth of data about the movements of individuals and populations. This in turn has sparked renewed interest in studying large-scale human mobility patterns [3] with applications in city/traffic planning, epidemiology and location-driven advertising.

Despite being valuable for research, organizations that collect detailed trajectories are wary about publicly releasing them due to concerns over the privacy of individuals therein. Home and work locations can be easy to discern (based on the frequency with which locations are visited). Additionally, these trajectories may also reveal the health status of individuals (specialities of, and frequency of visits to, clinics), their habitual vices (visits to casinos, alcohol and tobacco shops), and their truthfulness (actual vs. claimed locations). However, anonymizing such data is hard – merely removing identifiers, or coarsening the data in space and time are not enough. A recent study [10] of mobility data of 1.5 million individuals over a period of fifteen months showed that raw trajectories are highly identifiable – 4 spatio-temporal points are enough to uniquely identify 95% of the individuals; further, the study showed that the uniqueness of human mobility trajectories decays insignificantly as their spatial and temporal resolution coarsens.

What is needed is a principled approach to generate a database of synthetic trajectories, with the *provable privacy guarantees* of differential privacy [11], while ensuring *high utility* in practice – i.e., the synthetic trajectories have similar aggregate properties as the original trajectories. In this paper, we present *Differentially Private Trajectories (DPT)*, our system for generating such synthetic trajectories. As in prior work in this area (e.g., [6]), DPT considers regular trajectories, or sequences of locations that are sampled at uniform time intervals. However, in a significant departure from prior work, DPT uses a novel *hierarchical reference systems (HRS)* model, rather than the straightforward Markov model, to parsimoniously capture correlations between adjacent locations in regular trajectories. While previous work [6] was shown to adequately deal with uniform speed trajectories in small spatial domains, it can result in a considerable loss in utility on trajectories with varying speeds over time even in moderately large spatial domains, such as the movement of taxi cabs in a large metropolitan area [1]. In contrast, the HRS model is designed to naturally deal with such realistic trajectories using a set of hierarchically organized reference systems: movement at slow speeds can be summarized using a fine granularity reference system, while movements at progressively higher speeds are summarized using coarser granularity reference systems. To guarantee differential privacy, DPT applies the standard Laplace mechanism to add noise to counts maintained in each reference system.

Our Contributions. The novelty of DPT is in the model selection strategy used to build a useful differentially private generative model, and in a direction weighted sampling technique to pro-

*Author is now at Google Inc.

Proceedings of the VLDB Endowment, Vol. 8, No. 11

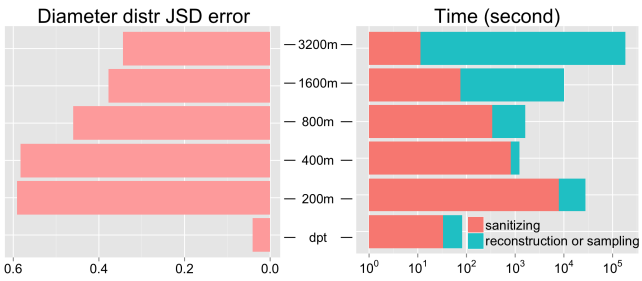

Figure 1: Comparing synthetic trajectories generation using seq (at varying granularities) vs DPT

duce high utility synthetic trajectories from the differentially private model. Consequently, we maintain that DPT is the first method that can handle realistic trajectories of millions of individuals over large spatial domains to generate useful trajectories with differential privacy guarantees.

Solution Overview. The input to our system is a database of regular trajectories, or sequences of (latitude, longitude) pairs sampled at some uniform rate, corresponding to the movements of a set of individuals. Synthetic trajectories are generated by first fitting a probabilistic model on the data from which new trajectories can be sampled. A popular choice of model for sequential data is the *order k Markov process* that generates the next location of trajectory based on the previous k locations [15]. The sufficient statistics of this model are given by a *prefix tree* of height $k+1$, where nodes at the i^{th} level of the tree record the number of times sequences of i consecutive locations corresponding to the path from the root of the tree up to the node at level i appear in the database. Recent work has proposed a differentially private approach to fit a Markov model by carefully adding noise drawn from a Laplace distribution to parts of the prefix tree [6]. This approach was shown to work well when the domain of locations is small, and can be applied to continuous spatial domains by discretizing locations (e.g., via a uniform coarse grid).

However, the aforementioned approach fails to scale to realistic trajectory databases that span even moderately large geographical regions. Note that the size of the prefix tree is $O(|\Sigma|^{k+1})$, where Σ is the domain of possible locations. A sufficiently fine discretization of the spatial domain that captures all the mobility patterns in the data could result in very large domain sizes (of several tens of thousands) making the model fitting procedure not only very slow, but also overfitting the data. As the number of nodes in the tree increases, the amount of noise added to ensure differential privacy also grows. On the other hand, while a coarse discretization of the space results in a small prefix tree, much of the spatial correlation information in the original trajectories is lost.

This is quantified in Figure 1. We considered the movement of taxi cabs in a large metropolitan area [1] spanning a geographical region of roughly 34km $\times$ 40km. We discretized this space using uniform grids with varying resolutions of 200m, 400m, 800m, 1600m and 3200m. The number of cells in the discretized domain under the 200m grid is 34000 while the size under the 3200m grid is less than 150. We synthesized trajectories under each of these discretizations using the method of Chen et al. [6] (which we call seq), and measured the Jensen Shannon divergence (JSD) between the distribution of the diameter, i.e., the distance traveled by trajectories (see Metric I in Section 5.1), in the original database to that of the trajectories in the synthetic database. In all cases, the divergence is in the range [0.33, 0.59] (a high value considering that

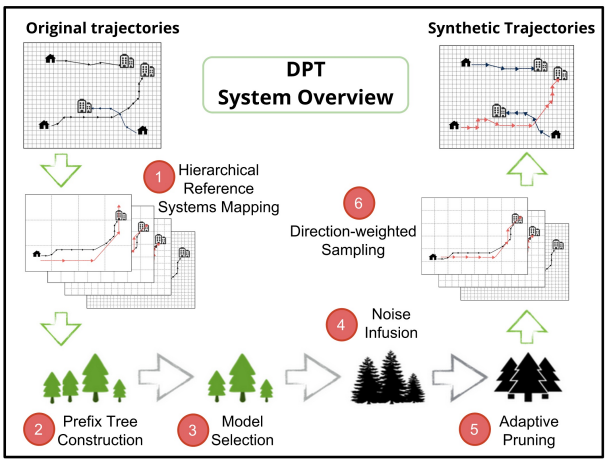

Figure 2: DPT Framework Overview

the maximum possible JSD is $\ln(2) = 0.69$). Further, for an input of over 4 million trajectories, the time taken to compute the relevant prefix trees and then reconstruct a synthetic trajectory database with least error is over a day.

Our system Differentially Private Trajectories (DPT) is a scalable end-to-end system for synthesizing regular trajectories with provable privacy properties. DPT can synthesize trajectories spanning large geographical areas with significantly more utility (JSD error on the diameter of less than 0.05) than Chen et al.'s method and is orders of magnitude faster (fitting the model and sampling trajectories take on the order of 100 seconds for an input of over 4 million trajectories). Figure 2 gives a schematic overview of the steps of DPT, which incorporates the following key novel ideas:

Hierarchical Reference Systems: DPT discretizes the spatial domain at multiple resolutions (see Step 1 in Figure 2) using a hierarchy of reference systems, maintaining one prefix tree for each resolution (Step 2). Different reference systems capture movements at different speeds. Within each reference system individuals are restricted to move from each point to only a small number of neighboring points in one step. Thus, while there are a *larger* number of prefix trees, each prefix tree has a much *smaller* branching factor resulting in a big reduction in the number of counts maintained by the model.

Model Selection: Materializing prefix trees with too large a height or at resolutions that do not correspond to viable speeds can significantly hurt utility. The reference systems (corresponding to resolutions at which the spatial domain is discretized) for which prefix trees are materialized, and the heights of trees are hard to set without looking at the database. Hence, DPT uses a novel model selection algorithm (Step 3) to set these parameters in a differentially private manner. As in prior work, noise drawn from the Laplace distribution is added to the chosen prefix trees (Step 4), and these noisy trees are pruned adaptively (Step 5) to further improve utility.

Direction Weighted Sampling: Most real world trajectories have an inherent directionality. This is captured to some extent by the Markov model, and trajectories sampled from the true model tend to retain this property. However, this directionality could be lost due to the noise added to the counts of the prefix trees in the private model. In line with prior work on enforcing constraints in differentially private query answering [13], we present a novel postprocessing strategy to restore directionality while sampling synthetic trajectories from the noisy model.

Organization: We first discuss related work in Section 2 and present our hierarchical reference systems in Section 3. Section 4 describes our differentially private techniques for learning the model from the data, including model selection and direction weighted sampling. We present a thorough experimental evaluation in Section 5, using real and model-generated data sets.

2. RELATED WORK

Studies on human mobility patterns [10, 23] have shown that individual trajectories are both highly *unique* and *predictable*. The study by Montjoye et al. [10] shows that the uniqueness of human mobility traces decays only slightly as their spatial and temporal resolution coarsens. This high identifiability indicates the difficulty of achieving properties such as k-anonymity, where at least k individuals have the same trajectory, using generalization, bucketization, and suppression techniques. Nevertheless, there have been numerous attempts in this direction [2, 8, 14, 18, 25, 28]. At the same time, individual patterns exhibit high predictability, as in the study of Song et al. [23] who showed that each individual has more than 90 percent potential predictability, largely independent of the user's typical radius. Hence, the release of an individual's mobility data can make them susceptible to privacy attacks, such as user identification, sensitive location and sequential tracking attacks discussed by Kopanaki et al. [16].

Because of these two properties, we aim to provide a differentially private algorithm, first to give a more robust privacy guarantee than k-anonymity and second to synthesize traces resembling the repetitiveness of an individual's trajectories. Several prior works have offered approaches to this question [6, 7, 20, 22], but these do not adapt well to general trajectory data. Shao et al. [22] develop sampling and interpolation strategies to a variant (weaker) form of differential privacy. A line of work achieves ε-differential privacy by publishing synthetic traces with hierarchical structure [6, 7] and frequency vectors [20]. The hierarchical framework captures the dependency of sequential data with a Markov assumption and achieves scalability with a set of pruning strategies. Chen et al. [7] build a tree by grouping trajectories with the same prefix, but due to the high uniqueness of patterns, the counts for leaf nodes are very sparse. In subsequent work [6], substrings are taken into account for building an exploration tree based on a Markov assumption, so that leaf counts are higher and hence offer better utility.

