Private Spatial Data Aggregation in the Local Setting

Rui Chen
Samsung Research America
Mountain View, CA
rui.chen1@samsung.com

Haoran Li
Emory University
Altanta, GA
hli57@emory.edu

A. K. Qin
RMIT University
Melbourne, Australia
kai.qin@rmit.edu.au

Shiva P. Kasiviswanathan
Samsung Research America
Mountain View, CA
shiva.k01@samsung.com

Hongxia Jin
Samsung Research America
Mountain View, CA
hongxia@acm.org

***Abstract*—With the deep penetration of the Internet and mobile devices, privacy preservation in the *local setting* has been an increasing demand in many real-life applications. The local setting refers to the scenario where a user trusts nobody else and is willing to share his/her information only if it has been properly sanitized before leaving his/her own device. Moreover, a user may hold only a single data element to share, instead of a database. Despite its ubiquitousness, the above facts make the local setting substantially more challenging than the traditional centralized or distributed settings.**

In this paper, we for the first time study the problem of private spatial data aggregation in the local setting, which finds its way in many real-world applications, such as Waze and Google Maps. In response to users' different privacy requirements that are natural in the local setting, we propose a new privacy model called *personalized local differential privacy* (PLDP) that allows to achieve desirable utility while still providing rigorous privacy guarantees. We design an efficient personalized count estimation protocol as a building block for achieving PLDP and give theoretical analysis of its utility, privacy and complexity. We then present a novel framework that allows an untrusted server to accurately learn the user distribution over a spatial domain while achieving PLDP for each user. This is mainly achieved by designing a novel user group clustering algorithm tailored to our problem. We confirm the effectiveness and efficiency of our framework through extensive experiments on multiple real benchmark datasets.

I. Introduction

The fast penetration of the Internet and mobile devices has allowed users to easily contribute their personal data to different mobile apps that are changing the way we live. For example, *WeatherSignal* (http://weathersignal.com) provides weather information in any given region by using a user' mobile as a weather station; *Stereopublic* (http://www.stereopublic.net) creates a sound map of a city by soliciting audio data from users so that it can lead users to quiet urban areas and at the meantime create an audio archive of the cityscape; *Placemeter* (https://www.placemeter.com/) collects videos from users' mobile devices and turns them into meaningful aggregated data that could be used by local businesses, marketers and urban planners; *Waze* (https://www.waze.com) provides real-time traffic conditions by collecting users' GPS locations.

The process of soliciting contributions from a large group of users, as in the above examples, has resulted in the new *crowdsourcing* business model. It is expected that crowdsourcing-based apps will keep booming and further benefit our lives. However, the personal data contributed by users brings huge privacy threats, which have been a major impediment to the wide acceptance of crowdsourcing [22]. Indeed, the willingness of users, often unpaid volunteers, to contribute is directly related to the privacy risks of their shared data. Since gaining a sufficiently large number of data contributors is critical to the success of any crowdsourcing-based app, developing practical privacy technologies with rigorous privacy guarantees is an urgent need.

Differential privacy [7] is the state-of-the-art privacy notion that provides provable privacy protection independent of an adversary's background knowledge and computational power. Unfortunately, crowdsourcing-based apps lead to a new data sharing setting—the *local setting*, which is rarely studied under differential privacy. Distinctive from the centralized setting, in the local setting, a user trusts nobody else and demands that his/her data be properly sanitized before leaving his/her own device. Moreover, the user's data may be a single element, rather than a database, which is a common assumption in the traditional distributed setting [12], [18], [19]. As a result, the local setting is much more challenging. In particular, the well-established Laplace mechanism [7] and the exponential mechanism [16] we rely on to achieve differential privacy are no longer capable of obtaining reliable results.

In this paper, we for the first time study the practical problem of differentially private spatial data aggregation in the local setting, which underpins many real-world applications, such as *Waze* and *Google Maps*. In our problem, given a large number of users who have a location generated by their mobile devices, an untrusted server would like to learn users' distribution over a spatial domain (so that, for example, it can generate a traffic congestion map) without learning any user's true location. The spatial domain is composed of a set of disjoint locations, which we also call *location universe*. In the rest of this paper, we use spatial domain and location universe interchangeably. A user defends his/her privacy by deploying an *on-device* perturbation mechanism that sanitizes the location before sending it to the server. The system architecture of our problem is illustrated in Figure 1.

Such a problem renders several non-trivial technical challenges. First, how can we properly capture users' privacy requirements in the local setting? The *local differential privacy* (LDP) model [15] proposed for the local setting requires that, by seeing a user's perturbed information, an adversary not be able to statistically distinguish his/her true location from *any* other location in the universe. That is, any user altering his/her true location into any other location should have a negligible

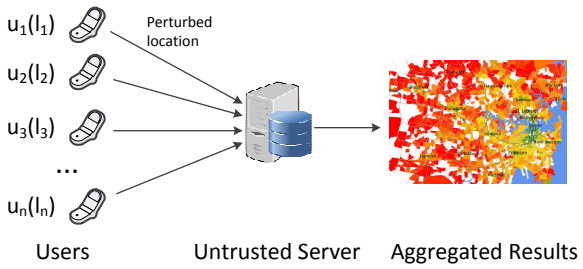


Fig. 1. The system architecture of our problem

effect on the output of the on-device perturbation mechanism. When the location universe is large, as is the case for most real-life applications, LDP, unfortunately, is too restrictive to obtain reasonable accuracy. Even worse, such a utility loss does not necessarily accompany privacy gain. In reality, an adversary could learn the rough area of a victim with high certainty. For example, by seeing a victim's physical presence in the workplace, an adversary almost ascertains that the victim would be in the city for the day and thus can easily filter out the perturbed locations anywhere else. Moreover, we observe that the same location is not equally sensitive to every user and that a user has the best understanding of the sensitivity of his/her location. These observations drive us to advocate a personalized privacy notion. We propose the *personalized local differential privacy* (PLDP) that provides high flexibility for users to specify personalized privacy requirements and that allows each user to take full control of his/her privacy, independent of other users' privacy settings.

Second, since the Laplace mechanism and the exponential mechanism are incapable of achieving desirable accuracy in the local setting, how can we design a basic mechanism that accurately computes users' aggregated information while respecting PLDP? The key challenge comes from the fact that each user may hold only a single location that needs to be shared. This could be possibly solved by secure multi-party computation and/or other encryption-based techniques. However, these techniques violate the basic setting of the existing applications, for example, users normally cannot directly communicate with each other, and thus require substantial changes to the existing architecture. Furthermore, these techniques face known scalability issues that restrain them from real-world deployments. In view of these shortages, we resort to *randomized response*-based techniques for provable privacy guarantee. More specifically, we propose the personalized count estimation protocol (PCEP) that adapts the private frequency oracle in [3] to accommodate personalized privacy requirements. We also put efforts to improve the efficiency and communication cost so that it can suit the limited resources on mobile devices.

Third, with the PCEP protocol, how can we build a unified framework for an untrusted server to learn accurate aggregated statistics while respecting users' personalized privacy requirements? We observe that the crux is how to group users to feed into different copies of PCEP. This procedure is critical to the resultant privacy and utility. Motivated by two extreme solutions, we formulate an optimization problem to identify the fruitful middle ground that minimizes the maximum absolute error over all locations in the universe and solve it by designing novel clustering techniques tailored to our problem.

Contributions. Motivated by real-life demands, in this paper we for the first time study the problem of private spatial data aggregation in the challenging local setting. Our contributions are four-fold:

- We formulate the private spatial data aggregation problem in the local setting. In response to users' natural personalized privacy requirements in such a problem, we propose a novel personalized local differential privacy (PLDP) model that provides a better trade-off between privacy and utility. While being proposed in the context of spatial data, our PLDP model is also applicable to other types of data.
- We design an efficient personalized count estimation protocol PCEP that serves as the basic building block for achieving PLDP. We theoretically analyze its privacy and utility guarantees as well as its complexity. We show that it is a lightweight protocol that is suitable for mobile devices.
- We develop a unified private spatial data aggregation (PSDA) framework for an untrusted server. It features a novel user group clustering component that guides how to feed different users into different PCEP protocols in order to maximize utility. We also develop post-processing techniques by enforcing the constraints placed by users' personalized privacy requirements.
- We experimentally evaluate the performance of our framework over large-scale real benchmark datasets. We show the superiority of our PLDP model to provide a better trade-off between privacy and utility. We also demonstrate the effectiveness and efficiency of our PSDA framework for practical use.

The rest of the paper is organized as follows. Section II reviews the related literature. Section III formulates our problem with some necessary background. Section IV discusses our sanitization framework in detail and theoretically analyzes its performance. Section V presents our experimental results. Finally, Section VI concludes our work.

II. Related Work

Private Spatial Data Aggregation. Differentially private spatial data aggregation has been previously studied in the *centralized* setting. Cormode et al. [5] design differentially private spatial decomposition techniques to generate the number of data points within each spatial region in order to answer range queries over arbitrary query regions. This is achieved by building various differentially private tree structures that recursively partition the space into smaller regions. Qardaji et al. [20] propose a two-level adaptive grid method whose key challenge is to choose the suitable grid granularity. This method first places a coarse-grained grid over the spatial domain and then partitions the cells based on their noisy densities.

Mir et al. [17] present a differentially private approach to model human mobility patterns based on call-detail-records (CDRs). The core idea is to calculate five types of noisy distributions that can be used to generate synthetic CDRs, which are then used to estimate population densities. Acs and Castelluccia [2] conduct a case study of privately releasing

the number of individuals in 989 different areas in Paris every hour for an entire week based on CDRs. They propose several optimization mechanisms based on the public characteristics of the dataset and the application to improve data utility. To et al. [22] study the problem of private spatial decomposition in the context of spatial crowdsourcing. Specifically, they adapt the approaches in [5], [20] to meet the specific requirements of the spatial crowdsourcing framework. As explained before, the techniques developed in the above studies, unfortunately, cannot be applied to the local setting.

