Privacy Amplification by Sampling under User-level Differential Privacy

JUANRU FANG, Hong Kong University of Science and Technology, Hong Kong
KE YI, Hong Kong University of Science and Technology, Hong Kong

Random sampling is an effective tool for reducing the computational costs of query processing in large databases. It has also been used frequently for private data analysis, in particular, under differential privacy (DP). An interesting phenomenon that the literature has identified, is that sampling can amplify the privacy guarantee of a mechanism, which in turn leads to reduced noise scales that have to be injected.

All existing privacy amplification results only hold in the standard, record-level DP model. Recently, user-level differential privacy (user-DP) has gained a lot of attention as it protects all data records contributed by any particular user, thus offering stronger privacy protection. Sampling-based mechanisms under user-DP have not been explored so far, except naively running the mechanism on a sample without privacy amplification, which results in large DP noises. In fact, sampling is in even more demand under user-DP, since all state-of-the-art user-DP mechanisms have high computational costs due to the complex relationships between users and records. In this paper, we take the first step towards the study of privacy amplification by sampling under user-DP, and give the amplification results for two common user-DP sampling strategies: simple sampling and sample-and-explore. The experimental results show that these sampling-based mechanisms can be a useful tool to obtain some quick and reasonably accurate estimates on large private datasets.

CCS Concepts: • **Security and privacy** → **Database and storage security**.

Additional Key Words and Phrases: Differential privacy, Random sampling

ACM Reference Format:
Juanru Fang and Ke Yi. 2024. Privacy Amplification by Sampling under User-level Differential Privacy. *Proc. ACM Manag. Data* 2, 1 (SIGMOD), Article 34 (February 2024), 26 pages. https://doi.org/10.1145/3639289

1 INTRODUCTION

We first recall the definition of *differential privacy (DP)*:

Definition 1.1. Let $\mathcal{I}$ denote the set of all database instances and $\mathbb{P}(\mathcal{Z})$ the set of probability distributions over $\mathcal{Z}$. A mechanism $\mathcal{M} : \mathcal{I} \to \mathbb{P}(\mathcal{Z})$ maps each instance $I \in \mathcal{I}$ to the distribution $\mathcal{M}(I)$, and outputs a random variable drawn from $\mathcal{M}(I)$. A mechanism $\mathcal{M}$ is said to satisfy (ε, δ)-DP if for any *neighboring* instances $I \sim I'$ and any measurable subset $Z \subseteq \mathcal{Z}$,

$$\int_Z \mathcal{M}(I)(z)dz \le e^{\varepsilon} \cdot \int_Z \mathcal{M}(I')(z)dz + \delta.$$

In the definition above, ε is the privacy parameter, also called the privacy budget, typically set to a constant, while δ should be negligible. Especially, when $\delta = 0$, the mechanism is said to satisfy pure DP.

Authors' addresses: Juanru Fang, Hong Kong University of Science and Technology, Hong Kong, Hong Kong, jfangad@cse.ust.hk; Ke Yi, Hong Kong University of Science and Technology, Hong Kong, Hong Kong, yike@cse.ust.hk.

ACM 2836-6573/2024/2-ART34
https://doi.org/10.1145/3639289

Note that this definition is not complete until the neighboring relationship $\sim$ is specified. Indeed, it depends on what information should be protected, and different neighboring relationships lead to different DP models or policies. In the standard DP model, $I \sim I'$ if one can be obtained from the other by adding or deleting one record, so this DP model protects the privacy of individual records. In this model, there is no distinction between "users" and "records", or equivalently speaking, each user contributes at most one record. However, the relationship between users and records is more complicated in many real-world situations: A record may be jointly contributed by a group of users and a user may contribute to multiple records, and we aim to protect the privacy of individual users.

1.1 User-level Differential Privacy

In this paper, we adopt the user-DP model introduced in [14], which directly captures the relationship between users and records.

Let $\mathcal{U}$ denote the domain of all users and $\mathcal{R}$ the domain of data records. An instance $I = (U, V)$ consists of a set of users $U \subseteq \mathcal{U}$ and a mapping $V : \mathcal{U}^\ell \to 2^\mathcal{R}$ from tuples of ℓ users to sets of records, representing the data jointly contributed by these users. Specifically, for each tuple $x = (u_1, \dots, u_\ell) \in \mathcal{U}^\ell$, $V(x)$ is the set of records contributed by $u_1, \dots, u_\ell$ in this particular order. It is required that the $V(x)$'s are pairwise disjoint, and $V(x) = \emptyset$ if x contains a user not in U. We write $u \in x$ if user u appears in x. Let $R(V) = \bigcup_x V(x)$ be all the records, $n = |U|$ the number of users, and $m = |R(V)|$ the total number of records. For any user u, let $R(V, u) = \bigcup_{x \ni u} V(x)$ be the records that are (partly) contributed by user u.

We can now define the neighboring relationship between instances. For any two instances $I = (U, V)$ and $I' = (U', V')$, we say that I' is a *sub-instance* of I, denoted $I' \preceq I$, if $U' \subseteq U$ and $V'(x) \subseteq V(x)$ for all x. We call I' a *down neighbor* of I, if I' can be obtained from I by deleting all or part of the contributions of a user u^*, called the *witness*, i.e., $U \setminus \{u^*\} \subseteq U' \subseteq U$ and

$$V'(x) \subsetneq V(x) \Rightarrow u^* \in x.$$

Then, I and I' are *neighbors* if one is a down neighbor of the other, denoted as $I \sim_{u^*} I'$, where u^* is the witness. The distance between I and I', denoted by $d(I, I')$, is the length of the shortest sequence $(I_0 = I, I_1, \dots, I_d = I')$ such that $I_{i-1} \sim I_i$ for all $i = 1, \dots, d$.

For $\ell = 1$, the model is equivalent to a simple user-DP model used frequently in the machine learning community [2, 13], where each user contributes multiple records, but they do not jointly contribute any records. If we further restrict $|V(u)| \in \{0, 1\}$ for all $u \in \mathcal{U}$, then the model degenerates to the standard record-level DP. We are more interested in the $\ell \geq 2$ case, where users may interact and jointly contribute records.

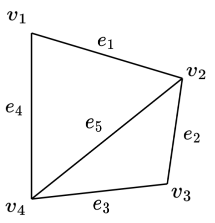
We also note that the user-DP model adopted in this paper is slightly more general than that in [14], where V maps each unordered set of ℓ users to a set of records. Here, we use ordered ℓ-tuples, which allow us to handle graphs (both directed and undirected) and relational data easily; some examples are provided below. Furthermore, for the sampling-based mechanism introduced in Section 5, the ordered model allows us to sample the users based on their position in the tuples, which improves privacy amplification and sampling error for certain problems.

1.2 Sum Estimation under User-DP

In this work, we focus on the sum estimation problem under user-DP. In this problem, each record t is associated with a weight $w(t)$, and the query returns the total weight $f(I) = \sum_{t \in R(V)} w(t)$. It has been shown [10, 14, 23] that by appropriately defining the set of users U, the mapping V, and the weights $w(\cdot)$, any select-join-aggregate (SJA) queries with foreign-key constraints [10, 23], as well as any subgraph counting problem under node/edge-DP [6, 8, 20, 22, 26, 37], can be framed as a sum estimation problem in the user-DP model defined above.

Example 1.1. Consider an SJA query on the TPC-H data that computes the total revenue from all lineitems where the ship date is within the last year and the customer and supplier are within the same nation. Suppose we wish to protect the privacy of both customers and suppliers. This query can be captured in our user-DP model as follows. We first perform the (natural) join of `Lineitem`, `Orders`, `Customer`, `Supplier`, and `Nation`. Then set $\ell = 2$; U includes all customers and suppliers; V maps each ordered (c, s) pair to the set of lineitems that customer c purchases from supplier s while meeting the two selection conditions; for each lineitem t, $w(t)$ is the revenue from this lineitem. □

This user-DP model also captures subgraph counting problems under both node-DP and edge-DP, two common DP policies for graph data, as special cases. Given a graph and a pattern that we want to count, we can construct an instance in our user-DP model as follows: Under node-DP (resp. edge-DP), set ℓ as the number of nodes (resp. edges) in the pattern; U includes all nodes (resp. edges); V maps each ordered tuple of ℓ nodes (resp. edges) to a record t iff the nodes (resp. edges) form the pattern in a specific order; for each record t, set $w(t) = 1$. Then the function $f(I)$ exactly returns the number of the patterns in the graph.



Edge, Node DP

$U : \{v_1, v_2, v_3, v_4\}$

$V : V(v_1, v_2) = \{t_1\}$ $\quad V(v_2, v_3) = \{t_2\}$

$V(v_3, v_4) = \{t_3\}$ $\quad V(v_1, v_4) = \{t_4\}$

$V(v_2, v_4) = \{t_5\}$

$w(\cdot) : w(t_i) = 1, i = 1, \dots, 5$

Triangle, Edge DP

$U : \{e_1, e_2, e_3, e_4, e_5\}$

$V : V(e_1, e_4, e_5) = \{t_1\}$

$V(e_2, e_3, e_5) = \{t_2\}$

$w(\cdot) : w(t_i) = 1, i = 1, 2$

Fig. 1. Edge counting under node-DP and triangle counting under edge-DP

Example 1.2. Figure 1 gives two examples: edge counting under node-DP and triangle counting under edge-DP. For the former, we set $\ell = 2$ and take the nodes as the users. Then we set $V(u, v) = \{1\}$ iff there is an edge between u and v and $u < v$ (assuming each user is identified by a unique ID). Note that we introduce the condition $u < v$ to avoid double counting. If the graph is directed, we just set $V(u, v) = \{1\}$ for each directed edge (u, v) without the $u < v$ restriction. For triangle counting under edge-DP, we set $\ell = 3$ and take the edges as the users. Then we set $V(e, e', e'') = \{1\}$ iff the three edges e, e', e'' form a triangle and $e < e' < e''$. □

Problems become computationally expensive under user-DP, especially when $\ell \geq 2$. First, the set of records $R(V)$ is often represented implicitly. For example, in the triangle counting problem, $R(V)$ consists of all triangles in a graph, whose size can be much larger than the size of the graph. For a relational query, $R(V)$ is all the join results, which can be large for non-key joins. All prior user-DP mechanisms compute $R(V)$ explicitly, which is the first source of computational inefficiency. Moreover, each user may contribute to an arbitrary number of records, which makes the sensitivity (see Section 3.1 for the formal definition) of the query unbounded. A widely used strategy is the truncation mechanism, i.e., for some truncation threshold C, the contributions from any user are discarded if they exceed C. When $\ell = 1$, the query after truncation will have sensitivity C, so some standard DP mechanisms, e.g. the Laplace mechanism, can be applied. However, when $\ell \geq 2$, it has been shown that this simple truncation mechanism fails to lower the sensitivity [10] due to the complex relationship between users and records. Therefore, when $\ell \geq 2$, more complicated algorithms are needed. These mechanisms either solve a larger number of linear programs [8, 10, 14] or compute a complex smooth sensitivity [22, 26], both of which incur a high computational cost.

1.3 Sampling and Privacy Amplification

Since the output of any DP mechanism $\mathcal{M}(I)$ must be randomized and approximate, there is no point insisting that $\mathcal{M}(I)$ should be computed exactly. Thus, random sampling has been widely used to reduce the computational cost of $\mathcal{M}(I)$. In fact, for many problems, e.g., approximate query processing, stochastic gradient descent, and statistical estimation, random sampling is a common practice even if privacy is not a concern.

However, when we run $\mathcal{M}$ on a sample of I, the DP noise gets amplified. Specifically, when the sample is taken with probability η, we need to scale back the output of $\mathcal{M}$ by a factor of $\frac{1}{\eta}$ (this factor can be even larger under user-DP; see Section 4), which also amplifies the DP noise by a factor of $\frac{1}{\eta}$. Interestingly, in recent years, the literature has observed that this scaling back of DP noise can be avoided. More formally, let $\mathcal{S} : \mathcal{I} \rightarrow \mathbb{P}(\mathcal{I})$ be a sampling strategy and $\mathcal{M}$ an (ε, δ)-DP mechanism, we denote $\mathcal{M}^{\mathcal{S}} : \mathcal{I} \rightarrow \mathbb{P}(\mathcal{Z})$ as the mechanism running on a sample produced by $\mathcal{S}$. Then, it has been shown that $\mathcal{M}^{\mathcal{S}}$ satisfies $(\varepsilon' = \Theta(\eta\varepsilon), \delta')$-DP (assuming $\varepsilon < O(1)$). This phenomenon is known as *privacy amplification*, as the privacy parameter has decreased, hence better privacy, from ε to $\varepsilon' = \Theta(\eta\varepsilon)$ after sampling. Conversely, if the given privacy budget is ε', we can run $\mathcal{M}$ with parameter $\varepsilon = \Theta(\frac{\varepsilon'}{\eta})$. Since for most mechanisms (e.g., the Laplace mechanism), the DP noise is inversely proportional to the privacy parameter, this means that the DP noise injected to $\mathcal{M}^{\mathcal{S}}$ can be a factor-$(\frac{1}{\eta})$ smaller, which precisely cancels the effect of the noise amplification during the scaling back. The total error of $\mathcal{M}^{\mathcal{S}}$ is thus the sampling error, which is unrelated to privacy, plus a DP noise whose scale is the same as that injected by $\mathcal{M}$ running on the full dataset. Privacy amplification is therefore an important tool to get the performance boost from the sampling without sacrificing the utility (of the DP part).

