# LF-GDPR: A Framework for Estimating Graph Metrics With Local Differential Privacy

Qingqing Ye, *Member, IEEE*, Haibo Hu, *Senior Member, IEEE*, Man Ho Au, *Member, IEEE*, Xiaofeng Meng, *Member, IEEE*, and Xiaokui Xiao, *Member, IEEE*

**Abstract**—Local differential privacy (LDP) is an emerging technique for privacy-preserving data collection without a trusted collector. Despite its strong privacy guarantee, LDP cannot be easily applied to real-world graph analysis tasks such as community detection and centrality analysis due to its high implementation complexity and low data utility. In this paper, we address these two issues by presenting LF-GDPR, the first LDP-enabled graph metric estimation framework for graph analysis. It collects two atomic graph metrics—the adjacency bit vector and node degree—from each node locally. LF-GDPR simplifies the job of implementing LDP-related steps (e.g., local perturbation, aggregation and calibration) for a graph metric estimation task by providing either a complete or a parameterized algorithm for each step. To address low data utility of LDP, it optimally allocates privacy budget between the two atomic metrics during data collection. To demonstrate the usage of LF-GDPR, we show use cases on two common graph analysis tasks, namely, clustering coefficient estimation and community detection. The privacy and utility achieved by LF-GDPR are verified through theoretical analysis and extensive experimental results.

**Index Terms**—Local differential privacy, graph metric, privacy-preserving graph analysis

## 1 Introduction

With the prevalence of big data and machine learning, graph analytics has received great attention and nurtured numerous applications in web, social network, transportation, and knowledge base. However, recent privacy incidents, particularly the Facebook privacy scandal, pose real-life threats to any *centralized* party who needs to safeguard graph data of individuals while providing graph analysis service to third parties. In that scandal, a third-party developer Cambridge Analytica retrieves the personal profiles of 87 million Facebook users through the Facebook Graph API for third-party apps [1], [2]. The main cause is that this API allows these apps to access the *friends list* of a user by a simple authorization, through which these apps propagate like virus in the social network. Unfortunately, most existing privacy models on graph assume a centralized trusted party to release the graph data that satisfies certain privacy metrics, for example, the $k$-neighborhood anonymity [3], $k$-degree anonymity [4], $k$-automorphism [5], $k$-isomorphism [6], and differential privacy [7], [8]. However, in practice even Facebook cannot be fully trusted or is in the centralized position to release graph data on behalf of each user. For decentralized graphs in which each user or party locally maintains a limited view of the graph, there is even no such a central party. These graphs, such as the World Wide Web, federated knowledge graphs, peer-to-peer (e.g., vehicular and mobile ad-hoc) and blockchain networks, and contact tracing graph for COVID-19, are in a more compelling need to find alternative privacy models without a trusted party [9].

A promising model is local differential privacy (LDP) [10], where each individual user *locally perturbs her share of graph metrics* (e.g., node degree and adjacency list, depending on the graph analysis task) before sending them to the data collector for analysis. As such, the data collector does not need to be trusted. A recent work *LDPGen* [11] has also shown the potential of LDP for graph analytics. In that work, LDP is used to collect node degree for synthetic graph generation. However, such solution is usually task specific—for different tasks, such as centrality analysis and community detection, dedicated LDP solutions must be designed from scratch. To show how complicated it is, an LDP solution usually takes four steps: (1) selecting graph metrics to collect from users for the target metric (e.g., clustering coefficient, modularity, or centrality) of this task, (2) designing a local perturbation algorithm for users to report these metrics under LDP, (3) designing a collector-side aggregation algorithm to estimate the target metric based on the perturbed data, (4) designing an optional calibration algorithm for the target metric if the estimation is biased. Step (4) is important as locally perturbed data often causes bias (i.e., deviation from the true mean) in the collector-side


- *Qingqing Ye is with the Department of Electronic and Information Engineering, Hong Kong Polytechnic University, Hong Kong. This work was completed when Qingqing Ye was with Renmin University of China, Beijing 100872, China. E-mail: qqing.ye@polyu.edu.hk.*
- *Haibo Hu is with the Department of Electronic and Information Engineering, Hong Kong Polytechnic University, Hong Kong , and also with the PolyU Shenzhen Research Institute, Shenzhen 518057, China. E-mail: haibo.hu@polyu.edu.hk.*
- *Man Ho Au is with the Department of Computer Science, The University of Hong Kong, Hong Kong. E-mail: allenau@cs.hku.hk.*
- *Xiaofeng Meng is with the School of Information, Renmin University of China, Beijing 100872, China. E-mail: xfmeng@ruc.edu.cn.*
- *Xiaokui Xiao is with the School of Computing, National University of Singapore, Singapore 119077. E-mail: xkxiao@nus.edu.sg.*


*Manuscript received 27 Feb. 2020; revised 18 Nov. 2020; accepted 16 Dec. 2020. Date of publication 24 Dec. 2020; date of current version 12 Sept. 2022. (Corresponding author: Haibo Hu.)*
*Recommended for acceptance by Benjamin C. M. Fung.*
*Digital Object Identifier no. 10.1109/TKDE.2020.3047124*

statistics. Obviously, working out such a solution *requires in-depth knowledge of LDP*, which hinders the embrace of LDP by more graph applications.

In this paper, we address this challenge by presenting Local Framework for Graph with Differentially Private Release (LF-GDPR), the first LDP-enabled graph metric estimation framework for general graph analysis. It simplifies the job of a graph application to design an LDP solution for a graph metric estimation task by providing complete or parameterized algorithms for steps (2)-(4) as above. As long as the target graph metric can be derived from the two atomic metrics, namely, the adjacency bit vector and node degree, the parameterized algorithms in steps (2)-(4) can be completed with ease. Furthermore, LF-GDPR features an optimal allocation of privacy budget between the two atomic metrics. To illustrate the usage of LF-GDPR, we will also show use cases on two common graph analysis tasks, namely, clustering coefficient estimation and community detection. To summarize, our main contributions in this paper are as follows.

1) This is the first LDP-enabled graph metric estimation framework for a variety of graph analysis tasks.
2) We provide complete or parameterized algorithms for local perturbation, collector-side aggregation, and calibration.
3) We present an optimal solution to allocate the privacy budget between adjacency bit vector and node degree.
4) We show two use cases of LF-GDPR and compare their performance with existing methods on real datasets.

The rest of the paper is organized as follows. Section 2 introduces preliminaries on local differential privacy and its application in graph analytics. Section 3 presents an overview of LF-GDPR. Section 4 describes the implementation details of this framework. Sections 5 and 6 show the detailed usage of LF-GDPR in two use cases. Section 7 presents the experimental results, followed by Section 8 which reviews related work. Section 9 draws a conclusion with future work.

## 2 Preliminaries

### 2.1 Local Differential Privacy

Differential privacy [12] (DP) is defined on a randomized algorithm $\mathcal{A}$ of a sensitive database. $\mathcal{A}$ is said to satisfy $\epsilon$-differential privacy, if for any two neighboring databases $D$ and $D'$ that differ only in one tuple, and for any possible output $s$ of $\mathcal{A}$, we have $\frac{\Pr[\mathcal{A}(\mathrm{D})=\mathrm{s}]}{\Pr[\mathcal{A}(\mathrm{D}')=\mathrm{s}]} \leq e^{\epsilon}$. In essence, DP guarantees that after observing any output of $\mathcal{A}$, an adversary cannot infer with high confidence whether the input database is $D$ or $D'$, thus hiding the existence or non-existence of any individual tuple.

Centralized DP requires the real database stored in a trusted server where the randomized algorithm $\mathcal{A}$ can execute. However, this assumption does not hold in many real-world applications. *Local differential privacy* [10], [13] is proposed to assume each individual is responsible for her own tuple in the database. In LDP, each user locally perturbs her tuple using a randomized algorithm before sending it to the untrusted data collector. Formally, a randomized algorithm $\mathcal{A}$ satisfies $\epsilon$-local differential privacy, if for any two input tuples $t$ and $t'$ and for any output $t^*$, $\frac{\Pr[\mathcal{A}(\mathrm{t})=\mathrm{t}^*]}{\Pr[\mathcal{A}(\mathrm{t}')=\mathrm{t}^*]} \leq e^{\epsilon}$ holds. In essence, LDP guarantees that after observing any output tuple $t^*$, the untrusted data collector cannot infer with high confidence whether the input tuple is $t$ or $t'$.

### 2.2 Local Differential Privacy on Graphs

In this paper, a graph $G$ is defined as $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$ is the set of nodes, and $E \subseteq V \times V$ is the set of edges. For the node $i$, $d_i$ denotes its degree and $\boldsymbol{B}_i = \{b_1, b_2, \ldots, b_n\}$ denotes its *adjacency bit vector*, where $b_j = 1$ if and only if edge $(i, j) \in E$, and otherwise $b_j = 0$. The adjacency bit vectors of all nodes constitute the *adjacency matrix* of graph $G$, or formally, $\boldsymbol{M}_{n\times n} = \{\boldsymbol{B}_1, \boldsymbol{B}_2, \ldots, \boldsymbol{B}_n\}$.

As with existing LDP works, we concern attacks where an adversary can infer with high confidence whether an edge exists or not, which compromises a user's relation anonymity in a social network. As a graph has both nodes and edges, LDP can be applied to either of them, which leads to *node local differential privacy* [14] and *edge local differential privacy* [15]. Node LDP (resp. edge LDP) guarantees the output of a randomized algorithm does not reveal whether any individual node (resp. edge) exists in $G$.

**Definition 2.1. (Node local differential privacy).** *A randomized algorithm $\mathcal{A}$ satisfies $\epsilon$-node local differential privacy (a.k.a., $\epsilon$-node LDP), if and only if for any two adjacency bit vectors $\boldsymbol{B}, \boldsymbol{B}'$ and any output $s \in range(\mathcal{A})$, $\frac{\Pr[\mathcal{A}(\boldsymbol{B})=s]}{\Pr[\mathcal{A}(\boldsymbol{B}')=s]} \leq e^{\epsilon}$ holds.*

**Definition 2.2. (Edge local differential privacy).** *A randomized algorithm $\mathcal{A}$ satisfies $\epsilon$-edge local differential privacy (a.k.a., $\epsilon$-edge LDP), if and only if for any two adjacency bit vectors $\boldsymbol{B}$ and $\boldsymbol{B}'$ that differ only in one bit, and any output $s \in range(\mathcal{A})$, $\frac{\Pr[\mathcal{A}(\boldsymbol{B})=s]}{\Pr[\mathcal{A}(\boldsymbol{B}')=s]} \leq e^{\epsilon}$ holds.*

Both node and edge LDP satisfy sequential composition.

**Theorem 2.3. (Sequential Composition)** *[11]. Given $c$ randomized algorithms $\mathcal{A}_i (1 \leq i \leq c)$, each satisfying $\epsilon_i$-node (resp. edge) LDP, the collection of these algorithms $\mathcal{A}_i (1 \leq i \leq c)$ satisfies $(\sum \epsilon_i)$-node (resp. edge) LDP.*

Edge-LDP is a relaxation of node-LDP, which limits the definition of neighbors from any two adjacency bit vectors to those that differ only in one bit (i.e., one edge). Nonetheless, edge-LDP can still achieve strong indistinguishability of each edge's existence, which suffices for many graph applications such as social networks while preserving high utility [14]. As such, in this paper we assume edge-LDP as with all existing graph LDP works.

## 3 LF-GDPR: Framework Overview

In this section, we first introduce the rationale behind LF-GDPR for privacy-preserving graph analytics and then overview its workflow. Finally, we introduce two use cases of LF-GDPR.

### 3.1 Design Principle

The core of privacy-preserving graph analytics often involves *estimating some target graph metric* without accessing the original graph. Under the DP/LDP privacy model,

TABLE 1
Popular Graph Analysis Tasks and Metrics

| Graph Analysis Task | Graph Metric Concerned | Derivation from $B$, $M$, and $D$ |
|---|---|---|
| synthetic graph generation | clustering coefficient | $cc_i = \frac{M_{ii}^3}{d_i(d_i-1)}$ |
| community detection, graph clustering | modularity | $Q_c = \frac{\|M_c\|}{\|D\|} - \frac{\|D_c\|^2}{\|D\|^2}$ |
| node role, page rank | degree centrality | $c_i = d_i$ |
| | eigenvector centrality | $c_i = B_i M^k$ |
| connectivity analysis (clique / hub) | structural similarity | $\tau(i,j) = \frac{\|B_i \cap B_j\|}{\sqrt{d_i d_j}}$ |
| node similarity search | cosine similarity | $\tau(i,j) = \frac{B_i B_j'}{\sqrt{d_i d_j}}$ |

there are two solution paradigms, namely, generating a synthetic graph to calculate this metric [11], [16], [17], [18], [19] and designing a dedicated DP/LDP solution for such metric [7], [14], [20], [21], [22]. The former provides a general solution but suffers from low estimation accuracy as *the neighborhood information in the original graph is missing* from the synthetic graph. The latter can achieve higher estimation accuracy but cannot generalize such a dedicated solution to other problems—it works poorly or even no longer works if the target graph metric or graph type (e.g., undirected graph, attributed graph, and DAG) is changed [8], [18].

LF-GDPR is our answer to both solution generality and estimation accuracy under the LDP model. It collects from each node $i$ two atomic graph metrics that can derive a wide range of common metrics. The first is the *adjacency bit vector* $\boldsymbol{B}$, where each element $j$ is 1 only if $j$ is a neighbor of $i$. $\boldsymbol{B}$ of all nodes collectively constitutes the adjacency matrix $\boldsymbol{M}$ of the graph. The second metric is *node degree vector* $\boldsymbol{D} = \{d_1, d_2, \ldots, d_n\}$, which is frequently used in graph analytics to measure the density of connectivity [21]. Table 1 lists some of the most popular graph analysis tasks in the literature [23], [24], [25] and their graph metrics, all of which can be derived from $\boldsymbol{B}$, $\boldsymbol{M}$, and $\boldsymbol{D}$.

Intuitively, for each node, $d$ can be estimated from $\boldsymbol{B}$. However, given a large graph and limited privacy budget, the estimation accuracy could be too noisy to be meaningful. To illustrate this, let us assume each bit of the adjacency bit vector $\boldsymbol{B}$ is perturbed independently by the classic Randomized Response (RR) [26] algorithm with privacy budget $\epsilon$. As stated in [26], the variance of the estimated node degree $\widetilde{d}$ is

$$Var[\widetilde{d}] = n \cdot \left[ \frac{1}{16(\frac{e^\epsilon}{e^\epsilon+1} - \frac{1}{2})^2} - \left(\frac{d}{n} - \frac{1}{2}\right)^2 \right]. \quad (1)$$

Even for a moderate social graph with extremely large privacy budget, for example, $d = 100$, $n = 1M$, and $\epsilon = 8$ (the largest $\epsilon$ used in [11] is 7), $Var[\widetilde{d}] \approx 435 > 4d$, which means the variance of the estimated degree is over 4 times that of the degree itself. As such, we choose to spend some privacy budget on an independently perturbed degree. This further motivates us to design an optimal privacy budget allocation between adjacency bit vector $\boldsymbol{B}$ and node degree $d$, to minimize the distance between the target graph metric and the estimated one.