Distributed Differential Privacy. There have been some recent studies that apply differential privacy to the distributed setting. Mohammed et al. [18] propose an algorithm for privately integrating data vertically partitioned between two parties in the semi-honest adversary model. This is achieved by a two-party protocol for the exponential mechanism. Pathak et al. [19] propose a differentially private protocol for learning an aggregate classifier from a collection of distributed private data. In the protocol, the individual parties locally compute an optimal classifier with their local data. These individual classifiers are then averaged by an untrusted curator to obtain the aggregate classifier by using a secure protocol based on asymmetric key additively homomorphic encryption. Hong et al. [12] design a collaborative search log sanitization (CELS) protocol to enable distributed parties to collaboratively generate sanitized search logs. The CELS protocol satisfies the weaker (ϵ, δ)-differential privacy notion to trade for boosted utility.

Shi et al. [21] propose a construction that allows an untrusted server to learn the sum statistics from distributed time-series data, which mainly relies on cryptographic techniques. Acs and Castelluccia [1] introduce a novel cryptographic protocol for smart metering systems, whose merit is not requiring message exchange among users. Hardt and Nath [11] apply differential privacy to personalized mobile advertising. Specifically, they improve the protocol in [1] by assuming the existence of a server and a proxy that do not collude with each other. Chen et al. [4] develop a distributed differentially private system to support statistical queries over multiple clients' data via an honest-but-curious proxy.

All these solutions lean on secure multi-party computation, which is known to be computationally prohibitive to deploy in real-world applications and have robustness issues in face of party failures. Furthermore, many of these solutions assume each distributed party holds a database. In contrast, the local setting we consider is substantially more challenging in that each user has only a single data point (e.g., a location).

Local Differential Privacy. In spite of its practical use, local differential privacy is rarely studied in the literature. The very few existing studies are mainly of theoretical interest only. It is still not clear how to apply local differential privacy to real-world problems. The notion of local differential privacy was first proposed in [15]. In this paper, Kasiviswanathan et al. study the classes of learning problems that are solvable in the local setting. Duchi et al. [6] provide general techniques for deriving minimax bounds under local differential privacy. In particular, they illustrate the use of these techniques for two canonical problems, mean estimation and convex risk minimization. Bassily and Smith [3] propose protocols to produce a succinct histogram of the input data under local differential privacy. They design a concise local randomizer and a frequency oracle, which will be adapted in this paper as the basic building blocks in our solution.

Kairouz et al. [14] introduce a family of extremal mechanisms to address the trade-off between local differential privacy and utility. In particular, they present two simple extremal mechanisms, the binary and randomized response mechanisms. Though they are of theoretical interest, these results, unfortunately, cannot provide meaningful utility in real-world applications. To our best knowledge, the only application of local differential privacy to a real-world problem is the Randomized Aggregatable Privacy-Preserving Ordinal Response (RAPPOR) protocol [8], which privately collects statistics about an arbitrary set of strings from end users. Its key idea is to apply randomized response to Bloom filters. However, it has been shown that the utility provided by RAPPOR is less desirable than the technique in [3].

To our best knowledge, we are the first to apply local differential privacy to spatial data aggregation. We will show that with careful design it is possible to simultaneously achieve high utility and privacy for practical use.

Personalized Privacy Models. Personalized privacy notions originate from the observation that different users may have different privacy expectations. Gedik and Liu [9] present a personalized k-anonymity model for sharing location information. It enables each mobile node to specify the minimum anonymity level (i.e., k value) it desires. Xiao and Tao [23] propose the concept of *personalized anonymity* that allows a user to specify his/her privacy requirement in terms of a guarding node in the taxonomy of a sensitive attribute. The published table guarantees that for all users an adversary cannot associate them with any leaf value of their guarding nodes. Yuan et al. [24] define three levels of privacy protection in the context of social networks by assuming three levels of background knowledge. The very recent effort by Jorgensen et al. [13] is to personalize differential privacy. Users are allowed to specify their own privacy parameter, and the key idea of achieving this personalized differential privacy is a non-uniform sampling procedure at the user level.

All these personalized privacy models assume a trusted server. In contrast, our PLDP is tailored to the local setting where users perform on-device perturbation to protect their privacy.

III. Preliminaries and Problem Formulation

A. Local Differential Privacy

The traditional differential privacy [7] was proposed for the *centralized* model, which assumes the existence of a *trusted* data curator that has access to the private data. However, this is often not the case in many real-world applications. With the increasing awareness of the erosion of privacy in the Internet era, users tend to protect their privacy at the root. That is, they trust nobody else and are willing to share their information only if it has been properly sanitized before leaving their own devices. Moreover, due to the prevalence of mobile devices, often users may hold only a single data element to share (e.g., the current locations), rather than a database. These facts correspond to the *local setting*.

The notion of local differential privacy (LDP) [15] was consequently proposed for the local setting. Intuitively, it requires that no matter which data value a user has, the data recipient (e.g., an untrusted server) should receive almost the same information. In other words, by seeing the perturbed information, an adversary with arbitrary background knowledge is not able to distinguish the user's original value. Formally, LDP is given below.

Definition 3.1 (Local Differential Privacy [15]): A randomized algorithm $\mathcal{A}$ satisfies ϵ-local differential privacy (or ϵ-*LDP*), if for any pair of values $l, l' \in \mathcal{L}$, and for any $O \subseteq Range(\mathcal{A})$,

$$P[\mathcal{A}(l) \in O] \leq \exp(\epsilon) \cdot P[\mathcal{A}(l') \in O],$$

where the probability is over the coin flips of $\mathcal{A}$.

While LDP provides rigorous privacy protection in the local setting, its drawback is on the utility side. In real-world applications, where the universe size $|\mathcal{L}|$ is large, achieving a reasonable trade-off between privacy and utility under LDP is almost a chimera, as we will show later in Section V. For this reason, we explore more practical privacy notions that strike a better trade-off in the local setting.

B. Personalized Local Differential Privacy

A salient property of the local setting is that a user can take full control of his/her privacy by independently perturbing data to an extent that matches his/her own privacy preference. Since a user is the one who understands his/her privacy requirement best and different users may have different privacy preferences, it is natural to advocate a *personalized* privacy notion in the local setting. To this end, we propose a novel privacy model called *personalized local differential privacy* (PLDP), which is a generalized version of LDP.

In PLDP, to formulate his/her privacy preference, a user specifies two privacy parameters. The first is the minimum region τ he/she feels comfortable to disclose. We refer to this region as the user's *safe region*. For example, by setting his safe region to New York, Joe states that he does not mind others learning that he is in New York, but not any more fine-grained region in New York. The specification of a safe region could be done by using a *public* spatial *taxonomy*, for example, the administrative divisions of a country as shown in Figure 2. The taxonomy can be customized for different applications provided that it is constructed in a way independent of any user's location (otherwise it may leak users' true locations). A user specifies a node in the taxonomy that covers his/her true location as the safe region and expects that an adversary cannot distinguish the true location from any other location in the safe region. The user can then use the second parameter ϵ to limit an adversary's capability of distinguishing any two locations within the safe region. The role of the parameter ϵ is analogous to its counterpart in LDP. Since a user's privacy preference is collectively determined by the pair of his/her safe region τ and an adversary's ability ϵ of distinguishing any two locations in τ, we call this pair the user's *privacy specification*. Now we are ready to formally define our PLDP model.

Definition 3.2 (Personalized Local Differential Privacy): Given the personalized privacy specification (τ, ϵ) of a user u, a randomized algorithm $\mathcal{A}$ satisfies (τ, ϵ)-personalized local differential privacy (or (τ, ε)-*PLDP*) for u, if for any two locations $l, l' \in \tau$ and any $O \subseteq Range(\mathcal{A})$,

$$\Pr[\mathcal{A}(l) \in O] \leq \exp(\epsilon) \cdot \Pr[\mathcal{A}(l') \in O],$$

where the probability space is over the coin flips of $\mathcal{A}$.

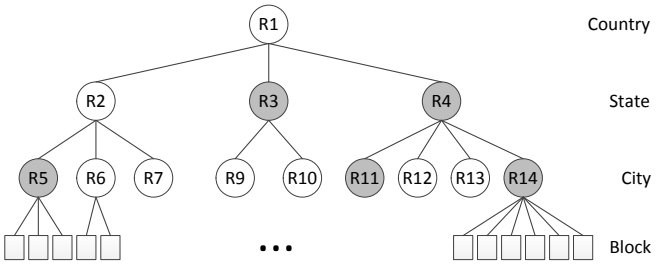


Fig. 2. A sample spatial taxonomy structure

It is easy to observe that PLDP is a generalized version of LDP. If a user sets his/her safe region to the location universe $\mathcal{L}$, PLDP becomes the standard LDP. Otherwise, PLDP represents a meaningful relaxation of LDP. We argue that the introduction of the concept of safe region does not necessarily degrade the extent of privacy protection because in reality an adversary could learn the rough area of a victim with high certainty without accessing the published information. For example, an adversary almost ascertains that the victim would be in the city for the day by seeing his/her physical presence in the workplace. In this case, setting $\tau = \mathcal{L}$ does not really help. On the other hand, personalized safe regions play a key role in improving the accuracy of aggregated results. In Section V, we will show that our PLDP model indeed achieves a much better trade-off between privacy and utility than LDP.

It is worth noting that while we introduce (τ, ϵ)-PLDP in the context of spatial data, it can be readily extended to another data domain where a user's privacy can be meaningfully defined via a data-independent taxonomy structure. One such example is relational data in which each categorical attribute is accompanied by a taxonomy [23].

C. Threat Model

In general, the objective of an adversary is to learn a user's true location. An adversary could be either the untrusted server, a participating user or an outside attacker. We consider an adversary that may have arbitrary background knowledge and that may collude with other adversaries. Our goal is to enforce personalized local differential privacy for each user so that his/her location is not learnable by any adversary. In particular, the untrusted server could behave dishonestly. This does not compromise a user's privacy in our solution, but will degrade the final utility. Therefore, it is of the server's interest to correctly follow the protocol. As such, our solution provides very strong protection of a user's privacy. Data pollution attacks by malicious users (e.g., malicious users could fake their locations to bias the distribution obtained by the server) are possible, but are beyond the scope of this paper.