This line of work gives good insight into synthesizing the mobility patterns of individuals, but fails to address challenges faced in building Markov models for real unconstrained trajectory data. First, the large spatial and temporal domain size in trajectory data usually results in a large number of transition possibilities even for a 1st-order Markov model [20]. Pruning strategies proposed in [6, 7] help to scale, but their impact on accuracy is unknown. The domain studied in [6, 7] is short trajectories logged at subway stops, which are both shorter and over a smaller domain than the cases we wish to study. These works truncate trajectories to bound the sensitivity of queries, but there is no discussion on the optimal methods to extract the most useful information of the trajectories. Last, they propose post-processing algorithms on the hierarchical counts, but in an ad hoc way, with no connection to a desired utility metric.

Pratesi et al. [20] define adjacency for reference points, and remap observations to step between adjacent points in order to reduce the number of possible transitions. However, only one reference system of homogeneous reference points is used, meaning that many steps take place between the same reference point, and so are discarded. In our work, we use a hierarchical reference system to capture different types of movement, and so minimize the loss of information during the mapping. Related work on adaptive spatial partitioning [7, 9, 21] has explored the adaptive decomposition of space under differential privacy, but this work looked primarily at point distributions rather than trajectories. Chen et al. [7] build a prefix tree with hybrid-granularity, where each level of the tree has two sub-levels of generalized locations and non-generalized locations. This hybrid granularity helps prune out nodes associated with zero-frequency patterns, but it is not clear how to generalize nodes at each level. Qardaji et al. [21] present an adaptive grid by laying a coarse-grained grid over the dataset, where each cell is then refined according to its noisy counts to answer range queries over the space. Cormode et al. [9] compute private medians for building data-dependent trees. These methods are effective for spatial decomposition when assuming no sequential dependency among locations. We extend the idea of adaptive decomposition of space to hierarchical reference systems in order to capture trajectories with different speeds and sampling rates.

Recent work [4, 6] on providing differential privacy for sequential data under Markov models represent the trajectories as strings (of locations). Similar to Chen et al. [6], Bonomi et al. [4] constrain the amount of information contributed to the query output in order to bound the sensitivity of queries, but the difference is that Bonomi et al. has a stable transformation over each object in the sequential data by selecting top patterns of each object instead of taking the first few occurrence of sequential data. Taking only top patterns for each object may miss the global information across all individuals while the first few observations of sequential data may capture some global information, but miss information at the end of long trajectories.

D	Trajectory database
PT	Personal Trajectory table of an individual
t	A single trajectory
$\mathbf{\Sigma}$	A set of latitude-longitude coordinates
σ	Latitude-longitude coordinate of a location
Σ	Reference system, a set of anchor points
a	Anchor point in reference system Σ
$\top$	Starting symbol
$\perp$	Ending symbol
Σ^k	k-gram pattern
$c(t,x)$	The number of occurrences of $x \in \Sigma^k$ in t
$c(PT,x)$	The number of occurrences of $x \in \Sigma^k$ in PT
$c(D,x)$	The number of occurrences of $x \in \Sigma^k$ in D

Table 1: Notation Summary

3. HIERARCHICAL RS MODEL

We next describe a Markov model for mobility trajectories and build up to our novel hierarchical reference systems (HRS) model.

3.1 Notation and Background

Given a spatial domain $\mathbf{\Sigma}$, a regular trajectory t is a sequence of locations (σ_i) observed and recorded at regular time intervals, where $\sigma_i \in \mathbf{\Sigma}$ for $i = 1, 2, \cdots$. The spatial domain is usually a set of latitude-longitude coordinates. We consider a trajectory database D consisting of $|D|$ individuals. Each individual has a personal trajectory table, denoted by PT with each tuple representing a regular trajectory, t. It is possible for an individual to have multiple trajectories, of varying lengths, where length is defined as the number of points in the trajectory. We call any sequence $x \in \mathbf{\Sigma}^k$ of k locations as a k-gram. If x is a k-gram and y is an ℓ-gram, then we denote by xy the $(k+\ell)$-gram obtained by appending the sequence y to sequence x. We denote by $c(t,x)$ the number of occurrences of x in t,

$c(PT,x)$ the total number of occurrences of x in the trajectory table PT, i.e. $c(PT,x) = \sum_{t \in PT} c(t,x)$, and $c(D,x)$ the total number of occurrences of x in the database D, i.e. $c(D,x) = \sum_{PT \in D} c(PT,x)$. We summarize the notation of the paper in Table 1. Markov processes [19] to model correlations between contiguous locations in a regular mobility trajectory are defined as follows.

DEFINITION 3.1 (MARKOV PROCESS). *A regular trajectory $(\sigma_1 \sigma_2 \cdots \sigma_n) \in \Sigma^n$ is said to follow an order ℓ Markov process if for every $\ell \leq i < n$, $\sigma \in \Sigma$*

$$\Pr[\sigma_{i+1} = \sigma | \sigma_1 \cdots \sigma_i] = \Pr[\sigma_{i+1} = \sigma | \sigma_{i-\ell+1} \cdots \sigma_i]. \quad (1)$$

We refer to the probability $\Pr[\sigma_{i+1} = \sigma | \sigma_{i-\ell+1} \cdots \sigma_i]$ as a *transition probability* of the Markov process. The collection of transition probabilities for all $x = \sigma_{i-\ell+1} \cdots \sigma_i \in \Sigma^\ell$ can be estimated using the set of all ℓ- and $\ell+1$-gram counts, i.e.

$$\Pr[\sigma_{i+1} = \sigma | \sigma_{i-\ell+1} \cdots \sigma_i] = \frac{c(D, x\sigma)}{c(D,x)}. \quad (2)$$

Typically, a starting symbol ($\top$) and a stopping symbol ($\bot$) are prepended and appended (respectively) to the original trajectories so that the starting and stopping probabilities are captured in the model (and are treated as special locations in Σ). A synthetic trajectory is initialized with a starting symbol ($\top$) and is continuously sampled by picking the next location from the Markov process till reaching the stopping symbol ($\bot$). Formally, let $x = (\top \sigma_1 \cdots \sigma_i)$ denote the prefix of a trajectory sampled so far and x' the longest suffix of x of length $\ell \leq (k-1)$, where $(k-1)$ is the maximum order of Markov process considered. The next location is chosen using the transition probability given x' in the order ℓ model. Thus, sampling synthetic trajectories requires us to maintain all ℓ-gram counts for $1 \leq \ell \leq k$. These counts can be stored in a *prefix tree* $\mathcal{T}$ of height k — nodes in $\mathcal{T}$ are $\Sigma^1 \cup \cdots \cup \Sigma^k$, and edges connect each ℓ-gram x to an $(\ell+1)$-gram $x\sigma$, for all $\sigma \in \Sigma$.

3.2 Discretization and Reference Systems

A trajectory is usually recorded as a sequence of points in some continuous domain Σ, e.g. latitude-longitude coordinates. One common way to analyze trajectories on a continuous domain (and to limit the size of the model) is via discretization of the space using a *reference system* [24].

DEFINITION 3.2 (REFERENCE SYSTEM). *A reference system (RS) is a set of anchor points $\Sigma \subset \Sigma$, associated with a mapping function $f : \Sigma \to \Sigma$.*

In this work, we use a reference system constructed by imposing a uniform grid over the space and choosing the centroids as anchor points. We denote such a reference system by Σ_v, where v denotes the resolution or the length of the side of each grid cell. Varying the resolution v results in different reference systems; coarse grids (large v) correspond to fewer anchor points, and fine grids (small v) result in a larger number of anchor points. The function considered here for the reference system Σ_v is represented by f_v and maps a point in Σ to its closest anchor point in Σ_v (if a point is equidistant to more than one anchor point, ties are broken in a consistent way).

3.3 Hierarchical Reference Systems

Even with the use of a reference system Σ, the number of parameters (transition probabilities) to specify a Markov model can be very large. The model requires $\approx |\Sigma|^{k+1}$ counts (for large $|\Sigma|$) in a prefix tree $\mathcal{T}$ of height k (i.e., ℓ-gram counts for $1 \leq \ell \leq k$). For instance, if Σ is a coarse 16×16 grid partitioning the space into 256 cells, then for $k = 3$, $|\Sigma|^{k+1} \approx 4 \cdot 10^9$. Yet, grids of much finer resolution are needed to model all the transitions in the data.

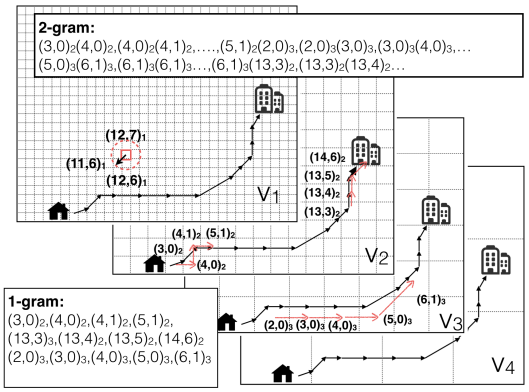

Figure 3: Map trajectory with different granularity

Therefore, rather than building a Markov model using a single reference system ([6, 7, 20, 24]), we adopt a novel approach in DPT. First, given a reference system Σ, we choose to estimate transition probabilities of ℓ-grams, $x \in \Sigma^\ell$ to *only* the anchor points that are "close" to the last observed location in x. In the context of a uniform grid based reference system Σ_v, we only consider transitions from a grid cell a to either itself or to its 8 adjacent cells. We call them the *neighboring* cells. All other transition probabilities are forced to be 0. Under this constraint, we only need to maintain $|\Sigma_v|$ 1-gram counts, $9 \cdot |\Sigma_v|$ 2-gram counts, $9^2 \cdot |\Sigma_v|$ 3-gram counts and so on, thus reducing the total number of counts in the prefix tree $\mathcal{T}$ from $|\Sigma_v|^{k+1}$ to $O(9^k \cdot |\Sigma_v|)$.

However, restricting transitions to only neighboring anchor points cannot capture all the transitions in the original trajectories under any single reference system. This is because, even within a single trajectory, objects tend to move at different speed ranges. For instance, a taxi may travel at greater speeds on a highway, but with much slower speeds in downtown, and both types of driving can occur in a single trip. Hence, we use a set of reference systems to capture motion at different speed ranges, and transitions in each of these reference systems occur between anchor points that are close.