To summarize, in LF-GDPR each node sends two perturbed atomic metrics, namely, the adjacency bit vector $\widetilde{\boldsymbol{B}}$ (perturbed from $\boldsymbol{B}$) and node degree $\widetilde{d}$ (perturbed from $d$), to the data collector, who then aggregates them to estimate the target graph metric.

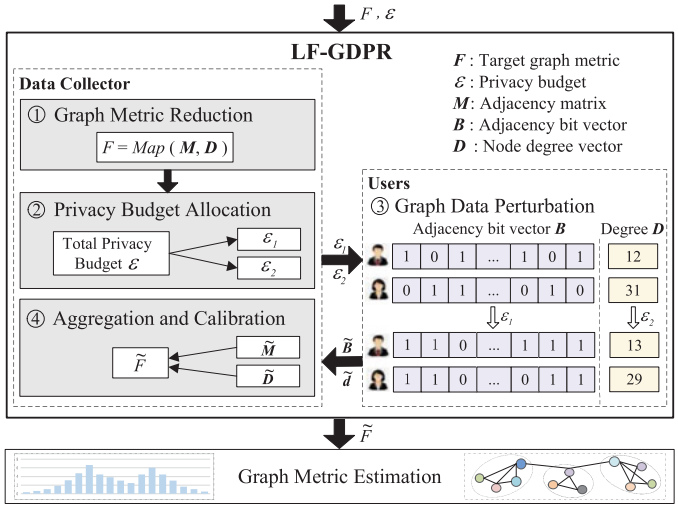



Fig. 1. An overview of LF-GDPR.

### 3.2 LF-GDPR Overview

LF-GDPR works as shown in Fig. 1. A data collector who wishes to estimate a target graph metric $F$ first reduces it from the adjacency matrix $\boldsymbol{M}$ and node degree vector $\boldsymbol{D}$ of all nodes by deriving a mapping function $F = Map(\boldsymbol{M}, \boldsymbol{D})$ (step ①). Based on this reduction, LF-GDPR optimally allocates the total privacy budget $\epsilon$ between $\boldsymbol{M}$ and $\boldsymbol{D}$, denoted by $\epsilon_1$ and $\epsilon_2$, respectively (step ②). Then each node locally perturbs its adjacency bit vector $\boldsymbol{B}$ into $\widetilde{\boldsymbol{B}}$ to satisfy $\epsilon_1$-edge LDP, and perturbs its node degree $d$ into $\widetilde{d}$ to satisfy $\epsilon_2$-edge LDP (step ③). According to the composability of LDP, each node then satisfies $\epsilon$-edge LDP. Note that this step is challenging as both $\boldsymbol{B}$ and $d$ are correlated among nodes. For $\boldsymbol{B}$, the $j$th bit of node $i$'s adjacency bit vector is the same as the $i$th bit of node $j$'s adjacency bit vector. For $d$, whether $i$ and $j$ has an edge affects both degrees of $i$ and $j$. Sections 4.2 and 4.3 solve this issue and send out the perturbed $\boldsymbol{B}$ and $d$, i.e., $\widetilde{\boldsymbol{B}}$ and $\widetilde{d}$. The data collector receives them from all nodes, aggregates them according to the mapping function $Map(\cdot)$ to obtain the estimated target metric $\widetilde{F}$, and further calibrates it to suppress estimation bias and improve accuracy (step ④). The resulted $\widetilde{F}$ is then used for graph analysis. The detailed implementation of LF-GDPR for steps ①②③④ will be presented in Section 4. Note that the algorithms in steps ①②④ are parameterized, which can only be determined when the target graph metric $F$ is specified.

**Example 3.2. LF-GDPR against Facebook Privacy Scandal.** Facebook API essentially controls how a third-party app accesses the data of each individual user. To limit the access right of an average app (e.g., the one developed by Cambridge Analytica) while still supporting graph analytics, Facebook API should have a new permission rule that only allows such app to access the perturbed adjacency bit vector and degree of a user's friends list under $\epsilon_1$ and $\epsilon_2$-edge LDP, respectively. In the Cambridge Analytica case, the app is a personality test, so the app developer may choose structural similarity as the target graph metric and use the estimated value for the personality test. To estimate structural similarity, the app then implements steps ①②④ of LF-GDPR. On the user side, each user $u$ has a privacy budget $\epsilon_u$ for her friends list.

If $\epsilon_u \geq \epsilon_1 + \epsilon_2$, the user can grant access to this app for perturbed adjacency bit vector and degree; otherwise, the user simply ignores this access request.

### 3.3 Two Cases of Graph Analytics Using LF-GDPR

To illustrate LF-GDPR, we show two use cases throughout this paper. In this subsection, we introduce their background and target graph metrics $F$. Their usage details, including the reduction of $F$ (step ①), the optimal privacy budget allocation (step ②), and the aggregation and calibration (step ④), are presented in Sections 5 and 6 respectively.

#### 3.3.1 Clustering Coefficient Estimation

The clustering coefficient of a node measures the connectivity in its *neighborhood*, i.e., the subgraph of its neighbors. Formally, the clustering coefficient $cc_i$ of node $i$ is defined as

$$cc_i = \frac{2t_i}{d_i(d_i-1)},$$

where $t_i$ denotes the number of edges in the neighborhood of node $i$, or equivalently, the number of triangles incident to node $i$. A clustering coefficient is in the range of [0,1], and a high value indicates its neighbors tend to directly connect to each other. It is an important measure of graph structure, and is widely used in graph analytics. For example, the graph model BTER [11], [27] needs clustering coefficient (as well as node degree) to generate a synthetic graph. As it depends on the neighborhood information and thus cannot be calculated locally in each node, existing LDP techniques for values, such as [28], [29], [30], cannot work. The detailed solution by LF-GDPR will be shown in Section 5.

#### 3.3.2 Modularity Estimation and Community Detection

Communities (i.e., densely connected subgraphs) are commonly used in graph analytics to understand the underlying structure of a graph. The criterion of a good community is similar to a graph partition—with many intra-community edges and only a few inter-community edges. Many popular community detection methods are based on modularity maximization [31], which iteratively improves modularity, a widely-adopted metric to measure the quality of detected communities. Formally, the modularity $Q$ of a graph is defined as the sum of individual modularities $q_c$ of all communities $\mathcal{C}$

$$Q = \sum_{c=1}^{r} q_c = \sum_{c=1}^{r} \left[ \frac{L_c}{L} - \left( \frac{K_c}{2L} \right)^2 \right], \tag{2}$$

where $r$ is the number of communities in the graph, $L$ is the total number of edges, $L_c$ is the total number of edges in community $\mathcal{C}$, and $K_c$ is the total degree of all nodes in $\mathcal{C}$. $Q$ is in the range of $[-1, 1]$, where a higher value is more desirable. As with clustering coefficient, neither individual nor overall modularity can be estimated by dedicated LDP techniques which do not send the adjacency bit vectors. Section 6 will elaborate on how to use LF-GDPR to estimate it.

## 4 LF-GDPR: Implementation

In this section, we present the implementation details of LF-GDPR. We first discuss graph metric reduction (step ①), followed by the perturbation protocols for adjacency bit vector and node degree, respectively (step ③). Then we elaborate on the aggregation and calibration algorithm (step ④). Finally, we present the optimal allocation of privacy budget between adjacency bit vector and node degree (step ②).

### 4.1 Graph Metric Reduction

The reduction outputs a polynomial mapping function $Map(\cdot)$ from the target graph metric $F$ to the adjacency matrix $\boldsymbol{M} = \{\boldsymbol{B}_1, \boldsymbol{B}_2, \ldots, \boldsymbol{B}_n\}$ and degree vector $\boldsymbol{D} = \{d_1, d_2, \ldots, d_n\}$, i.e., $F = Map(\boldsymbol{M}, \boldsymbol{D})$. Without loss of generality, we assume $F$ is a polynomial of $\boldsymbol{M}$ and $\boldsymbol{D}$. That is, $F$ is a sum of terms $F_l$, each of which is a multiple of $\boldsymbol{M}$ and $\boldsymbol{D}$ of some exponents. Since $F$ and $F_l$ are scalars, in each term $F_l$, we need functions $f$ and $g$ to transform $\boldsymbol{M}$ and $\boldsymbol{D}$ with exponents to scalars, respectively. Formally,

$$F = \sum\nolimits_l F_l = \sum\nolimits_l f_{\phi_l}(\boldsymbol{M}^{k_l}) \cdot g_{\psi_l}(\boldsymbol{D}), \tag{3}$$

where $\boldsymbol{M}^{k_l}$ is the $k_l$th power of adjacency matrix $\boldsymbol{M}$ whose cell $(i, j)$ denotes the number of paths between node $i$ and $j$ of length $k_l$, $\phi_l$ projects a matrix to a cell, a row, a column or a sub-matrix, and $f_{\phi_l}(\cdot)$ denotes an aggregation function $f$ (e.g., sum) after projection $\phi_l$. Likewise, $\psi_l$ projects a vector to a scalar or a sub-vector, and $g_{\psi_l}(\cdot)$ denotes an aggregation function $g$ after $\psi_l$.

As such, the metric reduction step is to determine $k_l$, $f_{\phi_l}(\cdot)$, and $g_{\psi_l}(\cdot)$ for each term $F_l$ in Eq. (3).

### 4.2 Adjacency Bit Vector Perturbation

An intuitive approach, known as *Randomized Neighbor List* (*RNL*) [11], perturbs each bit of the vector independently by the classic Randomized Response [26]. Formally, given an adjacency bit vector $\boldsymbol{B} = \{b_1, b_2, \ldots, b_n\}$, and privacy budget $\epsilon_1$, the perturbed vector $\widetilde{\boldsymbol{B}} = \{\widetilde{b}_1, \widetilde{b}_2, \ldots, \widetilde{b}_n\}$ is obtained as follows:

$$\widetilde{b}_i = \begin{cases} b_i & \text{w.p. } \frac{e^{\epsilon_1}}{1+e^{\epsilon_1}} \\ 1 - b_i & \text{w.p. } \frac{1}{1+e^{\epsilon_1}} \end{cases}. \tag{4}$$

Note that here basic RR rather than OUE [32] is adopted. This is because adjacency bit vector is a binary vector, and according to [33], RR can achieve better accuracy than OUE.

Note that in Eq. (4), the probability of preserving an edge (bit '1') or non-edge (bit '0'), i.e., $p = \frac{e^{\epsilon_1}}{1+e^{\epsilon_1}}$, is not proportional to the amount of edge information disclosed to the collector. In fact, the success rate of the collector inferring an observed edge is a true edge is $\frac{\gamma p}{\gamma p + (1-\gamma)(1-p)}$, where $\gamma$ is the edge density in a graph. Although the edge density $\gamma$ is not considered in the definition of edge LDP, but it contributes to the posterior probability for the collector to infer the truth from an observed edge or non-edge. As such, a high edge density $\gamma$ also plays an important role in raising the success rate. But it is normally very small in social networks, and furthermore, such statics are generally not precisely owned by the collector.

*RNL* is proved to satisfy $\epsilon_1$-edge LDP for each user. However, *for undirected graphs,* RNL *can only achieve* $2\epsilon_1$*-edge LDP for the collector*, because the data collector witnesses the same edge perturbed twice and independently. Let $\widetilde{\boldsymbol{M}} = \{\widetilde{\boldsymbol{B}}_1, \widetilde{\boldsymbol{B}}_2, \ldots, \widetilde{\boldsymbol{B}}_n\}$ denote the perturbed adjacency matrix.

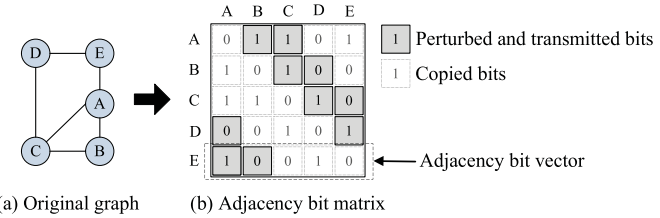



Fig. 2. Illustration of $RABV$ protocol.

The edge between nodes $i$ and $j$ appears in both $\widetilde{M}_{ij}$ and $\widetilde{M}_{ji}$, each perturbed with privacy budget $\epsilon_1$. Then according to the theorem of composability, *RNL* becomes a $2\epsilon_1$-edge LDP algorithm for an undirected graph, which is less private. A formal proof is as follows.

For the original adjacency matrix $\boldsymbol{M}$ of an undirected graph, $M_{ij} = M_{ji}$ always holds for any two nodes $i$ and $j$. By observing two perturbed bits $\widetilde{M}_{ij}$ and $\widetilde{M}_{ji}$ in the perturbed adjacency matrix $\widetilde{\boldsymbol{M}}$, the posterior probability that there exists an edge between nodes $i$ and $j$ can be denoted by $\Pr[M_{ij} = M_{ji} = 1 \,|\widetilde{M}_{ij}, \widetilde{M}_{ji}]$. Further, we have

$$\begin{aligned}
&\frac{\Pr[M_{ij} = M_{ji} = 1 \,|\widetilde{M}_{ij}, \widetilde{M}_{ji}]}{\Pr[M_{ij} = M_{ji} = 0 \,|\widetilde{M}_{ij}, \widetilde{M}_{ji}]} \\
&\leq \frac{\Pr[M_{ij} = M_{ji} = 1 \,|\widetilde{M}_{ij} = \widetilde{M}_{ji} = 1]}{\Pr[M_{ij} = M_{ji} = 0 \,|\widetilde{M}_{ij} = \widetilde{M}_{ji} = 1]} \\
&= \frac{\Pr[M_{ij} = 1 \,|\widetilde{M}_{ij} = 1] \cdot \Pr[M_{ji} = 1 \,|\widetilde{M}_{ji} = 1]}{\Pr[M_{ij} = 0 \,|\widetilde{M}_{ij} = 1] \cdot \Pr[M_{ji} = 0 \,|\widetilde{M}_{ji} = 1]} \\
&= \frac{\frac{e^{\epsilon_1}}{1+e^{\epsilon_1}} \cdot \frac{e^{\epsilon_1}}{1+e^{\epsilon_1}}}{\frac{1}{1+e^{\epsilon_1}} \cdot \frac{1}{1+e^{\epsilon_1}}} = e^{2\epsilon_1},
\end{aligned}$$

which proves that *RNL* only provides $2\epsilon_1$-edge LDP.

Furthermore, *RNL* requires each user to perturb and send all $n$ bits in the adjacency bit vector to data collector, which incurs a high computation and communication cost.