D. Problem Definition

In this paper, we consider the following problem: *Given a set of n users $\mathcal{U}$, each having a private location $l_i \in \mathcal{L}$ and a privacy specification (τ_i, ϵ_i), the untrusted server would*

Algorithm 1 Personalized count estimation protocol PCEP

Input: Users' locations $\{l_i \in \tau \subseteq \mathcal{L} : 1 \leq i \leq n\}$
Input: Users' privacy specifications $\{(\tau, \epsilon_i) : 0 \leq i \leq n\}$
Input: Confidence parameter $0 < \beta < 1$
Output: User counts $\mathbf{f}$ at different locations

1: Server calculates $\delta = \sqrt{\frac{\ln(2|\tau|/\beta)}{n}}$;
2: Server calculates $m = \frac{\ln(|\tau|+1)\ln(2/\beta)}{\delta^2}$;
3: Server generates a random matrix $\Phi \in \{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^{m \times |\tau|}$;
4: Server initializes $\mathbf{z}$ and $\mathbf{f}$;
5: **for** each user u_i **do**
6: Server randomly generates j from $\{1, \cdots, m\}$;
7: Server sends j-th row $\Phi_{j\cdot}$ of Φ to u_i;
8: u_i returns $z_i = \mathtt{LR}(\Phi_{j\cdot}, l_i, \epsilon_i)$ to server;
9: Server adds z_i to j-th bit of $\mathbf{z}$;
10: **end for**
11: **for** each location $l_k \in \tau$ **do**
12: Server sets k-th element of $\mathbf{f}$ to $\langle \Phi \mathbf{e}_{l_k}, \mathbf{z} \rangle$;
13: **end for**
14: **return** $\mathbf{f}$;

like to accurately learn the users' distribution over the spatial domain $\mathcal{L}$ while satisfying (τ_i, ϵ_i)-PLDP for each user $u_i \in \mathcal{U}$.

For the real-world applications mentioned before, it is of critical importance to bound the error of the estimated user count at any single location. Therefore, the utility of a solution is measured by the *maximum absolute error* (MAE) of all locations in the universe, $\max_{l \in \mathcal{L}} |\widetilde{s}_l - s_l|$, where $\widetilde{s}_l$ and s_l are the true and estimated counts at l, respectively. While our theoretical analysis is based on MAE, we experimentally show that our solution is accurate for several other popular data analysis tasks, such as KL divergence and range queries, in Section V.

IV. Our Sanitization Framework

In this section, we present our private spatial data aggregation (PSDA) framework, which is composed of two major components, an efficient personalized count estimation protocol PCEP that is a basic building block for achieving PLDP and a user group clustering algorithm that guides how to apply PCEP to different users in order to maximize data utility by taking full advantage of users' personalized privacy requirements. Finally, we show how to integrate these two components together to construct the unified PSDA framework.

A. Efficient Personalized Count Estimation Protocol

The well-established *Laplace mechanism* [7] and *exponential mechanism* [16] are no longer suitable to the local setting in which a user may have only a single element to release. In this section, we design an efficient personalized count estimation protocol PCEP that accurately estimates the number of users at each location in a given region while satisfying PLDP. Our protocol is built upon the private frequency oracle in [3], and differs in two fundamental aspects. First, we accommodate the *personalized* privacy requirements and accordingly establish its utility guarantee. Second, we significantly improve the efficiency and communication cost for practical use.

The general idea of PCEP is to apply dimensionality reduction techniques (e.g., the Johnson-Lindenstrauss transform) to project the input locations into a space of lower dimension and collect the users' locations in this new space by "diluting" the presence at a location into a probability distribution. Algorithm 1 provides the details of PCEP. It takes as input n users' locations $\{l_i \in \tau \subseteq \mathcal{L} : 1 \leq i \leq n\}$, their personalized privacy specifications $\{(\tau, \epsilon_i) : 1 \leq i \leq n\}$ and a confidence parameter β that affects the accuracy level, and returns the estimated user counts for all locations in τ. For now, we assume that all input users are of the same safe region τ. In the next sections, we will show how to handle users with different safe regions. At the beginning, the server calculates m, the dimension of the new space. Essentially, m could be any value that is large enough to make the Johnson-Lindenstrauss lemma hold.

Theorem 4.1 (Johnson-Lindenstrauss Lemma): Given $0 < \delta < 1$ and a set $\mathcal{V}$ of t points in $\mathbb{R}^d$, there exists a linear map $\Phi : \mathbb{R}^d \to \mathbb{R}^m$ for $m = O\left(\frac{\ln t}{\delta^2}\right)$ and all $\mathbf{u}, \mathbf{v} \in \mathcal{V}$ such that,

$$(1-\delta)\|\mathbf{u}-\mathbf{v}\|_2^2 \leq \|\Phi\mathbf{u} - \Phi\mathbf{v}\|_2^2 \leq (1+\delta)\|\mathbf{u}-\mathbf{v}\|_2^2.$$

We expose the intermediate variable δ on purpose of utility analysis (Line 1). Dimensionality reduction is conducted by first building a random $m \times |\tau|$ matrix Φ with entries uniformly randomly drawn from $\{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}$ (Line 3), where $|\tau|$ is the number of locations in τ. With very high probability, each column $\Phi_{\cdot i}$ of Φ represents a unique *encoding* of a location $l \in \tau$. In Line 4, the server initializes two variables, $\mathbf{z}$ that contains intermediate information used to estimate the user distribution and $\mathbf{f}$ that contains the counts to return, into size-m vectors of 0's. For each user u_i participating in the protocol, the server generates an index j uniformly at random from $\{1, \cdots, m\}$ and sends the j-th row $\Phi_{j\cdot}$ of Φ to u_i, which u_i perturbs according to his/her true location by a *local randomizer* (Lines 6-8).

Local Randomizer. The local randomizer [3] is a basic tool for achieving (personalized) local differential privacy. We describe its use in our setting in Algorithm 2. It first generates the standard basis vector $\mathbf{e}_{l_i}$ that has a 1 in the i-th position and 0's elsewhere. Using $\mathbf{e}_{l_i}$, it selects the bit $x_{l_i} = \mathbf{x}^\top \mathbf{e}_{l_i}$ that is a random bit from the encoding of l_i in Φ (i.e., the i-th column $\Phi_{\cdot i}$ of Φ)[1]. Then it randomizes x_{l_i} into a new bit z_i in a way that satisfies (τ, ϵ_i)-PLDP (Line 3 of Algorithm 2), and returns z_i to the server.

Next, we give the privacy and utility analysis of the local randomizer LR. Its privacy guarantee only depends on Line 3 of Algorithm 2, and we have the following theorem.

Theorem 4.2: For any user u_i with privacy specification (τ, ϵ_i) and any $\mathbf{x} \in \{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^{|\tau|}$, LR is (τ, ϵ_i)-PLDP for u_i.

Proof: By definition, we would like to prove that, for any $\mathbf{x}$, any location pairs $l_i, l_i' \in \tau$ and any z_i, $\frac{\Pr[\mathtt{LR}(\mathbf{x}, l_i, \epsilon_i) = z_i]}{\Pr[\mathtt{LR}(\mathbf{x}, l_i', \epsilon_i) = z_i]} \leq e^{\epsilon_i}$. Note that $x_{l_i}, x_{l_i'} \in \{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}$ and $z_i \in \{-c_{\epsilon_i}\sqrt{m}, c_{\epsilon_i}\sqrt{m}\}$. If $x_{l_i'}$ of l_i' equals x_{l_i}, by the construction of Line 3 of Algorithm 2, it is easy to see that the probabilities of generating the same z_i from x_{l_i} and $x_{l_i'}$ are identical, that is $\frac{\Pr[\mathtt{LR}(\mathbf{x}, l_i, \epsilon_i) = z_i]}{\Pr[\mathtt{LR}(\mathbf{x}, l_i', \epsilon_i) = z_i]} = 1$. If $x_{l_i'}$

[1] x_{l_i} can be directly selected without generating $\mathbf{e}_{l_i}$, but we deliberately introduce the notion of $\mathbf{e}_{l_i}$ for ease of theoretical analysis.

Algorithm 2 Local randomizer LR

Input: d-bit string $\mathbf{x} \in \{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^d$
Input: User u_i's location l_i
Input: User u_i's privacy parameter ϵ_i
Output: Sanitized bit z_i
1: Generate the standard basis vector $\mathbf{e}_{l_i} \in \{0,1\}^d$;
2: $x_{l_i} = \mathbf{x}^\top \mathbf{e}_{l_i}$;
3: Randomize x_{l_i} as:

$$z_i = \begin{cases} c_{\epsilon_i} m x_{l_i} \text{ with probability } \dfrac{e^{\epsilon_i}}{e^{\epsilon_i}+1} \\ -c_{\epsilon_i} m x_{l_i} \text{ with probability } \dfrac{1}{e^{\epsilon_i}+1} \end{cases}$$

where $c_{\epsilon_i} = \frac{e^{\epsilon_i}+1}{e^{\epsilon_i}-1}$;
4: **return** z_i;