1.4 Our Results

However, all past work on privacy amplification by sampling has only studied the standard, record-level DP model. In this paper, we take the first step towards investigating how much privacy can be amplified by sampling under user-DP.

A direct migration from record-level DP sampling to user-level DP is to sample the users, i.e., we first randomly sample a subset U_s of users from U, and construct the sampled instance $I_{|U_s} = (U_s, V_{|U_s})$, where $V_{|U_s}$ consists of all records contributed by U_s. Then we feed $I_{|U_s}$ to a user-DP mechanism $\mathcal{M}$. For example, the triangle count can be estimated by sampling a set of nodes U_s and counting the number of triangles in the subgraph induced by U_s. In Section 4, we first study the privacy amplification effect of this simple sampling strategy. For this case, we obtain a result similar to the standard privacy amplification result, i.e., the sampled mechanism $\mathcal{M}^{\mathcal{S}}$ satisfies (ε', δ')-DP for $\varepsilon' = \Theta(\eta\varepsilon)$.

Unfortunately, the amount of privacy amplification is *not* sufficient to cancel the noise amplification, because the scaling-back factor becomes $\frac{1}{\eta^\ell}$ under user-DP. Thus, the DP noise is still amplified by a factor of $\frac{1}{\eta^{\ell-1}}$. In addition, even in the non-private setting, simple sampling is known to have large sampling errors, and the literature has identified many more accurate sampling methods, mostly for graph data.

The main technical result of this paper is a method called *sample-and-explore*, which can be considered as an extension of a popular graph sampling method to user-DP. It first samples one user and then takes all the neighboring users (i.e., those with joint contributions) into the sampled instance. Intuitively, this sampling method is much better than simple sampling as it is more focused and ensures that all users sampled are useful; technically, it is better as its scaling-back factor does not depend on ℓ. However, it introduces challenges to the privacy amplification analysis, since the users are no longer sampled independently: There is a high correlation between neighboring users, which must be taken into consideration in the privacy amplification analysis.

Section 5 contains our main privacy amplification result for sample-and-explore. We first analyze the case where a single user is sampled, for which we show that the privacy parameter can be reduced by roughly a factor of $\frac{\tau}{n}$, where τ is an upper bound on the number of neighbors a user may have. Then we use advanced composition to extend it to the case where k users are sampled. The final result implies that the DP noise is amplified by a factor of $O(\tau \cdot \sqrt{\log(1/\delta)})$. Thus, it significantly improves that of simple sampling and sample-and-explore without privacy amplification. However, it does not quite match the amplification result from record-level DP, where the DP noise remains the same as without sampling. But this is intuitively inherent: The extra τ accounts for the correlation between users under the user-DP model, while the $\sqrt{\log(1/\delta)}$ factor roots from the advanced composition.

In Section 6, we conduct an extensive set of experiments on subgraph counting queries under node-DP, as well as relational queries on TPC-H data. The results show that the sampling-based mechanisms can significantly reduce the running time of existing mechanisms that run on full datasets, by 10^4 to 10^6 times. The trade-off is, of course, that the sampling methods incur a higher error, mostly the sampling error, as the DP noise has been greatly reduced by our privacy amplification analysis. Thus, the sampling-based mechanisms can be a useful tool to obtain some quick estimates on a large private dataset, before a (much) more expensive mechanism should be invoked on the full dataset.

2 RELATED WORK

The study of user-DP began with the simple case where $\ell = 1$ [2, 23, 30, 35]. The existing works have studied problems like sum estimation, quantile estimation, machine learning, etc. For the case $\ell \geq 2$, [10] studies the sum estimation problem, and [14] extends it to estimating any monotonic function. Due to the correlation between the users, linear programming is often needed in these mechanisms, resulting in long running times. Node-DP and edge-DP are two special cases of user-DP, which have been extensively studied [6, 8, 9, 22, 26, 27, 37], but none of them has considered the sampling amplification issue.

Privacy amplification by sampling under record-level DP has been studied since 2008 [21]. Balle et al. [3] analyze the privacy amplification results for Poisson sampling and sampling with/without replacement under (ε, δ)-DP. For Renyi DP, privacy amplification for Poisson sampling [1, 25, 38] and sampling without replacement [34] is also well studied. Bun et al. [7], on the other hand, focuses on concentrated DP and analyzes the amplification results for Poisson sampling. However, due to the correlation between users, these results do not hold under user-DP.

Besides sampling, other techniques that can achieve privacy amplification include shuffling [15, 16, 32], iteration [17], stochastic post-processing [4], etc. It would be interesting to consider their counterparts in the user-DP model as well.

3 PRELIMINARIES

3.1 Differential Privacy

Definition 1.1 is not convenient for privacy amplification analyses. Below we introduce an equivalent definition of DP in terms of α-divergences. For some $\alpha \geq 1$, the α-divergence between two probability distributions $\mu, \mu' \in \mathbb{P}(\mathcal{Z})$ is

$$D_\alpha(\mu || \mu') = \int_{\mathcal{Z}} \max\left(\frac{d\mu}{d\mu'}(z) - \alpha, 0\right) d\mu'(z),$$

where $\frac{d\mu}{d\mu'}$ is the Radon-Nikodym derivative. If $\mathcal{Z}$ is a discrete space, it can be simplified to

$$D_\alpha(\mu || \mu') = \sum_{I \in \mathcal{I}} \max(\mu(I) - \alpha\mu'(I), 0).$$

The following definition has been shown to be equivalent to Definition 1.1 [5]:

Definition 3.1. Let $\delta_{\mathcal{M}}(\varepsilon) = \sup_{I \sim I'} D_{e^\varepsilon}(\mathcal{M}(I) || \mathcal{M}(I'))$ denote the privacy profile of $\mathcal{M}$ at ε. A mechanism $\mathcal{M}$ is (ε, δ)-DP if $\delta_{\mathcal{M}}(\varepsilon) \leq \delta$.

We further use $\delta_{\mathcal{M},k}(\varepsilon) = \sup_{d(I,I') \leq k} D_{e^\varepsilon}(\mathcal{M}(I) || \mathcal{M}(I'))$ to denote the k-th group privacy profile of $\mathcal{M}$ at ε. Note that $\delta_{\mathcal{M},0}(\varepsilon) = 0, \delta_{\mathcal{M},1}(\varepsilon) = \delta_{\mathcal{M}}(\varepsilon)$. For any $k \geq 1$, we always have

$$\delta_{\mathcal{M},k}(\varepsilon) \leq \frac{e^\varepsilon - 1}{e^{\frac{\varepsilon}{k}} - 1} \cdot \delta_{\mathcal{M}}\left(\frac{\varepsilon}{k}\right).$$

The following are some important properties of DP [12]:

Lemma 3.2 (Composition). *Given (ε, δ)-DP mechanisms $\mathcal{M}_1, \ldots, \mathcal{M}_k$, the mechanism $\mathcal{M} = (\mathcal{M}_1, \ldots, \mathcal{M}_k)$ satisfies $(k\varepsilon, k\delta)$-DP. Furthermore, for any $\delta_0 > 0$, $\mathcal{M}$ also satisfies $(\varepsilon', k\delta + \delta_0)$-DP for*

$$\varepsilon' = \sqrt{2k \ln\left(\frac{1}{\delta_0}\right)} \cdot \varepsilon + k\varepsilon(e^\varepsilon - 1).$$

Lemma 3.3 (Post-processing). *Given an (ε, δ)-DP mechanism $\mathcal{M}$, any post-processing on the output of $\mathcal{M}$ still preserves (ε, δ)-DP.*

For any function $f : \mathcal{I} \to \mathbb{R}$, define its (global) sensitivity $\mathrm{GS}_f = \sup_{I \sim I'} |f(I) - f(I')|$.

Lemma 3.4 (Laplace Mechanism). *Given an instance I, the Laplace mechanism outputs*

$$\mathcal{M}(I) = f(I) + \mathrm{GS}_f \cdot \mathrm{Lap}(\frac{1}{\varepsilon}),$$

where $\mathrm{Lap}(b)$ *is the Laplace distribution centered at* 0 *with scale b, and the mechanism preserves pure ε-DP.*

3.2 Privacy Amplification

Let $\mathcal{S}$ be a sampling procedure, $\mathcal{M}$ an (ε, δ)-DP mechanism. We use $\mathcal{M}^{\mathcal{S}}$ to denote running $\mathcal{M}$ on a sample returned from $\mathcal{S}$. Let M be the Markov kernel associated with the mechanism $\mathcal{M}$, and let

$$\mathrm{TV}(\mu, \mu') = 1 - \sum_{\bar{I} \in \mathcal{I}} \min(\mu(\bar{I}), \mu'(\bar{I}))$$

denote the total variation distance between two probability distributions $\mu, \mu' \in \mathbb{P}(\mathcal{I})$. The privacy amplification framework from [3] can be summarized as follows.

Lemma 3.5 ([3]). *Let $I \sim I'$ be any two neighboring instances. For any $\mathrm{TV}(\mathcal{S}(I), \mathcal{S}(I')) \leq \eta \leq 1$, there exist distributions $\omega_0, \omega_1, \omega_1' \in \mathbb{P}(\mathcal{I})$ such that*

$$\mathcal{S}(I) = (1-\eta)\cdot\omega_0 + \eta\cdot\omega_1, \tag{1}$$
$$\mathcal{S}(I') = (1-\eta)\cdot\omega_0 + \eta\cdot\omega_1'. \tag{2}$$

Furthermore,

$$D_{e^{\varepsilon'}}(\mathcal{M}^{\mathcal{S}}(I)||\mathcal{M}^{\mathcal{S}}(I')) \leq (1-e^{\varepsilon'-\varepsilon})\cdot\eta\cdot D_{e^{\varepsilon}}(\omega_1 M||\omega_0 M) + e^{\varepsilon'-\varepsilon}\cdot\eta\cdot D_{e^{\varepsilon}}(\omega_1 M||\omega_1' M), \tag{3}$$

for $\varepsilon' = \ln(1+\eta\cdot(e^{\varepsilon}-1))$.

Together with Definition 3.1, the lemma above implies that the privacy parameter ε can be reduced (privacy is amplified) to ε', if we can decompose $\mathcal{S}(I), \mathcal{S}(I')$ as in (1), (2) so that (3) is negligible. The amount of amplification directly depends on η; in particular, for $\varepsilon < O(1)$, we have $\varepsilon' = \Theta(\eta\varepsilon)$.

In order to bound (3), we need to bound $D_{e^{\varepsilon}}(\omega_1 M||\omega_0 M)$ and $D_{e^{\varepsilon}}(\omega_1 M||\omega_1' M)$. This is done using a technique called *coupling*. A coupling between ω_1 and ω_0 is a distribution $\pi \in \mathbb{P}(\mathcal{I}\times\mathcal{I})$ whose marginal distributions are ω_1 and ω_0, respectively.

Lemma 3.6 ([3]). *For any coupling π between ω_1 and ω_0 and any DP mechanism $\mathcal{M}$,*

$$\begin{aligned} D_{e^{\varepsilon}}(\omega_1 M||\omega_0 M) &\leq \sum_{I_1, I_0} \pi(I_1, I_0)\cdot D_{e^{\varepsilon}}(\mathcal{M}(I_1)||\mathcal{M}(I_0)) \\ &\leq \sum_{I_1, I_0} \pi(I_1, I_0)\cdot \delta_{\mathcal{M}, d(I_1, I_0)}(\varepsilon). \end{aligned} \tag{4}$$

Then we replace ω_0 by ω_1' in Lemma 3.6 to bound $D_{e^{\varepsilon}}(\omega_1 M||\omega_1' M)$ using any coupling between ω_1 and ω_1'. Note that (4) ranges over all I_1, I_0 such that $\pi(I_1, I_0) > 0$, so it is not necessary that $I_1 \sim I_0$. But a good coupling should pair up I_1, I_0 such that $d(I_1, I_0)$ is small so as to minimize (4).