DEFINITION 3.3 (HIERARCHICAL REFERENCE SYSTEMS). *Let Σ_v denote the set of centroids of cells in a uniform grid of resolution v. The* hierarchical reference systems *$HRS = \{\Sigma_{v_1}, \ldots, \Sigma_{v_M}\}$ is a set of reference systems that have a hierarchical relationship, where $v_1 < \cdots < v_M$. For any anchor point $a \in \Sigma_{v_m}$, we define (1) its parent in $\Sigma_{v_{m'}}$, where $m' > m$ as the closest point to a in $\Sigma_{v_{m'}}$,*

$$\text{par}(a, \Sigma_{v_{m'}}) = \text{argmin}_{a' \in \Sigma_{v_{m'}}} d(a', a)$$

and (2) its children in $\Sigma_{v_{m'}}$, where $m' < m$ as

$$\text{children}(a, \Sigma_{v_{m'}}) = \left\{ a' \in \Sigma_{v_{m'}} \,|\, \text{par}(a', \Sigma_{v_m}) = a \right\}.$$

If multiple anchor points can be the parent of a single anchor point, ties are broken in a consistent way. In this paper we consider a set of reference systems with geometrically increasing resolutions, i.e. $\{\Sigma_{v_1}, \ldots, \Sigma_{v_M}\}$, where $v_m / v_{m-1} = 2$, for all $1 < m \leq M$. Such an HRS can capture motions at different speed ranges in the raw trajectory t — long steps (pairs of consecutive points which are far away) are mapped with coarser reference systems while shorter steps are mapped with finer reference systems. We assume the longest step in the whole database is less than v_M.

Given an *HRS*, we want to replace each point on the raw trajectory t by its closest anchor point in one of the reference systems. The goal is to choose the appropriate reference system for each raw point, such that the resulting trajectory has the following property: consecutive points on it are either neighbour anchors in

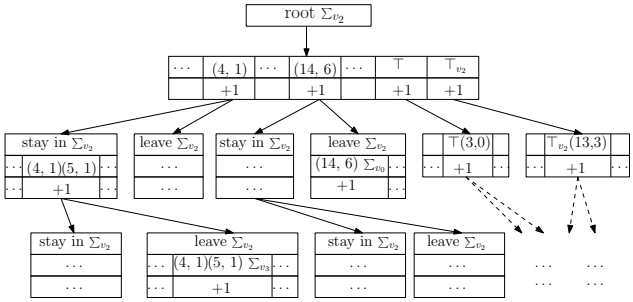

Figure 4: Prefix tree $\mathcal{T}_2$ for Σ_{v_2}

the same system, or have a parent-child relationship. (These properties ensure that we can model the trajectories by an HRS with bounded fanout.) Algorithm 1 shows how to achieve this, but we start by illustrating it via Example 3.1. The resulting trajectory t_{rs} then becomes a sequence of segments, where a segment is defined as contiguous anchor points from the same reference system. We introduce a new start symbol $\top_{v_m}$ as a special location in each reference system Σ_{v_m} to indicate the start of a segment falling into that reference system. Σ_{v_0} can represent the stopping symbol $\perp$.

EXAMPLE 3.1. *Figure 3 considers four grid-based reference systems $\Sigma_{v_1}, \Sigma_{v_2}, \Sigma_{v_3}, \Sigma_{v_4}$, where $v_1 = v_2/2 = v_3/4 = v_4/8$. Each point is represented in the form of $(x,y)_m$, where m indicates that the reference system is Σ_{v_m}, and x and y are the horizontal and vertical indexes in Σ_{v_m}. The trajectories shown here refer to the same trajectory. The first four points in t are mapped to anchor points in Σ_{v_2} as $(3,0)_2(4,0)_2(4,1)_2(5,1)_2$ (neighboring cells in Σ_{v_2}), as each step crosses more than one cell in Σ_{v_1}. Similarly, the next step after $(5,1)_2$ crosses more than one cell in Σ_{v_2}, and hence a coarser reference system Σ_{v_3} is used for the mapping, resulting in the anchor point $(2,0)_3$. Though the next step after $(2,0)_3$ could be mapped to a finer reference system Σ_{v_2} as $(7,1)_2(8,1)_2$, this is not preferable, since the trajectory is not slowing down. Greedily changing the reference system to a finer resolution is not favored. Hence, we adopt the rule that only when at least 3 consecutive points are mapped to the same cell of the current reference system do we change to a finer reference system. This gives more contiguous points in Σ_{v_3}: $(2,0)_3(3,0)_3(4,0)_3(5,0)_3(6,1)_3(6,1)_3(6,1)_3$. As the last three points remain in the same cell of Σ_{v_3}, this implies slowing down, and so the following points are mapped to a finer reference system Σ_{v_2} as $(13,3)_2(13,4)_2(13,5)_2(14,6)_2$. This results in three segments: (1)$\top(3,0)_2(4,0)_2(4,1)_2(5,1)_2(2,0)_3$; (2)$\top_{v_3}(2,0)_3(3,0)_3(4,0)_3(5,0)_3(6,1)_3(6,1)_3(6,1)_3(13,3)_2$; (3)$\top_{v_2}(13,3)_2(13,4)_2(13,5)_2(14,6)_2\perp$.*

Algorithm 1 summarizes our proposed transformation process. This process starts with adding start symbol $\top$ to indicate the beginning of a trajectory in Line 1. m^* represents the optimal reference system used for the current mapping when iterating through the sequence of points in Line 3-10. Line 4 specifies that every step (pair of consecutive points) in the raw trajectory is mapped to neighboring cells (a_1, a_2) within the same reference system at the most detailed resolution. This ensures that the finest resolution at the same or coarser level is used (lines 5-6). However, this greedy mapping may lead to many short segments. Hence, in Lines 7-9, before updating the optimal reference system $\Sigma_{v_{m^*}}$ to a finer reference system $\Sigma_{v_{m'}}$, we check if both current and previous steps are sufficiently small (the three consecutive anchor points in the previous optimal reference system are the same). The intuition for this

Input: $t = \sigma_1\sigma_2...\sigma_{|t|}$: regular trajectory,
$HRS = \{\Sigma_{v_1}, \ldots, \Sigma_{v_M}\}$: M hierarchical reference systems.
Output: t_{rs}: sequence of anchor points in HRS
1: Initialize $t_{rs} = (\top)$ *// start symbol*
2: Initialize optimal reference system index $m^* = 1$
3: **for** $i = 2 : |t|$ **do**
4: $\quad m' \leftarrow$ finest reference system Σ_{v_m} s.t. $d(f_{v_m}(\sigma_{i-1}), f_{v_m}(\sigma_i)) \leq v_m$
5: $\quad$ **if** $v_{m'} \geq v_{m^*}$ **then**
6: $\quad\quad m^* \leftarrow m'$ *// coarser or the same reference system*
7: $\quad$ **else if** $f_{v_{m^*}}(\sigma_i) = f_{v_{m^*}}(\sigma_{i-1}) = f_{v_{m^*}}(\sigma_{i-2})$ **then**
8: $\quad\quad$ *// stay in the same cell (assume $\sigma_0 = \sigma_1$)*
9: $\quad\quad m^* \leftarrow m'$ *// finer reference system*
10: $\quad$ Add $f_{v_{m^*}}(\sigma_{i-1})$ to t_{rs}
11: Add $f_{v_{m^*}}(\sigma_{|t|})$ and $\perp$ to t_{rs} *// last point and stopping symbol*
12: **return** t_{rs}

Algorithm 1: HRS-based Transformation

is that if the travelling speed is halved to the previous speed range v_{m^*}, at least 2 consecutive steps may fall into the same cell in the current reference system because of the geometrically increasing setting of HRS, i.e. $v_m/v_{m-1} = 2$.

3.4 Hierarchical RS Model

We now present a probabilistic generative model for trajectories based on hierarchical reference systems $HRS = \{\Sigma_{v_1}, \ldots, \Sigma_{v_M}\}$. Let $x = (\top a_1 \cdots a_i)$ denote the prefix of a trajectory that has been mapped to HRS using Algorithm 1. Thus x may contain anchor points from $\Sigma_{v_1} \cup \ldots \cup \Sigma_{v_M}$. Given x, there are two ways to extend x to the next anchor point a_{i+1} where (1) a_{i+1} is a neighboring cell of a_i in the same reference system; or (2) a_{i+1} is the parent or a child of a_i in a different reference system. The count for each extension is maintained in a set of prefix trees $\{\mathcal{T}_1, \ldots, \mathcal{T}_M\}$, where each $\mathcal{T}_m$ has a height k, similar to the basic Markov process described in Section 3.1, We name this set of prefix trees as a forest $\mathcal{F}$ or *forest counting query* as defined in Definition 3.4. Nodes in the prefix tree $\mathcal{T}_m$ correspond to ℓ-grams in Σ_{v_m} for $1 \leq \ell \leq k$, as well as ℓ-grams appended with a move to a different reference system $m' \neq m$. Thus, every node in $\mathcal{T}_m$ has 9 children representing moves to neighboring cells in the same reference system (same speed range), and M children representing moves to different reference systems including $\perp$ as Σ_{v_0} (change speed range). In Example 3.2, we give more details on $\mathcal{F}$ built from the trajectory shown in Example 3.1. Then we will continue illustrating how to sample trajectories from $\mathcal{F}$ in the next section (Section 3.5).

