AsgLDP: Collecting and Generating Decentralized Attributed Graphs With Local Differential Privacy

Chengkun Wei, Shouling Ji, *Member, IEEE*, Changchang Liu, Wenzhi Chen, *Member, IEEE*, and Ting Wang

***Abstract*— A large amount of valuable information resides in a decentralized attributed social graph, where each user locally maintains a limited view of the graph. However, there exists a conflicting requirement between publishing an attributed social graph and protecting the privacy of sensitive information contained in each user's local data. In this paper, we aim to collect and generate attributed social graphs in a decentralized manner while providing local differential privacy (LDP) for the collected data. Existing LDP-based synthetic graph generation methods either fail to preserve important graph properties (such as modularity and clustering coefficient) due to excessive noise injection or are unable to process attribute data, thus limiting their adoption and applicability. To overcome these weaknesses, we propose AsgLDP, a novel technique to generate privacy-preserving attributed graph data while satisfying LDP. AsgLDP preserves various graph properties through carefully designing the injected noise and estimating the joint distribution of attribute data. There are two key steps in AsgLDP: 1) collecting and generating graph data while satisfying LDP, and 2) optimizing the privacy-utility tradeoff of the generated data while preserving general graph properties such as the degree distribution, community structure and attribute distribution. Through theoretical analysis as well as experiments over 6 real-world datasets, we demonstrate the effectiveness of AsgLDP in preserving general graph properties such as degree distribution, community structure and attributed community search, while rigorously satisfying LDP. We also show that AsgLDP achieves a superior balance between utility and privacy as compared to the state-of-the-art approaches.**

***Index Terms*— Decentralized social network, synthetic attributed graph generation, local differential privacy, community discovery.**

I. Introduction

SOCIAL network data have been explored in a variety of applications, such as marketing [60], [63], commodity recommendation [64], [65] and disease detection [67], [68]. However, these social network data often contain sensitive information, such as trusted friendships between people, important interactions between friends, business transactions between companies, which thus raises privacy concerns if published directly. In the literature, several centralized privacy-preserving graph generation techniques have been proposed while satisfying rigorous privacy guarantees such as Differential Privacy (DP), which has been accepted as the de facto standard for data privacy in both academia and industry [2], [18], [35]–[37]. Different from centralized differential privacy, we focus on protecting privacy for decentralized social networks where each user only possesses a portion of the graph. Moreover, users in the social network may be associated with various sensitive attributes (e.g., age, location and sexual preference). For example, in an infectious disease surveillance system, we need to collect the report of each user's health condition (e.g., pneumonia, influenza and bronchitis), and each user's contact network, which is the web of interactions through which diseases spread. Thus, to gain knowledge of decentralized social graphs without leaking sensitive information, it is essential to collect these sensitive local views with strong privacy guarantees.

Local differential privacy (LDP) has been widely adopted for collecting distributed data by technology companies such as Google [11], Apple [15], Samsung [13], etc. However, existing works on LDP graph learning and synthesis focused on modeling network structure alone, without taking into account node attributes and their relations with the graph structure [5], [74]. Real-world social graphs are usually associated with node attributes and do exhibit correlation between node attributes. For example, attributed social graphs are well known to have the properties of *homophily* [38] and *social influence* [28], where the homophily indicates that nodes with

Manuscript received October 18, 2019; revised March 15, 2020; accepted March 16, 2020. Date of publication April 3, 2020; date of current version April 29, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB0804102, in part by NSFC under Grant 61772466, Grant U1936215, and Grant U1836202, in part by the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under Grant LR19F020003, in part by the Zhejiang Provincial Natural Science Foundation under Grant LSY19H180011, in part by the Zhejiang Provincial Key Research and Development Program under Grant 2019C01055, in part by the Ant Financial Research Funding, in part by the Information Technology Center of Zhejiang University, and in part by the Alibaba-ZJU Joint Research Institute of Frontier Technologies. The work of Changchang Liu was supported in part by the U.S. Army Combat Capabilities Development Command Army Research Laboratory and was accomplished (ARL Cyber Security CRA) under Agreement W911NF-13-2-0045. The work of Ting Wang was supported in part by the National Science Foundation under Grant 1910546, Grant 1953813, and Grant 1846151. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Mauro Conti. *(Corresponding authors: Shouling Ji; Wenzhi Chen.)*

Chengkun Wei and Wenzhi Chen are with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China (e-mail: weichengkun@zju.edu.cn; chenwz@zju.edu.cn).