To address the problems of *RNL*, we propose a more private and efficient protocol *Randomized Adjacency Bit Vector* ($RABV$) to perturb edges in undirected graphs. As shown in Fig. 2b, the adjacency matrix is composed of $n$ rows, each corresponding to the adjacency bit vector of a node. For the first $1 \leq i \leq \lfloor \frac{n}{2} \rfloor$ nodes, $RABV$ uses RR as in Eq. (4) to perturb and transmit $t = \lfloor \frac{n}{2} \rfloor$ bits (i.e., bits in grey)—from the $(i+1)$th bit to the $(i+1+t \bmod n)$th bit; for the rest nodes, $RABV$ uses RR to perturb and transmit $t = \lfloor \frac{n-1}{2} \rfloor$ bits in the same way. In essence, *RABV perturbs one and only one bit* for each pair of symmetric bits in the adjacency matrix. The data collector can then obtain the whole matrix by copying bits in grey to their symmetric positions.

Following the same proof of $RNL$, $RABV$ is guaranteed to satisfy $\epsilon_1$-edge LDP for the collector. Further, since each node only perturbs and transmits about half of the bits in an adjacency bit vector, $RABV$ significantly reduces computation and communication cost of *RNL*.

### 4.3 Node Degree Perturbation

Releasing the degree of a node while satisfying edge $\epsilon$-LDP is essentially a centralized DP problem because all edges incident to this node, or equivalently, all bits in its adjacency bit vector, form a database and the degree is a count function. In the literature, *Laplace Mechanism* [12] is the predominant technique to perturb numerical function values such as counts. As such, LF-GDPR adopts it to perturb the degree $d_i$ of each node $i$. According to the definition of edge LDP, two adjacency bit vectors $\boldsymbol{B}$ and $\boldsymbol{B}'$ are two neighboring databases if they differ in only one bit. As such, the sensitivity of degree (i.e., count function) is 1, and therefore adding Laplace noise $Lap(\frac{1}{\epsilon_2})$ to the node degree can satisfy $\epsilon_2$-LDP. That is, $\widetilde{d}_i = d_i + Lap(\frac{1}{\epsilon_2})$.

Similar to perturbing adjacency bit vector, however, in the above naive approach the data collector witnesses two node degrees $d_i$ and $d_j$ perturbed independently, but they share the same edge between $i$ and $j$. As such, whether this edge exists or not contributes to both $d_i$ and $d_j$. In the most extreme case where there are only two nodes and one edge in the graph, $d_1 = 1$ and $d_2 = 1$, both of which indicate the existence of this edge. If it is removed, both $d_1$ and $d_2$ will decrease by 1, causing the sensitivity of node degree perturbation to be 2. As DP or LDP does not refrain an adversary from possessing any background knowledge, in the worst case the collector already knows all edges except for this one. As such, witnessing the two node degrees $d_i$ and $d_j$ is degenerated to witnessing the edge between $i$ and $j$ twice and independently.

Unfortunately, the remedy that works for perturbing adjacency bit vector cannot be adopted here, as direct bit copy is not feasible for degree. As such, we take an alternative approach to increase the Laplace noise. The following theorem proves that if we add Laplace noise $Lap(\frac{2}{\epsilon_2})$ to every node degree, $\epsilon_2$-LDP can be satisfied for the collector.

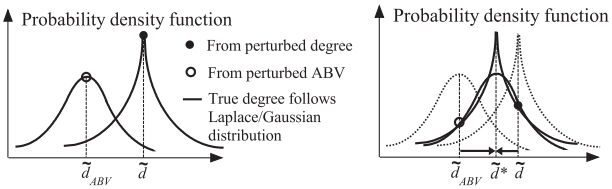
**Theorem 4.1.** *A perturbation algorithm $\mathcal{A}$ satisfies $\epsilon_2$-LDP for the collector if it adds Laplace noise $Lap(\frac{2}{\epsilon_2})$ to every node degree $d_i$, i.e., $\widetilde{d}_i = \mathcal{A}(d_i) = d_i + Lap(\frac{2}{\epsilon_2})$.*

**Proof** By adding Laplace noise $Lap(\frac{2}{\epsilon_2})$ to any node degree $d_i$, i.e., $\widetilde{d}_i = d_i + Lap(\frac{2}{\epsilon_2})$, the perturbation algorithm $\mathcal{A}$ satisfies $\frac{\epsilon_2}{2}$-LDP for node $i$. For the collector, whether there is an edge between any two nodes $i$ and $j$ can be derived from both perturbed degrees $\widetilde{d}_i$ and $\widetilde{d}_j$. Then according to the composability property of Theorem 2.3, the perturbation algorithm $\mathcal{A}$ satisfies $\epsilon_2$-LDP for the collector. □

The perturbed degree $\widetilde{d}$ is a coarse estimation of the true degree. Now that we have both $\widetilde{d}$ and $\widetilde{d}_{ABV}$, the degree estimated from the perturbed adjacency bit vector $\widetilde{\boldsymbol{B}}$,[1] we can use *Maximum Likelihood Estimation* (MLE) [34] to obtain a refined estimation $\widetilde{d}^*$. The rationale of this refinement is illustrated in Fig. 3. Before refinement (Fig. 3a), as each bit of $\boldsymbol{B}$ follows Bernoulli distribution, according to De Moivre-Laplace Central Limit Theorem, the probability density function of $\widetilde{d}_{ABV}$ can be approximated by a Gaussian distribution $f_1(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-d^*)^2}{2\sigma^2}}$, where the variance $\sigma^2 = n \cdot [\frac{1}{16(p-\frac{1}{2})^2} - (\frac{\widetilde{d}}{n} - \frac{1}{2})^2]$ is derived in Eq. (1).[2] On the other hand, as $\widetilde{d}$ is obtained by adding Laplace noise to $d$, the

1. A naive and biased estimation is $\widetilde{d}_{ABV} = \sum_{j=1}^{n} \widetilde{b}_j$. In Example 4.4, we show a calibrated and unbiased estimation $\widetilde{d}_{ABV} = \frac{\sum_{j=1}^{n} \widetilde{b}_j}{2p-1} + \frac{(p-1)n}{2p-1}$, where $p = \frac{e^{\epsilon_1}}{e^{\epsilon_1}+1}$.

2. Here we replace $d$ with $\widetilde{d}$ in Eq. (1) for simplicity.




(a) Before refinement (b) After refinement

Fig. 3. Refining $\widetilde{d}$ to $\widetilde{d}^*$ by MLE.

probability density function of $\widetilde{d}$ follows a Laplace distribution $f_2(x) = \frac{\epsilon_2}{4} e^{-\frac{|x-d^*|\epsilon_2}{2}}$. The refinement, as shown in Fig. 3b, shifts both distributions to share the same mean, i.e., the true degree, as they are both drawn from it. To estimate this mean $d^*$ by MLE, we derive the joint likelihood of observing both $\widetilde{d}$ and $\widetilde{d}_{ABV}$, and maximize it. Since they are both independently perturbed, the joint likelihood is the multiplication of individual probabilities. Formally,

$$\begin{aligned}\widetilde{d}^* &= \arg\max_{\widetilde{d}^*} f_1(\widetilde{d}_{ABV}) \cdot f_2(\widetilde{d}) \\ &= \arg\max_{\widetilde{d}^*} \frac{\epsilon_2}{\sigma \cdot 4\sqrt{2\pi}} e^{-\frac{(\widetilde{d}_{ABV}-\widetilde{d}^*)^2+\sigma^2|\widetilde{d}-\widetilde{d}^*|\epsilon_2}{2\sigma^2}} \\ &\approx \arg\min_{\widetilde{d}^*} \Big((\widetilde{d}_{ABV}-\widetilde{d}^*)^2 + \sigma^2|\widetilde{d}-\widetilde{d}^*|\epsilon_2\Big).\end{aligned}$$

By solving the above equation, we have

$$\widetilde{d}^* = median\Big(\widetilde{d}_{ABV} - \frac{\sigma^2 \cdot \epsilon_2}{2}, \widetilde{d}, \widetilde{d}_{ABV} + \frac{\sigma^2 \cdot \epsilon_2}{2}\Big). \tag{5}$$

### 4.4 Aggregation and Calibration

Upon receiving the perturbed adjacency matrix $\widetilde{M}$ and degree vector $\widetilde{D}$,[3] the data collector can estimate the target graph metric $\widetilde{F}$ by aggregation according to Eq. (3) with a calibration function $\mathcal{R}(\cdot)$

$$\widetilde{F} = \sum\nolimits_l \mathcal{R}\Big(f_{\phi_l}(\widetilde{M}^{k_l})\Big) \cdot g_{\psi_l}(\widetilde{D}). \tag{6}$$

The calibration function aims to suppress the aggregation bias of $\widetilde{M}$ propagated by $f_{\phi_l}$. On the other hand, no calibration is needed for $g_{\psi_l}(\widetilde{D})$ as $\widetilde{D}$ is already an unbiased estimation of $D$, thanks to the Laplace Mechanism.

To derive $\mathcal{R}(\cdot)$, we regard $\mathcal{R}$ as the mapping between $f_{\phi_l}(M^{k_l})$ and $f_{\phi_l}(\widetilde{M}^{k_l})$. In other words, $\mathcal{R}$ estimates $f_{\phi_l}(M^{k_l})$ after observing $f_{\phi_l}(\widetilde{M}^{k_l})$. Formally,

$$\mathcal{R} : f_{\phi_l}(\widetilde{M}^{k_l}) \rightarrow f_{\phi_l}(M^{k_l}).$$

The following shows a concrete example for aggregation and calibration when estimating the number of edges in a graph. The result of this example will be used in Section 6 to estimate $L_c$ in Eq. (2) of modularity definition.

**Example 4.4.** For a graph with $n$ nodes, there are $N = \frac{1}{2}n(n-1)$ bits in its upper/lower triangular matrix, each indicating whether an edge exists or not. Let $s$ denote the number of edges in the original graph, i.e., the number of "1"s in these $N$ bits. These $N$ bits are then perturbed according to $RABV$ protocol by randomized response [26] with flipping probability $p$. To estimate $s$, the data collector takes the following two steps.

(1) *Aggregation.* It aggregates the number of "1"s in the perturbed $N$ bits and uses it as an initial estimation $\tilde{s}$.

(2) *Calibration.* Since the mapping between $s$ and $\widetilde{s}$ can be captured by $\widetilde{s} = sp + (N-s)(1-p)$, the collector then calibrates $\tilde{s}$ by $\mathcal{R}(\widetilde{s}) = \frac{s}{2p-1} + \frac{p-1}{2p-1}N$, which is derived by solving the mapping function.

We can further show $\mathcal{R}(\widetilde{s})$ is an unbiased estimation of $s$, because $\mathbb{E}[\mathcal{R}(\widetilde{s})] = \frac{1}{2p-1}[sp + (N-s)(1-p) + (p-1)N] = s$.

If both $\mathcal{R}(f_{\phi_l}(\widetilde{M}^{k_l}))$ and $g_{\psi_l}(\widetilde{D})$ are unbiased estimation of $f_{\phi_l}(M^{k_l})$ and $g_{\psi_l}(D)$ respectively, the following theorem guarantees $\widetilde{F}$ is an unbiased estimation of the target metric $F$.

**Theorem 4.2.** *If $\mathcal{R}(f_{\phi_l}(\widetilde{M}^{k_l}))$ and $g_{\psi_l}(\widetilde{D})$ are unbiased estimation of $f_{\phi_l}(M^{k_l})$ and $g_{\psi_l}(D)$ respectively, the estimated graph metric $\widetilde{F}$ is unbiased.*

**Proof.** According to the assumption of unbiased estimation, we have

$$\begin{aligned}\mathbb{E}\Big[\mathcal{R}\Big(f_{\phi_l}(\widetilde{M}^{k_l})\Big)\Big] &= f_{\phi_l}(M^{k_l}) \\ \mathbb{E}\Big[g_{\psi_l}(\widetilde{D})\Big] &= g_{\psi_l}(D).\end{aligned}$$

Since the adjacency bit vector and the degree of each node are perturbed independently, we have

$$\begin{aligned}\mathbb{E}\Big[\widetilde{F}\Big] &= \sum\nolimits_l \mathbb{E}\Big[\mathcal{R}\Big(f_{\phi_l}(\widetilde{M}^{k_l})\Big) \cdot g_{\psi_l}(\widetilde{D})\Big] \\ &= \sum\nolimits_l \mathbb{E}\Big[\mathcal{R}\Big(f_{\phi_l}(\widetilde{M}^{k_l})\Big)\Big] \cdot \mathbb{E}\Big[g_{\psi_l}(\widetilde{D})\Big] \\ &= \sum\nolimits_l f_{\phi_l}(M^{k_l}) \cdot g_{\psi_l}(D) \\ &= F.\end{aligned}$$

Therefore, $\widetilde{F}$ is unbiased. □

### 4.5 Optimal Privacy Budget Allocation

The final problem in LF-GDPR is to allocate the privacy budget (step ② in Fig. 1). Formally, it divides $\epsilon$ into $\epsilon_1 = \alpha\epsilon$ and $\epsilon_2 = (1-\alpha)\epsilon$, where $\alpha \in (0,1)$, for adjacency bit vector and node degree perturbation, respectively.

Our objective is to find the optimal $\alpha$ that minimizes the distance between the graph metric $F$ and our estimation $\widetilde{F}$. Without loss of generality, we adopt the $L_2$ distance [35] and set the loss function for optimization as the expectation of this distance, i.e., $\alpha = \arg\min_{\alpha\in(0,1)} \mathbb{E}[\|\widetilde{F} - F\|_2^2]$.

Assuming $\widetilde{F}$ is unbiased, we have

$$\begin{aligned}\mathbb{E}[\|\widetilde{F} - F\|_2^2] &= \mathbb{E}[F^2 - 2F\widetilde{F} + \widetilde{F}^2] \\ &= \mathbb{E}[F^2] - 2\mathbb{E}[F] \cdot \mathbb{E}[\widetilde{F}] + \mathbb{E}[\widetilde{F}^2] \\ &= \mathbb{E}[\widetilde{F}^2] - F^2.\end{aligned}$$

Since $F^2$ is constant, we only need to minimize $\mathbb{E}[\widetilde{F}^2]$ with respect to $\alpha$

$$\mathbb{E}[\widetilde{F}^2] = \mathbb{E}\Big[\Big(\sum\nolimits_l \mathcal{R}\Big(f_{\phi_l}(\widetilde{M}^{k_l})\Big) \cdot g_{\psi_l}(\widetilde{D})\Big)^2\Big]. \tag{7}$$

3. In the sequel, $\widetilde{D}$ denotes the refined degree $\widetilde{D}^*$ to simplify the notation.

In the next two sections, we will demonstrate how to derive the terms in Eq. (7) with respect to $\alpha$. Then we can apply numerical methods, e.g., Newton's method [36], to find $\alpha$ that minimizes Eq. (7). Further, the following theorem shows the accuracy guarantee of LF-GDPR.