DEFINITION 3.4 (FOREST COUNTING QUERY $\mathcal{F}(\cdot)$). *The term 'Forest counting query' refers to the set of counting queries on the database D to build model $\mathcal{F}$, denoted by*

$$\mathcal{F}(D) = \{c(D,x) | x \in \mathcal{T}, \mathcal{T} \in \mathcal{F}\}. \quad (3)$$

EXAMPLE 3.2. *Given the hierarchical reference systems in Example 3.1, $HRS = \{\Sigma_{v_1}, \Sigma_{v_2}, \Sigma_{v_3}, \Sigma_{v_4}\}$, we build a forest consisting of 4 prefix trees: $\mathcal{F} = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4\}$. In Figure 4, we materialize some of the nodes in $\mathcal{T}_2$ whose counts are contributed to by the segments resulting from the transformation of trajectory t in Figure 3. The highest level of $\mathcal{T}_2$ consists of all the anchor points in Σ_{v_2}, the starting symbol $\top$, and the start of a segment $\top_{v_2}$. The 1-grams listed in Figure 3 add counts to their corresponding nodes in $\mathcal{T}_2$. For instance, $(4,1)_2$ increments the count of node $(4,1)$. All the nodes except $\top$ and $\top_{v_2}$ have two types of children: those that remain with the same step size v_2, and those with a different*

step size from their parent. For instance, the 2-gram $(4,1)_2(5,1)_2$ *corresponds to the node* $(4,1)(5,1)$ *which stays in the same reference system* Σ_{v_2}*; the 2-gram* $(14,6)_2\perp$ *corresponds to the node* $(14,6)\Sigma_{v_0}$ *which changes from* Σ_{v_2} *to* Σ_{v_0} *(a stop) and the 3-gram* $(4,1)_2(5,1)_2(2,0)_3$ *corresponds to the node* $(4,1)(5,1)v_3$ *which changes from* Σ_{v_2} *to* Σ_{v_3}*.* $\top$ *indicates the start of a trajectory and* $\top_{v_2}$ *indicates the start of a segment (changing from another reference system). They could be followed by any anchor points in* Σ_{v_2}*. The 2-gram* $\top(3,0)_2$ *and* $\top_{v_2}(13,3)_2$ *increment the counts of nodes* $\top(3,0)$ *and* $\top_{v_2}(13,3)$ *in* $\mathcal{T}_2$ *respectively.*

The desiderata for anchor points constituting DPT's hierarchical model are that: (a) anchor points have a small ($O(1)$) number of neighbors in each reference system, (b) a parent anchor point in one reference system has a small number of children anchor points in the next finer reference system, and (c) the average distance between neighboring anchor points increases by constant factors in each reference systems. While we make use of grid-based systems, these requirements could also be achieved by systems based on e.g. clustering and pruning points of interest. The choice of reference systems does not affect the privacy guarantee, but may impact utility; to remain focused, we confine this study to using grids only.

3.5 Sampling Trajectories

Given a forest $\mathcal{F} = \{\mathcal{T}_1, , ..., \mathcal{T}_M\}$ of height k built upon $HRS = \{\Sigma_{v_1}, ..., \Sigma_{v_M}\}$, trajectories are sampled independently of each other. We sample the starting reference system Σ_{v_m} using probabilities proportional to the count of start symbol $\top$ in each $\mathcal{T}_m$. Let $x = (\top a_1 \cdots a_i)$ denote the prefix of the trajectory generated thus far, and x' denote the longest suffix of x (of length $\ell < k$) from the same reference system. Given x', the next location is chosen in two steps: (1) choose a reference system Σ_{v_m} from HRS or stopping symbol $\perp$ (represented by Σ_{v_0}), and then (2) transition to a neighboring cell of a_i in the chosen Σ_{v_m}. Both probabilities depend on the immediate history (of $\ell \leq k$ events) in the *same reference system as* a_i. Thus, the first probability can be estimated from $\mathcal{F}$ as:

$$\Pr[\Sigma_{v_m}|x] = \frac{c(D,(x'\Sigma_{v_m}))}{c(D,x')}, \tag{4}$$

$c(D,(x'\Sigma_{v_m}))$ is the number of occurrences of x' followed by a move in reference system Σ_{v_m}, for $m \leq M$, in the database D. Given x', we first sample the next reference system Σ_{v_m} based on the transition probability (4). For a neighboring cell a in the new reference system Σ_{v_m}, the second probability can be estimated as:

$$\Pr[a|x,\Sigma_{v_m}] = \frac{c(D,(x'a))}{c(D,x')}. \tag{5}$$

If a move to the same reference system is sampled, sampling continues using the first 9 children of x'; if a move to a coarser reference system is sampled, sampling continues using the ancestor of the last location in x'; if a move to a finer reference system is sampled, sampling continues using one of the children of the last location in x'. For the last case, rather than uniformly sampling a child of the last location a_i, we sample a_{i+1} from the children of a_i in the new reference system Σ_{v_m} using probabilities proportional to the count of $\top_{v_m}a_{i+1}$ in $\mathcal{T}_m$. The trajectory generation is stopped when a maximum length $L_{\max}$ is reached, or if a stopping symbol $\perp$ is sampled as the reference system Σ_{v_0}.

4. PRIVATE MODEL LEARNING

Two aspects of the HRS model need to be learned: (i) the *structure* of the model, i.e., the set of reference systems to be included in the model, and the height of the prefix trees for the chosen reference systems, and (ii) the *transition probabilities*, within and across the chosen reference systems. Transition probabilities are learned from the data by estimating counts in corresponding prefix trees. Differentially private methods for prefix tree construction have been studied in prior work [6, 7], and our prefix tree construction algorithm (Section 4.3) follows the approach in prior work with a few notable additions. We normalize the counts in the original trajectory table of each individual to bound the sensitivity (see Defn. 4.2) of the forest counting query. Adding noise to subtrees of the prefix tree with low counts can lead to low signal to noise ratios. We propose an adaptive pruning strategy to eliminate such subtrees, which outperforms prior prefix tree pruning strategies.

The main privacy contributions of our work, however, are (a) a novel differentially private *model selection algorithm* for learning the structure (Section 4.2), and (b) a *direction weighted sampling* strategy that recovers the global directional properties from the data that might be lost due to the noisy transition probability learning phase (Section 4.4). Structure learning is crucial, since the set of appropriate reference systems is not known apriori. A range of candidate reference systems that capture most of the transitions can be learned purely using domain knowledge (of typical speeds) and without access to the data. However, as we show in our experiments, even within this range, only a few of the reference systems may have sufficient data support. Hence, structure learning based on the data can significantly improve the signal to noise ratio, and thus the utility of the learned model.

In this section, we summarize the model of differential privacy and describe our novel model selection algorithm, followed by a brief overview of our methods for estimating transition probabilities privately. We then outline direction weighted sampling. We conclude with an end-to-end privacy evaluation of the system.

4.1 Differential Privacy

We define ε-differential privacy [12] in the context of trajectory databases. Let D_1, D_2 be two neighboring trajectory databases, i.e. D_1 and D_2 differ in only one individual's trajectory table PT, written as $\|D_1 - D_2\| = 1$. This paper consistently takes this to mean that PT is present in only one of the two databases.

DEFINITION 4.1 (DIFFERENTIAL PRIVACY). *Let* D_1*,* D_2 *be two neighboring databases. Let M denote a randomized algorithm over databases, and O be a possible output of M. Mechanism M is said to be* ε*-differentially private, for* $\varepsilon > 0$*, if*

$$\Pr[M(D_1) = O] \leq \exp(\varepsilon)\Pr[M(D_2) = O]. \tag{6}$$

In our setting, differential privacy guarantees that no individual's trajectory table can significantly affect the released information — the output distribution generated by M is nearly the same, whether or not the individual's trajectory table is present in the database. We make use of the concept of global sensitivity [12].

DEFINITION 4.2 (GLOBAL SENSITIVITY). *The global sensitivity of a function* $q : \mathcal{D} \to \mathbb{R}^d$*, denoted by* $S(q)$ *is defined as the largest* L_1 *difference* $\|q(D_1) - q(D_2)\|_1$*, where* D_1 *and* D_2 *are neighboring databases that differ in one personal trajectory table. More formally,*

$$S(q) = \max_{D_1,D_2:\|D_1-D_2\|=1} \|q(D_1) - q(D_2)\|_1. \tag{7}$$

One common technique used to achieve differential privacy is the Laplace mechanism proposed in [12]:

DEFINITION 4.3 (LAPLACE MECHANISM). *The Laplace mechanism privately computes a function:* $q : \mathcal{D} \to \mathbb{R}^d$ *by computing*

$q(D)+\eta$, *where* $\eta \in \mathbb{R}^d$ *is a vector of independent random variables, where* η_i *is drawn from the Laplace distribution with parameter* $S(q)/\varepsilon$. *That is,* $\Pr[\eta_i = z] \propto e^{-z\cdot\varepsilon/S(q)}$.

4.2 Private Model Selection

To ensure differential privacy, Laplace noise is added to the counts of the prefix trees in $\mathcal{F}$. The noisy count for each node x in the prefix tree is denoted by $\tilde{c}(D,x)$. Given a privacy budget ε to learn the noisy counts in $\mathcal{T}_m \in \mathcal{F}$ of height k (see Section 4.3), the total amount of noise can be measured as:

$$N_m(\varepsilon,k) = \mathsf{E}\left[\sum_{x\in\mathcal{T}_m}(c(D,x)-\tilde{c}(D,x))^2\right] = \sum_{x\in\mathcal{T}_m}\mathsf{Var}(\varepsilon,k,x), \quad (8)$$

where $\mathsf{Var}(\varepsilon,k,x) = \frac{2}{\varepsilon(x)^2}$ and $\varepsilon(x)$ is the privacy budget used to compute the noisy count for x and is dependent on ε and k. The total noise added to $\mathcal{F}$ corresponds to the *variance* of the model $\mathcal{F}$, which grows linearly in the number of trees and quadratically in the tree height k. Thus, the variance introduced by the Laplace mechanism can drown out the signal for large forests.

On the other hand, we would like our hierarchical reference system set to be *full*; i.e., every move in the trajectory dataset D can be captured as a move between neighboring cells in some reference system. Based on domain knowledge one can construct a full reference system set $\{\Sigma_{v_1},...,\Sigma_{v_M}\}$ in a data independent fashion (for instance, by picking v_m's as powers of 2 that capture the smallest and largest step sizes). We denote the forest $\mathcal{F}$ built on the full model by $\mathcal{F}_{\text{full}}$. Moreover, the trees in the forest are preferred to have height k large enough to cover high order correlations.

Our approach to solve this problem is to drop some of the trees (resulting in $\mathcal{F}^+ \subset \mathcal{F}_{\text{full}}$) and/or choose a proper height k such that the least *bias* is introduced into the model (since some of the transitions may no longer be accurately captured). We approximate the bias which arises from dropping a tree by the squared counts of moves from the corresponding reference system, since we are effectively representing these counts by zeros. Let $c_m(D)$ be the number of transitions in D that occur in reference system Σ_{v_m} (the sum of 1-gram counts in $\mathcal{T}_m$). Since $c_m(D)^2$ is an upper bound for the squared counts of i-grams for $i > 1$, it can be used as an estimate of the bias at level i in $\mathcal{T}_m$. Thus, $\mathcal{T}_m$ of height k has a total bias of

$$k \cdot c_m(D)^2. \quad (9)$$

We use a small part of the privacy budget ε_s to compute the noisy $c_m(D)$ to ensure the privacy guarantee of the model selection process. Hence, the variance for the tree $\mathcal{T}_m \in \mathcal{F}^+$ of height k is computed as $N_m(\varepsilon_r,k)$, where $\varepsilon_r = \varepsilon - \varepsilon_s$.