is different from x_{l_i}, we consider all four possible cases. *Case 1:* if $z_i = -c_{\epsilon_i}\sqrt{m}$, $x_{l_i} = -\frac{1}{\sqrt{m}}$ and $x_{l_i'} = \frac{1}{\sqrt{m}}$, $\frac{\Pr[\texttt{LR}(\mathbf{x},l_i,\epsilon_i)=z_i]}{\Pr[\texttt{LR}(\mathbf{x},l_i',\epsilon_i)=z_i]} = \frac{\frac{e^{\epsilon_i}}{e^{\epsilon_i}+1}}{\frac{1}{e^{\epsilon_i}+1}} = e^{\epsilon_i}$; *Case 2:* if $z_i = -c_{\epsilon_i}\sqrt{m}$, $x_{l_i} = \frac{1}{\sqrt{m}}$ and $x_{l_i'} = -\frac{1}{\sqrt{m}}$, $\frac{\Pr[\texttt{LR}(\mathbf{x},l_i,\epsilon_i)=z_i]}{\Pr[\texttt{LR}(\mathbf{x},l_i',\epsilon_i)=z_i]} = \frac{\frac{1}{e^{\epsilon_i}+1}}{\frac{e^{\epsilon_i}}{e^{\epsilon_i}+1}} = e^{-\epsilon_i}$; *Case 3:* if $z_i = c_{\epsilon_i}\sqrt{m}$, $x_{l_i} = -\frac{1}{\sqrt{m}}$ and $x_{l_i'} = \frac{1}{\sqrt{m}}$, $\frac{\Pr[\texttt{LR}(\mathbf{x},l_i,\epsilon_i)=z_i]}{\Pr[\texttt{LR}(\mathbf{x},l_i',\epsilon_i)=z_i]} = e^{-\epsilon_i}$; *Case 4:* if $z_i = c_{\epsilon_i}\sqrt{m}$, $x_{l_i} = \frac{1}{\sqrt{m}}$ and $x_{l_i'} = -\frac{1}{\sqrt{m}}$, $\frac{\Pr[\texttt{LR}(\mathbf{x},l_i,\epsilon_i)=z_i]}{\Pr[\texttt{LR}(\mathbf{x},l_i',\epsilon_i)=z_i]} = e^{\epsilon_i}$. Combining these cases, we have $\frac{\Pr[\texttt{LR}(\mathbf{x},l_i,\epsilon_i)=z_i]}{\Pr[\texttt{LR}(\mathbf{x},l_i',\epsilon_i)=z_i]} \le \max\{e^{-\epsilon_i}, e^{\epsilon_i}, 1\} = e^{\epsilon_i}$. Therefore, LR is (τ,ϵ_i)-PLDP for u_i. ∎

The utility of the local randomizer needs to be analyzed by taking into consideration the randomness of j (Line 6 of Algorithm 1), which determines its input $\Phi_{j\cdot}$. In essence, for a user u_i's true location l_i, LR perturbs the j-th bit x_{l_i} of l_i's encoding $\Phi_{\cdot i}$ into z_i. Let $\mathbf{z}_i = (0,\cdots,0,z_i,0,\cdots,0)$, where z_i is the j-th bit of $\mathbf{z}_i$ and $|\mathbf{z}_i| = m$. We show that $\mathbf{z}_i$ is an unbiased estimator of l_i's encoding $\Phi_{\cdot i}$.

Theorem 4.3: For any location encoding $\Phi_{\cdot i}$, $\mathbb{E}[\mathbf{z}_i] = \Phi_{\cdot i}$.

Proof: Due to the randomness of j, a bit x_{l_i} in $\Phi_{\cdot i}$ is selected with probability $\frac{1}{m}$. If x_{l_i} is chosen, with probability $\frac{1}{m}\cdot\frac{e^{\epsilon_i}}{e^{\epsilon_i}+1}$, $z_i = c_{\epsilon_i} m x_{l_i}$, and with probability $\frac{1}{m}\cdot\frac{1}{e^{\epsilon_i}+1}$, $z_i = -c_{\epsilon_i} m x_{l_i}$; otherwise $z_i = 0$. Thus we have:

$$\begin{aligned}\mathbb{E}[z_i] &= \frac{e^{\epsilon_i}}{m(e^{\epsilon_i}+1)}\cdot c_{\epsilon_i} m x_{l_i} + \frac{1}{m(e^{\epsilon_i}+1)}\cdot(-c_{\epsilon_i}) m x_{l_i} \\ &= \frac{e^{\epsilon_i}}{e^{\epsilon_i}+1}\cdot\frac{e^{\epsilon_i}+1}{e^{\epsilon_i}-1}x_{l_i} - \frac{1}{e^{\epsilon_i}+1}\cdot\frac{e^{\epsilon_i}+1}{e^{\epsilon_i}-1}x_{l_i} \\ &= x_{l_i}\end{aligned}$$

This completes the proof. ∎

Theorem 4.3 explains why the untrusted server can learn useful information regarding a user's true location from a single perturbed bit. The server aggregates all users' $\mathbf{z}_i$ into $\mathbf{z}$ and finally estimates the user count of each location $l_k \in \tau$ by $\langle \Phi \mathbf{e}_{l_k}, \mathbf{z}\rangle$, where $\mathbf{e}_{l_k}$ is the standard basis vector of l_k.

Theoretical Analysis. Now we are ready to theoretically analyze the privacy, utility and complexity of PCEP. The privacy guarantee is given in the following theorem.

Theorem 4.4: For each participating user u_i with privacy specification (τ,ϵ_i), PCEP provides (τ,ϵ_i)-PLDP.

The proof directly follows the privacy guarantee of the local randomizer (i.e., Theorem 4.2) because in the entire protocol each user discloses only a single bit perturbed by LR.

Recall that our utility goal is to bound the maximum absolute error (MAE) of all locations in τ. Let s_l and $\widetilde{s}_l$ be the true and estimated user counts at $l \in \tau$, respectively. We have the following utility guarantee.

Theorem 4.5: For any set of n participating users $\mathcal{U} = \{u_1,\cdots,u_n\}$ with privacy specification (τ,ϵ_i), with probability at least $1-\beta$, the maximum absolute error of PCEP is

$$\max_{l\in\tau}|\widetilde{s}_l - s_l| \le \sqrt{2\sum_{i=1}^{n} c_{\epsilon_i}^2 \cdot \ln\left(\frac{4|\tau|}{\beta}\right)} + \sqrt{n\ln\left(\frac{2|\tau|}{\beta}\right)},$$

where $c_{\epsilon_i} = \frac{e^{\epsilon_i}+1}{e^{\epsilon_i}-1}$.

Proof: For any location $l \in \tau$, we have $s_l = \langle \sum_{i=1}^n \mathbf{e}_{l_i}, \mathbf{e}_l\rangle$ and $\widetilde{s}_l = \langle \Phi\mathbf{e}_l, \mathbf{z}\rangle$. Thus the MAE of PCEP is

$$\max_{l\in\tau}|\widetilde{s}_l - s_l| = \max_{l\in\tau}\left|\langle \Phi\mathbf{e}_l, \mathbf{z}\rangle - \left\langle \sum_{i=1}^n \mathbf{e}_{l_i}, \mathbf{e}_l\right\rangle\right|.$$

By Theorem 4.3, $\mathbb{E}[\mathbf{z}] = \sum_{i=1}^n \mathbb{E}[\mathbf{z}_i] = \sum_{i=1}^n \Phi_{\cdot i} = \Phi\sum_{i=1}^n \mathbf{e}_{l_i}$. By the linearity of inner product and the triangle inequality, we can rewrite the MAE as

$$\begin{aligned}&\max_{l\in\tau}|\widetilde{s}_l - s_l| \\ &= \max_{l\in\tau}\left|\langle \Phi\mathbf{e}_l, \mathbf{z} - \mathbb{E}[\mathbf{z}]\rangle + \left\langle \Phi\mathbf{e}_l, \Phi\sum_{i=1}^n \mathbf{e}_{l_i}\right\rangle - \left\langle \sum_{i=1}^n \mathbf{e}_{l_i}, \mathbf{e}_l\right\rangle\right| \\ &\le \max_{l\in\tau}|\langle \Phi\mathbf{e}_l, \mathbf{z} - \mathbb{E}[\mathbf{z}]\rangle| + \\ &\quad \max_{l\in\tau}\left|\left\langle \Phi\mathbf{e}_l, \Phi\sum_{i=1}^n \mathbf{e}_{l_i}\right\rangle - \left\langle \sum_{i=1}^n \mathbf{e}_{l_i}, \mathbf{e}_l\right\rangle\right| \\ &= \max_{l\in\tau}\left|\sum_{i=1}^n \langle \mathbf{z}_i - \mathbb{E}[\mathbf{z}_i], \Phi\mathbf{e}_l\rangle\right| + \\ &\quad \max_{l\in\tau}\left|\left\langle \Phi\sum_{i=1}^n \mathbf{e}_{l_i}, \Phi\mathbf{e}_l\right\rangle - \left\langle \sum_{i=1}^n \mathbf{e}_{l_i}, \mathbf{e}_l\right\rangle\right| \qquad (1)\end{aligned}$$

To analyze the first term of Equation 1, we rely on the following lemma.

Lemma 4.1: Let $\mathbf{x}_1,\ldots,\mathbf{x}_n \in \{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^m$ be the encodings of the users' locations. For any $\mathbf{y} \in \{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^m$, with probability at least $1-\frac{\beta}{2|\tau|}$, we have

$$\left|\sum_{i=1}^n \langle \mathbf{z}_i - \mathbf{x}_i, \mathbf{y}\rangle\right| \le \sqrt{2\sum_{i=1}^n c_{\epsilon_i}^2 \cdot \ln\left(\frac{4|\tau|}{\beta}\right)}.$$

We prove this claim as follows. Recall that $\mathbf{z}_i = (0,\ldots,0,z_i,0,\ldots,0)$, where z_i is the j-th bit of $\mathbf{z}_i$ and $z_i \in \{-c_{\epsilon_i}\sqrt{m}, c_{\epsilon_i}\sqrt{m}\}$. For any $i \in \{1,\cdots,n\}$, $\langle \mathbf{z}_i, \mathbf{y}\rangle = z_i y_j$, where y_j is the j-th bit of $\mathbf{y}$, is a set of independent random variables taking values in $\{-c_{\epsilon_i}, c_{\epsilon_i}\}$. By the general form

of Hoeffding's inequality in which each random variable is bounded by a fixed interval, we can obtain the following:

$$\begin{aligned}
&\Pr\left(\left|\sum_{i=1}^{n}\langle \mathbf{z}_i-\mathbf{x}_i,\mathbf{y}\rangle\right|\geq t\right)\\
=&\Pr\left(\left|\sum_{i=1}^{n}\langle \mathbf{z}_i,\mathbf{y}\rangle-\sum_{i=1}^{n}\langle \mathbf{x}_i,\mathbf{y}\rangle\right|\geq t\right)\\
=&\Pr\left(\left|\sum_{i=1}^{n}\langle \mathbf{z}_i,\mathbf{y}\rangle-\mathbb{E}[\sum_{i=1}^{n}\langle \mathbf{z}_i,\mathbf{y}\rangle]\right|\geq t\right)\\
\leq&2\exp\left(-\frac{2t^2}{\sum_{i=1}^{n}(2c_{\epsilon_i})^2}\right)
\end{aligned}$$

By setting $2\exp\left(-\frac{2t^2}{\sum_{i=1}^{n}(2c_{\epsilon_i})^2}\right)=\frac{\beta}{2|\tau|}$, we have $t=\sqrt{2\sum_{i=1}^{n}c_{\epsilon_i}^2\ln\left(\frac{4|\tau|}{\beta}\right)}$. This establishes the claim.