Shouling Ji is with the Institute of Cyberspace Research, Zhejiang University, Hangzhou 310027, China, also with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China, and also with the Alibaba-Zhejiang University Joint Institute of Frontier Technologies (AZFT), Hangzhou 310058, China (e-mail: sji@zju.edu.cn).

Changchang Liu is with the Department of Distributed AI, IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598 USA (e-mail: changchang.liu33@ibm.com).

Ting Wang is with the College of Information Science and Technology, Pennsylvania State University, University Park, PA 16802 USA (e-mail: inbox.ting@gmail.com).

Digital Object Identifier 10.1109/TIFS.2020.2985524

similar attributes are likely to form connections, and the social influence indicates that the connected nodes are likely to have similar attributes. In addition, information diffusion in social networks can be influenced by both the graph structure and node attributes [18]. However, these correlation relationships have never been explored by existing LDP based privacy-preserving graph data generation methods.

In this work, we develop a novel attributed graph synthesizing method named as AsgLDP. To the best of our knowledge, AsgLDP is the first work that leverages LDP to protect both graph structural characteristics and node attributes.

Through carefully designing the injected noise, AsgLDP first collects local graph structure and node attributes in a decentralized manner while rigorously satisfying LDP. Then, it computes the unbiased degree distribution of the original graph and estimates the joint distribution of node attributes. Next, AsgLDP constructs a seed attributed graph based on the collected aggregate information, and further optimizes the utility-privacy trade off of the generated data in order to preserve general graph properties. Furthermore, we leverage 6 real-world social network datasets to demonstrate the effectiveness of AsgLDP in preserving general graph properties: 1) graph structure properties such as degree distribution, modularity and clustering coefficient; and 2) attribute properties such as distribution of node attributes and attributed community search.

In summary, our work makes the following contributions.

1) We present AsgLDP, a novel and effective framework to collect and generate attributed social graph (social graph structure associated with attribute information) under LDP in a decentralized manner, which is the first such attempt to our best knowledge.
2) AsgLDP first collects unbiased aggregate information of the original decentralized attributed graph while rigorously satisfying LDP. Specifically, we uniquely propose Random Jump to collect node degrees and leverage random attribute list to collect attribute information. Based on the collected aggregate information, AsgLDP further optimizes the utility of the generated graph so that general graph properties can be preserved.
3) Through theoretical analysis as well as experiments over multiple real-world datasets, we demonstrate that AsgLDP framework can be easily combined with existing LDP mechanisms and graph models for generating synthetic attributed social graphs with LDP guarantees. Furthermore, AsgLDP has shown significant advantages over the state-of-the-art methods in preserving degree distribution, community structure, attribute distribution and attributed community search.

II. Preliminaries

A. Attributed Graph Model

The rich information in a social network can be described by a graph in which nodes represent the users, edges represent the relations between them, and feature vectors associated with the nodes represent the attributes [10]. Such a graph is often referred to as an *attributed graph* [19].

An attributed graph $G = \langle V, E, X\rangle$ comprises a set of n_v nodes V, a set of n_e edges $E \subset V \times V$, and a set of n_v w-dimensional feature vectors X where w is the dimension of node attributes. An edge e_{ij} indicates a relationship existing between nodes v_i and v_j. The degree d_i of a node v_i is defined by the number of nodes that it is connected to. A node v_i's *neighbor list* can be denoted as an n_v-dimension bit binary vector $L_i = [l_1, \cdots, l_{n_v}]$, i.e., $l_j = 1/0$, where $j = 1, \cdots, n_v$. A node v_i is associated with a w-dimensional attribute vector $X_i = [x_{1i}, \cdots, x_{wi}]$ where x_{wi} is the w-th attribute value of node v_i.

A graph $G = \langle V, E, X\rangle$ can be divided into multiple communities $C = [C_1, \cdots, C_K]$. A *community* $C_k \in C, k \in [1, K]$ of graph G is defined as a subgraph consisting of a set of nodes in C_k, and their corresponding edges $e_{ij} \in E, \{v_i, v_j\} \in C_k$. We denote by $B_k \in B$ the set of *boundary* nodes of community C_k, which do not belong to C_k but are connected to at least one node in C_k, i.e., $e_{cb} \in E, v_c \in C_k, v_b \in B_k, C_k \cap B_k = \emptyset$.