Summing up the variance and bias terms specified in Eqns. (8) and (9) gives the total error of $\mathcal{F}^+$ of height k:

$$\text{Error}(\mathcal{F}^+,\varepsilon_r,k,D) = \sum_{\mathcal{T}_m\in\mathcal{F}^+} N_m(\varepsilon_r,k) + \sum_{\mathcal{T}_m\in(\mathcal{F}_{\text{full}}-\mathcal{F}^+)} k\cdot c_m(D)^2. \quad (10)$$

Our goal is to find the optimal model $\mathcal{F}^{+*}$ of height k which minimize the overall error specified in Eqn. (10) w.r.t. $\mathcal{F}^+$ and k,

$$\mathcal{F}^{+*}(\varepsilon_r,D) = \underset{\mathcal{F}^+\subseteq\mathcal{F}_{\text{full}},k\geq 2}{\text{argmin}} \text{Error}(\mathcal{F}^+,\varepsilon_r,k,D). \quad (11)$$

Eqn. (11) satisfies an interesting, and intuitive monotonicity property of the optimal reference systems as a function of privacy budget ε_r, and data size $|D|$: (1) given fixed data size $|D|$, a larger ε_r allows either more reference systems to be chosen, *or* trees to have larger heights; (2) given fixed privacy budget ε_r, a larger dataset allows either more reference systems to be chosen, *or* trees to have larger heights. We omit formal statement and proof of these claims for brevity. This monotonicity property means that given fixed dataset D and privacy budget ε_r,

$$\underset{\mathcal{F}^+\subseteq\mathcal{F}_{\text{full}}}{\text{argmin}} \text{Error}(\mathcal{F}^+,\varepsilon_r,k+1,D) \subseteq \underset{\mathcal{F}^+\subseteq\mathcal{F}_{\text{full}}}{\text{argmin}} \text{Error}(\mathcal{F}^+,\varepsilon_r,k,D). \quad (12)$$

Hence, we propose Algorithm 2 to search for the optimal $\mathcal{F}^{+*}$ and also k by decrementing the value of k from its maximum possible value. In this way, the search space of the optimal model at k is not the subsets of $\mathcal{F}_{\text{full}}$ any more, but the subsets of the optimal model at $k+1$ (as shown in Line 8). In Algorithm 2, Lines 1-2 use a privacy budget of ε_s to compute the noisy bias for each reference system if the corresponding system is dropped. Line 3 chooses the coarsest possible reference system $\Sigma_{v_{M^*}}$, which has a noisy root count square greater than the variance of the noise $\frac{2}{\varepsilon_s^2}$. Lines 4-5 initialize the full model's maximum allowed tree height k and remaining budget ε_r. Then the variance in $\mathcal{F}^{+*}$ is estimated based on the remaining budget ε_r . Lines 6-14 identify the best model with the lowest $\text{Error}(\mathcal{F}^+,\varepsilon_r,k,D)$ given the height k. The coarsest reference system is included in final output (in Line 10 and 14) in order to record any large steps which the optimal model fails to capture. This algorithm may not guarantee the optimal solution for Eqn. (11), but reduces the searching space based on Eqn. (12) and is shown to be effective in our evaluation (Section 5.4).

Input: D: database, $\mathcal{F}_{\text{full}}$: full model of size M, ε: total privacy budget, ε_s: privacy budget for model selection, $k_{\max}$: the maximum tree height
Output: $(\mathcal{F}^{+*},k)$: a subset of $\mathcal{F}_{\text{full}}$ of order k

1: Count moves in each reference system: $(c_1(D),...,c_M(D))$
2: $(\tilde{c}_1(D),...,\tilde{c}_M(D)) \leftarrow (c_1(D),...,c_M(D)) + \text{Lap}(1/\varepsilon_s)^M$
3: $M^* \leftarrow$ largest index m in $\{\mathcal{T}_1,\ldots,\mathcal{T}_M\}$ s.t. $\tilde{c}_m(D) \geq \frac{2}{\varepsilon_s^2}$
4: Initialize $k = k_{\max}$, $\mathcal{F}^{+*} = \{\mathcal{T}_1,\cdots,\mathcal{T}_{M^*}\}$
5: $\varepsilon_r = \varepsilon - \varepsilon_s$
6: **while** $k \geq 2$ **do**
7: Compute $N_m(\varepsilon_r,k)$, $\tilde{C}_m(D,k)$ $\forall\mathcal{T}_m \in \mathcal{F}^{+*}$
8: $\mathcal{F}^{+*}(\varepsilon_r,D) = \text{argmin}_{\mathcal{F}^+\subseteq\mathcal{F}^{+*}} \text{Error}(\mathcal{F}^+,\varepsilon_r,k,D)$
9: **if** $\mathcal{F}^{+*}(\varepsilon_r,D) = \emptyset$ **then**
10: **return** $(\mathcal{F}^{+*}\cup\{\mathcal{T}_{M^*}\},k)$
11: **else**
12: $\mathcal{F}^{+*} = \mathcal{F}^{+*}(\varepsilon_r,k,D)$
13: $k = k-1$
14: **return** $(\mathcal{F}^{+*}\cup\{\mathcal{T}_{M^*}\},k)$

Algorithm 2: Private Model Selection

4.3 Learning Transitions

Noisy Forest Counts. The transition probabilities in the optimal model $(\mathcal{F}^{+*},k)$ can be learned by adding Laplace noise to the counts of the prefix trees corresponding to $\mathcal{F}^{+*}$. Each transition maps to exactly one prefix tree in $\mathcal{F}^{+*}$, and affects the counts of exactly one node at each of the k levels of the tree. Thus, adding or removing a user's trajectory table with at most L transitions affects at most $L\cdot k$ nodes. However, there is no apriori bound on L, resulting in an unbounded sensitivity for the forest counting query.

Rather than resorting to truncating a user's trajectories [6, 7] or sampling a bounded number of transitions from a user's trajectory table [4, 27], we choose to normalize the counts such that the total weight contributed by all transitions in a user's trajectory table is 1. That is, if PT has L transitions, then each transition will contribute a weight of $1/L$ to the counts in the prefix tree. Thus, a user's trajectory table contributes at most weight 1 to level i of all prefix trees, resulting in a sensitivity of k for the forest counting query.

We follow Cormode et al.'s [9] strategy of splitting the privacy budget ε_r geometrically across the different levels of the prefix trees. The roots of the prefix trees are assigned a privacy budget ε_0. Level i nodes receive a budget of $\varepsilon_i = \varepsilon_0 \cdot \gamma^i$, where $\gamma = 2^{\frac{1}{3}}$ is a multiplicative factor that increases the budget (resulting in smaller noise) as we move down the prefix tree (to account for lower counts). ε_0 is set such that $\sum_{i=0}^{k} \varepsilon_i = \varepsilon_r$. For a node at level i in a prefix tree, a noisy count $\tilde{c}(D,x) = c(D,x) + \eta$ is released, where η is drawn from Laplace$(1/\varepsilon_i)$.

Adaptive Pruning. Model selection prunes away prefix trees $\mathcal{T}$ with low support and chooses a height k. However, in each remaining tree $\mathcal{T}$, a large fraction of the nodes may have a count of 0. Releasing noisy counts for nodes with zero counts increases the size of the tree and dilutes the signal to noise ratio. Hence, as in prior work [6], after adding Laplace noise to nodes in the prefix tree, (including nodes with a count of 0), we determine some subtrees to remove. Our pruning strategy is different from prior work: Nodes in tree $\mathcal{T}$ are traversed breadth-first. With each node $x \in \mathcal{T}$, we associate a threshold $\tilde{\theta}(x)$ based on signal to noise ratio. If the noisy count $\tilde{c}(D,x)$ is greater than the threshold $\tilde{\theta}(x)$, we release the noisy count and continue. Otherwise, we prune away all the descendants of x. If the privacy budget (ε') that would have been used for a descendant of x is greater than that for node x, then we recompute the noisy count for x using ε'. We could potentially combine the two counts using techniques from [26] to improve the accuracy.

Choice of Threshold $\tilde{\theta}(x)$: There are many choices for setting the thresholds $\tilde{\theta}(x)$. A simple data independent method, which we call *fixed pruning*, is to apply a fixed threshold θ for all the nodes in the tree. When $\tilde{\theta}$ is too small, we would add noise to many nodes with a true count of 0. If $\tilde{\theta}$ is too large, we may prune away nodes with large true counts. Both scenarios result in poor utility.

Hence, we propose an *adaptive pruning* strategy to select different thresholds for different nodes such that pruning results in a reduction in total error with high probability. The intuition behind adaptive pruning is the same as that for our model selection strategy: if the count of a node x is much smaller than the number of its descendants, then the bias introduced by pruning the subtree below x is much smaller than the variance due to noisy counts (if the subtree is not pruned). This is formalized in the following lemma:

LEMMA 4.1. *Let $\mathcal{T}$ be an input tree of height k, and x a node at level ℓ. Let* $\mathrm{desc}(x,i)$ *denote the number of descendants of x in level i, ε_i the privacy budget allocated to nodes at level i by geometric budgeting, and $\varepsilon' = \varepsilon_r - \sum_{x' \text{ is a prefix of } x} \varepsilon(x')$. Let $\tilde{\mathcal{T}}$ be the noisy tree from the Laplace mechanism (with geometric budgeting). Let $\tilde{\mathcal{T}}_{\text{pruned}}$ be the result of pruning the subtree under x whenever*

$$c(D,x) < \theta(x) = \sqrt{\frac{2}{k-\ell+1}\left(\sum_{i=\ell+1}^{k} \frac{\mathrm{desc}(x,i)}{\varepsilon_i^2} + \frac{1}{\varepsilon(x)^2} - \frac{1}{\varepsilon'^2}\right)}. \tag{13}$$

Then, $\tilde{\mathcal{T}}_{\text{pruned}}$ has lower error (in expectation) than $\tilde{\mathcal{T}}$.

Note that for regular trees (where all nodes have the same degree), nodes in the same level are assigned the same threshold $\theta(x)$. However, adaptive pruning works after noise has been added to a node, and hence has no access to the true count. We use $\tilde{\theta}(x) = \theta(x) - \ln(1/\delta)/\varepsilon(x)$ to prune x's descendants when the noisy count $\tilde{c}(D,x) < \tilde{\theta}(D,x)$. This is because with probability $1-\delta$ if the noisy count $\tilde{c}(D,x)$ is smaller than $\tilde{\theta}(x)$ then true count $c(D,x)$ is smaller than the threshold $\theta(x)$ by Chernoff bounds. This pruning choice seems effective on real datasets in our evaluation.