Since there are a total of $|\tau|$ locations in τ, by Lemma 4.1 and the union bound, the first item of Equation 1 is bound by $\sqrt{2\sum_{i=1}^{n}c_{\epsilon_i}^2\cdot\ln\left(\frac{4|\tau|}{\beta}\right)}$ with probability $1-\frac{\beta}{2}$.

We next analyze the second term of Equation 1, which represents the inherent error due to the Johnson-Lindenstrauss transform. For ease of presentation, let $\sum_{i=1}^{n}\mathbf{e}_{l_i}=\mathbf{u}$ and $\mathbf{e}_l=\mathbf{v}$. We first prove that

$$\max_{l\in\tau}\left|\left\langle\Phi\left(\frac{1}{n}\mathbf{u}\right),\Phi\mathbf{v}\right\rangle-\left\langle\frac{1}{n}\mathbf{u},\mathbf{v}\right\rangle\right|\leq\sqrt{\frac{\ln(2|\tau|/\beta)}{n}}.$$

From Theorem 4.1, it is easy to derive that when $m=O\left(\frac{\ln(|\tau|+1)\ln(2/\beta)}{\delta^2}\right)$ (Line 3 of Algorithm 1), with probability $1-\frac{\beta}{2}$,

$$\max_{l\in\tau}\left|\left\langle\Phi\left(\frac{1}{n}\mathbf{u}\right),\Phi\mathbf{v}\right\rangle-\left\langle\frac{1}{n}\mathbf{u},\mathbf{v}\right\rangle\right|\leq\frac{\delta}{2}\left(\left\|\frac{1}{n}\mathbf{u}\right\|_2^2+||\mathbf{v}||_2^2\right).$$

Since $\left\|\frac{1}{n}\mathbf{u}\right\|_2^2+||\mathbf{v}||_2^2\leq 2$ and, by construction (Line 1 of Algorithm 1), $\delta=\sqrt{\frac{\ln(2|\tau|/\beta)}{n}}$, we have

$$\max_{l\in\tau}\left|\left\langle\Phi\left(\frac{1}{n}\mathbf{u}\right),\Phi\mathbf{v}\right\rangle-\left\langle\frac{1}{n}\mathbf{u},\mathbf{v}\right\rangle\right|\leq\sqrt{\frac{\ln(2|\tau|/\beta)}{n}}.$$

Therefore, we can learn that, with probability $1-\frac{\beta}{2}$, the second term of Equation 1 is bounded by

$$\max_{l\in\tau}|\langle\Phi\mathbf{u},\Phi\mathbf{v}\rangle-\langle\mathbf{u},\mathbf{v}\rangle|\leq\sqrt{n\ln(2|\tau|/\beta)}.$$

Again, by the union bound, with probability $1-\beta$, the sum of the two terms is bounded by $\sqrt{2\sum_{i=1}^{n}c_{\epsilon_i}^2\ln(4|\tau|/\beta)}+\sqrt{n\ln(2|\tau|/\beta)}$. ■

Next we discuss the time complexity of PCEP on both the server side and the user side. Since the user-side computation is normally done on mobile devices, we in general advocate the *thin client* design pattern so as to minimize the user-side computational overhead. On the user side, the only computation is to perturb a random bit using the local randomizer (Algorithm 2). While, for ease of theoretical analysis, we introduce and construct the standard basis vector, it is easy to verify that in practice we can access and perturb the bit in $O(1)$ time. On the server side, the server needs to compute the random matrix Φ, which takes $O(m|\tau|)$ time. For each client, the server extracts a random row from Φ and adds the perturbed bit returned by the client to $\mathbf{z}$, which can be done in $O(|\tau|)$ time. Thus the total time complexity of the server side is $O(m|\tau|+n|\tau|)=O(n|\tau|)$, where by construction $m=O(n)$.

We finally analyze the communication cost. There are two types of messages being exchanged between the server and the users. The first type is the random rows sent to each user, which total a cost of $O(n|\tau|)$; the second type is the perturbed bits sent by the users, which total $O(n)$. Therefore, the total communication cost is $O(n|\tau|)$. Specifically, for a single user, his/her communication cost is only $O(|\tau|)$. This confirms that PCEP is a lightweight protocol suitable for deployment to mobile devices.

B. User Group Clustering

With our PCEP protocol, the next question is how to feed the users into different copies of PCEP so that we can maximize the resulting utility. Recall that our utility goal is to minimize the MAE of the estimated user counts at all locations in the given location universe $\mathcal{L}$, that is $\max_{l\in\mathcal{L}}|\widetilde{s}_l-s_l|$, where $\widetilde{s}_l$ and s_l are the true and estimated counts, respectively. This could be done by two "extreme" solutions. One is to directly feed all users and the entire location universe into the protocol, which represents the "coarsest" solution. However, this scheme is sub-optimal as it does not take full benefit of users' personalized privacy requirements (i.e., the more fine-grained safe regions specified by users).

Another extreme solution is to apply PCEP to each set of users who specify the same node in the spatial taxonomy as their safe region, representing the "finest" solution. We call such a set of users a *user group*. Figure 2 illustrates five user groups marked by the shaded nodes. For example, the user group at R_3 consists of all users who specify R_3 as their safe region. Note that *different user groups are disjoint* (i.e., a user cannot belong to multiple user groups), and therefore the estimated user count at a location l would be the sum of the estimated user counts at l from all user groups whose safe regions contain l. For example, in Figure 2, the estimated user count at a block under node R_{14} would be the sum of the estimates from the user groups at R_4 and R_{14}. We denote the user group at a node R by U_R. Unfortunately, this scheme is also sub-optimal because, as shown in the example below, it is possible to cluster some user groups to achieve a smaller error.

Example 4.1: Consider the taxonomy shown in Figure 2. Assume that there are only two user groups at R_4 and R_{14} (ignore the other user groups). Given a fixed confidence parameter $\beta=0.2$, we can estimate the number of users in each block under R_{14} in the following two ways. First, we can apply PCEP to the user groups U_{R_4} and $U_{R_{14}}$, respectively. Note that in this case we need to use a smaller confidence parameter (i.e., $\frac{\beta}{2}$) for each PCEP in order to achieve the confidence level β on their sum. Assume $|R_4|=20$, $|U_{R_4}|=60000$, $|R_{14}|=6$, $|U_{R_{14}}|=20000$, and $\epsilon_i=1.0$ for all users. By Theorem 4.5,

with probability $1-\beta$, the MAE of this scheme is 4637. An alternative approach is to merge these two user groups $U = U_{R_4} \cup U_{R_{14}}$ and apply PCEP to U (with $|U| = 80000$ and $|R_4| = 20$[2]). With probability $1-\beta$, the resulting MAE is 3327, which is much smaller than the first scheme.

This example corroborates that there exist fruitful middle grounds between the two aforementioned extremes, which necessitates the need of clustering user groups. Now we formulate the *user group clustering* problem with the goal of minimizing the MAE. Recall that, by Theorem 4.5, the error of applying PCEP to a user group U_R with safe region R is determined by four parameters: the confidence level β, the number n of users in U_R, the size $|R|$ of region R and $\sum_{i=1}^{n} c_{\epsilon_i}^2$ derived from users' personalized ϵ_i values. For ease of exposition, we call $\sum_{i=1}^{n} c_{\epsilon_i}^2$ the *privacy factor* and denote it by ς for short. The resulting MAE is then a function of these four parameters, that is $err(U_R) = err(\beta, n, |R|, \varsigma)$.

We first give the MAE of applying PCEP to a cluster C formed by merging user groups $U_1, U_2, \cdots, U_k$. Let n_i, d_i and ς_i be the number of users, the safe region size and the privacy factor of the user group U_i, respectively. The number of users and the privacy factor of C are simply the sum of those of the user groups. The safe region of C, however, needs some special treatment by considering the spatial relationships among the safe regions of the user groups. By construction, the spatial relationships of any two safe regions can be either *disjoint* or *full containment*. If a safe region is contained in another safe region in the cluster, it will be "absorbed". For example, in Example 4.1, the safe region of the cluster is R_4, not $R_4 \cup R_{14}$. To this end, we introduce a new variable o_i to indicate whether the safe region of U_i is contained in the safe region of another user group in the cluster. If yes, $o_i = 0$, otherwise $o_i = 1$. Then, the region size of the cluster is $\sum_{i=1}^{k} o_i d_i$. Therefore, given the confidence level β, with probability $1-\beta$, the MAE of applying PCEP to C is

$$err(C) = err(\beta, \sum_{i=1}^{k} n_i, \sum_{i=1}^{k} o_i d_i, \sum_{i=1}^{k} \varsigma_i).$$

Now we can formally formulate the user group clustering problem as the following optimization problem.

Definition 4.1 (User Group Clustering): Given a spatial taxonomy $\mathcal{T}$, a confidence parameter β, and a set of user groups $U_1, U_2, \cdots, U_k$, partition the k user groups into m $(1 \le m \le k)$ clusters $\mathcal{C} = \{C_1, C_2, \cdots, C_m\}$ such that

$$\max_{p \in \mathcal{T}} \sum_{C_i \in \mathcal{C}_p} err\left(\frac{\beta}{m}, \sum_{U_j \in C_i} n_j, \sum_{U_j \in C_i} o_j d_j, \sum_{U_j \in C_i} \varsigma_j\right)$$

is minimum w.r.t all possible m values, where p is a path of $\mathcal{T}$, $\mathcal{C}_p \subseteq \mathcal{C}$ is the set of clusters in p, and $C_s \cap C_t = \emptyset$ for any $s \neq t$.