In this work, we focus our analysis on binary attributes in undirected attributed graphs, while our method can be easily extended to general attributes and directed attributed graphs.

B. Local Differential Privacy

Differential privacy [1]–[3] was originally designed for a *centralized* setting, where a trusted curator aims to randomize the query output so that the privacy risk to an individual record is bounded to a given level. However, the data curator may not be trusted in many practical scenarios. For example, a malicious curator may sell data for profit or the curator may be attacked by hackers who is thus unable to properly protect the data. To address this issue, local differential privacy (LDP) [4] has been proposed where there is no trusted data curator. To protect privacy, each data owner locally perturbs his/her data using a randomized mechanism, and then sends the sanitized version to the curator.

Formally, let us denote the whole database as D, a randomized algorithm as M which takes a value t as input and z as output. We name the input value domain of M as the perturbation domain. Under a given privacy parameter $\epsilon > 0$, ϵ-LDP is defined as follows.

Definition 1 (ϵ-LDP [4]): A randomized algorithm M satisfies ϵ-LDP, if and only if for any two input values $t, t' \in D$ and any output z, the following inequality always holds.

$$Pr[M(t) = z] \leq e^{\epsilon} \times Pr[M(t') = z]$$

Intuitively, ϵ-LDP means that by observing the output z, the data curator cannot infer whether the input value is t or t' with high confidence (controlled by ϵ). Comparing to the setting that requires a trusted data curator, the local setting offers a stronger level of protection, because the aggregator sees only perturbed data. Even if the aggregator is malicious and colludes with all other participants, one individual's private data is still protected according to the guarantee of LDP. However, this enhancement of privacy protection usually leads to worse utility performance, which constitutes one disadvantage of LDP. Another disadvantage of LDP is that

it requires each individual user to perturb his/her local data before sharing while satisfying LDP, thus incurring extra overhead in practical deployment.

1) Random Response: Randomized response (RR) has been widely adopted for achieving LDP [11], [12]. Specifically, RR asks each user a sensitive question whose answer can be either *yes* or *no*. Each user gives the genuine answer with probability p and the opposite answer with probability $1-p$. The objectives of RR are that (i) each user answers the question with plausible deniability, and (ii) the data curator can compute an unbiased estimate of the ratio of users whose answer is *yes* (resp. *no*). It has been proven that RR satisfies ϵ-LDP if $p = \frac{e^\epsilon}{1+e^\epsilon}$ [11].

Note that if the curator simply counts the number of *yes* among the noisy answers (denoted as c), the results will be biased. To obtain an unbiased estimation, the curator needs to calibrate c and report

$$c' = \frac{c}{2p-1}. \tag{1}$$

Similarly, to obtain an unbiased estimation, the frequency of *yes* (denoted as f) also needs to be adjusted and reported as

$$f' = \frac{p-1+f}{2p-1}. \tag{2}$$

2) Locally Differential Private Protocols for Frequency Estimation: An LDP frequency estimation protocol consists of three algorithms: encode, perturb and aggregation [39]. The state-of-the-art protocols for frequency estimation under LDP are RAPPOR [11] and Random Matrix Projection [15]. Some researchers use frequency estimation protocols as primitives to solve other problems in LDP setting (e.g., [12]–[14], [39]). Based on encoding methods, we can organize LDP frequency estimation protocols into Direct Encoding (DE), Histogram Encoding (HE), Unary Encoding (UE) and Local Hashing (LH). Wang *et al.* [39] have summarized and carefully analyzed the existing LDP frequency estimation protocols.

3) Local Differential Privacy on Graphs: There are two categories in the existing research of LDP-based graph data generation: *edge LDP* and *node LDP* [5]–[7]. The former ensures that a randomized mechanism does not reveal the inclusion or removal of a particular edge in a neighbor list, while the later hides the inclusion or removal of a node together with all its edges.

Definition 2 (Edge LDP [6]): A randomized mechanism M satisfies ϵ-edge LDP if and only if for any two neighbor lists L and L', such that L and L' only differ in one bit, and any $z \in range(M)$, we have $\frac{Pr[M(L)=z]}{Pr[M(L')=z]} \leq e^\epsilon$.