Stochastic Transition Probabilities. In order for a prefix tree to represent stochastic transition probabilities, the counts in the tree should be non-negative. Additionally, the counts should satisfy *prefix consistency* i.e. the count at a node x must equal to the sum of the counts at its children. Consistency is ensured via constrained inference using least squares minimization [13]. We modify the original algorithm and proofs to work for the non-regular trees that result from pruning(we omit details for brevity). Non-negativity is ensured by (a) setting negative counts to 0 (if a node is not a leaf, all its descendants are set to 0); (b) normalizing the counts among children of the same parent node to obtain transition probabilities from the parent to the children; (c) computing the counts in a top-down approach from the original root count (obtained by consistency), given the stochastic transition probabilities from (b).

4.4 Direction Weighted Sampling

The order ℓ Markov process captures global properties of the trajectories, including direction, since the next value depends on the previous ℓ values. However, the noise added for ensuring privacy can mask this behavior. We introduce a novel direction weighted sampling scheme to recover the directionality in the data.

We encode the neighboring cells in a grid relative to the current cell. For instance, in Figure 3, the cell $(12,7)_1$ in reference system Σ_{v_1} has eight neighboring cells, one in each compass direction: $(11,6)_1$ in the southwest direction, $(12,6)_1$ in the south direction, etc. In order to avoid sudden unrealistic changes of direction, we modify our sampling procedure to remember the recent trend. Each direction, dir, has an equal weight when we start sampling a new trajectory, and the weight is updated along with the sampling process. Given the prefix of the trajectory generated so far, denoted by x, we set the weight of the direction dir to $w_{\text{dir}}(x) = \alpha^{c(\text{dir},x)}$, where α is a weighting factor greater than 1. We count the number of times x has followed direction dir by $c(\text{dir},x)$. However, in order to allow a long trajectory to turn locally, we use only the window of the last ω moves of x, denoted by $x_{[\omega]}$, to compute the weight of direction dir, i.e. $w_{\text{dir}}(x) = \alpha^{c(\text{dir},x_{[\omega]})}$. Sampling with window size $\omega = 0$ corresponds to the basic sampling method without considering the direction of the prefix of the trajectory. We show in our experiments that direction-weighted sampling with a reasonable window size ω can improve the utility of synthetic trajectories.

4.5 End-to-End Privacy Analysis

THEOREM 4.2. *The DPT system satisfies ε-differential privacy.*

PROOF. As we maintain normalized counts in prefix trees, not the number of transitions, adding or removing a user's trajectory table results in a total change of 1 in the i^{th} level of all prefix trees (for all i). This normalization does not require global knowledge and can be performed by each user independently. Thus, it is a *1-stable transformation* [17], and does not violate differential privacy.

The private model selection procedure in Algorithm 2 uses the noisy count at the root of all the prefix trees in $\mathcal{F}_{\text{full}}$, and thereafter does not use the trajectory database. Since we normalize trajectories such that the sum of the weights associated with the transitions of a single user sum to 1, adding or removing a user results in a total change of 1 to the set of root counts. Hence, by adding Lap$(1/\varepsilon_s)$ to each root count Algorithm 2 satisfies ε_s-differential privacy.

Noisy prefix trees are constructed by adding Laplace noise with sensitivity 1 to each of the prefix trees. The ε_r-budget is split geometrically across the levels. This again ensures ε_r-differential privacy. Adaptive pruning does not alter the differential privacy guarantee, since pruning decisions are made after noise is added. After pruning the subtree below x, the count of x maybe re-estimated using unused privacy budget. By composition properties of differential privacy, DPT ensures $\varepsilon_s + \varepsilon_r = \varepsilon$-differential privacy. □

5. EMPIRICAL EVALUATION

In this section, we empirically evaluate DPT over two large trajectory datasets. The evaluation results are presented in two parts:

- Synthetic trajectories released using DPT are more useful and scalable compared to prior work in our end-to-end evaluations. We show that the synthetic trajectories mirror the original trajectories on three utility metrics – distribution of diameter (i.e., distance traveled), conditional distributions of destinations given starting regions, and frequent patterns.
- The individual algorithms powering DPT (namely differentially private model selection, adaptive pruning and direction weighted sampling) outperform alternate solutions derived from prior work.

We first describe our datasets and utility metrics in Section 5.1, then list the design choices for DPT in Section 5.2. Next, we present empirical end-to-end scalability and utility results in Section 5.3 followed by utility evaluation on components of DPT in Section 5.4.

5.1 Datasets and Utility Metrics

Table 2 summarizes the two large trajectory datasets used in our experiments. Given Σ_{v_1}, a full set of hierarchical reference systems of size 6 are considered for both datasets, i.e. $\Sigma_{v_1}, \Sigma_{v_2}, \ldots, \Sigma_{v_6}$, where $v_m = v_{m+1}/2$. We will define Σ_{v_1} for each dataset below.

Taxi dataset: A set of GPS trajectories was recorded by 8602 taxi cabs in Beijing, China, during May 2009 [1]. The trajectories cover the region of Beijing within the bounding box (39.788N, 116.148W) and (40.093N, 116.612W) – approximately 34 km × 40 km. The finest resolution v_1 considered is 100 m × 100 m. The raw sampling rate of these trajectories ranges from 30 seconds to 5 minutes. The dataset consists of approximately 4.3 million trips with passengers. Each trip is linearly interpolated into a sequence of location points at 30-second sampling intervals.

Network dataset: A set of trajectories was synthesized by Thomas Brinkhoff's network-based generator [5] for moving objects. The data consists of 300 files of 50,000 individuals' trajectories of length up to 1,000 time units, sampled at equal time intervals. This data covers the region of Oldenburg, a city in Germany, within a bounding box of approximately 9 km × 10 km. The finest resolution v_1 considered is about 50 m × 50 m.

We developed three utility metrics Q_d, Q_t, Q_p^m to evaluate whether synthetic trajectories D_{syn} preserve aggregate properties of the original trajectories D_{raw}. We sampled 5 sets of 50,000 trajectories from the DPT model for D_{syn} and evaluated against 5 sets from D_{raw} each of which consists of 50000 randomly selected raw trajectories. We then reported the relevant mean error or accuracy with its standard deviation. The three metrics are described below.

Metric I: Q_d - Diameter Distribution.
The diameter for a trajectory $t = \sigma_1 \cdots \sigma_n$ is the maximum distance between any pair of locations in t, i.e. $\max_{i,j} d(\sigma_i, \sigma_j)\ \forall i, j = 1, \ldots, n$. Let $Q_d(D)$ denote the empirical distribution of the diameter on trajectory database D, where the diameter is quantized into 25 equal width buckets $\{[0,x), [x,2x), \ldots, [24x, 25x)\}$. x is set to 800m for Taxi dataset, and 400m for Network dataset (both correspond to the resolution Σ_{v_4}). The error in Q_d is measured by:

$$\mathcal{E}_{\text{diameter}} = JSD(Q_d(D_{\text{syn}}), Q_d(D_{\text{raw}})), \tag{14}$$

where $JSD(\cdot,\cdot)$ is the Jensen Shannon divergence with range $[0, \ln 2]$.

Metric II: Q_t - Trip Distribution.
Given a coarse grid Σ_v, a starting region a_s is a square consisting of 3×3 cells from Σ_v. Let $D_{a_s \to a}$ be the set of trajectories in D which starts from the region a_s and ends at the cell $a \in \Sigma_v$. Trip distribution Q_t measures the distribution of the ending point a in $D_{a_s \to a}$ given a_s, i.e. $\Pr[\cdot|a_s, D]$. A set of 1000 starting regions S are randomly chosen. v is set to 4km for Taxi and 1km for Network dataset. Based on $JSD(\cdot,\cdot)$, the error in Q_t is measured as

$$\mathcal{E}_{\text{trip}} = \frac{1}{|S|} \sum_{a_s \in S} JSD(\Pr[\cdot|a_s, D_{\text{syn}}], \Pr[\cdot|a_s, D_{\text{raw}}]). \tag{15}$$

Metric III: Q_p^m - Frequent Patterns.
We first project the path taken by each trajectory in D on a grid Σ_{v_m}, where the path is formed by connecting consecutive points with a straight line. We obtain the sequence of cells the trajectory passes through, and represent this sequence of cells with their corresponding anchor points in Σ_{v_m}, for $m \in \{2, \ldots, 6\}$. The query Q_P^m takes in D and outputs the top p patterns from $\Sigma_{v_m}^k$ with the highest count, for $k = 2, \ldots, 6$. We consider p to be 1000 and 10000 in our evaluations. The utility in Q_p^m of D_{raw} w.r.t. D_{syn} is measured as

$$\mathcal{A}_{\text{pattern}} = F_1(Q_p^m(D_{\text{raw}}), Q_p^m(D_{\text{syn}})), \tag{16}$$

where $F_1(\cdot,\cdot)$ is the F_1 score with range $[0,1]$ (i.e., harmonic mean of precision and recall) – a similarity measure between item sets.

Dataset	$\lvert D\rvert$	$L_{\max}$	L_{mean}	Σ_{raw} (km^2)	v_1 (m)	$\lvert\Sigma_{v_1}\rvert$
Taxi	4,268,780	1919	20.03	34×40	100	306×463
Network	1.5×10^7	977	75.12	9×10	50	177×201

Table 2: Datasets Summary

settings	default	alt	parameters
HRS	opt	full; Σ_{v_5}	maximum tree height: $k_{max} = 8$
pruning	adapt	fixed; ngram; np	privacy budget ε: 2.0,1.0,0.5,0.3,0.1; model selection budget: $\varepsilon_s = 0.1\varepsilon$; pruning parameter δ : 0.1
sampling	dir	nodir	factor α: 1.2; window size ω: $0, 4, \ldots, 32$

Table 3: Design choices

5.2 Design Choices

We consider alternative implementations of DPT by varying the structure of the HRS model (namely the reference systems used), the pruning strategy and the sampling method as shown in Table 3. In the default setting [opt, adapt, dir], opt refers to the model output by private model selection, adapt refers to adaptive pruning, and dir refers to direction weighted sampling. Two alternate models are presented to opt: full and Σ_{v_5}, where full refers to the full model consisting of all the reference systems from Σ_{v_1} to Σ_{v_5}. Σ_{v_6} is not in full as we do not expect to see transitions in Σ_{v_6} since transitions in Σ_{v_6} corresponds to movements at the speed of greater than 240 miles per hour. The maximum tree height k_{max} in private model selection is set to 8, but it can be set arbitrarily large and will not affect the output of model selection. The alternate models use the same height as the optimal model in the evaluation. Alternatives to the pruning strategy are fixed which uses a fixed threshold of 0 (described in Section 4.3), and ngram, the pruning strategy used in [6]. np refers to the non-private model without noise. Lastly, direction-weighted sampling is studied by varying the window size ω. nodir refers to the case when $\omega = 0$.