4 PRIVACY AMPLIFICATION BY SIMPLE SAMPLING

As a warm-up, we first apply the privacy amplification framework above to the analysis of a simple sampling strategy $\mathcal{S}$ under user-DP, where each user $u \in U$ is sampled independently with probability η. Let $U_s \subseteq U$ denote the set of sampled users. The sub-instance of I induced by U_s is $I_{|U_s} = (U_s, V_{|U_s})$, where

$$V_{|U_s}(x) = \begin{cases} V(x), & \text{if } \forall u \in x, u \in U_s \\ \varnothing, & \text{otherwise.} \end{cases}$$

Namely, $V_{|U_s}$ consists of all records contributed by users in U_s. Then, we feed $I_{|U_s}$ to some user-DP mechanism $\mathcal{M}$. The output of $\mathcal{M}(I_{|U_s})$ should be appropriately post-processed to obtain a good estimate of $f(R(V))$.

Example 4.1. For the triangle counting problem under node-DP, this sampling strategy is precisely the subgraph sampling algorithm described in [31, 36], where each node is sampled to U_s with probability η. Then we count the number of triangles in the subgraph $I_{|U_s}$ induced by the sampled nodes using some node-DP mechanism $\mathcal{M}$ [6, 10, 11, 37]. Since each triangle appears in $I_{|U_s}$ with probability η^3, we need to scale the output of $\mathcal{M}(I_{|U_s})$ back by a factor of $\frac{1}{\eta^3}$. Edge-DP can be similarly handled by sampling the edges instead of nodes. □

4.1 Privacy Amplification

Without privacy amplification, the DP noise injected to $\mathcal{M}(I_{|U_s})$ is in general multiplied by a factor of $\frac{1}{\eta^\ell}$ during the scaling back. Below we show how this can be mitigated by privacy amplification.

We need to apply Lemma 3.5 and 3.6 on every two neighboring instances $I \sim I'$. First, consider the case where $I' \preceq I$, and let u^* denote the witness. We observe the following properties of $I_{|U_s}$: For any set of sampled users U_s,

(1) if $u^* \notin U_s$, then $I_{|U_s} \preceq I'$;
(2) if $I_{|U_s} \preceq I'$, then $I_{|U_s} = I'_{|U_s}$, otherwise, $\mathcal{S}(I')(I|_{U_s}) = 0$.

These properties imply that the common support of $\mathcal{S}(I)$ and $\mathcal{S}(I')$ must be the sub-instances of I'. In addition, for each $\bar{I} \preceq I'$, we have $\mathcal{S}(I)(\bar{I}) \leq \mathcal{S}(I')(\bar{I})$. This is because, by property (2), any set of sampled users U_s such that $I_{|U_s} = \bar{I}$ also lead to $I'_{|U_s} = \bar{I}$. This allows us to bound $\mathrm{TV}(\mathcal{S}(I), \mathcal{S}(I'))$:

$$\begin{aligned}\mathrm{TV}(\mathcal{S}(I), \mathcal{S}(I')) =& 1 - \sum_{\bar{I} \in \mathcal{I}} \min(\mathcal{S}(I)(\bar{I}), \mathcal{S}(I')(\bar{I})) \\ =& 1 - \sum_{\bar{I} \preceq I'} \mathcal{S}(I)(\bar{I}) \\ \leq& 1 - \Pr[u^* \notin U_s] \\ =& \eta\end{aligned}$$

Next, we decompose $\mathcal{S}(I)$ and $\mathcal{S}(I')$ as in (1), (2). Let $v(U_s)$ be the sampling probability of U_s over U, i.e.,

$$v(U_s) = \eta^{|U_s|}(1-\eta)^{|U|-|U_s|}.$$

Then we define ω_0 by setting

$$\begin{aligned}\omega_0(\bar{I}) =& \frac{1}{1-\eta} \cdot \sum_{U_s : u^* \notin U_s, I_{|U_s} = \bar{I}} v(U_s) \\ =& \frac{1}{1-\eta} \cdot \sum_{U_s : u^* \notin U_s, I'_{|U_s} = \bar{I}} v(U_s)\end{aligned}$$

for every $\bar{I} \in \mathcal{I}$. The second equality is due to the two properties above, namely, when u^* is not sampled, the induced sub-instances $I_{|U_s}$ and $I'_{|U_s}$ are the same. The other two distributions ω_1, ω'_1 then follow from ω_0 and (1), (2), i.e.,

$$\omega_1(\bar{I}) = \frac{1}{\eta} \cdot \sum_{U_s : u^* \in U_s, I_{|U_s} = \bar{I}} v(U_s),$$

$$\omega'_1(\bar{I}) = \frac{1}{\eta} \cdot \sum_{U_s : u^* \in U_s, I'_{|U_s \cap U'} = \bar{I}} v(U_s).$$

Example 4.2. Consider the triangle counting problem under node-DP. Let I' be a down neighbor of I with witness u^*, namely, all extra triangles in I involve u^*. The three distributions above are as follows:

- ω_0 samples each node in $U \setminus \{u^*\}$ with probability η and constructs the induced subgraph of I' on the sampled nodes.
- ω_1 samples each node in $U \setminus \{u^*\}$ with probability η, adds u^* to the sample and constructs the induced subgraph of I.

- ω_1' samples each node in $U \setminus \{u^*\}$ with probability η, adds u^* to the sample and constructs the induced subgraph of I'. □

Next, we bound $D_{e^\varepsilon}(\omega_1 M || \omega_0 M)$. We can construct a coupling π between ω_1 and ω_0 iteratively as follows. We initialize $\pi(I_1, I_0) = 0$ for any pair of samples I_1, I_0. Next, we go through all sets of users $U_s \subseteq U \setminus \{u^*\}$ and increase $\pi(I_{|U_s \cup \{u^*\}}, I_{|U_s})$ by

$$\frac{1}{\eta} \cdot v(U_s \cup \{u^*\}) = \frac{1}{1-\eta} \cdot v(U_s).$$

It can be verified that π is a valid coupling. Moreover, from the construction process, we can see that if $\pi(I_1, I_0) > 0$, we have $d(I_1, I_0) \leq 1$ since the witness u^* is the only different user such that all different records, if there are any, are contributed by u^*. Then by Lemma 3.6, we have $D_{e^\varepsilon}(\omega_1 M || \omega_0 M) \leq \delta_{\mathcal{M}}(\varepsilon)$.

Example 4.3. Continuing with the example above, the coupling π corresponds to the following random process that produces $(I_1, I_0) \in \mathcal{I} \times \mathcal{I}$: Take each node in $U \setminus \{u^*\}$ into the sample U_s with probability η independently. Then set I_0 as the subgraph of I' induced by U_s, and I_1 as the subgraph of I induced by $U_s \cup \{u^*\}$. It is easy to see that the marginal distributions of π are ω_1 and ω_0, respectively, and any I_1, I_0 coupled together (i.e., $\pi(I_1, I_0) > 0$) only differ in u^*. □

For $D_{e^\varepsilon}(\omega_1 M || \omega_1' M)$, we can construct a coupling π similarly: We initialize $\pi(I_1, I_1') = 0$ for any pair of samples I_1, I_1'. Next, we go through all sets of users $U_s \subseteq U$ such that $u^* \in U_s$, and increase $\pi(I_{|U_s}, I'_{|U_s \cap U'})$ by $\frac{1}{\eta} \cdot v(U_s)$. Any I_1, I_1' coupled together still have $d(I_1, I_1') \leq 1$. Thus, we can conclude $D_{e^\varepsilon}(\omega_1 M || \omega_1' M) \leq \delta_{\mathcal{M}}(\varepsilon)$. Then by Lemma 3.5, we obtain $D_{e^{\varepsilon'}}(\mathcal{M}^{\mathcal{S}}(I) || \mathcal{M}^{\mathcal{S}}(I')) \leq \eta \cdot \delta_{\mathcal{M}}(\varepsilon)$.

Example 4.4. The coupling between ω_1 and ω_1' corresponds to the following random process that produces $(I_1, I_1') \in \mathcal{I} \times \mathcal{I}$: Take each node in $U \setminus \{u^*\}$ into the sample U_s with probability η independently. Then set I_1 as the subgraph of I induced by $U_s \cup \{u^*\}$, set I_1' as the subgraph of I' induced by $U_s \cup \{u^*\}$. □

For the other case where $I \leq I'$, the decomposition is symmetric. More specifically, we now sample a set of users $U_s \subseteq U'$, and the decomposition looks as follows. Let $v(U_s)$ be the sampling probability of U_s over U', we have

$$\begin{aligned}
\omega_0(\bar{I}) &= \frac{1}{1-\eta} \cdot \sum_{U_s : u^* \notin U_s, I'_{|U_s} = \bar{I}} v(U_s) = \frac{1}{1-\eta} \cdot \sum_{U_s : u^* \notin U_s, I_{|U_s} = \bar{I}} v(U_s), \\
\omega_1(\bar{I}) &= \frac{1}{\eta} \cdot \sum_{U_s : u^* \in U_s, I_{|U_s \cap U} = \bar{I}} v(U_s), \omega_1'(\bar{I}) = \frac{1}{\eta} \cdot \sum_{U_s : u^* \in U_s, I'_{|U_s} = \bar{I}} v(U_s).
\end{aligned}$$

The analyses for $D_{e^\varepsilon}(\omega_1 M || \omega_0 M)$ and $D_{e^\varepsilon}(\omega_1 M || \omega_1' M)$ are the same as above. So we conclude with the following privacy amplification result:

Theorem 4.1. *For the simple sampling strategy $\mathcal{S}$ and any (ε, δ)-DP mechanism $\mathcal{M}$, $\mathcal{M}^{\mathcal{S}}$ satisfies (ε', δ')-DP, where*

$$\begin{aligned}
\varepsilon' &= \ln(1 + \eta \cdot (e^\varepsilon - 1)), \\
\delta' &= \eta \cdot \delta.
\end{aligned}$$

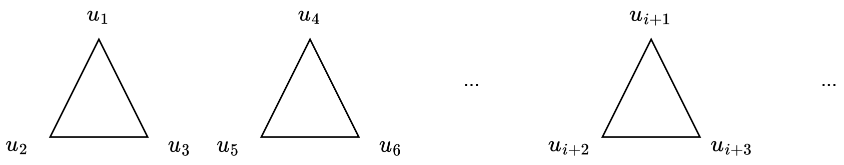


Fig. 2. Triangle counting in a sparse graph.

4.2 Error Analysis

Recall that the error in $\mathcal{M}^{\mathcal{S}}$ consists of two parts: the sampling error and the DP noise. Theorem 4.1 implies that, given a privacy budget of (ε', δ'), we can invoke $\mathcal{M}$ with privacy parameter $\varepsilon \approx \varepsilon'/\eta, \delta = \delta'/\eta$. Thus, the DP noise after scaling back is

$$\frac{1}{\eta^{\ell}} \cdot \text{Noise}_{\mathcal{M}}\left(\frac{\varepsilon'}{\eta}, \frac{\delta'}{\eta}\right) \approx \frac{1}{\eta^{\ell}} \cdot \eta \cdot \text{Noise}_{\mathcal{M}}(\varepsilon', \delta') = \frac{1}{\eta^{\ell-1}} \cdot \text{Noise}_{\mathcal{M}}(\varepsilon', \delta'), \tag{5}$$

where $\text{Noise}_{\mathcal{M}}(\varepsilon', \delta')$ denotes the DP noise injected by $\mathcal{M}$ with privacy parameter (ε', δ').

The analysis of the sampling error, which is unrelated to privacy, is detailed in [33, 36]. It depends on certain properties of the instance and can be very large on sparse instances.

Example 4.5. Consider the triangle counting problem on the instance in Figure 2. Since each of the $n/3$ triangles is sampled with probability η^3, the variance of the triangle count in the sample is $O(\eta^3 \cdot n)$, or an error of $O(\eta^{1.5} \cdot \sqrt{n})$. After scaling back, the sampling error becomes $O(\sqrt{n}/\eta^{1.5})$, which turns into $O(\sqrt{n}/\eta^{\ell/2})$ for a graph pattern with ℓ nodes, e.g., an ℓ-cycle. □

Besides Bernoulli sampling, other variants of simple sampling, such as taking a sample of size $\eta \cdot n$ without replacement, have also been considered in [31, 36]. The privacy amplification analyses for these variants are technical, while the results are not as clean as Theorem 4.1, hence omitted. Most importantly, all of them have high sampling errors and DP noises.

5 PRIVACY AMPLIFICATION BY SAMPLE-AND-EXPLORE

The intuitive reason why simple sampling has high errors (both sampling error and DP noise), especially for large ℓ and on sparse instances as Example 4.5 illustrates, is that the probability of forming a record (e.g., a triangle) by blindly sampling the users is very low. For the subgraph counting problem, there are many methods with much better accuracy [24, 29, 36], all of which take a sample-and-explore approach. We adopt one version of it and extend it to user-DP, as follows.