**Theorem 4.3.** *For a graph metric $F$ and our estimation $\widetilde{F}$, with at least $1-\beta$ probability, we have*

$$|F-\widetilde{F}| = O(\sqrt{\mathbb{E}[\widetilde{F}^2]\cdot log(1/\beta)}).$$

**Proof.** For a graph metric $F$, and its estimated one $\widetilde{F}$, the variance of $F-\widetilde{F}$ is

$$\begin{aligned} Var[F-\widetilde{F}] &= Var[\widetilde{F}] = \mathbb{E}[F^2] - (\mathbb{E}[F])^2 \\ &= \mathbb{E}[\widetilde{F}^2] - F^2 \leq \mathbb{E}[\widetilde{F}^2]. \end{aligned}$$

By Benstein's inequality,

$$\begin{aligned} \Pr[|F-\widetilde{F}|\geq\lambda] &\leq 2\cdot\exp\left(-\frac{\lambda^2}{2Var[F-\widetilde{F}]+\frac{2}{3}\lambda}\right) \\ &\leq 2\cdot\exp\left(-\frac{\lambda^2}{2\mathbb{E}[\widetilde{F}^2]+\frac{2}{3}\lambda}\right). \end{aligned}$$

By the union bound, there exists $\lambda = O(\sqrt{\mathbb{E}[\widetilde{F}^2]\cdot log(1/\beta)})$ such that $|F-\widetilde{F}| < \lambda$ holds with at least $1-\beta$ probability. □

As will be shown in the next two sections, $\mathbb{E}[\widetilde{F}^2]$ can be further expressed by $\epsilon$, $n$ or $d$ for a specific graph metric.

### 4.6 Summary

Algorithm 1 summarizes the overall protocol of LF-GDPR. It takes three inputs—the target graph metric $F$, the privacy budget $\epsilon$, and the true adjacency bit vector $B_i$ of each node $i$, and returns an estimation of graph metric $\widetilde{F}$ under $\epsilon$-LDP. In Line 1, the data collector reduces $F$ to adjacency matrix and node degree. Based on the reduction, in Line 2 the privacy budget $\epsilon$ is divided into $\alpha\epsilon$ and $(1-\alpha)\epsilon$ by the optimal privacy budget allocation algorithm (see Section 4.5 for details), and then $\alpha$ is sent to each node (Line 3). On each node $i$, $RABV$ perturbs its adjacency bit vector (Lines 5-6, see Section 4.2 for details). For each bit to perturb, it adopts RR with privacy budget $\alpha\epsilon$. Then node $i$ further perturbs its degree $d_i$ by adding a Laplace noise with privacy budget $(1-\alpha)\epsilon$ (Line 7). Finally, the perturbed adjacency bit vector and node degree are sent to the data collector (Line 8). After the collector receives the perturbed adjacency matrix $\widetilde{M}$ and degree vector $\widetilde{D}$, it first completes the whole adjacency matrix by copying bits to their symmetric ones in $\widetilde{M}$ (Line 9), and then refines each node degree $\widetilde{d}_i$ to $\widetilde{d}_i^*$ (Line 10, see Section 4.3 for details). Finally, it applies aggregation and calibration to estimate the graph metric $\widetilde{F}$ (Line 11).

*Security of Correlation.* It is known that the privacy provided by differential privacy decrease significantly under correlations [37], [38]. However, correlation between adjacency bit vectors and node degrees does not compromise LDP in LF-GDPR. First, there is pairwise correlation between the adjacency bit vectors of any two users, but the proposed RABV protocol is able to well address it by avoiding "double dose" of the same edge information. Second, there is correlation between the node degrees of two users who share an edge. But Theorem 4.1 proves that by setting sensitivity to 2 and adding $Lap(\frac{2}{\epsilon_2})$ noise, this correlation does not compromise $\epsilon_2$-LDP. Third, there is correlation between the adjacency bit vector and node degree of the same user. But since we divide the privacy budget between them, according to sequential composition, $\epsilon$-LDP is still achieved even if they have the strongest correlation (i.e., an equivalent or causal value).

**Algorithm 1.** Overall Protocol of LF-GDPR Framework

**Input:** Target graph metric $F$
Privacy budget $\epsilon$
True adjacency bit vector $\{B_1,\ldots,B_n\}$
**Output:** An estimation of the graph metric $\widetilde{F}$ under $\epsilon$-LDP
**Procedure:**
// ***Collector side***
1: Reduce graph metric $F$ to adjacency matrix $M=\{B_1,\ldots,B_n\}$ and degree vector $D$ derived from $M$
2: Calculate $\alpha$ for privacy budget allocation based on $F$ and $\epsilon$
3: Send $\alpha$ to each node
// ***User side***
4: **for** each node $i\in\{1,2,\ldots,n\}$ **do**
5: $t = i\leq\lfloor\frac{n}{2}\rfloor\ ?\ \lfloor\frac{n}{2}\rfloor : \lfloor\frac{n-1}{2}\rfloor$
6: **for** each $b_j\in B_i$, where $i+1\leq j\leq(i+1+t) \bmod n$ **do**

$$\text{Perturb}\,\widetilde{b}_j = \begin{cases} b_j & \text{w.p. } \frac{e^{\alpha\epsilon}}{1+e^{\alpha\epsilon}} \\ 1-b_j & \text{w.p. } \frac{1}{1+e^{\alpha\epsilon}} \end{cases}$$

7: Calculate the degree $d_i$ from $B_i$ and then perturb it as

$$\widetilde{d}_i = d_i + Lap(2/((1-\alpha)\epsilon))$$

8: Send $\widetilde{B}_i$ and $\widetilde{d}_i$ to the data collector
9: **end for**
// ***Collector side***
10: Copy symmetric bits in $\widetilde{M}=\{\widetilde{B}_1,\ldots,\widetilde{B}_n\}$
11: Refine $\widetilde{d}_i$ to $\widetilde{d}_i^*$ of each node $i$ according to Eq. (5)
12: Apply aggregation and calibration to estimate the graph metric $\widetilde{F}$ based on $\widetilde{M}$ and $\widetilde{D}=\{\widetilde{d}_1^*,\ldots,\widetilde{d}_n^*\}$
13: **return** $\widetilde{F}$

## 5 Clustering Coefficient Estimation With LF-GDPR

In this section, we show how to use LF-GDPR to estimate the clustering coefficients of all nodes in a graph. Based on the implementation framework in Section 4, we present the details of steps ①②④. Finally, Algorithm 2 summarizes the whole process.

### 5.1 Implementation Details

*Graph Metric Reduction* (step ① in LF-GDPR). Recall that the clustering coefficient of node $i$, $cc_i=\frac{2t_i}{d_i(d_i-1)}$, where $t_i$ is the number of triangles incident to $i$. To count $t_i$, we set $k_1=3$ so that $M^3$ denotes the number of 3-hop walks for all pairs of nodes. We then set projection $\phi_i$ to $M^3_{ii}$, the $i$th diagonal element of $M^3$ that denotes the number of 3-hop walks

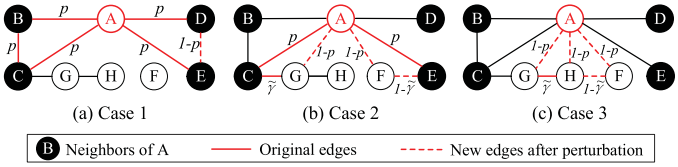



Fig. 4. Estimate number of triangles incident to node A.

starting and ending at node $i$.[4] Note that $M^3_{ii}$ counts the triangles incident to node $i$ twice (e.g., triangles $ijk$ and $ikj$), so $M^3_{ii} = 2t_i$, which is exactly the numerator in the above $cc_i$ definition. As such, the aggregation function $f(\cdot)$ can be simply set to an identity function. Formally,

$$f_{\phi_i}(\boldsymbol{M}^3) = f(M^3_{ii}) = M^3_{ii} = 2t_i.$$

To obtain the denominator in the above $cc_i$ definition, we set the projection $\psi_i$ to $d_i$, the $i$th element of degree vector $\boldsymbol{D}$. And the aggregation function $g(\cdot)$ is set according to the denominator in the definition of clustering coefficient

$$g_{\psi_i}(\boldsymbol{D}) = g(d_i) = \frac{1}{d_i(d_i-1)}.$$

To sum up, the clustering coefficient of any node $i$, denoted by $F_i$, can be reduced to $\boldsymbol{M}$ and $\boldsymbol{D}$ as

$$F_i = f_{\phi_i}(\boldsymbol{M}^3) \cdot g_{\psi_i}(\boldsymbol{D}). \tag{8}$$

*Aggregation and Calibration* (step ④ in LF-GDPR). The data collector receives the perturbed adjacency matrix $\widetilde{\boldsymbol{M}}$ and degree vector $\widetilde{\boldsymbol{D}}$. According to Eqs. (6) and (8), the estimated clustering coefficient of any node $i$ is

$$\widetilde{F}_i = \mathcal{R}\Big(f_{\phi_i}(\widetilde{\boldsymbol{M}}^3)\Big) \cdot g_{\psi_i}(\widetilde{\boldsymbol{D}}), \tag{9}$$

where the calibration function $\mathcal{R}(\cdot)$ estimates $f_{\phi_i}(\boldsymbol{M}^3)$, the number of triangles incident to node $i$ based on the perturbed number $f_{\phi_i}(\widetilde{\boldsymbol{M}}^3)$. In what follows, we derive $\mathcal{R}(\cdot)$.

According to Section 4.4, to derive $\mathcal{R}(\cdot)$ we need to estimate $f_{\phi_A}(\boldsymbol{M}^3)$, or equivalently $t_A = f_{\phi_A}(\boldsymbol{M}^3)/2$, the number of triangles incident to node $A$ in the original graph. Figs. 4a, 4b, and 4c enumerate all three cases of such triangles based on whether the other two nodes of this triangle are $A$'s neighbors in the original graph. Let $d$ denote its degree and $p = \frac{e^{\alpha\epsilon}}{1+e^{\alpha\epsilon}}$ the perturbation probability. In each case, the edges that constitute such triangles are highlighted by red color. In particular, the red solid lines denote the original edges, and each is retained in the perturbed graph with a probability of $p$. The red dashed lines denote the new edges after perturbation, and each appears with a probability of $1-p$.

1) Fig. 4a: both nodes are neighbors of $A$. There are two sub-cases based on whether there exists an edge between these two nodes in the original graph. For triangles such as $ABC$, there is an edge between $B$ and $C$ in the original graph. Such triangles will be retained in the perturbed graph with probability $p^3$. For triangles such as $ADE$, there is no edge between $D$ and $E$ in the original graph. Such triangles will be retained in the perturbed graph with probability $p^2(1-p)$. Summing up both sub-cases, the number of such triangles in the perturbed graph is $\widetilde{t}_{A,1} = t_A \cdot p^3 + \left(\frac{1}{2}d(d-1) - t_A\right) \cdot p^2(1-p)$.
2) Fig. 4b: only one node is a neighbor of $A$, for example triangles $ACG$ and $AEF$. Since $d$ nodes are adjacent to $A$ and $n-d-1$ nodes are not adjacent, there are $d(n-d-1)$ possible triangles. In such a triangle, the two edges incident to $A$ will be retained in the perturbed graph with probabilities $p(1-p)$. The probability of having the third edge (e.g., $CG$ or $EF$) in the perturbed graph can be approximated by the overall edge density after perturbation, i.e., $\widetilde{\gamma} = \gamma p + (1-\gamma)(1-p)$, where $\gamma = \frac{\sum_{i=1}^{n} d_i}{n(n-1)}$ denote the edge density in the original graph. As such, the number of triangles in this case is $\widetilde{t}_{A,2} = d(n-d-1) \cdot p(1-p)\widetilde{\gamma}$.
3) Fig. 4c: neither node is a neighbor of $A$, for example triangles $AGH$ and $AFH$. In such a triangle, the two edges incident to $A$ will be retained in the perturbed graph with probabilities $(1-p)^2$. The probability of having the third edge (e.g., $GH$ or $FH$) in the perturbed graph can also be approximated by $\widetilde{\gamma}$. Since there are $\binom{n-d-1}{2} = \frac{1}{2}(n-d-1)(n-d-2)$ possible triangles, the number of triangles in this case is $\widetilde{t}_{A,3} = \frac{1}{2}(n-d-1)(n-d-2) \cdot (1-p)^2\ \widetilde{\gamma}$.

By summing up $\widetilde{t}_{A,1}$, $\widetilde{t}_{A,2}$, and $\widetilde{t}_{A,3}$, we obtain $\widetilde{t}_A$. Since the calibration function $\mathcal{R}(\cdot)$ maps $\widetilde{t}_A$ to $t_A$, i.e., $\mathcal{R}(\widetilde{t}_A) = t_A$, we can solve $t_A$ from $\widetilde{t}_A$ and derive $\mathcal{R}(\cdot)$ as[5]

$$\begin{aligned}\mathcal{R}(\widetilde{t}_A) = \frac{1}{p^2(2p-1)}\Big(&\widetilde{t}_A - \frac{1}{2}\widetilde{d}(\widetilde{d}-1)p^2(1-p)\\ &- \widetilde{d}(n-\widetilde{d}-1)p(1-p)\widetilde{\gamma}\\ &- \frac{1}{2}(n-\widetilde{d}-1)(n-\widetilde{d}-2)(1-p)^2\widetilde{\gamma}\Big).\end{aligned} \tag{10}$$

*Privacy Budget Allocation* (step ④ in LF-GDPR). According to Section 4.5, to solve $\alpha$ we derive and minimize $\mathbb{E}[\widetilde{F}^2]$ with respect to $\alpha$ in Eq. (7). Theorem 5.1 below shows the closed-form solution of $\alpha$.

**Theorem 5.1.** *The optimal $\alpha$ for clustering coefficient estimation can be approximated by*

$$\arg\min_{\alpha\in(0,1)} \frac{e^{\alpha\epsilon}+2}{e^{3\alpha\epsilon}(e^{\alpha\epsilon}-1)^2}\left(1 + \frac{8(10\hat{d}^2 - 10\hat{d} + 3)}{\hat{d}^2(\hat{d}-1)^2(1-\alpha)^2\epsilon^2}\right), \tag{11}$$

*where $\hat{d}$ is a representative degree (e.g., the mean, median, or most frequent degree) of all nodes in the original graph.*

**Proof.** According to Eq. (7), we have

$$\begin{aligned}\mathbb{E}[\widetilde{F}^2] &= \mathbb{E}\left[\left(\sum\nolimits_l \mathcal{R}\Big(f_{\phi_l}(\widetilde{\boldsymbol{M}}^{k_l})\Big) \cdot g_{\psi_l}(\widetilde{\boldsymbol{D}})\right)^2\right]\\ &= \left(f^2_{\phi}(\boldsymbol{M}^3) + Var\left[\mathcal{R}\Big(f_{\phi}(\widetilde{\boldsymbol{M}}^3)\Big)\right]\right) \cdot \mathbb{E}\left[g^2_{\psi}(\widetilde{\boldsymbol{D}})\right].\end{aligned}$$

4. The full notion of $\phi_i$ should be $\phi_{1,i}$. Since there is only one term in the definition of clustering coefficient, we omit the notation 1. The same applies to $\psi_i$.