5.3 End-to-End Evaluation

In the end-to-end evaluation, we compare DPT with the most relevant prior work [6] denoted by seq. This prior work computes a noisy prefix tree on the entire domain by adding Laplace noise followed by pruning (sanitization phase), and then uses the noisy prefix tree to reconstruct a database of sequences (reconstruction phase). For seq, we transform the first $l_{\max}$ sampled points of

each trajectory with a single reference system Σ_{v_m}, for $m = 1, \ldots, 6$, where $l_{\max}$ is the truncation length used in seq to bound sensitivity as the input. Then this input is processed by the original code of seq provided by [6], resulting in a differentially private synthetic trajectory database. For end-to-end evaluations, we use the default setting of DPT– [opt, adapt, dir].

Scalability. We ran experiments on a single machine with 2.40GHz 30MB Cache (256GB RAM - 48 hyperthreaded cores) 5 times, and report the average time for sanitization and reconstruction of 50,000 trajectory samples. In Figure 1 on page 2, we show the runtime of seq with increasing resolution from v_6 to v_2 (a $4\times$ increase in domain size each time) for $l_{\max} = 20$ and k-grams for $k \leq 4$ when $\varepsilon = 1.0$. The time for sanitization increases exponentially with the domain size, i.e. $O(|\Sigma|^k)$, since seq considers the transitions between all pairs of points in the domain. In comparison, DPT uses the optimal model $\{\Sigma_{v_4}, \Sigma_{v_5}\}$ with a fanout of constant size, which takes orders of magnitude less time to complete the sanitization process. In addition, seq aims to reconstruct a database of all possible trajectories by joining shorter trajectories to form longer ones. When a database has a large size, this approach takes a very long time. When resolution is Σ_{v_2}, the reconstruction takes more than 5 hours to complete. We observe that the reconstruction time decreases from v_2 to v_3 and then increases again from v_3 to v_6. This is because many consecutive points are mapped to the same cell indices in the coarse reference system. This allows the reconstruction process to form longer trajectories from shorter trajectories of repeated points. For instance, the maximum length of trajectories reconstructed in Σ_{v_6} reaches 20, while the maximum length in Σ_{v_2} is only 5. Therefore, a sampling approach is preferred as it can quickly produce a set of good representative trajectories with various lengths and can be parallelized easily. Results for seq at resolution v_1 are not reported as the program does not terminate even after a day.

Each of the steps in DPT (as shown in Figure 2) is efficient to implement. There are three main steps: preprocessing (HRS mapping and prefix construction), sanitizing (model selection, noise infusion, adaptive pruning) and sampling. The runtime complexity of the three steps are linear respectively in input data size, domain size and output size. On an input of over 4 million trajectories with domain size over 2000, preprocessing takes around 10 minutes while sanitizing takes around 35 seconds (adaptive pruning contributes to the fast speed of this process). Around 50 seconds are required to generate 50,000 samples. Due to space constraints, we omit detailed evaluation for each component.

Utility. In Figure 5, we show the JSD error for diameter distribution Q_d, and trip distribution Q_t, and F_1 score for frequent patterns Q_p^m respectively when $\varepsilon = 1.0$. For both error metrics, as the resolution gets finer (from v_6 to v_2), the error gets worse. DPT improves upon seq by $> 80\%$ on Q_d error and $> 60\%$ on Q_t error. For Q_p^m, given the resolution at which seq is processed, we report the accuracy score of the top $1,000$ frequent patterns at the same resolution: both trajectories generated by seq and DPT are projected on Σ_{v_m} for the evaluation of frequent patterns. The comparison shows that DPT obtains better F_1 score at resolutions v_6 to v_3, with 30% to 200% improvements, even though DPT is not necessarily generated at Σ_{v_m}. At v_2, DPT performs poorly because its optimal model selected is $\{\Sigma_{v_4}, \Sigma_{v_5}\}$, and hence it is unlikely to capture information at fine resolution. Experiments at smaller ε also show that DPT outperforms seq. We omit the figures due to space constraints.

5.4 Component Utility Evaluation

In this section, we report the evaluation on three components: 1) private model selection, 2) adaptive pruning and 3) direction-

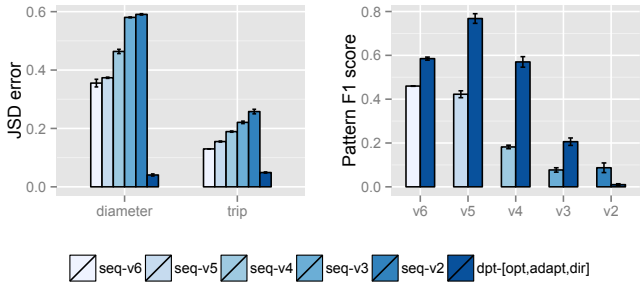

Figure 5: Utility of DPT v.s seq for Taxi, at $\varepsilon = 1.0$

ε	2	1	0.5	0.3	0.1
k=2	v_2, v_3, v_4, v_5	v_3, v_4, v_5	v_3, v_4, v_5	v_3, v_4, v_5	v_4, v_5
k=3	v_3, v_4, v_5	v_3, v_4, v_5	v_3, v_4, v_5	v_4, v_5	v_4, v_5
k=4	v_3, v_4, v_5	v_4, v_5	v_4, v_5	v_4, v_5	v_5
k=5	v_4, v_5	v_4, v_5	v_5	v_5	v_5
k=8	v_5	v_5	v_5	v_5	v_5

Table 4: Monotonicity of Taxi dataset in Section 4.2

weighted sampling on both Taxi (T) and Network (N) datasets. For each component, we will consider alternate choices defined in Section 5.2. The setting [full, np, dir] is chosen as the benchmark for the optimal non-private (np) model, because full is unbiased and close to optimal among all non-private models. All three utility metrics are measured and frequent patterns Q_p^m are reported at Σ_{v_4} for $p = 1,000$ and at Σ_{v_5} for $p = 10,000$ patterns.

Component I: Model Selection. Table 4 shows the optimal model selected by private model selection (Algorithm 2) on Taxi dataset. As k increases and ε decreases, the signal to noise ratio gets weaker, and only a smaller subset of reference systems can be supported. This validates the monotonicity property from Section 4.2. We observe similar trends for Network dataset.

We also find that the optimal model performs considerably better than its alternatives. In Figure 6, we present two obvious alternate models: the private full model [full, adapt, dir] and the private coarsest reference system [v_5, adapt, dir]. The optimal model opt approached the high accuracy of the non-private full model [full, np, dir] as ε increases. At $\varepsilon = 0.1$ opt achieves accuracy/error within 85% of [full, np, dir] for Taxi dataset.

Though the non-private full model preserves good utility, the optimal model consistently beats the private version of the full model [full, adapt, dir] as the private full model incurs too much noise. Moreover, opt outperforms Σ_{v_5} at most settings. When ε is large opt significantly outperforms Σ_{v_5}. For instance, for the Network dataset, the optimal model for $\varepsilon = 1$ is $\{\Sigma_{v_2}, \Sigma_{v_3}, \Sigma_{v_5}\}$, and thus captures more fine detail than Σ_{v_5}, resulting in low error. On the other hand, at small ε, Σ_{v_5} may be the optimal model (but opt picks a different model due to noise). In those cases, Σ_{v_5} can outperform opt. Furthermore, the good performance of Σ_{v_5} for Q_p^5 and Q_t can be explained by the fact that Q_p^5 and Q_t are measured at the same resolution as Σ_{v_5} or at a coarser resolution.

Component II: Adaptive pruning. We substitute the default setting of the second component (sanitizing method) — geometric noise with adaptive pruning methods (adapt), by its alternatives — geometric noise with fixed threshold pruning at threshold 0 (fixed), and n-gram methods (ngram) [6]. In Figure 7, we present the evaluation results on synthetic trajectories generated