5.1 Sample-and-explore under User-DP

We first consider the simple case where only one user u_s is sampled uniformly from U. We construct the induced sub-instance $I_{|u_s} = (U_{|u_s}, V_{|u_s})$ where

$$U_{|u_s} = \{u \in U \mid \exists x = (u_s, \dots), u \in x, V(x) \neq \varnothing\},$$

and

$$V_{|u_s}(x) = \begin{cases} V(x), & \text{if } x = (u_s, \dots); \\ \varnothing, & \text{otherwise.} \end{cases}$$

Namely, $V_{|u_s}$ consists of all records for which u_s is the first contributing user, while the user set $U_{|u_s}$ includes all involved users. To improve accuracy, this can be repeated, as illustrated in Section 5.3.

Example 5.1. For the triangle counting problem under node-DP, the sample-and-explore strategy instantiates into the following: First sample a random node u_s. Then take all the triangles where u_s is the smallest node (in terms of user id) into $I_{|u_s}$. Because each triangle appears in $I_{|u_s}$ with probability $1/n$, we scale the triangle count in $I_{|u_s}$ back by a factor of n.

On the instance in Figure 2, the number of triangles in $I_{|u_s}$ is 1 with probability 1/3 (when one of the $n/3$ nodes at the top of the triangles is sampled) and 0 with probability 2/3. Thus, the variance of the sample count is $O(1)$. After scaling back, this translates into a sampling error of $O(n)$. This does not appear to be much smaller than that of simple sampling in Example 4.5. However, we have only sampled one user so far. We will later extend it to sampling multiple users, which will reduce the sampling error significantly. □

5.2 Privacy Amplification

We first analyze the privacy amplification for sampling a single user. Given an instance $I = (U, V)$, for any user $u \in U$, we say that another user u' is a *neighbor* of u if there is an $x = (u, \dots) \in U^\ell$ such that $u' \in x$, and $V(x) \neq \emptyset$. Thus, sampling u causes u' to be also taken into the sample. Note that this neighboring relationship is defined between users in a given instance I, and should not be confused with the neighboring relationship between two instances.

Let $\Lambda(I, u)$ denote the set of neighbors of u on I. We will need an upper bound τ on $|\Lambda(I, u)|$ for any user $u \in \mathcal{U}$ and any instance $I \in \mathcal{I}$. For the problem of triangle counting, two nodes are neighbors if they appear in the same triangle, so they must be connected by an edge. Thus, we may use the degree upper bound as τ.

Now we analyze the privacy amplification for the sample-and-explore strategy $\mathcal{S}$. Let $I \sim_{u^*} I'$ be any two neighboring instances. We first consider the case where $I' \preceq I$. As in simple sampling, we observe that similar properties also hold for $I_{|u_s}$. Specifically, for any sampled user u_s,

(1) if $u_s \notin \Lambda(I, u^*)$, then $I_{|u_s} \preceq I'$;
(2) if $I_{|u_s} \preceq I'$, then $I_{|u_s} = I'_{|u_s}$, otherwise, $\mathcal{S}(I')(I_{|u_s}) = 0$.

Thus, we can still conclude that the common support of $\mathcal{S}(I)$ and $\mathcal{S}(I')$ must be the set $\{\bar{I} : \bar{I} \preceq I'\}$, and for any instance $\bar{I} \preceq I'$, $\mathcal{S}(I)(\bar{I}) \leq \mathcal{S}(I')(\bar{I})$. The total variation distance is thus

$$\begin{aligned}
\mathrm{TV}(\mathcal{S}(I), \mathcal{S}(I')) =& 1 - \sum_{\bar{I} \in \mathcal{I}} \min(\mathcal{S}(I)(\bar{I}), \mathcal{S}(I')(\bar{I})) \\
=& 1 - \sum_{\bar{I} \preceq I'} \mathcal{S}(I)(\bar{I}) \\
\leq& 1 - \Pr[u_s \notin \Lambda(I, u^*)] \\
\leq& \frac{\tau}{n}
\end{aligned}$$

We thus set $\eta = \frac{\tau}{n}$.

We now construct the decomposition of $\mathcal{S}(\mathcal{I})$ and $\mathcal{S}(\mathcal{I}')$. We add more users to $\Lambda(I, u^*)$ to obtain a set of τ users including all neighbors of u. Let $\Lambda_\tau(I, u^*)$ be the resulting set of users. Let $n' = |U'|$. We have $n' \leq n$ when $I' \preceq I$. Let v and v' denote the sampling distribution of u_s on I and I' respectively, i.e.,

$$v(u_s) = \frac{1}{n} \quad \text{and} \quad v'(u_s) = \begin{cases} \frac{1}{n'}, & \text{if } u_s \in U', \\ 0, & \text{otherwise.} \end{cases}$$

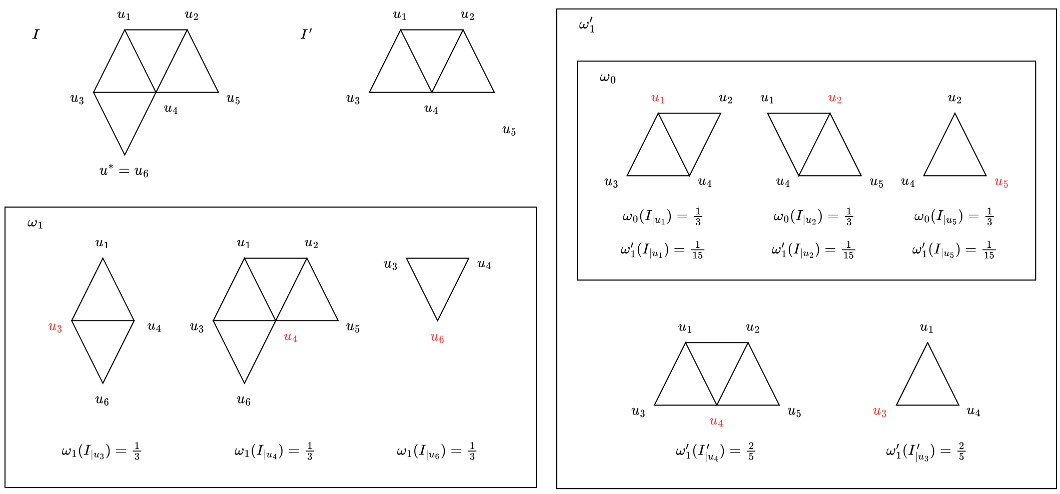

I
$u^* = u_6$
I'
ω_1
$\omega_1(I_{|u_3}) = \frac{1}{3}$
$\omega_1(I_{|u_4}) = \frac{1}{3}$
$\omega_1(I_{|u_6}) = \frac{1}{3}$
ω'_1
ω_0
$\omega_0(I_{|u_1}) = \frac{1}{3}$
$\omega'_1(I_{|u_1}) = \frac{1}{15}$
$\omega_0(I_{|u_2}) = \frac{1}{3}$
$\omega'_1(I_{|u_2}) = \frac{1}{15}$
$\omega_0(I_{|u_5}) = \frac{1}{3}$
$\omega'_1(I_{|u_5}) = \frac{1}{15}$
$\omega'_1(I'_{|u_4}) = \frac{2}{5}$
$\omega'_1(I'_{|u_3}) = \frac{2}{5}$

Fig. 3. Decomposition of $\mathcal{S}(I)$ and $\mathcal{S}(I')$ on example instances for triangle counting.

For any sample $\bar{I} \in \mathcal{I}$, we set

$$\omega_0(\bar{I}) = \frac{1}{1-\eta} \cdot \sum_{u_s : u_s \notin \Lambda_\tau(I,u^*), I_{|u_s} = \bar{I}} v(u_s).$$

Then ω_1 and ω'_1 can be set accordingly, namely,

$$\begin{aligned}
\omega_1(\bar{I}) =& \frac{1}{\eta} \cdot \sum_{u_s : u_s \in \Lambda_\tau(I,u^*), I_{|u_s} = \bar{I}} v(u_s), \\
\omega'_1(\bar{I}) =& \frac{1}{\eta} \cdot \sum_{u_s : u_s \in U', u_s \in \Lambda_\tau(I,u^*), I'_{|u_s} = \bar{I}} v'(u_s) \\
&+ \frac{1}{\eta} \cdot \sum_{u_s : u_s \in U', u_s \notin \Lambda_\tau(I,u^*), I'_{|u_s} = \bar{I}} (v'(u_s) - v(u_s)).
\end{aligned}$$

Example 5.2. Consider the triangle counting problem under node-DP on the two instances I and I' in Figure 3, where $u^* = u'_6$, $n = 6$, and $n' = 5$. Assume $\tau = 3$ so that $\eta = \frac{1}{2}$ and $\Lambda_\tau(I, u^*) = \Lambda(I, u^*) = \{u_3, u_4, u_6\}$. All samples in $\omega_0, \omega_1, \omega'_1$ with nonzero probabilities are shown in the figure. □

Next, we analyze $D_{e^\varepsilon}(\omega_1 M || \omega_0 M)$ and $D_{e^\varepsilon}(\omega_1 M || \omega'_1 M)$. For both cases, we can use an arbitrary coupling π, because every possible sample has only contributions from one user. It is tempting to claim that any two samples have a distance of 2, but this is not true, strictly speaking. For example, consider $I_{|u_3}$ in the support of ω_1 and $I_{|u_1}$ in the support of ω_0 in Figure 3. To change the former into the latter, we delete u_3 and all its contributions (the two triangles involving u_3) and then add u_1 and its contributions (the two triangles involving u_1). However, this is not all. After deleting u_3 and the two triangles involving u_3, we still have singleton nodes u_1, u_4, u_6 left. The node u_6 will need to be deleted in a separate step. Also, before adding the two triangles involving u_1, the node u_2 has to be added. Thus, we have $d(I_{|u_3}, I_{|u_1}) = 4$, which can be larger if u_1 and u_3 have larger degrees.

To address the issue, we introduce a mild assumption on the mechanism $\mathcal{M}$, that it does not depend on these singleton nodes, or more generally, users without contributions. This is reasonable as in our user-DP model, the goal is to approximate a function $f(I) = f(R(V))$, where $R(V)$ does not depend on users without contributions. However, as $\mathcal{M}$ may behave arbitrarily (it may not try to approximate $f(R(V))$ at all), we still need this condition for the privacy amplification analysis.

Under this assumption, $D_{e^\varepsilon}(\omega_1 M || \omega_0 M)$ and $D_{e^\varepsilon}(\omega_1 M || \omega_1' M)$ are both bounded by $\delta_{\mathcal{M},2}(\varepsilon)$ under an arbitrary coupling. Then by Lemma 3.5, we have

$$D_{e^{\varepsilon'}}(\mathcal{M}^{\mathcal{S}}(I) || \mathcal{M}^{\mathcal{S}}(I')) \le \eta \cdot \delta_{\mathcal{M},2}(\varepsilon).$$

The analysis for the other case $I \preceq I'$ is similar. The total variation distance is

$$\begin{aligned}\mathrm{TV}(\mathcal{S}(I), \mathcal{S}(I')) =& 1 - \sum_{\bar{I} \in \mathcal{I}} \min(\mathcal{S}(I)(\bar{I}), \mathcal{S}(I')(\bar{I})) \\ =& 1 - \sum_{\bar{I} \preceq I} \mathcal{S}(I')(\bar{I}) \\ \le& 1 - \Pr[u_s \notin \Lambda(I', u^*)] \\ \le& \frac{\tau}{n'}.\end{aligned}$$

Let $\eta' = \frac{\tau}{n'}$. We can decompose the distributions $\mathcal{S}(I)$ and $\mathcal{S}(I')$ into ω_0, ω_1 and ω_1' as in (1) and (2). The remaining proofs are the same, and we can replace η with η' to get the upper bounds for $D_{e^\varepsilon}(\omega_1 M, \omega_0 M)$ and $D_{e^\varepsilon}(\omega_1 M, \omega_1' M)$. Note that we have $n \le n'$ when $I \preceq I'$, so $\eta' \le \eta$, and $D_{e^{\varepsilon'}}(\mathcal{M}^{\mathcal{S}}(I) || \mathcal{M}^{\mathcal{S}}(I'))$ can also be bounded by $\eta \cdot \delta_{\mathcal{M},2}(\varepsilon)$.