5. We replace $d$ with $\widetilde{d}$, because the former is unknown to data collector and the latter is an unbiased estimation of the former.

For each node $i$, by setting

$$f_{\phi_i}(\boldsymbol{M}^3) = 2t_i \quad \text{and} \quad g_{\psi_i}(\boldsymbol{D}) = \frac{1}{d_i(d_i-1)},$$

we approximate $\mathbb{E}[\widetilde{F}_i^2]$ by

$$\mathbb{E}[\widetilde{F}_i^2] = 4(t_i^2 + Var[\mathcal{R}(\widetilde{t}_i)]) \cdot \mathbb{E}\left[\frac{1}{\widetilde{d}_i^2(\widetilde{d}_i-1)^2}\right], \tag{12}$$

where

$$\begin{aligned} &Var\big[\mathcal{R}\big(\widetilde{t}_i\big)\big] = \frac{Var[\widetilde{t}_i]}{p^4(2p-1)^2} \\ &= \frac{\frac{1}{2}(n-1)(n-2)Var\Big[\widetilde{M}_{it_1}\widetilde{M}_{t_1t_2}\widetilde{M}_{t_2i}\Big]}{p^4(2p-1)^2} \\ &\approx \frac{(n-d_i-1)^2(n-d_i-2)^2}{2(n-1)(n-2)} \cdot \frac{e^{\alpha\epsilon}+2}{e^{3\alpha\epsilon}(e^{\alpha\epsilon}-1)^2}. \end{aligned} \tag{13}$$

Since $t_i^2 \ll O(n^2) \sim Var[\mathcal{R}(\widetilde{t}_i)]$ for most cases, we omit the term of $t_i^2$. As for $\mathbb{E}[\frac{1}{\widetilde{d}_i^2(\widetilde{d}_i-1)^2}]$, by Taylor expansion at $\mathbb{E}[\widetilde{d}_i]$, we have

$$\mathbb{E}\left[\frac{1}{\widetilde{d}_i^2(\widetilde{d}_i-1)^2}\right] \approx \frac{1}{d_i^2(d_i-1)^2} + \frac{8(10d_i^2-10d_i+3)}{d_i^4(d_i-1)^4(1-\alpha)^2\epsilon^2}. \tag{14}$$

By substituting Eqs. (13) and (14) into Eq. (12), we have

$$\begin{aligned} \mathbb{E}[\widetilde{F}_i^2] =& \frac{4(n-d_i-1)^2(n-d_i-2)^2}{2(n-1)(n-2)d_i^2(d_i-1)^2} \\ &\cdot \frac{e^{\alpha\epsilon}+2}{e^{3\alpha\epsilon}(e^{\alpha\epsilon}-1)^2}\left(1+\frac{8(10d_i^2-10d_i+3)}{d_i^2(d_i-1)^2(1-\alpha)^2\epsilon^2}\right). \end{aligned}$$

To minimize $\mathbb{E}[\widetilde{F}_i^2]$, we omit the first item which is independent of $\alpha$. To unify $\alpha$ for all nodes, we replace $d_i$ with $\hat{d}$, a representative degree (e.g., the mean, median, or most frequent degree) of all nodes in the original graph. Therefore, we can derive $\alpha$ as

$$\arg\min_{\alpha\in(0,1)} \quad \frac{e^{\alpha\epsilon}+2}{e^{3\alpha\epsilon}(e^{\alpha\epsilon}-1)^2}\left(1+\frac{8(10d_i^2-10d_i+3)}{\hat{d}^2(\hat{d}-1)^2(1-\alpha)^2\epsilon^2}\right).$$

□

As for the representative degree $\hat{d}$, it can be estimated by a portion of privacy budget. The data collector can ask each node to consume some of its privacy budgets for a preliminary round of node degree perturbation and send back $\widetilde{D}$ to estimate $\hat{d}$.

### 5.2 Overall Algorithm

Algorithm 2 summarizes how the data collector estimates the clustering coefficients of all nodes, based on the perturbed adjacency matrix $\widetilde{M}$ and degree vector $\widetilde{d}$. It first computes $\widetilde{\gamma}$, the edge density in the perturbed graph from $\widetilde{d}$ (Line 1). Then for each node the collector calculates the number of triangles incident to it (Line 3) and then further calibrates this number based on Eq. (10) (Line 4). Finally, its clustering coefficient is estimated based on Eq. (9) (Line 5).

*Accuracy Guarantee*. According to Theorem 5.1, with at least $1-\beta$ probability, the error of clustering coefficient estimation is bounded by $O(\frac{\sqrt{log(1/\beta)}}{d\cdot\epsilon})$.

**Algorithm 2.** Collector-Side Clustering Coefficient Estimation

**Input:** Perturbed adjacency matrix $\widetilde{M} = \{\widetilde{B}_1, \ldots, \widetilde{B}_n\}$
Perturbed degree vector $\widetilde{D} = \{\widetilde{d}_1, \ldots, \widetilde{d}_n\}$
Percentage $\alpha$ for privacy budget allocation
**Output:** Estimated clustering coefficient $\boldsymbol{cc} = \{cc_1, \ldots, cc_n\}$
**Procedure:**
1: Calculate the edge density in perturbed graph $\widetilde{\gamma} = \frac{\sum_{i=1}^{n}\widetilde{d}_i}{n(n-1)}$
2: **for** each node $i \in \{1, 2, \ldots, n\}$ **do**
3: Calculate the number of triangles $\widetilde{t}_i$ incident to node $i$
4: Calibrate $\widetilde{t}_i$ to get an unbiased one $t_i$ according to Eq. (10), where $p = \frac{e^{\alpha\epsilon}}{1+e^{\alpha\epsilon}}$.
5: Estimate node $i$'s clustering coefficient $cc_i = \frac{2t_i}{\widetilde{d}_i(\widetilde{d}_i-1)}$
6: **end for**
7: **return** $\boldsymbol{cc} = \{cc_1, \ldots, cc_n\}$

## 6 Community Detection With LF-GDPR

In this section, we show how to use LF-GDPR to estimate the modularity of any community in the graph, with only a single round of $\widetilde{B}$ and $\widetilde{D}$ collection. Based on the implementation framework in Section 4, we present the details of steps ①②④. Finally, Algorithms 3 summarizes the process of modularity estimation, which serves for further community detection.

### 6.1 Implementation Details

*Graph Metric Reduction* (step ① in LF-GDPR). Recall in Eq. (2), the modularity of a community $\mathcal{C}$ is $q_c = \frac{L_c}{L} - \frac{K_c^2}{4L^2}$, where $L_c$ is the number of edges in $\mathcal{C}$, $K_c$ is the total degree of all nodes in $\mathcal{C}$, and $L$ is the total number of edges in the whole graph. As such, we can write the graph metric $F_c = q_c$ in the form of Eq. (3) as

$$F_c = q_c = f_{\phi_{1,c}}(\boldsymbol{M}) \cdot g_{\psi_{1,c}}(\boldsymbol{D}) - g_{\psi_{2,c}}(\boldsymbol{D}). \tag{15}$$

There are two terms in the above equation. In the first term, $\phi_{1,c}$ projects graph $G$ to community $\mathcal{C}$, i.e., a sub-matrix $\boldsymbol{M}_c$ of nodes in $\mathcal{C}$ only, and $f_{\phi_{1,c}}(\boldsymbol{M}) = \frac{1}{2}\|\boldsymbol{M}_c\|$, half of the summation of all elements in $\boldsymbol{M}_c$. As such, $f_{\phi_{1,c}}(\boldsymbol{M}) = L_c$. Similarly, $g_{\psi_{1,c}}(\boldsymbol{D}) = \frac{1}{L} = \frac{2}{\|\boldsymbol{D}\|}$. The second term does not involve $\boldsymbol{M}$, so we set $f_{\phi_{2,c}}(\boldsymbol{M}) = -1$. To project graph $G$ to community $\mathcal{C}$, we set $\psi_{2,c}$ to a sub-vector $\boldsymbol{D}_c$ of nodes in $\mathcal{C}$ only, and then $g_{\psi_{2,c}}(\boldsymbol{D}) = \frac{K_c^2}{4L^2} = \frac{\|\boldsymbol{D}_c\|^2}{\|\boldsymbol{D}\|^2}$.

*Aggregation and Calibration* (step ② in LF-GDPR). According to Eqs. (6) and (15), the data collector estimates the modularity $\widetilde{F}$ based on the perturbed adjacency matrix $\widetilde{M}$ and degree vector $\widetilde{D}$ as follows.

$$\widetilde{F} = \mathcal{R}\Big(f_{\phi_{1,c}}(\widetilde{\boldsymbol{M}})\Big) \cdot g_{\psi_{1,c}}(\widetilde{\boldsymbol{D}}) - g_{\psi_{2,c}}(\widetilde{\boldsymbol{D}}).$$

Note that only the first term needs calibration $\mathcal{R}(\cdot)$ as the second term does not involve $\widetilde{M}$. To derive $\mathcal{R}(\cdot)$, we

estimate $f_{\phi_{1,c}}(\boldsymbol{M})$ from $f_{\phi_{1,c}}(\widetilde{\boldsymbol{M}})$ based on the *RABV* algorithm and the fact that $f_{\phi_{1,c}}(\boldsymbol{M}) = \frac{1}{2}\|\boldsymbol{M}_c\| = L_c$. Example 4.4 shows the derivation of this estimation. By solving $f_{\phi_{1,c}}(\boldsymbol{M})$ in terms of $f_{\phi_{1,c}}(\widetilde{\boldsymbol{M}})$, we can derive $\mathcal{R}(\cdot)$ as

$$\mathcal{R}\Big(f_{\phi_{1,c}}(\widetilde{\boldsymbol{M}})\Big) = \frac{f_{\phi_{1,c}}(\widetilde{\boldsymbol{M}})}{2p-1} + \frac{1}{2}n_c(n_c-1)\frac{p-1}{2p-1}, \tag{16}$$

where $n_c = |\mathcal{C}|$ denotes the number of nodes in $\mathcal{C}$.

*Privacy Budget Allocation* (step ④ in LF-GDPR). Similar to clustering coefficient estimation, we derive and minimize $\mathbb{E}[\widetilde{F}^2]$ with respect to $\alpha$ in Eq. (7). Theorem 6.1 below shows the closed-form solution of $\alpha$.

**Theorem 6.1.** *The optimal $\alpha$ for modularity estimation can be approximated by*

$$\arg\min_{\alpha\in(0,1)} \frac{(1-\alpha)^2\epsilon^2L^2 + 6n^2}{(1-\alpha)^2\epsilon^2L^4}\left(\frac{1}{16(\frac{e^{\alpha\epsilon}}{1+e^{\alpha\epsilon}}-\frac{1}{2})^2} - \left(\frac{2L}{n(n-1)}-\frac{1}{2}\right)^2\right).$$

**Proof.** According to Eqs. (7) and (2), we have

$$\begin{aligned}\mathbb{E}[\widetilde{F}^2] &= \mathbb{E}\left[\left(\sum\nolimits_l \mathcal{R}\Big(f_{\phi_l}(\widetilde{\boldsymbol{M}}^{k_l})\Big)\cdot g_{\psi_l}(\widetilde{\boldsymbol{D}})\right)^2\right]\\ &= \left(f^2_{\phi_1}(\boldsymbol{M}) + Var\Big[\mathcal{R}\Big(f_{\phi_1}(\widetilde{\boldsymbol{M}})\Big)\Big]\right)\cdot\mathbb{E}\Big[g^2_{\psi_1}(\widetilde{\boldsymbol{D}})\Big]\\ &\quad + \mathbb{E}\Big[g^2_{\psi_2}(\widetilde{\boldsymbol{D}})\Big] + \mathbb{E}\Big[\mathcal{R}\Big(f_{\phi_1}(\widetilde{\boldsymbol{M}})\Big)\Big]\mathbb{E}\Big[g_{\psi_1}(\widetilde{\boldsymbol{D}})g_{\psi_2}(\widetilde{\boldsymbol{D}})\Big].\end{aligned}$$

For each community $\mathcal{C}$, note that $\mathbb{E}[L_c] = \frac{n_c^2}{n^2}L$ and by setting

$$f_{\phi_{1,c}}(\boldsymbol{M}) = L_c \quad\text{and}\quad g_{\psi_{1,c}}(\boldsymbol{D}) = \frac{1}{L},$$

$$f_{\phi_{2,c}}(\boldsymbol{M}) = -1 \quad\text{and}\quad g_{\psi_{2,c}}(\boldsymbol{D}) = \frac{K_c^2}{4L^2},$$

we can approximate $\mathbb{E}[\widetilde{F}_c^2]$ by

$$\begin{aligned}\mathbb{E}[\widetilde{F}_c^2] &= \left(\left(\frac{n_c^2L}{n^2}\right)^2 + Var\Big[\mathcal{R}\Big(f_{\phi_{1,c}}(\widetilde{\boldsymbol{M}})\Big)\Big]\right)\cdot\mathbb{E}\Big[g^2_{\psi_{1,c}}(\widetilde{\boldsymbol{D}})\Big]\\ &\quad + \mathbb{E}\Big[g^2_{\psi_{2,c}}(\widetilde{\boldsymbol{D}})\Big] - \frac{2n_c^2L}{n^2}\cdot\mathbb{E}\Big[g_{\psi_{1,c}}(\widetilde{\boldsymbol{D}})\cdot g_{\psi_{2,c}}(\widetilde{\boldsymbol{D}})\Big],\end{aligned} \tag{17}$$

where

$$\begin{aligned}&Var\Big[\mathcal{R}\Big(f_{\phi_{1,c}}(\widetilde{\boldsymbol{M}})\Big)\Big]\\ &= \frac{1}{2}n_c(n_c-1)\left(\frac{1}{16(p-\frac{1}{2})^2} - \left(\frac{2L}{n(n-1)}-\frac{1}{2}\right)^2\right).\end{aligned} \tag{18}$$