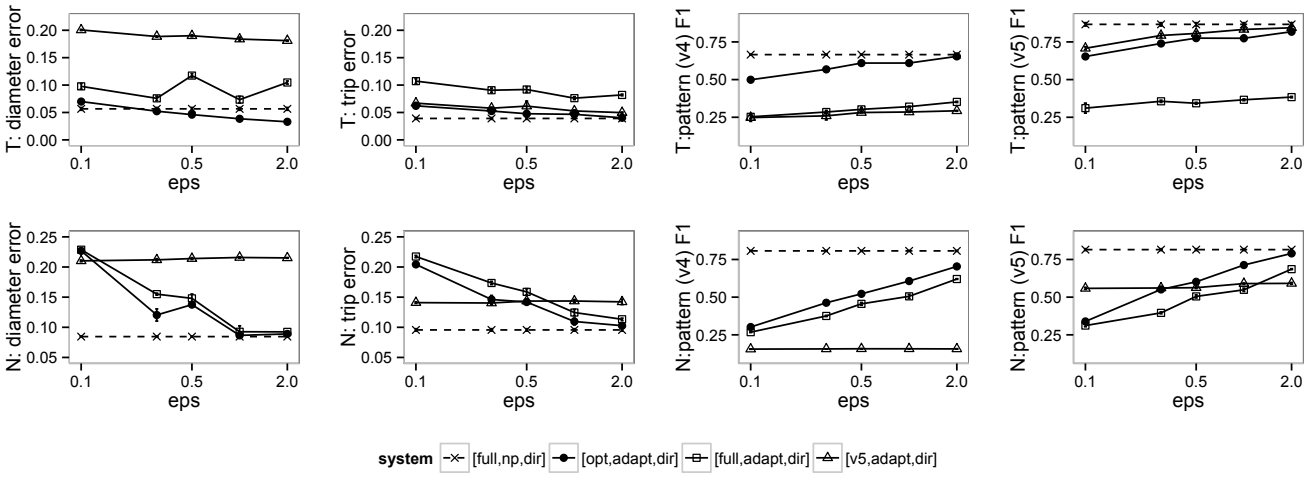

Figure 6: Differentially private model selection v.s. other models

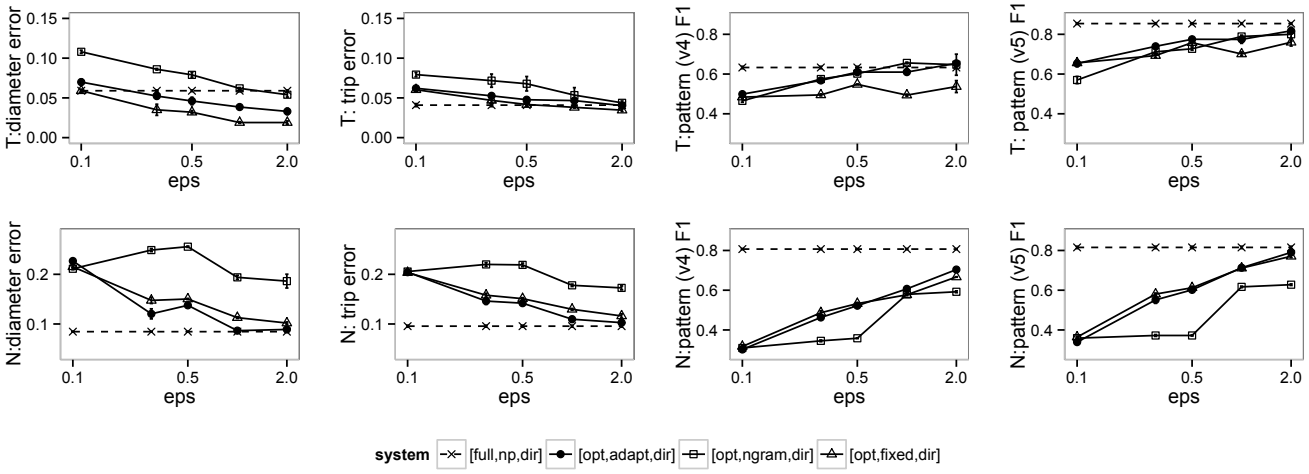

Figure 7: Differentially private pruning methods

from these three settings [opt, adapt, dir], [opt, fixed, dir], [opt, ngram, dir]. We observe that adapt performs just as well as ngram for Q_p^5, Q_p^4 on Taxi dataset, but gives better accuracy than ngram for other metrics. We conjecture two reasons for the poor performance of ngram: First, ngram can effectively estimate the right subtree height at each interval node and prune away subtrees with low support when the given tree height is large. However, our private model selection has considered the optimal tree height in opt and hence, the height estimation in ngram is no longer useful for opt. Second, the privacy budget is split equally amongst nodes on the root-to-leaf path which survive after ngram pruning. Uniform budget splits have been shown to be less accurate than geometric noise in [9]. Furthermore, fixed seems to perform as well as adapt. We further studied the noisy prefix trees output by the three pruning strategies. We find that fixed adds the most number of false nodes (nodes with true count 0) to the noisy trees leading to zigzag trajectories when sampled without dir. Direction-weighted sampling, however, lifts the accuracy of fixed.

Component III: Direction-weighted Sampling. The intuition behind direction-weighted sampling is to minimize unnecessary local turns in the synthetic trajectories. In this evaluation, we vary the direction setting (dir) in the default setting of DPT [opt,adapt,dir], by changing the window size ω from 0 to 32. $\omega = 0$ refers to no-direction weighted sampling (nodir). Default window sizes are set to 25% of the average length of raw trajectory dataset (6 and 16 for Taxi and Network respectively). As direction-weighted sampling has a larger impact on longer trajectories, we filter out Taxi synthetic trajectories consisting of more than 15 sampled points for evaluation, where 15 is the median number of location points in the ground truth. For Network dataset has an average length greater than the maximum window size evaluated, we use all synthetic trajectories for evaluation. As shown in Figure 8, direction-weighted sampling re-enforces the directionality by smoothing the local turns, and hence results in lower errors and higher F_1 scores. For non-private model (np), the improvement by this direction constraint is not as significant as in the private models. For $\varepsilon = 0.5, 0.1$, the utility of private models improves as window size increases and stabilizes at a window size slightly greater than 4 (the order of the Markov process). The JSD errors for Q_d and Q_t are reduced up to 20% and the F_1 scores are improved up to 30%.

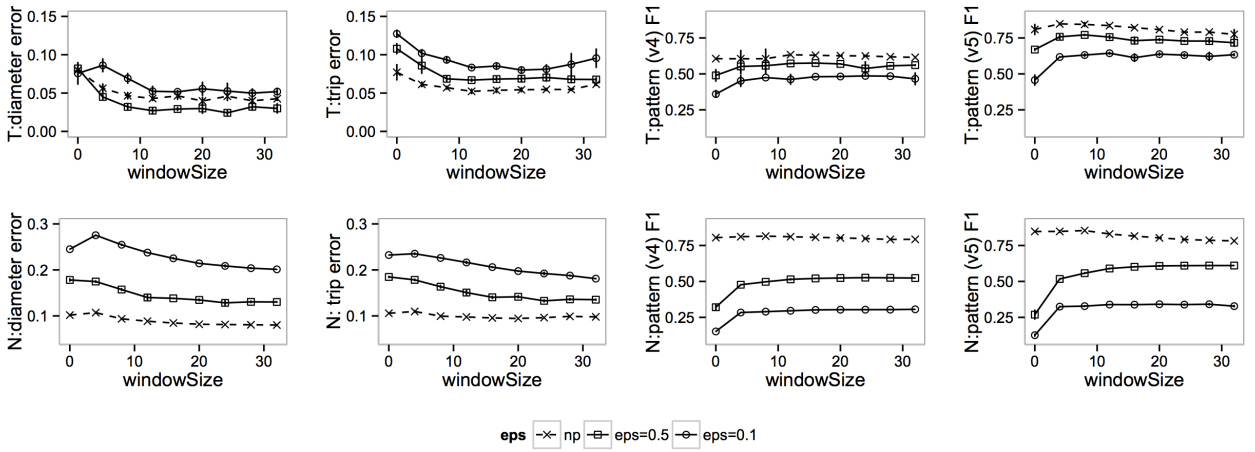

Figure 8: Direction-weighted sampling

6. CONCLUSIONS

The problem of releasing realistic but private trajectory data takes on new importance as more mobility data is produced, and more pressures arise to share versions of the data. With DPT, we introduce the first system devoted to doing so for natural, speed-varying trajectories under differential privacy. Our analysis and empirical study demonstrate that this is an effective way to reveal information, while provably protecting privacy.

There is much opportunity for extending this line of work. While effective in practice, we have yet to show strong analytic guarantees for the utility of data released via DPT. We have focused on reference systems based on multi-resolution grids, but other systems could lead to better results for certain trajectories where paths are more constrained. We believe that adapting existing interpolation techniques [24] or HRS based on sampling rates instead of speeds can allow DPT to work with irregular trajectories. Last, as airborne sensors and drones become more prevalent, we may extend our current focus on trajectories within the (2D) plane to three dimensional trajectories, where the data density further decreases.

7. REFERENCES

[1] Taxi trajectory open dataset, Tsinghua university, China. http://sensor.ee.tsinghua.edu.cn, 2009.

[2] O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *ICDE*, pages 376 – 385, 2008.

[3] R. A. Becker, R. Cáceres, K. Hanson, S. Isaacman, J. M. Loh, M. Martonosi, J. Rowland, S. Urbanek, A. Varshavsky, and C. Volinsky. Human mobility characterization from cellular network data. *Commun. ACM*, 56(1):74–82, 2013.

[4] L. Bonomi and L. Xiong. A two-phase algorithm for mining sequential patterns with differential privacy. In *CIKM*, pages 269–278, 2013.

[5] T. Brinkhoff. A framework for generating network-based moving objects. *GeoInformatica*, 6(2):153–180, 2002.

[6] R. Chen, G. Acs, and C. Castelluccia. Differentially private sequential data publication via variable-length n-grams. In *CCS*, pages 638–649, 2012.

[7] R. Chen, B. C. Fung, B. C. Desai, and N. M. Sossou. Differentially private transit data publication: a case study on the montreal transportation system. In *KDD*, pages 213–221, 2012.

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

[9] G. Cormode, M. Procopiuc, D. Srivastava, E. Shen, and T. Yu. Differentially private spatial decompositions. In *ICDE*, pages 20–31, 2012.

[10] Y.-A. de Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel. Unique in the crowd: The privacy bounds of human mobility. *Sci. Rep.*, 3(1376), 2013.

[11] C. Dwork. Differential privacy. In *ICALP*, pages 1–12, 2006.

[12] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[13] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private histograms through consistency. *PVLDB*, 3(1):1021–1032, 2010.

[14] H. Hu, J. Xu, S. T. On, J. Du, and J. K. Ng. Privacy-aware location data publishing. *ACM Trans. Database Syst.*, 35(3), 2010.

[15] H. Jeung, H. T. Shen, and X. Zhou. Mining trajectory patterns using hidden markov models. In *DaWaK*, pages 470–480. Springer, 2007.

[16] D. Kopanaki, N. Pelekis, A. Gkoulalas-Divanis, M. Vodas, and Y. Theodoridis. A framework for mobility pattern mining and privacy-aware querying of trajectory data. In *HDMS*, 2012.

[17] F. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *SIGMOD*, pages 19–30, 2009.

[18] A. Monreale, G. L. Andrienko, N. V. Andrienko, F. Giannotti, D. Pedreschi, S. Rinzivillo, and S. Wrobel. Movement data anonymity through generalization. *Transactions on Data Privacy*, 3(2):91–121, 2010.

[19] J. Norris. Discrete-time markov chains. *Markov Chains*, 2004.

[20] F. Pratesi, A. Monreale, H. Wang, S. Rinzivillo, D. Pedreschi, G. Andrienko, and N. Andrienko. Privacy-aware distributed mobility data analytics. In *SEBD*, 2013.

[21] W. Qardaji, W. Yang, and N. Li. Differentially private grids for geospatial data. In *ICDE*, pages 757–768, 2013.

[22] 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 *DEXA*, pages 357–365, 2013.

[23] C. Song, Z. Qu, N. Blumm, and A.-L. Barabsi. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, 2010.

[24] H. Su, K. Zheng, H. Wang, J. Huang, and X. Zhou. Calibrating trajectory data for similarity-based analysis. In *SIGMOD*, pages 833–844, 2013.

[25] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In *MDM*, pages 65–72, 2008.

[26] X. Xiao, G. Bender, M. Hay, and J. Gehrke. iReduct: Differential privacy with reduced related errors. In *SIGMOD*, pages 229–240, 2011.

[27] S. Xu, S. Su, X. Cheng, Z. Li, and L. Xiong. Differentially private frequent sequence mining via sampling-based candidate pruning. In *ICDE*, 2015.

[28] R. Yarovoy, F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Anonymizing moving objects: How to hide a mob in a crowd? In *EDBT*, pages 72–83, 2009.