Theorem 5.1. *For the single sample-and-explore strategy $\mathcal{S}$ and any (ε, δ)-DP mechanism $\mathcal{M}$ that does not depend on users without contributions, $\mathcal{M}^{\mathcal{S}}$ satisfies $(\varepsilon_1', \delta_1')$-DP, where*

$$\begin{aligned}\varepsilon_1' &= \ln\left(1 + \frac{\tau}{n} \cdot (e^\varepsilon - 1)\right), \\ \delta_1' &= \frac{\tau}{n} \cdot \delta_{\mathcal{M},2}(\varepsilon).\end{aligned}$$

One potential problem is $\delta_{\mathcal{M},2}(\varepsilon)$, hence δ_1', may not be negligible, even if $\delta = \delta_{\mathcal{M}}(\varepsilon)$ is. This problem can be fixed as follows. We view $\mathcal{M}$ as a $(2\varepsilon, \delta)$-DP mechanism $\mathcal{M}'$ satisfying $\delta_{\mathcal{M}'}(\varepsilon) \le \delta$. Then we apply Theorem 5.1 on $\mathcal{M}'$ with privacy parameters $2\varepsilon, \delta$. This leads to the following result:

Corollary 5.2. *For the simple sample-and-explore strategy $\mathcal{S}$ and any (ε, δ)-DP mechanism $\mathcal{M}$ that does not depend on users without contributions, $\mathcal{M}^{\mathcal{S}}$ satisfies $(\varepsilon_1', \delta_1')$-DP, where*

$$\begin{aligned}\varepsilon_1' &= \ln\left(1 + \frac{\tau}{n} \cdot (e^{2\varepsilon} - 1)\right), \\ \delta_1' &= \frac{\tau}{n} \cdot (e^\varepsilon + 1) \cdot \delta.\end{aligned}$$

Remark. The Corollary above aims for generality while losing a factor of 2. For many mechanisms, we can bound $D_{e^\varepsilon}(\omega_1 M || \omega_0 M)$ and $D_{e^\varepsilon}(\omega_1 M || \omega_1' M)$ more directly, thus obtaining a tighter privacy amplification result, as shown in Section 5.4.

5.3 Repeated Sample-and-explore

The privacy amplification result above only considers sampling a single user, which has a large sampling error. To reduce the sampling error, we repeat the sampling k times independently and take the average. More precisely, let $I_{|u_s}^{(i)}$ denote the i-th sample, then the repeated sample-and-explore mechanism is

$$\mathcal{M}^{\mathcal{S}}(I) = \frac{1}{k} \cdot \sum_{i=1}^{k} \mathcal{M}(I_{|u_s}^{(i)}).$$

The privacy of $\mathcal{M}^{\mathcal{S}}$ follows from Theorem 5.1 and Lemma 3.2 straightforwardly:

Theorem 5.3. *For the repeated sample-and-explore strategy $\mathcal{S}$, any (ε, δ)-DP mechanism $\mathcal{M}$ that does not depend on users without contributions, and any $\delta_0 > 0$, $\mathcal{M}^{\mathcal{S}}$ satisfies (ε', δ')-DP for*

$$\varepsilon' = \sqrt{2k \ln(\frac{1}{\delta_0})} \cdot \varepsilon_1' + k\varepsilon_1'(e^{\varepsilon_1'} - 1),$$
$$\delta' = k\delta_1' + \delta_0,$$

for $\varepsilon_1', \delta_1'$ stated in Theorem 5.1 or Corollary 5.2.

For $\varepsilon' < O(1)$, the result above can be simplified to $\varepsilon' = \Theta(\frac{\tau}{n} \cdot \sqrt{k \log(1/\delta_0)} \cdot \varepsilon)$, or $\varepsilon = \Theta(\varepsilon' \cdot \frac{n}{\tau} / \sqrt{k \log(1/\delta_0)})$. Thus, the DP noise after averaging over k invocations of $\mathcal{M}$ and scaling back is (assuming $\mathcal{M}$ adds zero-mean noise)

$$\begin{aligned} n \cdot \frac{\text{Noise}_{\mathcal{M}}(\varepsilon, \delta)}{\sqrt{k}} &= n \cdot \frac{\tau}{n} \cdot \sqrt{k \log(1/\delta_0)} \cdot \frac{\text{Noise}_{\mathcal{M}}(\varepsilon', \delta')}{\sqrt{k}} \\ &= \tau \cdot \sqrt{\log(1/\delta_0)} \cdot \text{Noise}_{\mathcal{M}}(\varepsilon', \delta'). \end{aligned}$$

Example 5.3. Continuing with Example 5.1, we repeat sample-and-explore k times and take the average. The sampling error is then reduced to $O(n/\sqrt{k})$. To compare it with simple sampling, we set $\eta = \frac{k}{n}$, then the sampling error of simple sampling is $O(n^2/k^{1.5})$, i.e., repeated sample-and-explore reduces the sampling error by a factor of $O(n/k)$, or $O((n/k)^{(\ell-1)/2})$ for a pattern with ℓ nodes. Intuitively, this is because in sample-and-explore, all users taken into the sample are "useful", whereas most users sampled under simple sampling are wasted as they do not form tuples with contributions (i.e., triangles).

In terms of DP noise, sample-and-explore reduces that of simple sampling (see Section 4.2) by a factor of $O((n/k)^{\ell-1}/\tau)$. Note that, without privacy amplification, we must use composition to allocate the given privacy budget ε' to each invocation of the mechanism $\mathcal{M}$ running on the k samples, each with $\varepsilon \approx \varepsilon'/\sqrt{k}$ (ignoring log factors). The DP noise on each sample is thus $\sqrt{k} \cdot \text{Noise}_{\mathcal{M}}(\varepsilon', \delta')$. The averaging reduces the noise by a factor of $\sqrt{k}$, to $\text{Noise}_{\mathcal{M}}(\varepsilon', \delta')$. Finally, the scaling-back puts the final DP noise at $n \cdot \text{Noise}_{\mathcal{M}}(\varepsilon', \delta')$. This is not necessarily better than the DP noise of simple sampling and may dominate the sampling error. Thus, it is important to derive privacy amplification results for sample-and-explore to continue to be useful under DP. □

5.4 Instantiation with Truncation Mechanism

In this section, we instantiate our sample-and-explore framework with $\mathcal{M}$ being the truncation mechanism [10, 22], which is the state-of-the-art mechanism under user-DP. Instead of a direct plug-in, we make the following three technical improvements: First, due to some nice properties of each sampled instance $I_{|u_s}$ in the sample-and-explore framework, we can simplify the mechanism without solving the linear program. Second, we tighten the privacy amplification analysis for the truncation mechanism by analyzing $D_{e^\varepsilon}(\omega_1 M || \omega_0 M)$ and $D_{e^\varepsilon}(\omega_1 M || \omega_1' M)$ directly. Thirdly, as the truncation mechanism truncates user contributions before adding noise, it introduces a (negative) bias. While repeated sample-and-explore can reduce variance by increasing k, it cannot reduce bias. Thus, some careful bias-variance trade-off needs to be done to minimize the total DP error.

We first review the truncation mechanism $\mathcal{M}$ in [10, 22] when applied on a sample $I_{|u_s} = (U_{|u_s}, V_{|u_s})$. A variable x_t is introduced for each record $t \in R(I_{|u_s})$. Given a truncation threshold C, $\mathcal{M}$ solves the linear program:

$$\begin{aligned} \text{maximize} \quad & \check{f}(I_{|u_s}, C) = \sum_{t \in R(I_{|u_s})} x_t \\ \text{subject to} \quad & \sum_{t \in R(I_{|u_s}, u)} x_t \le C, \forall u \in U_{|u_s}, \\ & 0 \le x_t \le w(t), \forall t \in R(I_{|u_s}). \end{aligned}$$

Since the constraints in the linear program ensure that each user contributes to at most C, the global sensitivity of $\check{f}(\cdot, C)$ is C. Then, $\mathcal{M}$ outputs $\check{f}(I_{|u_s}, C) + C \cdot \text{Lap}(1/\varepsilon)$.

Simplification. We note that in any sample $I_{|u_s}$, all the records are contributed by user u_s. Thus, one of the constraints, namely, $\sum_{t \in R(I_{|u_s}, u_s)} x_t \le C$, dominates all others, and the linear program can be simplified to a truncated sum, i.e.,

$$\check{f}(I_{|u_s}, C) = \min\left(f(I_{|u_s}), C\right).$$

Privacy analysis. It is easy to see that the truncation mechanism does not depend on users without contributions, so the results in Section 5.2 directly apply. Actually, we can tighten the privacy amplification analysis. The key observation is that for any instances I, I', not necessarily neighbors, we always have

$$|\check{f}(I, C) - \check{f}(I', C)| \le C.$$

Therefore, the Laplace noise can always mask the difference, i.e., $\delta_{\mathcal{M},k}(\varepsilon) = \delta_{\mathcal{M}}(\varepsilon)$ for all k. Then we can replace $\delta_{\mathcal{M},k}(\varepsilon)$ in Theorem 5.1 with $\delta_{\mathcal{M}}(\varepsilon)$ to get the tight privacy analysis for the truncation mechanism:

THEOREM 5.4. *For the simple sample-and-explore strategy $\mathcal{S}$ and the ε-DP truncation mechanism $\mathcal{M}$, $\mathcal{M}^{\mathcal{S}}$ satisfies $(\varepsilon', 0)$-DP for*

$$\varepsilon' = \ln(1 + \frac{\tau}{n} \cdot (e^{\varepsilon} - 1)).$$

Then we combine Theorem 5.4 and 5.3 for the repeated sample-and-explore instantiation with the truncation mechanism, as described in Algorithm 1. In the algorithm, we also allocate part of the privacy budget ε_n to compute a privatized $\tilde{n}$ to be used as the scaling-back factor. Algorithm 1 then satisfies $(\varepsilon_{\text{total}}, \delta_{\text{total}})$-DP.

Setting the truncation threshold C. It remains to show how to find a good value for C. Given a threshold C, the final output of Algorithm 1 is

$$\tilde{f}(I) = \frac{\tilde{n}}{k} \cdot \sum_{i=1}^{k} \min\left(f(I_{|u_s}^{(i)}), C\right) + \frac{\tilde{n}}{k} \cdot C \cdot \sum_{i=1}^{k} \text{Lap}\left(\frac{1}{\varepsilon}\right).$$

Let $\check{f}(I, C) = \sum_{u_s \in U} \min(f(I_{|u_s}), C)$, so that

$$|\check{f}(I, C) - f(I)| = \sum_{u_s \in U} \max(f(I_{|u_s}) - C, 0).$$

Algorithm 1: Sum estimation by repeated sample-and-explore

```
Input  : The instance $I$, the function $f$, the iteration number $k$, the upper bound $\tau$, the
         truncation threshold $C$, and the privacy budget $\varepsilon_{\text{total}}$ and $\delta_{\text{total}}$
Output : A privatized $f(I)$
/* Step 1: Compute the privacy budget $\varepsilon$ */
1 $\varepsilon_n \leftarrow \frac{1}{5}\varepsilon_{\text{total}}$, $\varepsilon' \leftarrow \frac{4}{5}\varepsilon_{\text{total}}$;
2 $\varepsilon'_1 \leftarrow \text{argmax}\{x : \sqrt{2k\ln(\frac{1}{\delta_{\text{total}}})} \cdot x + kx(e^x - 1) \leq \varepsilon'\}$;
3 $\varepsilon \leftarrow \ln(\frac{n}{\tau}(e^{\varepsilon'_1} - 1) + 1)$;
/* Step 2: Sample and explore */
4 for $i = 1, 2, \ldots, k$ do
5     Sample a user $u_s^{(i)} \in U$, construct $I^{(i)}_{|u_s}$, and compute $\min(f(I^{(i)}_{|u_s}), C)$;
6 end
/* Step 3: Estimate the sum */
7 $\tilde{n} \leftarrow n + \text{Lap}(\frac{1}{\varepsilon_n})$;
8 $\mathcal{M}^{\mathcal{S}}(I) \leftarrow \frac{1}{k} \cdot \sum_{i=1}^{k} \left(\min(f(I^{(i)}_{|u_s}), C) + \text{Lap}(\frac{C}{\varepsilon})\right)$;
9 return $\tilde{f}(I) = \tilde{n} \cdot \mathcal{M}^{\mathcal{S}}(I)$;
```

The expected error thus satisfies

$$\begin{aligned}\mathbb{E}[|\tilde{f}(I) - f(I)|] \leq &\mathbb{E}\left[\left|\frac{\tilde{n}}{k} \cdot \sum_{i=1}^{k} \min(f(I^{(i)}_{|u_s}), C) - \check{f}(I, C)\right|\right] \\ &+ \sum_{u_s \in U} \max(f(I_{|u_s}) - C, 0) \\ &+ \frac{\tilde{n}}{k} \cdot C \cdot \mathbb{E}\left[\left|\sum_{i=1}^{k} \text{Lap}(\frac{1}{\varepsilon})\right|\right]\end{aligned}$$

where the first term is the sample variance, the second term is the bias, and the last term is the DP noise. For the last term, we further have

$$\mathbb{E}\left[\left|\sum_{i=1}^{k} \text{Lap}(\frac{1}{\varepsilon})\right|\right] \leq \mathbb{E}\left[\left|\sum_{i=1}^{k} \text{Lap}(\frac{1}{\varepsilon})\right|^2\right]^{\frac{1}{2}} = \sqrt{2k} \cdot \frac{1}{\varepsilon}.$$

We next try to find a good truncation threshold C to balance the bias and the DP noise, i.e., to minimize

$$g(I, C) = \sum_{u_s \in U} \max(f(I_{|u_s}) - C, 0) + \sqrt{\frac{2}{k}} \cdot \frac{\tilde{n}}{\varepsilon} \cdot C.$$

By taking the derivative of $g(I, C)$, the minimum is achieved when

$$\frac{\partial g(I, C)}{\partial C} = \sqrt{\frac{2}{k}} \cdot \frac{\tilde{n}}{\varepsilon} - |\{u_s \in U : f(I_{|u_s}) \geq C\}| = 0,$$

Table 1. Basic information of graph data.