By Taylor expansion at $\mathbb{E}[\widetilde{d}_i]$, we have

$$\mathbb{E}[g^2_{\psi_{1,c}}(\widetilde{\boldsymbol{D}})] = \frac{1}{L^2} + \frac{6n^2}{(1-\alpha)^2\epsilon^2L^4} \tag{19}$$

$$\begin{aligned}\mathbb{E}[g^2_{\psi_{2,c}}(\widetilde{\boldsymbol{D}})] &= n_c^2\Bigg(\frac{1}{n^4} + \frac{32}{n^2(1-\alpha)^2\epsilon^2L^2}\\ &\quad + \frac{264}{(1-\alpha)^4\epsilon^4L^4} + \frac{480n^2}{(1-\alpha)^6\epsilon^6L^6}\Bigg)\end{aligned} \tag{20}$$

$$\begin{aligned}\mathbb{E}[g_{\psi_{1,c}}(\widetilde{\boldsymbol{D}})\cdot g_{\psi_{2,c}}(\widetilde{\boldsymbol{D}})] &= n_c^2\Bigg(\frac{1}{n^2L} + \frac{14}{(1-\alpha)^2\epsilon^2L^3}\\ &\quad + \frac{24n^2}{(1-\alpha)^4\epsilon^4L^5}\Bigg).\end{aligned} \tag{21}$$

By substituting Eqs. (18), (19), (20), and (21) into Eq. (17), we have

$$\begin{aligned}\mathbb{E}[\widetilde{F}_c^2] &\approx \frac{n_c(n_c-1)((1-\alpha)^2\epsilon^2L^2+6n^2)}{2(1-\alpha)^2\epsilon^2L^4}\Bigg(\frac{1}{16(\frac{e^{\alpha\epsilon}}{1+e^{\alpha\epsilon}}-\frac{1}{2})^2}\\ &\quad - \left(\frac{2L}{n(n-1)}-\frac{1}{2}\right)^2\Bigg) + \frac{n_c^2-n_c^4}{n^4}.\end{aligned}$$

To minimize $\mathbb{E}[\widetilde{F}_c^2]$, we omit items $\frac{n_c(n_c-1)}{2}$ and $\frac{n_c^2-n_c^4}{n^4}$ that are independent of $\alpha$, and derive $\alpha$ as

$$\arg\min_{\alpha\in(0,1)} \frac{(1-\alpha)^2\epsilon^2L^2 + 6n^2}{(1-\alpha)^2\epsilon^2L^4}\left(\frac{1}{16(\frac{e^{\alpha\epsilon}}{1+e^{\alpha\epsilon}}-\frac{1}{2})^2} - \left(\frac{2L}{n(n-1)}-\frac{1}{2}\right)^2\right).$$

□

Similar to obtaining $\hat{d}$ for clustering coefficient estimation in Section 5, the total number of edges $L$ in graph can be obtained by using a portion of privacy budget for a preliminary round of node degree perturbation to collect $\widetilde{\boldsymbol{D}}$ and estimate $L$.

### 6.2 Overall Algorithm

Algorithm 3 summarizes how the data collector estimates the modularity of a given community $\mathcal{C}$, according to the perturbed adjacency matrix $\widetilde{\boldsymbol{M}}$ and the degree vector $\widetilde{\boldsymbol{D}}$. First, it obtains $\widetilde{L}_c$, the number of edges in $\mathcal{C}$, by counting and halving the number of "1"s in $\widetilde{\boldsymbol{M}}_c$, the sub-matrix of $\mathcal{C}$ extracted from $\widetilde{\boldsymbol{M}}$ (Lines 1-2). It then calibrates $\widetilde{L}_c$ to an unbiased estimation $L_c$ based on Eq. (16) (Line 3). Finally, the estimated modularity is calculated according to Eq. (15) (Line 6), which is based on $L_c$, $L$ (obtained from Line 4) and $K_c$ (obtained from Line 5).

**Algorithm 3.** Collector-Side Modularity Estimation

**Input:** A community $\mathcal{C}$
Perturbed adjacency matrix $\widetilde{\boldsymbol{M}} = \{\widetilde{\boldsymbol{B}}_1, \ldots, \widetilde{\boldsymbol{B}}_n\}$
Perturbed degree vector $\widetilde{\boldsymbol{D}} = \{\widetilde{d}_1, \ldots, \widetilde{d}_n\}$
Percentage $\alpha$ for privacy budget allocation
**Output:** $q_c = EstMod(\cdot)$, the estimated modularity of $\mathcal{C}$
**Procedure:**
1: Extract a sub-matrix $\widetilde{\boldsymbol{M}}_c$ from $\widetilde{\boldsymbol{M}}$
2: Obtain $\widetilde{L}_c$ by counting and halving the number of "1"s in $\widetilde{\boldsymbol{M}}_c$
3: Calibrate $\widetilde{L}_c$ to get an unbiased one $L_c$ according to Eq. (16), where $p = \frac{e^{\alpha\epsilon}}{1+e^{\alpha\epsilon}}$.
4: Calculate the total number of edges in the whole graph $L = \frac{1}{2}\sum_{i=1}^{n}\widetilde{d}_i$
5: Calculate the total degree of all node in $\mathcal{C}$: $K_c = \sum_{c\in\mathcal{C}} d_c$
6: Calculate the estimated modularity of $\mathcal{C}$: $q_c = \frac{L_c}{L} - \frac{K_c^2}{4L^2}$
7: **return** $q_c$

*Accuracy Guarantee*. According to Theorem 6.1, with at least $1-\beta$ probability, the error of modularity estimation is bounded by $O(\frac{\sqrt{log(1/\beta)}}{n^2\cdot\epsilon})$.

Now that the modularity of any community can be estimated by Algorithm 3, we can adopt existing community detection methods that are based on modularity maximization [31]. In essence, they attempt to find a graph partition with the highest overall modularity of all communities. For ease of reference, Algorithm 4 presents the detailed implementation of *Louvain* method [31], a popular community detection method under LF-GDPR, where Algorithm 3 serves as the routine for modularity estimation.

As shown in Algorithm 4, there are two iterative phases in *Louvain*. In the first phase, the data collector assigns a different community to each node and calculates its modularity by invoking $EstMod(\cdot)$, i.e., Algorithm 3 (Lines 1-2). Then for each node $i$, the data collector calculates the gain of modularity that would take place by moving $i$ to the community of its neighbor $j$ (Line 5). Here $EstMod(\cdot)$ is invoked again to estimate the modularity of community $\{i, j\}$. Node $i$ is then moved into the community in which this gain is positive and maximum (Line 7), and then the modularity of this community is also updated (Line 8). This process is repeated for all nodes until no individual move can improve the total modularity of the graph. The result of the first phase is a new set of communities (Line 10). In the second phase, a new graph is formed from this set of communities, and the data collector repeats the process in the first phase to detect the final set of communities (Line 11).

**Algorithm 4.** Community Detection Under LF-GDPR With *Louvain* Method

**Input:** Perturbed adjacency matrix $\widetilde{\boldsymbol{M}} = \{\widetilde{\boldsymbol{B}}_1, \ldots, \widetilde{\boldsymbol{B}}_n\}$
Perturbed degree vector $\widetilde{\boldsymbol{D}} = \{\widetilde{d}_1, \ldots, \widetilde{d}_n\}$
Privacy budget for adjacency bit vector perturbation $\epsilon_1$
**Output:** A set of detected communities $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots\}$
**Procedure:**

1: Initialize $n$ communities $\{\mathcal{C}_i | 1 \leq i \leq n\}$, each consisting of only one node
2: Estimate the modularity of each $\mathcal{C}_i$: $q_i = EstMod(\mathcal{C}_i, \widetilde{\boldsymbol{M}}, \widetilde{\boldsymbol{D}}, \epsilon_1)$
3: **for** each node $i \in \{1, 2, \ldots, n\}$ **do**
4: **for** each node $j$ so that $\widetilde{M}_{ij} = 1$ **do**
5: Calculate gain of modularity:

$$\Delta q_{ij} = EstMod(\{i \cup \mathcal{C}_j\}, \widetilde{\boldsymbol{M}}, \widetilde{\boldsymbol{D}}, \epsilon_1) - q_i - q_j$$

6: **end for**
7: Move $i$ to the community of $j$, where $j = \arg\max\{\Delta q_{ij} | \Delta q_{ij} > 0\}$
8: Update the modularity of $\mathcal{C}_j$: $q_j = EstMod(\mathcal{C}_j, \widetilde{\boldsymbol{M}}, \widetilde{\boldsymbol{D}}, \epsilon_1)$
9: **end for**
10: Repeat Lines 3-7 until no individual move can improve the total modularity, and obtain a new set of communities $\mathcal{C}^*$
11: Build a graph from $\mathcal{C}^*$, and repeat Lines 3-8 to obtain $\mathcal{C}$
12: **return** $\mathcal{C}$

## 7 Experimental Evaluation

In this section, we compare the performance of LF-GDPR with two alternative methods, i.e., *RABV-only* and *LDPGen* [11] in both use cases, namely, clustering coefficient estimation and modularity estimation for community detection. In LF-GDPR, the optimal $\alpha$ for clustering coefficient estimation and modularity estimation is derived Theorems 5.1 and 6.1, respectively. Since the derivation is independent of the ground-truth data, we use this optimal $\alpha$ unless stated otherwise. *RABV-only* is a baseline solution where each node spends all its privacy budget in the *RABV* protocol and then derives her node degree from the perturbed adjacency bit vector. As for *LDPGen*, since it needs the clustering coefficient to generate a synthetic graph, we choose the most favorable one for it, i.e., the ground truth value. To have a fair comparison with *RABV-only* and *LDPGen*, for LF-GDPR, we use 10 percent of the privacy budget to estimate the domain knowledge in both use cases, i.e., the representative degree in Theorem 5.1 and the total number of graph edges in Theorem 6.1. All experiments run in Java on a desktop computer with Intel Core i7-8700K CPU, 64G RAM running Windows 10. The code of LF-GDPR and datasets are available in GitHub at https://github.com/Vicky-cs/LF-GDPR.

*Performance Measures*. For the first use case, we measure the *Mean Square Error* (MSE) of the clustering coefficients of all nodes, i.e., $\frac{1}{n}\sum_{i=1}^{n}(cc_i - \widetilde{cc}_i)^2$. For the second use case, to evaluate the modularity estimation, we measure the *Relative Error* (RE) between the ground-truth modularity $q$ and estimated modularity $\widetilde{q}$ of one community or all communities in a graph partition, i.e., $\frac{|q-\widetilde{q}|}{q}$. To evaluate the final community detection results, we adopt the same classic metrics for cluster validation as used in [11], namely *Adjusted Random Index* (ARI) [39] and *Adjusted Mutual Information* (AMI) [40]. They measure the similarity of two clusterings, and a larger ARI or AMI value indicates more similarity between them.

*Datasets*. We use four public datasets [41]. The first two are used in [11], and the rest two are added to evaluate on denser and larger graphs.

(1) *Facebook*—an undirected social network of 4,039 nodes and 88,234 edges, from a survey of participants in Facebook app.
(2) *Enron*—an undirected email communication network of 36,692 nodes and 183,831 edges.
(3) *AstroPh*—an undirected collaboration network of 18,772 authors and 198,110 edges indicating collaborations between authors in arXiv, who submitted papers to Astro Physical category.
(4) *Gplus*—an undirected social network of 107,614 Google+ users and 12,238,285 edges indicating shares of social circles.[6]

### 7.1 Clustering Coefficient Estimation

Fig. 5 shows the clustering coefficient estimation accuracy of LF-GDPR and two alternative methods over all datasets, with privacy budget $\epsilon$ varying from 1 to 8. In all cases, LF-GDPR is the most accurate. Furthermore, it always significantly outperforms *RABV-only*, which justifies our rationale in Section 3.1 that node degree derived from perturbed adjacency bit vector is too noisy. As $\epsilon$ increases, the accuracy of LF-GDPR and *RABV-only* improves significantly while *LDPGen* does not. This is because *LDPGen* is only affected by the Laplace noise added to node degree, which is already

6. The original *Gplus* dataset is a directed graph, and we convert it to an undirected graph to align with the other three datasets.

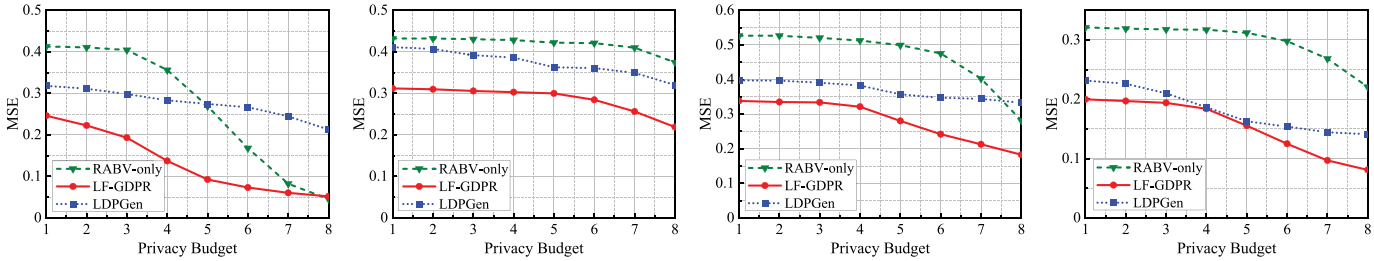



Fig. 5. Mean square error of clustering coefficient estimation.

very small when $\epsilon > 2$. In other words, *LDPGen* cannot fully exploit a large privacy budget.

To evaluate the impact of privacy budget allocation on the estimation accuracy, we compare LF-GDPR with optimal allocation (derived from Eq. (11)) against LF-GDPR with four constant $\alpha$, namely, 0.3, 0.5, 0.7 and 0.9 in Fig. 6. Due to the space limitation, we only show the results of *Facebook*. The optimal allocation achieves the lowest MSE in most cases. As for the constant $\alpha$, we observe that a large $\epsilon$ always favors a large $\alpha$, which indicates that the privacy budget needed by node degree perturbation is relatively stable, and therefore surplus budget should be mostly allocated to the adjacency bit vector. However, when privacy budget is small (e.g., $\epsilon < 2$), large $\alpha$ (e.g., $\alpha = 0.9$) leads to high MSE. The same observation is also made in modularity estimation, which is therefore omitted in the interest of space.

### 7.2 Modularity Estimation and Community Detection

In this experiment, we evaluate the modularity estimation and Louvain-based community detection of LF-GDPR against *RABV-only*, *LDPGen*, and the ground truth. To allow fair comparison, we use the same algorithms for the latter three except that the modularity is estimated from the perturbed adjacency matrix only (for *RABV-only*), or directly calculated from the synthetic graph (for *LDPGen*), or directly calculated from the original graph (for ground truth). Fig. 7 plots the RE of modularity by these three methods against ground truth in all datasets. LF-GDPR always outperforms the other two and its RE approaches 0 as $\epsilon$ increases, especially in *Facebook* and *Gplus* which have a higher mean degree than the other two datasets. *RABV-only* has the second lowest RE when $\epsilon$ is large, especially in *Facebook*, which means when the privacy budget is sufficient, adjacency bit vector alone can also estimate modularity fairly well. However, when $\epsilon$ is small, *RABV-only* has the highest RE among the three, which justifies our rationale in Section 3.1 that the estimated degree from a perturbed adjacency matrix could be too noisy to be meaningful. *LDPGen*, on the other hand, still has very high RE even when $\epsilon$ is large, which also justifies our rationale in Section 3.1 that the neighborhood information is lost in a synthetic graph.