Definition 3 (Node LDP [7]): A randomized mechanism M satisfies ϵ-node LDP if and only if for any two neighbor lists L and L' and any $z \in range(M)$, we have $\frac{Pr[M(L)=z]}{Pr[M(L')=z]} \leq e^\epsilon$.

It is well known that node-LDP imposes stronger constrains than edge-LDP as the insertion or deletion of a node can change nodes' neighbor list significantly. As such, node-LDP has to employ heavy perturbation to compensate the high sensitivity, which causes poor data utility [7]. On the other hand, although edge-LDP only protects the relationship between two nodes, it is sufficient for many graph analysis tasks and can preserve high utility. Therefore, in this paper, **we focus on edge-LDP**.

Similar to DP, the framework of LDP also satisfies the sequential composition property [8].

Theorem 1 (Sequential Composition [8]): Given h randomized algorithms $M_i (1 \leq i \leq h)$ each providing ϵ_i-LDP, the sequence of algorithms $M_i (1 \leq i \leq h)$ collectively provides $(\sum_1^h \epsilon_i)$-LDP.

III. AsgLDP

A. Design Motivation

Social graphs have been utilized for decades to study social environments and it has long been recognized that the structure of a social network alone may not be sufficient to identify social communities [9]. Recently, with the proliferation of information available for real-world social networks, nodes in social graphs are often associated with a number of attributes such as gender, adult, etc.

Traditional graph generation models have primarily focused on modeling the graph structure alone, which maintain structural characteristics of networks such as degree distribution and clustering coefficient [16], [17], [22], [23], [55], [56], [66]. Existing work for generating graph data fail to handle social graphs with correlated attributes [5], [16], [17], [20], [21], [61], [62]. A limited number of existing work can (potentially) be applied to consider node attributes, such as exponential random graph (ERG) [24], multiplicative atribute graph (MAG) [25], latent space (LS) approaches [26], mix membership stochastic blockmodels (SBM) [27], attributed graph model (AGM) [19] and TriCycLe [18].

However, there still exist challenges to apply these methods to the generation of attributed social graphs in a decentralized manner under LDP: 1) they cannot be directly applied to collect graph structures and node attributes under privacy protection in decentralized setting, where each user only has a limited local view of the network; 2) even if we generalize them to the decentralized setting, their mechanisms may fail to generate accurate attributed graph from the noisy data collected by the curator. To overcome these challenges, we propose AsgLDP to collect and generate highly-usable attributed graph data in a decentralized manner while satisfying LDP guarantees.

B. Approach Overview

We aim to design a framework to support the following functions: 1) a curator can collect network structure and attributes from each node while satisfying LDP, 2) the curator is able to learn aggregate information of graph structures and node attributes of the original data, and 3) by leveraging these aggregated information, the curator is able to generate synthetic attributed graph with high utility.

To achieve these objectives, we design AsgLDP which is composed of two phases: *collecting unbiased aggregate information of attributed graph data under LDP, generating synthetic attributed graph while optimizing utility-privacy tradeoff*. As shown in Fig. 1. we provide a brief overview for each phase below.

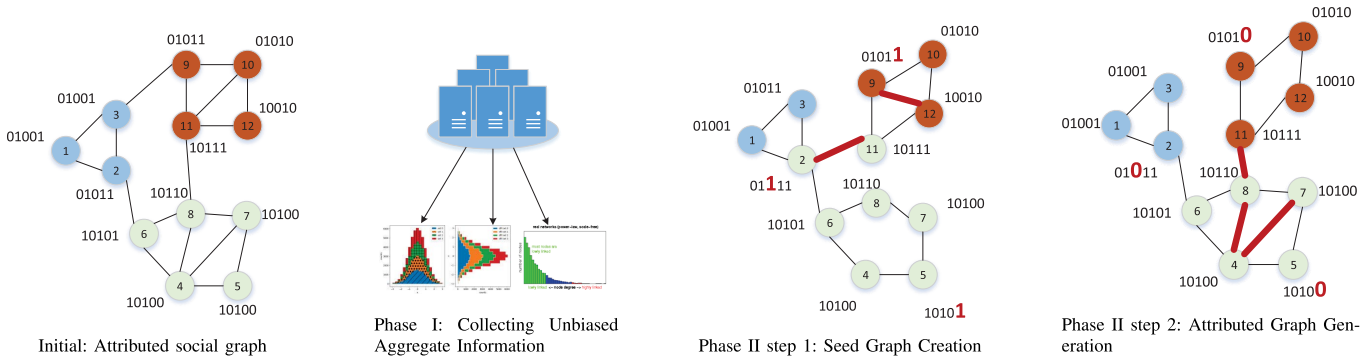


Fig. 1. The workflow of our AsgLDP framework.