Dataset	**Amazon**	**RoadnetCA**	**USRN**
Nodes	335,000	1,970,000	23,900,000
Edges	926,000	2,770,000	28,900,000
Maximum degree	549	12	9
Degree upper bound d	1,024	16	16
$\max_u \lvert\Lambda(I,u)\rvert$	290	11	9

Table 2. Basic information of TPC-H data.

Scale factor	**0.25**	**1**	**4**	**16**
Customers	37,500	150,000	600,000	2,400,000
Suppliers	2,500	10,000	40,000	160,000
$\max_u \lvert\Lambda(I,u)\rvert$	678	694	709	716
Upper bound τ	1024			

i.e., when C equals the $\sqrt{\frac{2}{k}} \cdot \frac{\tilde{n}}{\varepsilon}$-th largest $f(I_{|u_s})$. We can next approximate

$$\varepsilon' \approx \sqrt{2k\ln(\frac{1}{\delta_0})} \cdot \varepsilon_1', \varepsilon_1' \approx \eta \cdot \varepsilon, \tilde{n} \approx n.$$

The optimal truncation threshold C is thus the $\left(\sqrt{\ln(\frac{1}{\delta_0})} \cdot \frac{2\tau}{\varepsilon'}\right)$-th largest $f(I_{|u_s})$. However, we cannot use this exact C as it is also private information. Instead, we can choose a C based on some rough prior knowledge of the tail distribution of users' contributions.

6 EXPERIMENTS

To evaluate the efficiency and accuracy of our privacy amplification results, especially sample-and-explore, we conducted an extensive set of experiments with subgraph counting queries in real-world graph data [19, 28] under node-DP, as well as SQL queries on TPC-H data. For comparison, we also tested the following methods:

- Sample&Explore$_{\text{pub}}$ and SimpleSample$_{\text{pub}}$: These refer to the two sampling algorithms without adding privacy noise. These are meant to separate the sampling error from DP errors.
- The Laplace mechanism (Lap) and the R2T mechanism [10] without sampling: These apply to both subgraph counting and SQL queries.
- The recursive mechanism (RM) [8], the naive truncation with smooth sensitivity (NT) [22] and the smooth distance estimator (SDE) [6] without sampling: These are designed specifically for counting subgraphs under node-DP.

The experiments were conducted on a machine with a 2.2GHz Intel Xeon CPU and 256GB memory. We repeat each experiment 100 times, remove the best 20 and the worst 20 runs, and report the average error of the remaining runs. The default privacy budget is $\varepsilon_{\text{total}} = 4$ and $\delta_{\text{total}} = 10^{-8}$.

6.1 Setup

Datasets. We used 3 real-world graph datasets: **Amazon**, **RoadnetCA**, and **USRN**. **Amazon** is an Amazon product co-purchasing network. **RoadnetCA** is a road network of California and **USRN** is a road network of the U.S. The basic information of the graphs is given in Table 1. Similar to prior work [6, 10, 14], we choose the degree upper bound d to be higher than the actual maximum degree, as d should not depend on the instance.

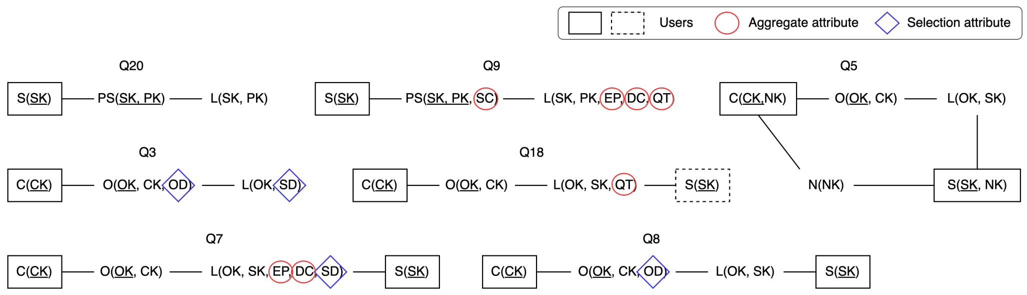


Fig. 4. Query structures.

The TPC-H dataset consists of eight relations: `Region(RK)`, `Nation(RK,NK)`, `Customer(NK,CK)`, `Orders(CK,OK)`, `Supplier(NK,SK)`, `Part(PK)`, `PartSupp(SK,PK)`, `Lineitem(SK,PK,OK,LN)`, where the underlined attributes are the primary keys. We use datasets with scale factors ranging from 0.25 to 16. The basic information is shown in Table 2.

Queries. For graph data, we count the number of length-2 paths, triangles, rectangles, and 2-triangles. For counting triangles ($\ell = 3$) and rectangles ($\ell = 4$), the mapping V is as defined in Section 1.2, i.e., $V(x) = \{1\}$ iff the users (nodes) appear in ascending order of their user id in x and they form a triangle or rectangle, respectively. For counting length-2 paths ($\ell = 3$), we use the following mapping: Suppose u_1-u_2-u_3 form a length-2 path, then we set $V(x) = \{1\}$ for $x = (u_2, u_1, u_3)$ such that $u_1 < u_3$. This way, $\tau = d$ is an upper bound on the number of neighbors any user u may have, which is needed for privacy amplification. Similarly, for counting 2-triangles ($\ell = 4$), suppose u_1, u_2, u_3, u_4 form a 2-triangle where (u_1, u_2, u_3) and (u_1, u_2, u_4) are both triangles. Then we set $V(x) = \{1\}$ for $x = (u_1, u_2, u_3, u_4)$ such that $u_1 < u_2, u_3 < u_4$. Then we can set $\tau = d$ as an upper bound on the number of neighbors any user u has.

For TPC-H data, we select 7 queries from the benchmark. We remove the group-by clause but keep all joins, selection conditions, and aggregations. The structures of the queries are shown in Figure 4. We view either customers, suppliers, or both, as users. When we only view customers or suppliers as users ($\ell = 1$), for each user u, V maps u to all the records contributed by u. When we view both as users ($\ell = 2$), we map each (customer, supplier) pair to all the records contributed by them.

Parameter settings. For `Sample&Explore`, we need an upper bound τ on the user contributions, and a truncation threshold C. For subgraph counting queries, we can verify that $\tau = d$ is a valid setting when we count subgraphs except for rectangles. For counting rectangles, a trivial setting is $\tau = O(d^2)$, however, such a τ is too large for us to obtain privacy amplification. Furthermore, we note that most real-world networks are not dense so that the exact $\max_u |\Lambda(V, u)|$'s, as shown in Table 1, can be well bounded by $\tau = d$. Thus, we use the same τ's for counting rectangles. For the truncation threshold C, we try different values $C = 2^i, i = 0, 1, \ldots$ and report the best one. The impact of the setting of C is shown in the last section of the experiments.

Both `R2T` and `Lap` require the global sensitivities GS_f's, which is computed from the degree upper bound d. For example, when we count triangles, we set $\mathrm{GS}_f = \frac{d(d-1)}{2}$. `NT` and `SDE` instead require a truncation threshold C (on the maximum degree in the truncated graph). Similar to `Sample&Explore`, we try different values $C = 2^i, i = 0, 1, \ldots$ and report the best one.

For `SimpleSample`, we adopt the Laplace mechanism as the DP mechanism due to its efficiency. To compute the global sensitivities for the samples, we first compute a degree upper bound d_s for the sample. For **RoadnetCA** and **USRN**, we always have $d_s = d$ as the degree upper bound d is small enough. For **Amazon**, we note that d is too large for the sample when a small sampling rate η is given. Therefore, we set $d_s = 2\eta \cdot d$ instead. All the global sensitivities GS_f's are then computed using the sampled degree upper bound d_s. For example, when we count triangles, we set $\mathrm{GS}_f = \frac{d_s(d_s-1)}{2} \approx 4\eta^2 \cdot \frac{d(d-1)}{2}$.

The parameter settings for the TPC-H dataset are similar. For `Sample&Explore`, when only customers or suppliers are viewed as users, we can verify that $\tau = 1$ is a valid setting. When both are viewed as users, $\max_u |\Lambda(I, u)|$'s for different scales are shown in Table 2, and we simply set $\tau = 1024$ for all scales. The clipping threshold C follows the same setting as before. Moreover, we assume we are given the prior knowledge that each user contributes to at most 10^3 records, and each record has weight as most 10^3. For `R2T` and `Lap`, the global sensitivity GS_f is thus 10^3 for counting queries and 10^6 for sum estimation queries. For `NT` and `SDE`, the clipping threshold C also follows the same setting as before. For `SimpleSample`, we scale the global sensitivity similarly as before. When customers or suppliers are viewed as users, we have $\mathrm{GS}_f = 10^3$ and 10^6 for counting and sum estimation queries, respectively, as all the records contributed by u_s are added to the sample. When both are viewed as users, the global sensitivity is $\mathrm{GS}_f = 2\eta \cdot 10^3$ and $\mathrm{GS}_f = 2\eta \cdot 10^6$ instead.

Implementation. As with most sampling-based approximate query processing systems [18, 24, 36], we pre-build necessary data structures to support sampling users and retrieve the contributions of the sampled user. In particular, for TPC-H data, we build an index on the primary key of each relation. This allows us to sample a user and retrieve all its contributions efficiently. For graph data, we store the graph in an adjacency list.

6.2 Experimental Results

For sampling-based algorithms, we are interested in the sample size-error trade-off. However, different sampling methods have different overheads, so we plot the time-error trade-off instead for a fair comparison. More precisely, in each experiment, we try different sample sizes (for sample-and-explore) or sampling rates (for simple sampling), measure the wall-clock time and the error, and plot the trade-off results. For non-sampling-based algorithms, their results just appear as single points in the plots. The errors are relative errors, so any error greater than 1 is meaningless, hence omitted.

Subgraph counting. The experimental results for various graph patterns are plotted in Figure 5. The first clear observation is that, in all cases, `Sample&Explore` is the first to return meaningful estimates in short running times, often 10^2 to 10^4 times faster than `SimpleSample`, and 10^2 to 10^6 times faster than the exact mechanisms. Of course, the latter can achieve a smaller error, as it only has DP noise but no sampling error. Anyway, the purpose of studying sampling-based DP mechanisms is to offer this time-error trade-off, which is useful for applications that desire a quick but not-so-accurate estimate. In fact, the two can be used in conjunction: The data analyst may first use some privacy budget to obtain a quick estimate, and if needed, a more accurate result can be computed, which would take (much) more time.

According to the analysis, we have shown that the error upper bound of our sampling mechanism is close to that of R2T, up to a factor of τ. Besides, the error of both R2T and our sampling mechanism consists of a bias term and a variance term. While the variance term approaches 0 as the sample size increases, the bias term does not, and the two mechanisms may have different biases.