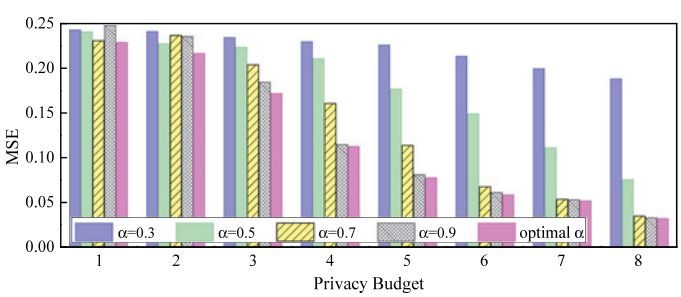



Fig. 6. Mean square error of clustering coefficient estimation, varying $\alpha$.

To compare the detected communities against ground truth, we plot ARI and AMI between the estimated and ground-truth graph partitions of each method[7] in Fig. 8. Due to space limitation, we only show the results of *Facebook* and *Enron*. LF-GDPR achieves higher ARI and AMI than *LDPGen* when $\epsilon > 1$, which means the detected communities by LF-GDPR are closer to the ground truth communities detected in the original graph. Particularly, in *Facebook* both ARI and AMI of LF-GDPR approach 1 for large $\epsilon$ (e.g., $\epsilon \geq 7$), which means that the detected communities are almost identical to the ground truth communities. We can also verify this observation from a visualization tool *Gephi* in Fig. 9, which illustrates three sets of communities detected from the original graph and from LF-GDPR ($\epsilon = 8$, $\epsilon = 1$) respectively. The sizes of top-3 communities in each set are also marked.

On the other hand, as with the RE results, *LDPGen* has steady ARI/AMI curves because it does not have the neighborhood information of the original graph. As such, it becomes significantly inferior to LF-GDPR when there is a large privacy budget to spend. Dataset-wise, both LF-GDPR and *LDPGen* perform better in *Facebook* than in *Enron*. This is because *Enron* is more sparse and therefore has more communities—1275 versus 16 in *Facebook*.

In addition, we evaluate the accuracy of modularity estimation with respect to the size of a community. For datasets *Facebook* and *Enron*, we randomly select 500 small (5 percent of the total nodes) communities and 500 large (20 percent of the total nodes) communities. Then we apply both LF-GDPR and *RABV-only* to estimate the modularity of each community and measure its RE against the ground truth modularity of that community. Due to space limitation, Fig. 10 only shows the results of *Facebook* and *Enron*. LF-GDPR significantly outperforms *RABV-only* in both small and large communities, due to the excessive noise in the node degree introduced by *RABV-only*. We also observe that both methods work better for smaller communities and

7. For *LDPGen**, we use the results directly from [11] because the calculation of ARI and AMI between partitions from two (similar) graphs requires an optimal node-to-cluster mapping, which is not specified in [11].

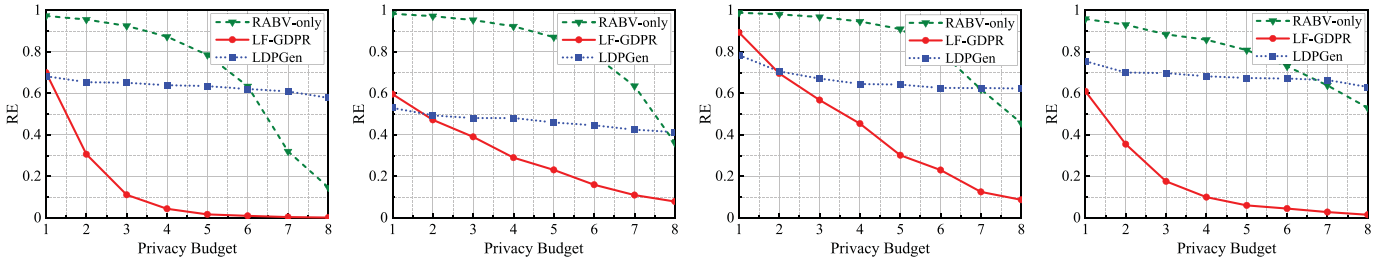



Fig. 7. Relative error of modularity of detected communities.

for the *Facebook* dataset (than the *Enron* dataset). We believe this indicates that LF-GDPR is more superior for denser graphs with more edges per node.

To evaluate the communication bandwidth cost, we show the number of kilobytes (kB) between a node and the data collector for all datasets in Table 2. We observe that all three methods are proportional to the node size $n$, whereas *LDPGen* is also logarithmic to the number of groups $g$, an internal parameter of *LDPGen*. This coincides with the asymptotic complexity—$O(2n + \lceil \log g \rceil n)$ of *LDPGen* versus $O(n)$ of LF-GDPR. As such, we expect *LDPGen* incurs even higher communication cost as the graph becomes larger due to an increasing $g$.

To evaluate the computation cost, we show the runtime of both metric estimation at the collector side in Table 3, with privacy budget $\epsilon$ ranging from 1 to 8. Due to the space limitation, we only show the results of *Facebook*. LF-GDPR and *RABV-only* have comparable runtime and decrease significantly with large $\epsilon$. For small $\epsilon$, the perturbed adjacency bit matrix is very dense, so the computation of metrics becomes time-consuming. On the other hand, *LDPGen* always needs to generate a synthetic graph with almost the same number of edges as the original one, so its runtime is independent of $\epsilon$ and is outperformed by LF-GDPR and *RABV-only* except for small $\epsilon$.

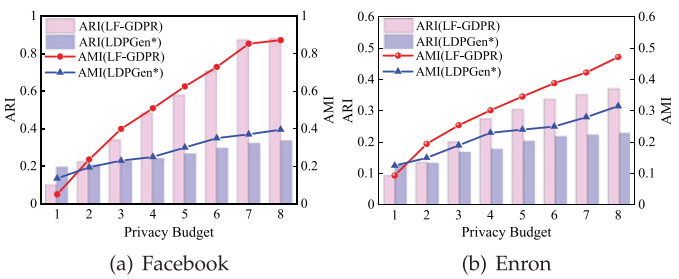



Fig. 8. Results of ARI and AMI.

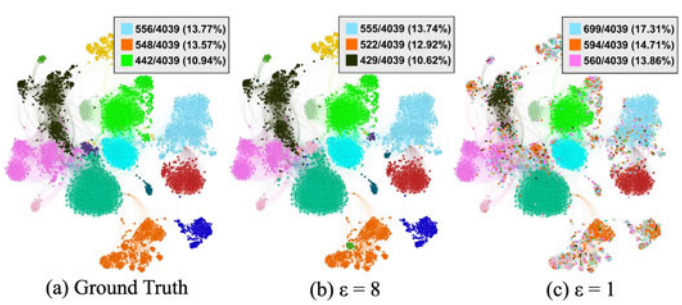



Fig. 9. Visualization of detected communities by *Gephi*.

## 7.3 LF-GDPR VS. Dedicated LDP Solutions

As mentioned in the introduction, dedicated LDP solutions may provide a better utility than a general framework as LF-GDPR. In this subsection, we conduct such a comparative study on clustering coefficient estimation (CCE) and modularity estimation for community detection (CD). The main challenge is the design of local perturbation mechanisms that can provide a global view which is needed for these two graph metrics. To address this, we equip each individual user with sufficient ground truth knowledge and design two optimistic dedicated solutions, i.e., *Dedicated-CCE* and *Dedicated-CD*. In Dedicated-CCE, we assume each user knows the entire ground-truth adjacency matrix, based on which her clustering coefficient is calculated and then perturbed by adding Laplace noise. In Dedicated-CD, we assume each user knows the ground-truth graph partition, based on which her number of edges linked to her community $\mathcal{C}$ is counted, perturbed by adding Laplace noise together with her node degree, and sent to the collector to calculate the modularity $q_c$ by Eq. (2). Note that these two dedicated solutions provide the same $\epsilon$-edge LDP guarantee as LF-GDPR. But since they optimistically assume to know the ground truth, their estimation accuracy only serves as the upper bound of dedicated solutions for CCE and CD.

Figs. 11a and 11b show the mean square error of clustering coefficient estimation of *Dedicated-CCE* and LF-GDPR on *Facebook* and *Enron* datasets. In the interest of space, the

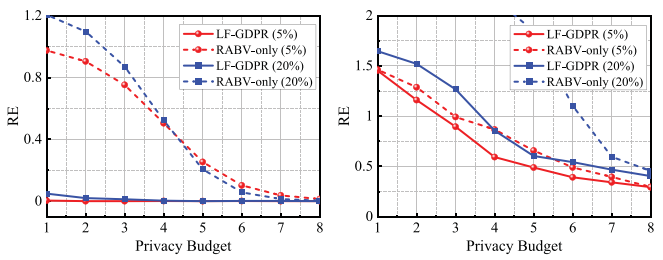



Fig. 10. Impact of community size on modularity estimation.

TABLE 2
Communication Bandwidth Cost (in kilobytes)

| Dataset | LF-GDPR | *RABV-only* | *LDPGen* |
|---|---|---|---|
| **Facebook** | 0.25 | 0.25 | 3.05 |
| **Enron** | 2.30 | 2.29 | 27.55 |
| **AstroPh** | 1.18 | 1.17 | 14.10 |
| **Gplus** | 6.73 | 6.73 | 80.73 |

TABLE 3
Runtime in Data Collector Side for Facebook(in milliseconds)

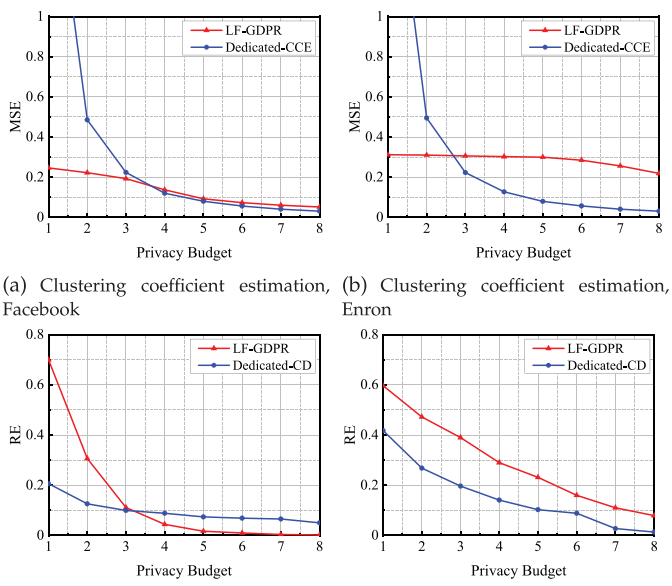
| Privacy Budget | Clustering Coefficient Estimation | | | Modularity Estimation | | |
|---|---|---|---|---|---|---|
| | LF-GDPR | *RABV-only* | LDPGen | LF-GDPR | *RABV-only* | LDPGen |
| 1 | 12,686 | 11,715 | 910 | 120 | 331 | 842 |
| 2 | 2,273 | 1,825 | 929 | 76 | 82 | 866 |
| 3 | 469 | 453 | 942 | 52 | 55 | 882 |
| 4 | 225 | 276 | 892 | 35 | 36 | 845 |
| 5 | 105 | 143 | 959 | 32 | 33 | 906 |
| 6 | 100 | 146 | 957 | 29 | 30 | 903 |
| 7 | 85 | 102 | 949 | 28 | 28 | 883 |
| 8 | 79 | 96 | 926 | 28 | 28 | 880 |

results on other datasets are omitted. We observe that *Dedicated-CCE* has very large MSE when the privacy budget is small, and it gradually outperforms LF-GDPR when $\epsilon \geq 4$ (on *Facebook*) or $\epsilon \geq 3$ (on *Enron*). But its MSE is at least 64 and 16 percent of the MSE of LF-GDPR on two datasets when $\epsilon = 8$. Figs. 11c and 11d show the relative error of modularity estimation of *Dedicated-CD* and LF-GDPR over *Facebook* and *Enron*. Similar to CCE, there is no all-winner—LF-GDPR performs better on *Facebook* when $\epsilon \geq 3$ whereas *Dedicated-CD* gains higher accuracy (but its RE is at least 20 percent of the RE of LF-GDPR) in other cases. To summarize, we conclude that LF-GDPR is able to obtain comparable estimation accuracy as dedicated LDP solutions for graph metric estimation.

## 8 Related Work

There are three related fields: privacy-preserving graph release, graph analytics with differential privacy, and local differential privacy.

*Privacy-Preserving Graph Release.* This field studies how a data owner publishes a privacy-preserving graph. Early works focus on anonymization techniques under those privacy models derived from *k*-anonymity [42]. Zhou *et al.* proposed *k*-neighborhood anonymity to defend against neighborhood attacks [3], Liu *et al.* proposed *k*-degree anonymity against degree attacks [4], Zou *et al.* and Cheng *et al.* proposed




(a) Clustering coefficient estimation, Facebook (b) Clustering coefficient estimation, Enron

(c) Modularity for community detection, Facebook (d) Modularity for community detection, Enron

Fig. 11. Comparison with dedicated LDP solutions.

*k*-automorphism [5] and *k*-isomorphism [6] respectively against structural attacks, and Xue *et al.* proposed random edge perturbation against walk-based structural identification [43]. As these approaches can be vulnerable to de-anonymization techniques [44], more rigorous privacy notions are proposed, such as *L*-opacity [45] and differential privacy [12]. The former ensures an adversary cannot infer whether the distance between two nodes is equal to or less than $L$. The latter uses a generative graph model to fit the original graph, and then produces a synthetic graph for analytics. Common graph models include *dK-series* [16], Stochastic Kronecker Graph (SKG) model [46], Exponential Random Graph Model (ERGM) [17], Attributed Graph Model (AGM) [18], Hierarchical Random Graph (HRG) [19], and BTER [11] (which adopts LDP).

*Graph Analytics With Differential Privacy.* This field studies how to estimate graph metric and statistics with differential privacy. Most of the existing work focuses on centralized differential privacy. Nissim *et al.* estimated the cost of the minimum spanning tree and the number of triangles in a graph [7]. This technique has been extended to subgraph counting queries [20], [47] such as *k*-stars, *k*-triangles and *k*-cliques, and frequent subgraph mining [8], [48]. Other works estimate the distribution of node degree [14], [21] and clustering coefficient [22]. In the local setting, Sun *et al.* [47] propose to estimate subgraph counts in a decentralized graph. In our previous work [49], we briefly introduce the LF-GDPR framework that estimates generic graph metrics with local differential privacy. This work has advanced our previous work in almost all aspects. First, this work materializes all algorithms in the LF-GDPR framework. Second, it proposes a refinement strategy for degree estimation and an optimal privacy budget allocation. Third, this work shows use cases on two common graph analysis tasks, namely, clustering coefficient estimation and community detection. Last but not the least, this work comprehensively evaluates the proposed algorithms on four public datasets.