Phase I (Collecting Unbiased Aggregate Information of Attributed Graph Data under LDP): This phase, as will be discussed in detail in Section IV, consists of three steps. Firstly, as the curator does not have any information about the graph structure, he/she needs to specify the number of participating users, allocate privacy budgets (e.g., ϵ_1 for protecting attribute list and ϵ_2 for protecting node degrees) and calculate degree perturbation domain. These parameters will be transmitted from the curator to each user (node). Secondly, each user perturbs his/her attribute list and the node degree according to the allocated privacy budget, and then sends the noisy attribute list and degree to the data curator. Finally, the data curator computes the unbiased degree distribution and the joint distribution of attributes.

Phase II (Generating Synthetic Attributed Graph While Optimizing Utility-Privacy Tradeoff): As will be described in detail in Section V, this phase is composed of two steps: *seed graph creation* and *attributed graph optimization*. In the first step, the curator creates a seed attributed graph based on the learned parameters from Phase I. The construction process consists of two stages: 1) we assign attributes to every node based on the joint distribution of attributes learned from Phase I; 2) we use the accept-reject sampling method [29] to construct the edges between nodes, so that the synthetic degree distribution is close to the original degree distribution.

For the next step of attributed graph optimization, we quantify the attribute consistency, the structural consistency and the community consistency between the synthesized graph and the original graph. Based on these metrics, we cluster the seed graph and then detect edge and attribute anomalies. Finally, we optimize the generation of the attributed graph according to the following criteria: 1) the distributions of degree and attributes should be similar to the original graph, and 2) nodes in the same community are more closely related (e.g., with more similar attributes and more edges connected) than those outside the community.

IV. Collecting Unbiased Aggregate Information of Attributed Graph Data Under LDP

A. Step 1: Collecting Attributed Data Under LDP

Our method for collecting nodes' randomized attribute lists (RAL) is similar to the randomized neighbor list (RNL) approach in [5]. Both collect a binary value list from each user. Consider that each node $v_i \in V$ is associated with a w-dimensional attribute vector $X_i = [x_{1i}, \cdots, x_{wi}]$, where x_{wi} is the w-th attribute value of node v_i and $x_{wi} \in [0, 1]$. Specifically, in RAL, given a privacy budget ϵ, each user flips each bit in his/her attribute list with probability $p_r = \frac{1}{1+e^{\epsilon}}$, and sends the perturbed attribute list to the data curator. The RAL approach satisfies ϵ-LDP (Theorem 3.1 in [5]).

However, compared with RNL, we do not have denser graph problem [5]. This is because we use RAL to collect the attributed data instead of the graph structure, and the RAL approach has two obvious advantages: First, the curator can have an unbiased estimation of the frequency of each attribute value. Specifically, the curator calculates the frequency of every attribute $[f_1, f_2, ..., f_w]$, and then obtains the unbiased frequency according to Eq. 2. Formally, the frequency of the w-th attribute, f_w, is defined as the portion of nodes who possess the w-th attribute, i.e., $f_w = \frac{\sum_{i=1}^{n_v} x_{wi}}{n_v}$. Second, the curator can obtain an unbiased estimation of the number of non-zero attributes owned by each user. Specifically, the curator counts the number of "1" (c_i) in each attribute list X_i, and then calibrates them based on Eq. 1. The collected noisy data and the unbiased estimation distributions obtained from RAL can be used in AsgLDP to calculate the joint distribution of attributes and correlations between nodes and attributes.

In the distributed scenario, each node has two key pieces of information (node attributes and degree) that need to be collected. The node attributes have been collected and protected by RAL. Next, we aim to develop an effective method to randomize the node degree while optimizing the utility-privacy tradeoff of the generated data.

B. Step 2: Collecting Node Degrees Under LDP

Wang et al. summarized the state-of-the-art LDP protocols for frequency estimation [39]. However, these methods cannot be directly applied to collect the degree distribution of distributed nodes, since they are based on the assumption that *the curator and each user know the value domain in advance.* (e.g., binary data value domain is $\{0, 1\}$, categorical data value domain is $\{1, 2, \cdots, n\}$ and numerical data value domain is $[-1, 1]$).