Next, we compare the two sampling-based mechanisms. For length-2 paths and triangles, `SimpleSample` performs better than `Sample&Explore` on **Amazon** after sufficient time. This is

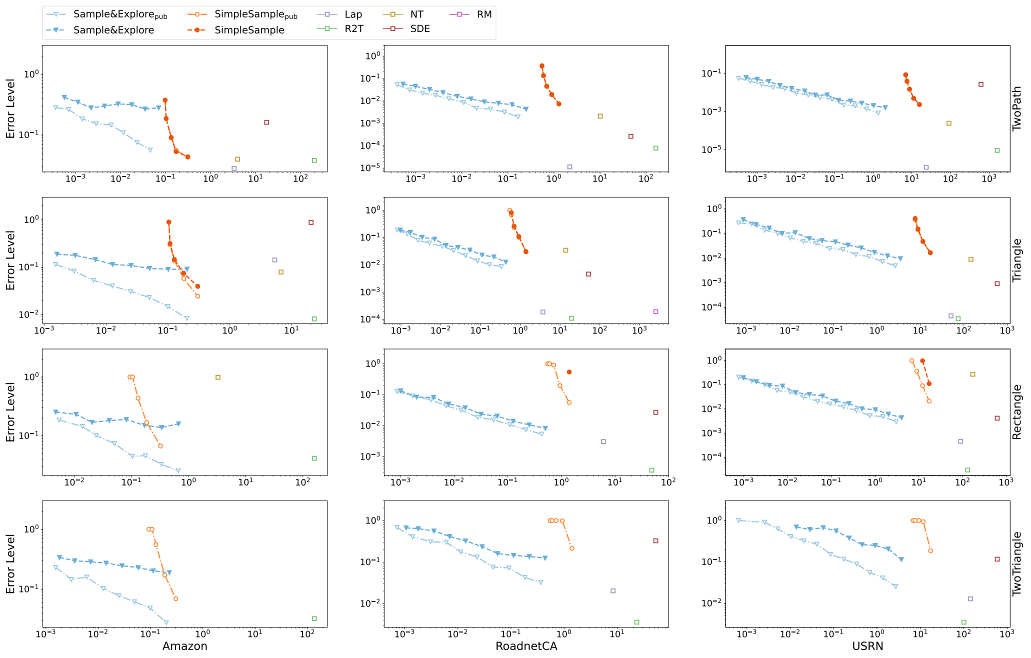


Fig. 5. Result of subgraph counting

because the pattern is simple and the graph is dense. For all other cases, `Sample&Explore` obtains a better time-error trade-off than that of `SimpleSample` or even `SimpleSample`$_{\text{pub}}$ especially in the early stage, i.e., small sample size k. This agrees with our analysis that, on sparse graphs, `Sample&Explore` has an advantage of $O((n/k)^{(\ell-1)/2})$ over `SimpleSample` in terms of sampling error, and $O((n/k)^{\ell-1}/\tau)$ in terms of DP noise.

We can also compare the sampling-based DP mechanisms with their non-private versions. We see that when the graph is sparse like **RoadnetCA** and **USRN**, the results of `Sample&Explore` are very close to those of `Sample&Explore`$_{\text{pub}}$, indicating a strong privacy amplification. The gap becomes larger for denser graphs like **Amazon**. In contrast, the gap between `SimpleSample` and `SimpleSample`$_{\text{pub}}$ depends on the sampling rate η and the global sensitivity GS_f. Thus, when the global sensitivity is small, e.g, when we count length-2 paths, the results of `SimpleSample` and `SimpleSample`$_{\text{pub}}$ almost coincide. Comparatively, when the global sensitivity is large, e.g., when we count rectangles, the gap can be so large that `SimpleSample` cannot achieve an error smaller by 1, hence omitted from the plot.

TPC-H queries. We conduct experiments on the datasets with scale 16, and the results are shown in Figure 6. With a time limit of 1000 seconds, `R2T` can only answer 6 of the 8 queries. The overall trends are similar to those on subgraph counting: `Sample&Explore` has a better error-time trade-off than `SimpleSample` in general, but `SimpleSample` may achieve better accuracy after sufficient time. It is also interesting to see the results from Q18 where we take the customers or both the customers and the suppliers as users. We can see that the error level increases for both sampling mechanisms when we protect the privacy of both customers and suppliers. For `Sample&Explore`, this is because the upper bound τ increases, and we obtain lower privacy amplification. For `SimpleSample`, this is

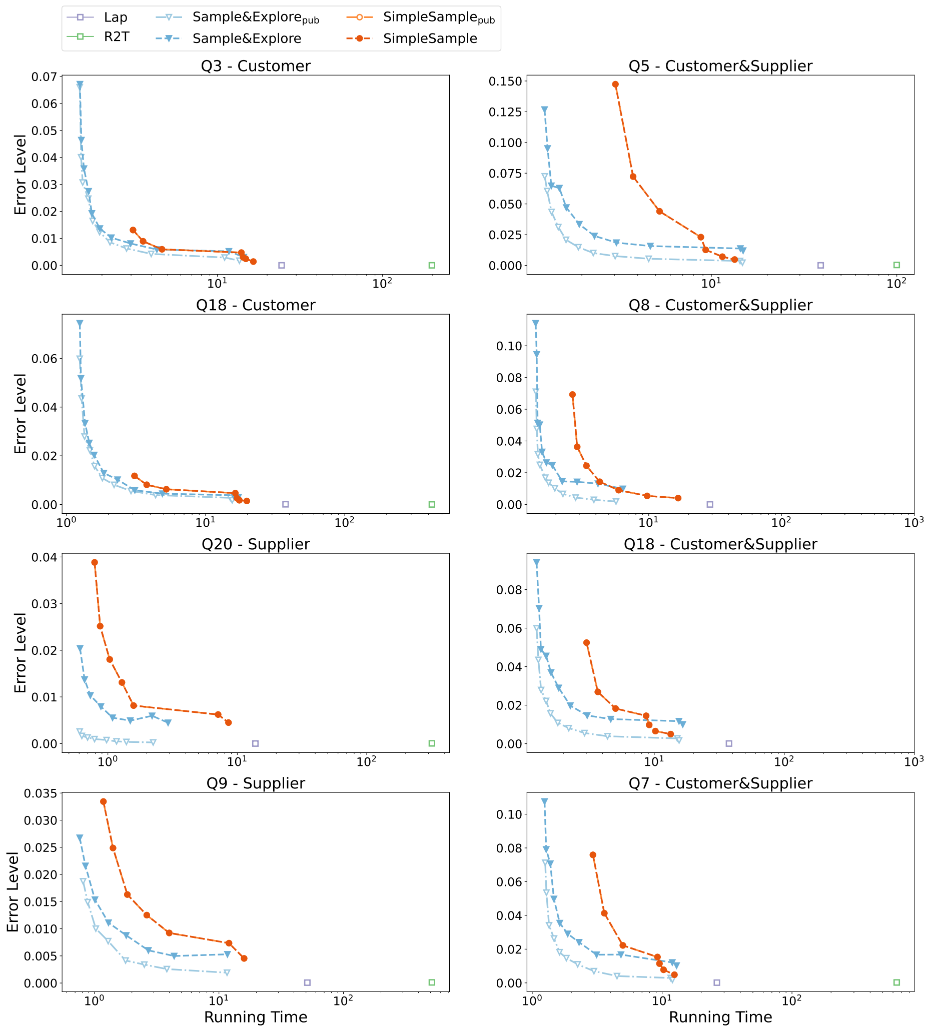


Fig. 6. Results of TPC-H queries

because a record only appears in the sample if the corresponding customer and supplier are both sampled, which is only with probability η^2. In addition, for R2T, the running time increases a lot due to the complex correlation of the users' contributions.

Sample size, accuracy and efficiency. In addition to plotting the error level against the running time, we have further generated plots that depict the error level and the running time in relation to the sample size for both Sample&Explore and SimpleSample. We use triangle counting as the problem and the results are shown in Figure 7. The results better illustrate why Sample&Explore outperforms SimpleSample, as shown in Figure 5. In terms of accuracy, Sample&Explore can achieve a relative error of less than 1 with a much smaller sample size, compared with SimpleSample, which aligns with our analysis of the sampling error. Moreover, in terms of efficiency, Sample&Explore and SimpleSample use operations of varying time costs. In Sample&Explore, the number of triangles

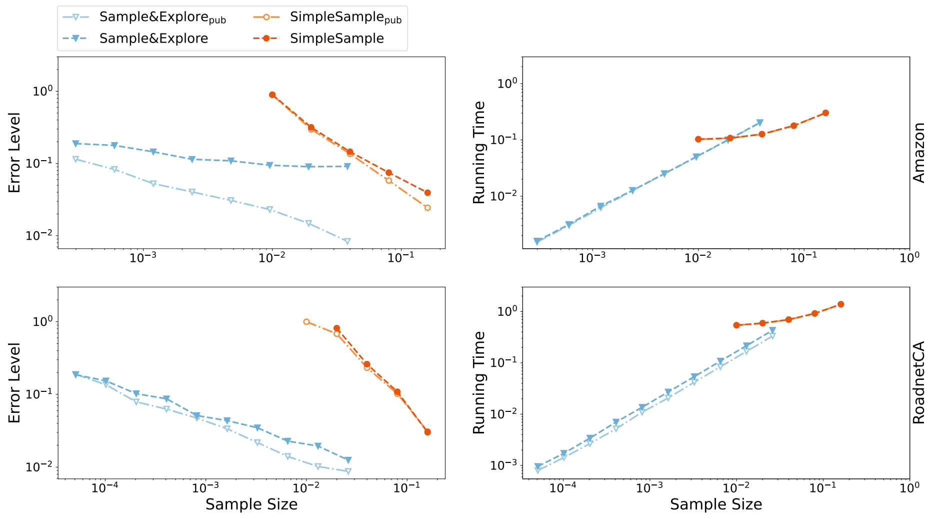


Fig. 7. Error level and running time of Sample&Explore and SimpleSample with respect to the sample size

contributed by a specific sampled node can be computed by accessing its one-hop neighbors. However, in SimpleSample, we need to construct the induced subgraph after sampling before determining the count. Therefore, both mechanisms experience varying speeds of increase in running time as the sample size grows, and for smaller sample sizes, Sample&Explore is more efficient compared to SimpleSample.

Scalability. We next analyze the impact of the size of the datasets on the results. We use Q3 and Q5 as examples, with dataset scales ranging from 0.25 to 16. The results are shown in Figure 8. We first note that the gap between Sample&Explore and Sample&Explore$_{pub}$ becomes smaller as the datasets become larger. This is because, in the TPC-H dataset, larger datasets lead to more users, and thus better privacy amplification. Therefore, we can infer that for larger datasets, Sample&Explore can still have good performance close to the ones of Sample&Explore$_{pub}$. In terms of accuracy, all mechanisms achieve smaller relative errors as the datasets become larger. However, in terms of efficiency, the running time of all mechanisms except Sample&Explore increases linearly with the size of the dataset, which makes these mechanisms inefficient on large datasets. In contrast, the running time of Sample&Explore is mainly dependent on the sample size. This enables Sample&Explore to work well on large datasets in a short time.

We also note that in general, the benefits of any sampling method only manifest when the scale is large enough. Unfortunately, it is hard to determine a specific threshold beyond which the sampling-based mechanisms would work better than the exact mechanisms because the running time of the latter varies a lot depending on the data. However, what is certain is that sampling-based methods will outperform exact mechanisms when the scale is large enough. For example, on the TPC-H data, R2T requires minutes to hours for different queries on a dataset with a scale of 16, and as the datasets grow larger, the running time exhibits a faster-than-linear increase. Consequently, for even larger scales, only sampling-based mechanisms will work in a reasonable amount of time.

Privacy budget ε. We further conduct experiments on how the privacy budget ε affects the performances of Sample&Explore and SimpleSample. We use triangle counting as the problem and the results are shown in Figure 9. As ε increases, we can see that the overall error either

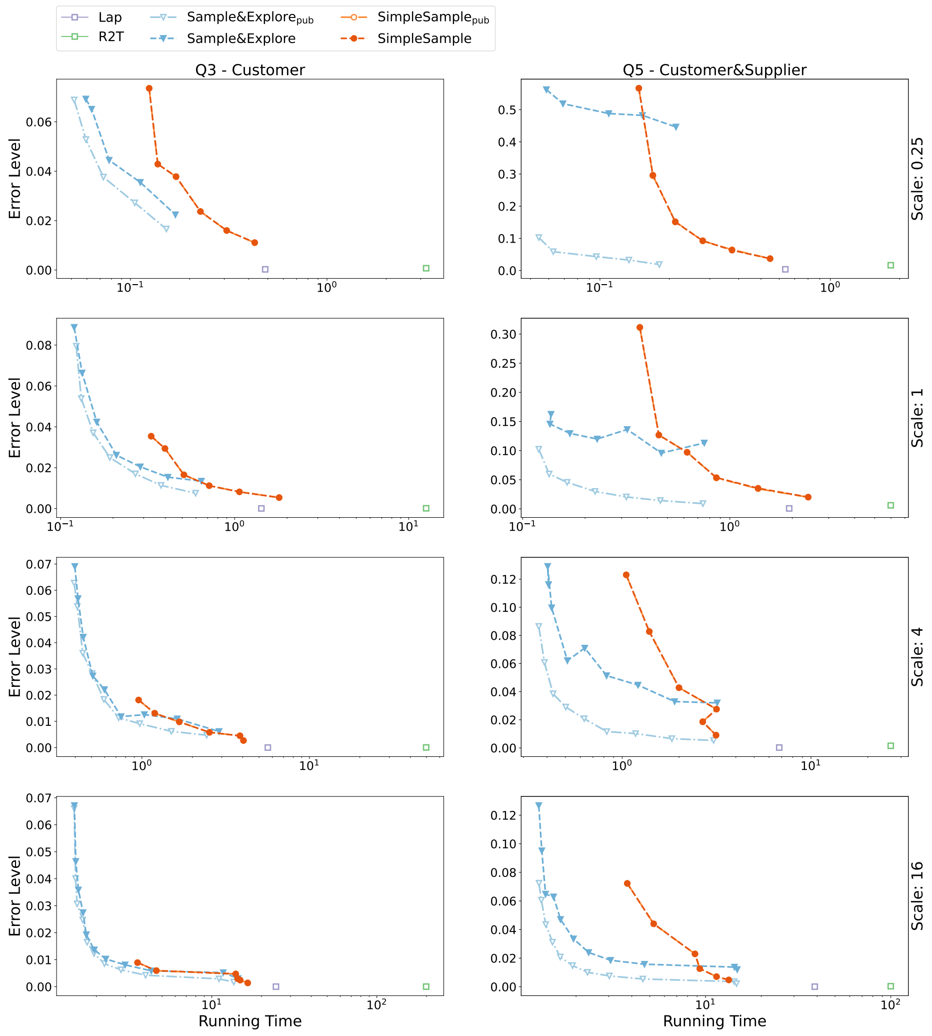


Fig. 8. Results of TPC-H queries on datasets of different scales

decreases or stays the same. This is because the privacy budget ε only affects the DP noise, but not the sample variance. More specifically, when the DP noise is much larger than the sample variance, a smaller DP noise leads to a smaller overall error. In contrast, when the sample variance is much larger than the DP noise, a smaller DP noise does not have much impact on the overall error. Note that the results are obtained at different times, so we cannot simply compare the errors in Figure 9.