In the above definition, for each cluster, its PCEP is assigned a confidence parameter $\frac{\beta}{m}$ so that we can achieve the overall confidence level β. Since each path p leads to a distinctive location in the universe, minimizing the objective function in Definition 4.1 is consistent with the utility measure (i.e., minimize the MAE) introduced in Section III-D. Unfortunately, this is an NP-hard optimization problem.

[2]This is because $R_{14} \subset R_4$.

Algorithm 3 User group clustering

Input: A set of user groups $U_1, \cdots, U_k$
Input: Confidence parameter β
Output: User clusters $\mathcal{C} = \{C_1, \cdots, C_m\}$
1: $\mathcal{C} = \{C_1, \cdots, C_k\}$ where $C_i = U_i$, for $i = 1, \cdots, k$;
2: Calculate the error $err(C_i)$ of each cluster $C_i \in \mathcal{C}$ using confidence parameter $\frac{\beta}{|\mathcal{C}|}$;
3: Calculate the error $err(p)$ of each path $p \in \mathcal{T}$ by $\sum_{C_i \in \mathcal{C}_p} err(C_i)$;
4: $l_{max} = \max_{p \in \mathcal{T}} err(p)$; //maximum error of all paths
5: **while** true **do**
6: Calculate $err(C_i)$ for each C_i using confidence level $\frac{\beta}{|\mathcal{C}|-1}$;
7: Calculate $err(p)$ for each valid path p by $\sum_{C_i \in \mathcal{C}_p} err(C_i)$;
8: **for** each valid path $p \in \mathcal{T}$ **do**
9: **for** each pair of clusters $C_s, C_t \in \mathcal{C}_p$, $s \neq t$ **do**
10: **if** the pair (C_s, C_t) were not visited **then**
11: //assume that the region of C_s contains that of C_t
12: Calculate the error $err'(p)$ of each valid path $p \in \mathcal{T}$ provided this pair would be merged:
$err'(p) = err(p)$
$err'(p) -= err(C_s) + err(C_s \cup C_t)$ if $C_s \in \mathcal{C}_p$
$err'(p) -= err(C_t)$ if $C_t \in \mathcal{C}_p$
13: $l_{s,t} = \max_{p \in \mathcal{T}} err'(p)$;
14: Mark (C_s, C_t) as visited;
15: **end if**
16: **end for**
17: **end for**
18: **if** the minimum of $l_{s,t}$ is smaller than l_{max} **then**
19: Merge the two clusters resulting in the minimum of $l_{s,t}$;
20: Set l_{max} as the minimum of $l_{s,t}$;
21: **else**
22: **break**;
23: **end if**
24: **end while**
25: **return** $\mathcal{C}$;

Theorem 4.6: The User Group Clustering problem is NP-hard.

Due to space limitation, we omit the proof here. Given the hardness of this problem, we resort to approximation algorithms. Specifically, we consider the agglomerative clustering approach. It regards each user group as a cluster in the very beginning, and iteratively merges two clusters which result in the maximum value decrease of the objective function (i.e., the maximum error reduction) defined in Definition 4.1 among all cluster pairs. This iterative process terminates when no further merging can decrease the value of the objective function. However, this general idea faces several unique challenges. First, our problem structure does not satisfy any natural monotonicity: merging two clusters may decrease the errors of some paths while increasing the errors of other paths. Second, we cannot only focus on optimizing the path p with the maximum error as merging clusters on other paths might also decrease the error of p. As such, we have to design a more sophisticated algorithm tailored to our problem, whose details are given in Algorithm 3.

In Lines 1-4 of Algorithm 3, we calculate the initial maximum absolute error of all paths. After that, we iteratively

evaluate all cluster pairs to determine which pair should be merged. This procedure, especially in face of the aforementioned challenges, could be computationally demanding. Therefore, we incorporate a unique problem-specific heuristic into the algorithm design: a cluster may only be merged with another cluster in the same path[3]. This heuristic allows us to substantially improve the efficiency and explains Line 9 of Algorithm 3. Note that a path not associated with any cluster does not need to be considered. We call a path associated with at least one cluster a *valid* path. As a result, in each iteration, we only need to consider all cluster pairs derived from each valid path p's $\mathcal{C}_p$. We further observe that some cluster pairs may repeatedly appear in multiple valid paths, and thus we avoid repeated computations by marking an already evaluated pair as visited (Lines 10 and 14 of Algorithm 3).

To address the aforementioned challenges, for each candidate cluster pair, we need to explicitly calculate the errors of all resulting clusters and then the errors of all valid paths provided the merging happens. This is a computationally expensive process. To this end, we devise an efficient scheme to compute the errors of all valid paths for each candidate pair (Line 12 of Algorithm 3). Specifically, let the number of clusters be w at a certain iteration. We first pre-compute the error $err(C_i)$ of each cluster C_i $(i = 1, \cdots, w)$ as $err\left(\frac{\beta}{w-1}, \sum_{U_j \in C_i} n_j, \sum_{U_j \in C_i} o_j d_j, \sum_{U_j \in C_i} \varsigma_j\right)$, and then calculate the error $err(p)$ of each valid path $p \in \mathcal{T}$ as $\sum_{C_i \in \mathcal{C}_p} err(C_i)$. Now consider a candidate pair of clusters C_s and C_t derived from $\mathcal{C}_p$. W.l.o.g., assume that the region of C_s contains that of C_t. If C_s and C_t are merged, we calculate the errors of all valid paths based on their pre-computed errors as follows: for the paths containing C_s, their errors $err'(p)$ after the merging can be calculated by $err(p) - err(C_s) + err(C_s \cup C_t)$; for the paths containing C_t, their errors $err'(p)$ after the merging will be further updated as $err'(p) - err(C_t)$; for the paths containing neither C_s nor C_t, their errors after the merging remain as $err(p)$.

With all these efforts, the efficiency of the proposed algorithm can be substantially improved. Indeed, our experiments on four real benchmark datasets (see Section V for their descriptions) indicate that Algorithm 3 reduces the MAE, on average, by 14.65% with only a small runtime.

Complexity. The time complexity of Algorithm 3 is in $O(lk^2h^2)$, where l is the total number of iterations, k is the number of user groups, and h is the height of the spatial taxonomy. This is because in each iteration for each valid path we consider at most $\frac{(h-1)(h-2)}{2} = O(h^2)$ candidate pairs and the number of valid paths is bounded by the number k of user groups. In practice, the total number of candidate pairs to be considered at each iteration is normally much smaller than $O(kh^2)$ because many pairs exist in multiple paths and thus are evaluated only once in Algorithm 3. Moreover, as the number of iterations goes up, the total number of candidate pairs per iteration decreases due to the merging. As such, our algorithm is efficient in practical settings. We experimentally confirm the efficiency of Algorithm 3 over different datasets in Section V-B.

[3]By Theorem 4.5, we can verify that, with high probability, merging two clusters in different paths increases the MAE.

Algorithm 4 Our PSDA framework

Input: A set of n users with privacy specifications $\{(\tau_i, \epsilon_i) : 1 \leq i \leq n\}$ and locations $\{l_i \in \tau_i : 1 \leq i \leq n\}$
Input: Confidence parameter β
Output: Sanitized counts for all locations $\{\widetilde{s}_l : l \in \mathcal{L}\}$
1: **for** each user u_i **do**
2: Send (τ_i, ϵ_i) to server;
3: **end for**
4: Server divides users into different groups;
5: Server partitions user groups into clusters $\mathcal{C}$;
6: **for** each cluster $C_i \in \mathcal{C}$ **do**
7: Server applies PCEP to C_i with confidence level $\frac{\beta}{|\mathcal{C}|}$;
8: **end for**
9: Server calculates the sanitized counts for all locations;
10: Server enforces consistency on the sanitized counts;
11: **return** $\{\widetilde{s}_l : l \in \mathcal{L}\}$;

C. Putting Things Together

Finally we show how to integrate the previous techniques into the unified private spatial data aggregation (PSDA) framework in Algorithm 4. Each user sends his/her personalized privacy specification to the untrusted server. After receiving the specifications from all users, the server divides them into different user groups according to their safe regions: all users with the same safe region form a user group. After that, the server partitions the user groups into different clusters, as described in Algorithm 3. For each cluster, the server applies the PCEP protocol to estimate the user distribution within the cluster. The confidence parameter $\frac{\beta}{|\mathcal{C}|}$ is used for each copy of PCEP so as to establish the overall confidence level β. By combining the estimates from all clusters, the server obtains the sanitized counts for all locations in the universe.

Due to the error of PCEP, there may exist some inconsistency in the estimated counts. To this end, we develop a post-processing technique to enforce the consistency constraints (Line 10 of Algorithm 4) as follows. For each node v in the taxonomy $\mathcal{T}$, we can calculate the lower and upper bounds of the true user count in its region. Let $dt(v)$ be the set of user groups associated with v's descendents (including v's user group if applicable) and $at(v)$ be the set of user groups associated with v's *proper* ancestors. The lower bound of v's true user count is $lb(v) = \sum_{U_i \in dt(v)} n_i$ and the upper bound $ub(v) = \sum_{U_i \in dt(v)} n_i + \sum_{U_i \in at(v)} n_i$, where n_i is the number of users in U_i. Our goal is to ensure that the estimated count at each node v falls in the range $[lb(v), ub(v)]$ in order to reduce errors. The first step is to calculate the estimated user count of each node, which can be done in a single bottom-up level order traversal by setting an internal node's count to the sum of its children's counts. Then we adjust the nodes' estimated counts in a top-down manner. If a node v's estimated count is $< lb(v)$, its count is set to $lb(v)$; if its count is $> ub(v)$, it is set to $ub(v)$[4]. Then the difference is uniformly distributed to its siblings that do not require an adjustment so that the sum of all children's counts equals their parent's count.