*Local Differential Privacy (LDP).* Due to its decentralized nature and no need of a trusted party, LDP becomes increasingly popular in privacy-preserving data collection [10], [50]. Existing works focus on estimating statistics such as frequency [32], [51], [52], mean [28], [30], heavy hitter [35], frequent itemset mining [53], *k*-way marginal release [54], [55], key-value data collection [29], [56] and time-series data collection [57]. Some works also focus on learning problems [30], [58], [59].

## 9 Conclusion

This paper presents a parameterized framework LF-GDPR for privacy-preserving graph metric estimation and analytics with local differential privacy. The building block is a user-side perturbation algorithm, and a collector-side aggregation and calibration algorithm. LF-GDPR simplifies the job of developing a practical LDP solution for a graph analysis task by providing a complete solution for all LDP steps. An optimal allocation of privacy budget between the two atomic metrics is also designed. Through theoretical and experimental analysis, we verify the privacy and data utility achieved by this framework.

As for future work, we plan to extend LF-GDPR to more specific graph types and graph analysis tasks, such as

attributed graph and DAG, and influential node analysis, to demonstrate its wide applicability. We will also investigate some relaxation of DP, such as Gaussian Mechanism and $(\epsilon, \delta)$-DP, to provide higher estimation accuracy and better utility. Graph-specific tighter bounds for the composition of DP [60] and the correlation of graph data will also be studied.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No: 62072390, U1636205, 91646203, 61941121 and 61972332), the Research Grants Council, Hong Kong SAR, China (Grant No: 15238116, 15222118, 15218919, 15203120 and C1008-16G), the Ministry of Education, Singapore (Grant No: MOE2018-T2-2-091).

## References

[1] B. Stephanie, Facebook Scandal a 'Game Changer' in Data Privacy Regulation, *Bloomberg*, Apr. 8, 2018. [Online]. Available: https://www.bloomberg.com/news/articles/2018-04-07/facebook-scandal-a-game-changer-in-dataprivacy-regulation

[2] Facebook, 2020. [Online]. Available: https://developers.facebook.com/docs/graph-api/

[3] B. Zhou and J. Pei, "Preserving privacy in social networks against neighborhood attacks," in *Proc. IEEE 24th Int. Conf. Data Eng.*, 2008, pp. 506–515.

[4] K. Liu and E. Terzi, "Towards identity anonymization on graphs," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2008, pp. 93–106.

[5] L. Zou, L. Chen, and M. T. Özsu, "K-automorphism: A general framework for privacy preserving network publication," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 946–957, 2009.

[6] J. Cheng, A. W. Fu, and J. Liu, "K-isomorphism: Privacy preserving network publication against structural attacks," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2010, pp. 459–470.

[7] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," in *Proc. 39th Annu. ACM Symp. Theory Comput.*, 2007, pp. 75–84.

[8] S. Xu, S. Su, L. Xiong, X. Cheng, and K. Xiao, "Differentially private frequent subgraph mining," in *Proc. IEEE 32nd Int. Conf. Data Eng.*, 2016, pp. 229–240.

[9] C. Wei, S. Ji, C. Liu, W. Chen, and T. Wang, "ASGLDP: Collecting and generating decentralized attributed graphs with local differential privacy," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 3239–3254, Apr. 2020.

[10] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *Proc. IEEE 54th Annu. Symp. Found. Comput. Sci.*, 2013, pp. 429–438.

[11] Z. Qin, T. Yu, Y. Yang, I. Khalil, X. Xiao, and K. Ren, "Generating synthetic decentralized social graphs with local differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 425–438.

[12] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. Theory Cryptogr. Conf.*, 2006, pp. 265–284.

[13] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith, "What can we learn privately?," *SIAM J. Comput.*, vol. 40, no. 3, pp. 793–826, 2011.

[14] S. P. Kasiviswanathan, K. Nissim, S. Raskhodnikova, and A. Smith, "Analyzing graphs with node differential privacy," in *Proc. Theory Cryptogr. Conf.*, 2013, pp. 457–476.

[15] J. Blocki, A. Blum, A. Datta, and O. Sheffet, "The johnson-lindenstrauss transform itself preserves differential privacy," in *Proc. IEEE 53rd Annu. Symp. Found. Comput. Sci.*, 2012, pp. 410–419.

[16] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Y. Zhao, "Sharing graphs using differentially private graph models," in *Proc. ACM SIGCOMM Conf. Internet Meas. Conf.*, 2011, pp. 81–98.

[17] W. Lu and G. Miklau, "Exponential random graph estimation under differential privacy," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 921–930.

[18] Z. Jorgensen, T. Yu, and G. Cormode, "Publishing attributed social graphs with formal privacy guarantees," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2016, pp. 107–122.

[19] Q. Xiao, R. Chen, and K. Tan, "Differentially private network data release via structural inference," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 911–920.

[20] V. Karwa, S. Raskhodnikova, A. Smith, and G. Yaroslavtsev, "Private analysis of graph structure," *Proc. VLDB Endowment*, vol. 4, no. 11, pp. 1146–1157, 2011.

[21] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate estimation of the degree distribution of private networks," in *Proc. 9th IEEE Int. Conf. Data Mining*, 2009, pp. 169–178.

[22] Y. Wang, X. Wu, J. Zhu, and Y. Xiang, "On learning cluster coefficient of private networks," *Soc. Netw. Anal. Mining*, vol. 3, no. 4, pp. 925–938, 2013.

[23] X. Xu, N. Yuruk, Z. Feng, and T. A. J. Schweiger, "SCAN: A structural clustering algorithm for networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2007, pp. 824–833.

[24] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd Ed., Burlington, MA, USA: Morgan Kaufmann, 2011.

[25] T. Martin, X. Zhang, and M. Newman, "Localization and centrality in networks," *Phys. Rev. E*, vol. 90, no. 5, 2014, Art. no. 052808.

[26] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *J. Amer. Statist. Assoc.*, vol. 60, no. 309, pp. 63–69, 1965.

[27] C. Seshadhri, T. G. Kolda, and A. Pinar, "Community structure and scale-free collections of erdős-rényi graphs," *Phys. Rev. E*, vol. 85, no. 5, 2012, Art. no. 056109.

[28] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 3574–3583.

[29] Q. Ye, H. Hu, X. Meng, and H. Zheng, "PrivKV: Key-value data collection with local differential privacy," in *Proc. IEEE Symp. Security Privacy*, 2019, pp. 317–331.

[30] N. Wang *et al.*, "Collecting and analyzing multidimensional data with local differential privacy," in *Proc. IEEE 35th Int. Conf. Data Eng.*, 2019, pp. 638–649.

[31] V. D. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Statist. Mech.: Theory Experiment*, vol. 2008, no. 10, 2008, Art. no. P10008.

[32] T. Wang, J. Blocki, N. Li, and S. Jha, "Locally differentially private protocols for frequency estimation," in *Proc. 26th USENIX Conf. Secur. Symp.*, 2017, pp. 729–745.

[33] P. Kairouz, K. Bonawitz, and D. Ramage, "Discrete distribution estimation under local privacy," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 2436–2444.

[34] S. S. Wilks, "The large-sample distribution of the likelihood ratio for testing composite hypotheses," *Ann. Math. Statist.*, vol. 9, no. 1, pp. 60–62, 1938.

[35] Z. Qin, Y. Yang, T. Yu, I. Khalil, X. Xiao, and K. Ren, "Heavy hitter estimation over set-valued data with local differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 192–203.

[36] R. Fletcher, *Practical Methods of Optimization*. Hoboken, NJ, USA: Wiley, 2013.

[37] R. Chen, B. C. Fung, S. Y. Philip, and B. C. Desai, "Correlated network data publication via differential privacy," *VLDB J.*, vol. 23, no. 4, pp. 653–676, 2014.

[38] B. Yang, I. Sato, and H. Nakagawa, "Bayesian differential privacy on correlated data," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2015, pp. 747–762.

[39] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *J. Amer. Statist. Assoc.*, vol. 66, no. 336, pp. 846–850, 1971.

[40] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Is a correction for chance necessary?," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 1073–1080.

[41] L. Jure and K. Andrej, "SNAP Datasets: Stanford large network dataset collection," 2014. [Online]. Available: http://snap.stanford.edu/data

[42] P. Samarati, "Protecting respondents identities in microdata release," *IEEE Trans. Knowl. Data Eng.*, vol. 13, no. 6, pp. 1010–1027, Nov./Dec. 2001.

[43] M. Xue, P. Karras, R. Chedy, P. Kalnis, and H. Pung, "Delineating social network data anonymization via random edge perturbation," in *Proc. 21st ACM Int. Conf. Inf. Knowl. Manage.*, 2012, pp. 475–484.

[44] A. Narayanan and V. Shmatikov, "De-anonymizing social networks," in *Proc. IEEE Symp. Secur. Privacy*, 2009, pp. 173–187.

[45] S. Nobari, P. Karras, H. Pang, and S. Bressan, "L-opacity: Linkage-aware graph anonymization," in *Proc. Int. Conf. Extending Database Technol.*, 2014, pp. 583–594.

[46] D. Mir and R. N. Wright, "A differentially private estimator for the stochastic kronecker graph model," in *Proc. Joint EDBT/ICDT Workshops*, 2012, pp. 167–176.
[47] H. Sun *et al.*, "Analyzing subgraph statistics from extended local views with decentralized differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2019, pp. 703–717.
[48] E. Shen and T. Yu, "Mining frequent graph patterns with differential privacy," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2013, pp. 545–553.
[49] Q. Ye, H. Hu, M. H. Au, X. Meng, and X. Xiao, "Towards locally differentially private generic graph metric estimation," in *Proc. IEEE 36th Int. Conf. Data Eng.*, 2020, pp. 1922–1925.
[50] Q. Ye and H. Hu, "Local differential privacy: Tools, challenges, and opportunities," in *Proc. Int. Conf. Web Inf. Syst. Eng.*, 2020, pp. 13–23.
[51] Ú. Erlingsson, V. Pihur, and A. Korolova, "RAPPOR: Randomized aggregatable privacy-preserving ordinal response," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2014, pp. 1054–1067.
[52] R. Bassily and A. Smith, "Local, private, efficient protocols for succinct histograms," in *Proc. 47th Annu. ACM Symp. Theory Comput.*, 2015, pp. 127–135.
[53] T. Wang, N. Li, and S. Jha, "Locally differentially private frequent itemset mining," in *Proc. IEEE Symp. Secur. Privacy*, 2018, pp. 127–143.
[54] G. Cormode, T. Kulkarni, and D. Srivastava, "Marginal release under local differential privacy," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2018, pp. 131–146.
[55] Z. Zhang, T. Wang, N. Li, S. He, and J. Chen, "CALM: Consistent adaptive local marginal for marginal release under local differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2018, pp. 212–229.
[56] X. Gu, M. Li, L. Xiong, and Y. Cao, "PCKV: Locally differentially private correlated key-value data collection with optimized utility," in *Proc. 29th USENIX Secur. Symp.*, 2020, pp. 967–984.
[57] Q. Ye, H. Hu, N. Li, X. Meng, H. Zheng, and H. Yan, "Beyond value perturbation: Local Differential privacy in the temporal setting," in *Proc. IEEE Int. Conf. Comput. Commun.*, to be published.
[58] H. Zheng, Q. Ye, H. Hu, C. Fang, and J. Shi, "BDPL: A boundary differentially private layer against machine learning model extraction attacks," in *Proc. Eur. Symp. Res. Comput. Secur.*, 2019, pp. 66–83.
[59] H. Zheng, Q. Ye, H. Hu, C. Fang, and J. Shi, "Protecting decision boundary of machine learning model with differentially private perturbation," *IEEE Trans. Dependable Secure Comput.*, to be published, doi: 10.1109/TDSC.2020.3043382.
[60] P. Kairouz, S. Oh, and P. Viswanath, "The composition theorem for differential privacy," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 1376–1385.

**Qingqing Ye** (Member, IEEE) received the PhD degree in computer science from the Renmin University of China, China, in 2020. She is a research assistant professor with the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong. She has received several prestigious awards, including China National Scholarship, Outstanding Doctoral Dissertation Award, and IEEE S&P Student Travel Award. Her research interests include data privacy and security, and adversarial machine learning.

**Haibo Hu** (Senior Member, IEEE) is an associate professor with the Department of Electronic and Information Engineering, Hong Kong Polytechnic University, Hong Kong. His research interests include cybersecurity, data privacy, Internet of Things, and machine learning. He has published more than 80 research papers in refereed journals, international conferences, and book chapters. As principal investigator, he has received more than 12 million HK dollars of external research grants from Hong Kong and mainland China. He is the recipient of a number of titles and awards, including IEEE MDM 2019 Best Paper Award, WAIM Distinguished Young Lecturer, VLDB Distinguished Reviewer, ACM-HK Best PhD Paper, Microsoft Imagine Cup, and GS1 Internet of Things Award.

**Man Ho Au** (Member, IEEE) is an associate professor of the Department of Computer Science, University of Hong Kong (HKU), Hong Kong. Before joining HKU, he was an associate professor with the Department of Computing, Hong Kong Polytechnic University, Hong Kong. His research interests include applied cryptography, information security, blockchain technology, and related industrial applications. He has published more than 170 refereed papers in top journals and conferences, including CRYPTO, ACM CCS, ACM SIGMOD, NDSS, IEEE TIFS, and TKDE. He is a recipient of the 2009 PET runner-up award for outstanding research in privacy-enhancing technologies, and best paper awards of ACISP 2016, ISPEC 2017, and ACISP 2018. He is a general chair of ASIACCS 2021, an expert member of the China delegation of ISO/IEC JTC 1/SC 27 working group two Cryptography and security mechanisms, and a committee member of the Hong Kong Blockchain Society R&D division.

**Xiaofeng Meng** (Member, IEEE) is a professor in the School of Information, Renmin University of China, China. He is a CCF fellow and the vice chair of the Special Interesting Group on Privacy of China Confidentiality Association(CCA). He has served on the program committee SIGMOD, ICDE, CIKM, MDM, DASFAA, etc., and editorial board of JCST, FCS, JoS, CRAD, etc. His research interests include web data management, cloud data management, mobile data management, and privacy protection.

**Xiaokui Xiao** (Member, IEEE) received the PhD degree in computer science and engineering from the Chinese University of Hong Kong, Hong Kong, in 2008. He is currently an associate professor at the School of Computing, National University of Singapore (NUS), Singapore. His research interests include data privacy and algorithms for large data.