Truncation threshold C. We use triangle counting as the problem and conduct experiments to investigate the effect of the truncation threshold C. We first verify the relationship between C and the sample size k. The results are shown in Figure 10, where the gray ones are the optimal thresholds as analyzed in Section 5.4. We can see that the analysis gives a good threshold. Moreover, the experimental results conform to the analysis that the optimal threshold is not affected by the sample size k much. Given that we have some prior knowledge of the tail distribution of users'

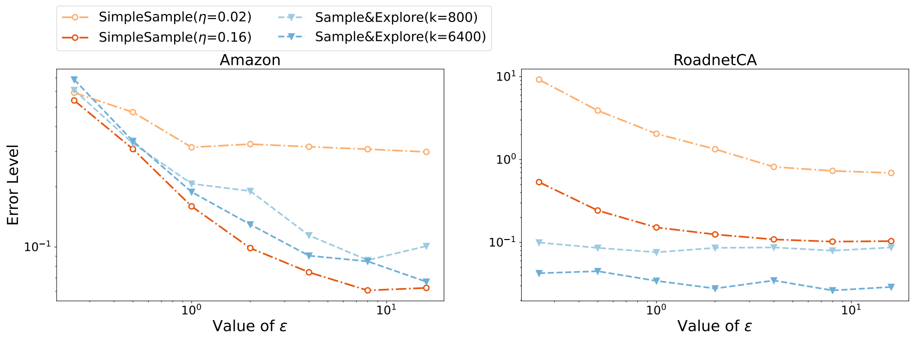


Fig. 9. Error level of Sample&Explore and SimpleSample with different privacy budget ε

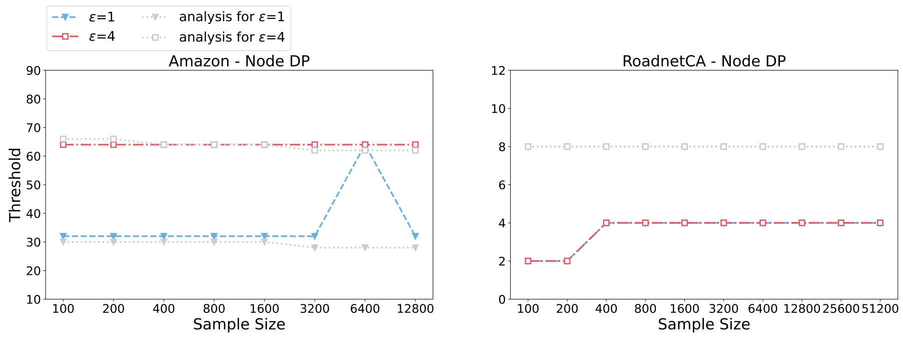


Fig. 10. The optimal threshold $C \in \{2^i : i = 1, 2, \cdots\}$ for Sample&Explore with respect to different sample size k

contributions, we can set a good C using the parameters. The difference is that the optimal threshold slightly increases as k increases while the analysis shows that it should not change. One possible reason is that, in the analysis, we do not consider the effect of the threshold C on the sample variance, however, in practice, a smaller C leads to a smaller sample variance.

7 FUTURE WORK

Both simple sampling and sample-and-explore sample the users. Another common sampling strategy in approximate query processing is to sample the records $R(V)$, which has been shown to offer better accuracy in terms of sampling errors. Note that the records $R(V)$ correspond to the join results, which are given implicitly. So sampling the records is equivalent to the problem of sampling over joins, which itself is already nontrivial [18, 24]. Privacy amplification for join sampling is thus a challenging but interesting problem worth further studying.

ACKNOWLEDGEMENTS

The research reported in this paper was supported by Hong Kong Research Grants Council under grants 16201819, 16205420, and 16205422. We appreciate the anonymous reviewers for their helpful comments on the manuscript.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. Association for Computing Machinery, New York, NY, USA, 308–318.

[2] Kareem Amin, Alex Kulesza, Andres Muñoz Medina, and Sergei Vassilvitskii. 2019. Bounding user contributions: A bias-variance trade-off in differential privacy. In *Proceedings of the 36th International Conference on Machine Learning*, Vol. 97. PMLR, 263–271.

[3] Borja Balle, Gilles Barthe, and Marco Gaboardi. 2018. Privacy Amplification by Subsampling: Tight Analyses via Couplings and Divergences. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Red Hook, NY, USA, 6280–6290.

[4] Borja Balle, Gilles Barthe, Marco Gaboardi, and Joseph Geumlek. 2019. Privacy Amplification by Mixing and Diffusion Mechanisms. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Red Hook, NY, USA, 13298–13308.

[5] Gilles Barthe and Federico Olmedo. 2013. Beyond Differential Privacy: Composition Theorems and Relational Logic for f-divergences between Probabilistic Programs. In *Automata, Languages, and Programming*. Springer Berlin Heidelberg, Berlin, Heidelberg, 49–60.

[6] Jeremiah Blocki, Avrim Blum, Anupam Datta, and Or Sheffet. 2013. Differentially Private Data Analysis of Social Networks via Restricted Sensitivity. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science*. Association for Computing Machinery, New York, NY, USA, 87–96.

[7] Mark Bun, Cynthia Dwork, Guy N. Rothblum, and Thomas Steinke. 2018. Composable and Versatile Privacy via Truncated CDP. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. Association for Computing Machinery, New York, NY, USA, 74–86.

[8] Shixi Chen and Shuigeng Zhou. 2013. Recursive mechanism: towards node differential privacy and unrestricted joins. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. Association for Computing Machinery, New York, NY, USA, 653–664.

[9] Wei-Yen Day, Ninghui Li, and Min Lyu. 2016. Publishing Graph Degree Distribution with Node Differential Privacy. In *Proceedings of the 2016 International Conference on Management of Data*. Association for Computing Machinery, New York, NY, USA, 123–138.

[10] Wei Dong, Juanru Fang, Ke Yi, Yuchao Tao, and Ashwin Machanavajjhala. 2022. R2T: Instance-optimal truncation for differentially private query evaluation with foreign keys. In *Proc. ACM SIGMOD International Conference on Management of Data*. Association for Computing Machinery, New York, NY, USA, 759–772.

[11] Wei Dong and Ke Yi. 2022. A Nearly Instance-optimal Differentially Private Mechanism for Conjunctive Queries. In *PODS*.

[12] Cynthia Dwork and Aaron Roth. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science* 9 (2014).

[13] Alessandro Epasto, Mohammad Mahdian, Jieming Mao, Vahab Mirrokni, and Lijie Ren. 2020. Smoothly bounding user contributions in differential privacy. In *Advances in Neural Information Processing Systems*.

[14] Juanru Fang, Wei Dong, and Ke Yi. 2022. Shifted Inverse: A General Mechanism for Monotonic Functions under User Differential Privacy. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. Association for Computing Machinery, New York, NY, USA, 1009–1022.

[15] Vitaly Feldman, Audra McMillan, and Kunal Talwar. 2022. Hiding Among the Clones: A Simple and Nearly Optimal Analysis of Privacy Amplification by Shuffling. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. 954–964.

[16] Vitaly Feldman, Audra McMillan, and Kunal Talwar. 2023. Stronger Privacy Amplification by Shuffling for Renyi and Approximate Differential Privacy. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 4966–4981.

[17] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. 2018. Privacy Amplification by Iteration. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. 521–532.

[18] Peter J. Haas and Joseph M. Hellerstein. 1999. Ripple Joins for Online Aggregation. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*. Association for Computing Machinery, New York, NY, USA, 287–298.

[19] Leskovec Jure and Krevl Andrej. 2016. SNAP datasets: Stanford large network dataset collection (2014). *URL http://snap. stanford. edu/data* (2016), 49.

[20] Vishesh Karwa, Sofya Raskhodnikova, Adam Smith, and Grigory Yaroslavtsev. 2011. Private analysis of graph structure. In *Proceedings of the VLDB Endowment*.

[21] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2008. What Can We Learn Privately?. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*. 531–540.

[22] Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2013. Analyzing graphs with node differential privacy. In *Proceedings of the 10th Theory of Cryptography Conference on Theory of Cryptography*. Springer-Verlag, Berlin, Heidelberg, 457–476.

[23] Ios Kotsogiannis, Yuchao Tao, Xi He, Maryam Fanaeepour, Ashwin Machanavajjhala, Michael Hay, and Gerome Miklau. 2018. PrivateSQL: a differentially private SQL query engine. In *Proceedings of the VLDB Endowment*.

[24] Feifei Li, Bin Wu, Ke Yi, and Zhuoyue Zhao. 2016. Wander Join: Online Aggregation via Random Walks. In *Proceedings of the 2016 International Conference on Management of Data*. Association for Computing Machinery, New York, NY, USA, 615–629.

[25] Ilya Mironov, Kunal Talwar, and Li Zhang. 2019. Rényi Differential Privacy of the Sampled Gaussian Mechanism. arXiv:1908.10530 [cs.LG]

[26] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2007. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*. 75–84.

[27] Sofya Raskhodnikova and Adam Smith. 2016. Lipschitz Extensions for Node-Private Graph Statistics and the Generalized Exponential Mechanism. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. 495–504.

[28] Ryan A. Rossi and Nesreen K. Ahmed. 2015. The Network Data Repository with Interactive Graph Analytics and Visualization. *URL https://networkrepository.com*.

[29] C. Seshadhri, Ali Pinar, and Tamara G. Kolda. 2013. Triadic Measures on Graphs: The Power of Wedge Sampling. In *Proceedings of the 2013 SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics.

[30] Yuchao Tao, Xi He, Ashwin MacHanavajjhala, and Sudeepa Roy. 2020. Computing local sensitivities of counting queries with joins. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. Association for Computing Machinery, New York, NY, USA, 479–494.

[31] Charalampos E. Tsourakakis, U. Kang, Gary L. Miller, and Christos Faloutsos. 2009. DOULION: Counting Triangles in Massive Graphs with a Coin. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, New York, NY, USA, 837–846.

[32] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. 2019. Amplification by Shuffling: From Local to Central Differential Privacy via Anonymity. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, USA, 2468–2479.

[33] David Vengerov, Andre Cavalheiro Menck, Mohamed Zait, and Sunil P. Chakkappen. 2015. Join Size Estimation Subject to Filter Conditions. *Proc. VLDB Endow.* 8 (2015).

[34] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. 2019. Subsampled Rényi Differential Privacy and Analytical Moments Accountant. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. 1226–1235.

[35] Royce J Wilson, Celia Yuxin Zhang, William Lam, Damien Desfontaines, Daniel Simmons-Marengo, and Bryant Gipson. 2020. Differentially private SQL with bounded user contribution. In *Proceedings on Privacy Enhancing Technologies Symposium*.

[36] Bin Wu, Ke Yi, and Zhenguo Li. 2016. Counting Triangles in Large Graphs by Random Sampling. In *IEEE Transactions on Knowledge and Data Engineering*. 2013–2026.

[37] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. 2015. Private release of graph statistics using ladder functions. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*. 731–745.

[38] Yuqing Zhu and Yu-Xiang Wang. 2019. Poission Subsampled Rényi Differential Privacy. In *Proceedings of the 36th International Conference on Machine Learning*. 7634–7642.

Received July 2023; revised October 2023; accepted November 2023