Privacy Analysis. Due to space limitation, we focus on analyzing the privacy guarantee of our PSDA framework. Its

[4]The estimated count at the root equals the total number of users.

TABLE I. DATASET STATISTICS

Dataset	Number of users	Coordinate range	Smallest granularity
road	1,634,165	$[-124.8, -103.0] \times [31.3, 49.0]$	1.0×1.0
checkin	1,000,000	$[-176.3, 177.46] \times [-48.2, 90.0]$	2.0×2.0
landmark	870,051	$[-124.4, -67.0] \times [24.6, 49.0]$	1.0×1.0
storage	8,938	$[-123.2, -70.3] \times [25.7, 48.8]$	1.0×1.0

utility and complexity can be easily derived from those given in Section IV-A and Section IV-B.

Theorem 4.7: For each participating user u_i with privacy specification (τ_i, ϵ_i), our PSDA *framework provides (τ_i, ϵ_i)-PLDP.*

Proof: In our framework, each user u_i participates in exactly one copy of the PCEP protocol with other users in the same cluster C. Note that the clustering procedure does not rely on a user's true location, but only his/her specified safe region, and thus does not incur any additional privacy threat. By design, the user's safe region τ_i is fully contained in C's region R. By Theorem 4.4, it is guaranteed that for u_i any two locations in R are indistinguishable to an extent bounded by $\exp(\epsilon_i)$. Since $\tau_i \subseteq R$, any two locations in τ_i is also guaranteed to be indistinguishable to an extent bounded by $\exp(\epsilon_i)$. In addition, the post-processing procedure only relies on users' public privacy specifications, but not their true locations, and therefore does not introduce any additional cost. This establishes the proof. ■

V. EXPERIMENTAL EVALUATION

In this section, we experimentally evaluate the performance of our PSDA framework and, in particular, validate our personalized local differential privacy model over four real benchmark datasets used in [20].

The first dataset *road* we use was drawn from the 2006 TIGER/Line dataset and represents the GPS coordinates of road intersections in Washington and New Mexico. The *checkin* dataset contains users' check-in locations from the Gowalla website over the period of Feb. 2009 - Oct. 2010. *landmark* consists of the landmarks in the 48 continental states in the U.S., and *storage* includes the locations of storage facilities in the U.S. We summarize the characteristics of the datasets in Table I, where the smallest granularity is the size of the leaf nodes in a spatial taxonomy. For each dataset, we construct its spatial taxonomy by using a fixed fanout of 4. We also tested with a wide range of other fanouts and observed similar results.

We construct the users' personalized privacy specifications as follows. Since a user's safe region is specified as a node in the spatial taxonomy, which is similar to the privacy requirement in [23], we follow a similar but more rigorous setting. The users are randomly divided into 4 groups with $p_1\%$, $p_2\%$, $p_3\%$ and $p_4\%$ of the total users, respectively. In the first group, the users specify their true locations as their safe regions (i.e., they are willing to share their true locations probably because the locations are not sensitive to them); in the second group, the users specify their true locations' parents as their safe regions; in the third group, the users specify their true locations' grandparents as their safe regions; in the last group, the users specify their true locations' great-grandparents as their safe regions. To study the effect of different safe region specifications, we create two sets of safe region specifications, $S_1 = \{p_1 = 10, p_2 = 20, p_3 = 40, p_4 = 30\}$ and $S_2 = \{p_1 = 30, p_2 = 40, p_3 = 20, p_4 = 10\}$, where S_1 represents a more stringent privacy requirement than S_2. Similarly, we create two sets of ϵ value specifications: for each user, his/her ϵ_i value is uniformly randomly drawn from $E_1 = \{0.25, 0.5, 0.75\}$ or $E_2 = \{0.75, 1.0, 1.25\}$, where E_1 represents a more stringent privacy requirement. As such, we have a total of four sets of privacy specifications, namely (S_1, E_1), (S_1, E_2), (S_2, E_1) and (S_2, E_2). The confidence parameter β is set to 0.1.

A. Benchmark Methods

To our best knowledge, there does not exist any prior technique that can directly address our problem. We analyze the two state-of-the-art techniques [5], [20] in attempt to adapt them to the local setting. In [5], both *data-dependent* and *data-independent* solutions are provided. Unfortunately, the data-dependent ones are not applicable as they rely on the exponential mechanism, which is not available in the local setting. Therefore, we only design a variant of the data-independent kd-tree-based technique in which we replace the Laplace mechanism with our PCEP protocol to learn aggregated counts. We refer to it as kdTree. For a fair comparison, we also make use of the personalized privacy specifications in kdTree to improve accuracy. Similarly, we could adapt the grid-based approaches in [20] by using our PCEP protocol. However, their performance heavily relies on the proper selection of numbers of grids in each level. Their guidelines based on the Laplace mechanism normally give poor results for PCEP. For comparison purposes, we also include the following two schemes. The first is to feed all users into a single PCEP protocol with their safe regions being the location universe while still having personalized ϵ_i values. The main purpose of this scheme is to justify the necessity of introducing the concept of safe region and thus the validity of PLDP. We refer to it as SR. The second is to apply PCEP to each user group with $\epsilon_i = 0$. Essentially, this method randomly returns a location in the safe region. This method shares the same spirit with *spatial cloaking* [10], a privacy technique widely used for location-based services. We use this method to show that adding the notion of differential privacy into this established technique enables substantial utility improvement while still providing rigorous privacy protection. We refer to it as Cloak.

B. Experimental Results

KL Divergence. Recall that our motivation is to enable an untrusted server (e.g., Waze) to accurately learn users' distribution over a spatial domain. This objective can be precisely measured by KL divergence. We give the KL divergences of different schemes over the four datasets in Table II. All experimental results reported in this paper are the average of 10 runs. It can be observed that PSDA substantially outperforms all competitors. We highlight the smallest KL divergences in Table II. Except *storage* whose number of users is very small, PSDA obtains reasonably small KL divergences for all other datasets under various privacy specifications. Roughly, the difference between PSDA and SR demonstrates the benefit of introducing the notion of safe region while the difference between PSDA and Cloak demonstrates the benefit of integrating differential privacy to spatial cloaking. These results together validate the importance of our personalized local differential privacy model for practical use.

TABLE II. KL DIVERGENCES OF DIFFERENT SCHEMES

(a) KL divergences under (S_1, E_1)

Dataset	PSDA	kdTree	Cloak	SR
road	**0.045**	0.116	0.475	0.564
checkin	**0.111**	0.270	0.796	2.051
landmark	**0.172**	0.504	0.641	3.593
storage	**0.877**	1.826	1.278	3.912

(b) KL divergences under (S_1, E_2)

Dataset	PSDA	kdTree	Cloak	SR
road	**0.014**	0.041	0.475	0.199
checkin	**0.042**	0.088	0.795	1.178
landmark	**0.066**	0.199	0.641	2.881
storage	**0.547**	0.978	1.272	3.768

(c) KL divergences under (S_2, E_1)

Dataset	PSDA	kdTree	Cloak	SR
road	**0.033**	0.112	0.239	0.554
checkin	**0.063**	0.303	0.367	1.978
landmark	**0.079**	0.599	0.318	3.572
storage	**0.454**	1.781	0.690	3.865

(d) KL divergences under (S_2, E_2)

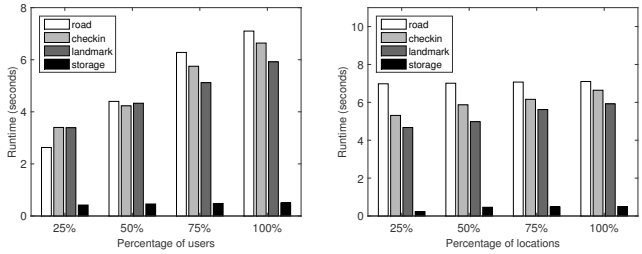
Dataset	PSDA	kdTree	Cloak	SR
road	**0.011**	0.038	0.24	0.203
checkin	**0.025**	0.084	0.368	1.161
landmark	**0.035**	0.195	0.317	2.890
storage	**0.271**	0.926	0.691	3.770

Range Queries. In the second set of experiments, we evaluate the utility of sanitized data for range queries. Following the setting in [5], [20], we use *relative error* as the accuracy measure. For a query q, let $A(q)$ be its true answer and $\widetilde{A}(q)$ be its answer on the sanitized data. The relative error is defined as $RE(q) = \frac{|A(q)-\widetilde{A}(q)|}{\max\{A(q),s\}}$, where s is the *sanity bound* that mitigates the effect of queries with extremely small selectivities [20]. We set $s = 0.001 \times |D|$ for *road*, *checkin* and *landmark*, and $s = 0.01 \times |D|$ for *storage* to compensate its overly small size, where $|D|$ is the number of users in a dataset. Similarly, we consider six different query sizes, with q_1 being the smallest. Each q_{i+1} increases the size of q_i by 1.5 times. The q_1 sizes of *road*, *checkin*, *landmark* and *storage* are 1.0×1.0, 4.0×4.0, 2.0×2.0 and 2.0×2.0, respectively. For each query size, we randomly generate 600 queries and report the average relative error.

We give the experimental results in Figures 3-6. We can make several interesting observations. First, as expected, the relative error decreases with the relaxation of the privacy requirements. This explains why we obtain the best accuracy under (S_2, E_2). In addition, we observe that the accuracy of kdTree is more sensitive to ϵ values because it needs to split the total privacy parameter for building different levels of the kd-tree. Second, PSDA achieves the best accuracy in almost all cases, confirming its superiority for solving our problem. In general, the error of PSDA is reasonably small for all settings. Third, again we can observe that neither Cloak nor SR can achieve desirable accuracy. The relative error of SR is too large to fit into Figure 5(c) and Figure 6(c). This fact again justifies our design of personalized privacy requirements.

Scalability. In the last set of experiments, we study the scalability of our framework. As per our theoretical analysis, the user-side runtime is negligible, and thus we only report the server-side runtime with respect to the number of users and the size of the location universe[5]. Figure 7 presents the runtimes

[5]The height of the taxonomy and the number of user groups are all pertinent to the universe size.



(a) Runtime vs. # of users (b) Runtime vs. # of locations

Fig. 7. Runtime under different settings

of PSDA under different numbers of users and locations. The experiments were conducted on a machine with an Intel i7 2.40GHz CPU and 8GB RAM. To form the test datasets, we uniformly extract 25%, 50%, 75%, and 100% of users and locations from each dataset at random. As can be observed, PSDA is highly efficient. In the worst case, it still takes less than 8 seconds. A further breakdown of the runtime indicates that file I/O (i.e., loading input user data) accounts for a significant fraction. This is why the difference between various numbers of locations is negligible. We can further observe that in practice the runtime of PSDA roughly grows linearly with the number of users and the number of locations. This confirms that our solution is scalable for real-world deployments.

VI. CONCLUSION

Private spatial data aggregation in the local setting has emerged as a potential impetus for many crowdsourcing-based businesses. However, this problem has not been touched upon before. In this paper, we presented the first solution to this challenging problem. Motivated by the traits of the local setting, we proposed the notion of personalized local differential privacy (PLDP) that achieves a more desirable trade-off between privacy and utility for practical use. We then designed an efficient PCEP protocol as the building block of achieving PLDP. With PCEP, we developed a unified PSDA framework that maximizes the resulting utility by dispatching users to different copies of PCEP. We theoretically and experimentally demonstrate the effectiveness and efficiency of our framework. We deem that our work represents an important step towards better privacy protection in crowdsourcing-based applications.

REFERENCES

[1] G. Acs and C. Castelluccia. I have a DREAM!: differentially private smart metering. In *IH*, 2012.

[2] G. Acs and C. Castelluccia. A case study: privacy preserving release of spatio-temporal density in paris. In *SIGKDD*, 2014.

[3] R. Bassily and A. Smith. Local, private, efficient protocols for succinct histograms. In *STOC*, 2015.

[4] R. Chen, A. Reznichenko, P. Francis, and J. Gehrke. Towards statistical queries over distributed private user data. In *NSDI*, 2012.

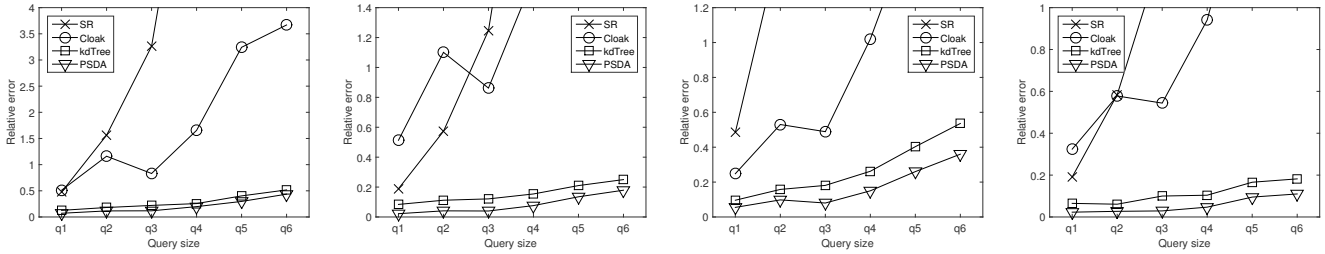
[5] G. Cormode, C. Procopiuc, D. Srivastava, E. Shen, and T. Yu. Differentially private spatial decompositions. In *ICDE*, 2012.

[6] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Local privacy and statistical minimax rates. In *FOCS*, 2013.

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

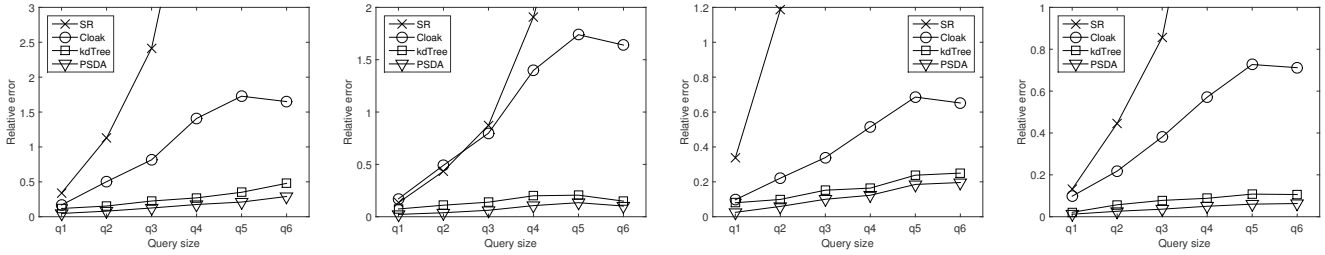
[8] U. Erlingsson, V. Pihur, and A. Korolova. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In *CCS*, 2014.

[9] B. Gedik and L. Liu. Location privacy in mobile systems: a personalized anonymization model. In *ICDCS*, 2005.



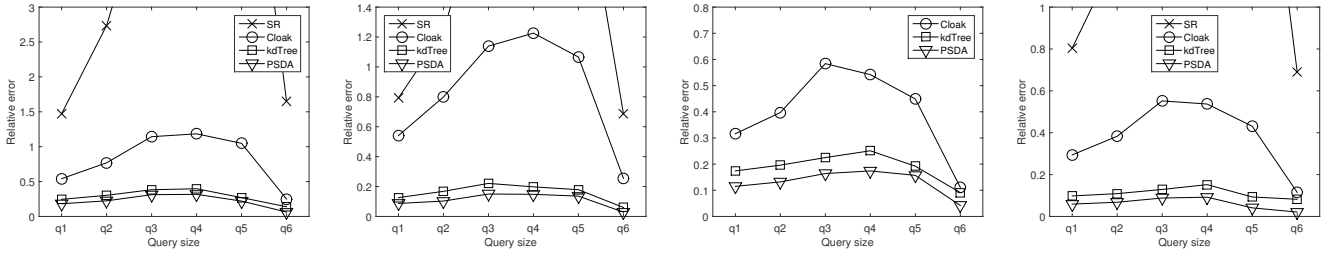
(a) (S_1, E_1) (b) (S_1, E_2) (c) (S_2, E_1) (d) (S_2, E_2)

Fig. 3. Relative errors of range queries on *road*



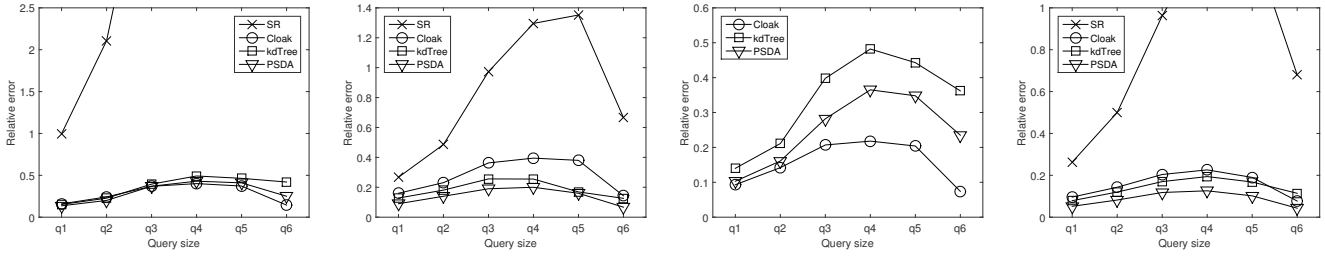
(a) (S_1, E_1) (b) (S_1, E_2) (c) (S_2, E_1) (d) (S_2, E_2)

Fig. 4. Relative errors of range queries on *checkin*



(a) (S_1, E_1) (b) (S_1, E_2) (c) (S_2, E_1) (d) (S_2, E_2)

Fig. 5. Relative errors of range queries on *landmark*



(a) (S_1, E_1) (b) (S_1, E_2) (c) (S_2, E_1) (d) (S_2, E_2)

Fig. 6. Relative errors of range queries on *storage*

[10] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *MobiSys*, 2003.

[11] M. Hardt and S. Nath. Privacy-aware personalization for mobile advertising. In *CCS*, 2012.

[12] Y. Hong, J. Vaidya, H. Lu, P. Karras, and S. Goel. Collaborative search log sanitization: Toward differential privacy and boosted utility. *TDSC*, 2015.

[13] Z. Jorgensen, T. Yu, and G. Cormode. Conservative or liberal? Personalized differential privacy. In *ICDE*, 2015.

[14] P. Kairouz, S. Oh, and P. Viswanath. Extremal mechanisms for local differential privacy. In *NIPS*, 2014.

[15] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately. In *FOCS*, 2008.

[16] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, 2007.

[17] D. J. Mir, S. Issacman, R. Caceres, M. Martonosi, and R. N. Wright. DP-WHERE: Differentially private modeling of human mobility. In *IEEE BigData*, 2013.

[18] N. Mohammed, D. Alhadidi, B. C. M. Fung, and M. Debbabi. Secure two-party differentially private data release for vertically partitioned data. *TDSC*, 11(1):59–71, 2014.

[19] M. A. Pathak, S. Rane, and B. Raj. Multiparty differential privacy via aggregation of locally trained classifiers. In *NIPS*, 2010.

[20] W. Qardaji, W. Yang, and N. Li. Differentially private grids for geospatial data. In *ICDE*, 2013.

[21] E. Shi, T.-H. H. Chan, E. Rieffel, R. Chow, and D. Song. Privacy-preserving aggregation of time-series data. In *NDSS*, 2011.

[22] H. To, G. Ghinita, and C. Shahabi. A framework for protecting worker location privacy in spatial crowdsourcing. *PVLDB*, 7(10):919–930, 2014.

[23] X. Xiao and Y. Tao. Personalized privacy preservation. In *SIGMOD*, 2006.

[24] M. Yuan, L. Chen, and P. S. Yu. Personalized privacy protection in social networks. *PVLDB*, 4(2):141–150, 2010.