However, in a distributed environment, the curator and each node have no knowledge of the network size. Thus, each user does not know the domain of its degree. For example, the curator aims to collect the friendship relationships within an online social network, which has 1 million users, under LDP guarantees. An active user (node) v_i has $1,000$ friends (with degree $d_i = 1,000$) in the social network. To protect privacy, user v_i needs to perturb his/her degree d_i to d_i' locally and send the perturbed value to the curator. However, in the absence of the value (degree) perturbation domain, there is no LDP method to guide user v_i to perturb d_i (1,000) to d_i' appropriately (90 may be too small and 10,000 may be too large).

To solve this problem, we propose random jump (RJ) to perturb the node degree in a decentralized manner while satisfying LDP. In our scenario, the curator and each user (node) in the graph do not know the number of nodes in the entire network. Thus, the first step of RJ is to negotiate important parameters between the curator and users (nodes) such as the privacy budget ϵ and the number of collected nodes n_v. Meanwhile, each user needs a value (degree) domain to guide the degree perturbation. We name the degree domain as *jump domian*. Specifically, given a degree d and radius r, the jump domain $JD(d, r)$ is an effective area taking d as the center and r as the radius. We quantify the utility loss and the privacy leakage of RJ method, and calculate the optimal jump radius r to achieve the best utility-privacy tradeoff.

For a given jump domain $JD(d, r)$, we apply the generalized randomized response (GRR) [32] method to perturb the input value (degree) d. In the general case, when $|JD| = 2r + 1$, the perturbation function is defined as:

$$\forall_{y \in JD_{d,r}} Pr[d' = y] = \begin{cases} p_{rj} = \dfrac{e^\epsilon}{e^\epsilon + 2r}, & \text{if } y = d \\ q_{rj} = \dfrac{1}{e^\epsilon + 2r}, & \text{if } y \neq d \end{cases} \tag{3}$$

Next, we prove that our RJ method as shown Algorithm 1 theoretically satisfies the LDP guarantees in Theorem 2. We further prove the utility preserving properties of RJ in Theorem 3 and Theorem 4.

Algorithm 1 Random Jump Algorithm (RJ)

Input: Privacy budget ϵ, degree $d \in \mathbf{Z}$, radius r
Output: Perturbed value d'
//Determine jump domain
$JD(d, r) = \{d - r, ..., d - 1, d, d + 1, ..., d + r\}$
Perturb d to obtain d' according to Eq. 3.
return d'

Theorem 2: Our RJ method as shown in Algorithm 1 satisfies ϵ-LDP.

Proof: For any inputs d_1,d_2 and output d', we have:

$$\frac{Pr[d'|d_1]}{Pr[d'|d_2]} \leq \frac{p_{rj}}{q_{rj}} = \frac{\frac{e^\epsilon}{e^\epsilon + 2r}}{\frac{1}{e^\epsilon + 2r}} = e^\epsilon \tag{4}$$

□

Theorem 3: Our RJ method is unbiased, i.e., for any value d and the perturbed d' generated from Algorithm 1, we have $\mathbb{E}[d'] = d$.

Proof: Based on Eq. 3, we have

$$\begin{aligned} \mathbb{E}[d'] &= q_{rj} \cdot (d-r) + q_{rj} \cdot (d-(r-1)) + ... + q_{rj} \cdot (d-1) \\ &\quad + p_{rj} \cdot d + q_{rj} \cdot (d+1) + ... + q_{rj} \cdot (d+r) \\ &= \frac{1}{e^\epsilon + 2r} \sum_{i=1}^{r} (d-i) + \frac{e^\epsilon}{e^\epsilon + 2r} d + \frac{1}{e^\epsilon + 2r} \sum_{i=1}^{r} (d+i) \\ &= \frac{e^\epsilon \cdot d + 2r \cdot d}{e^\epsilon + 2r} = d \end{aligned}$$

□

Combining Eq. 3 and Theorem 3, we know that the output of RJ mechanism is distributed as follows.

$$f_o = \frac{e^\epsilon - 1}{e^\epsilon + 2r} f + \frac{1}{e^\epsilon + 2r} \tag{5}$$

where f is the degree frequency of input data and f_o is the degree frequency of the output of RJ.

The empirical estimate of f under RJ method (denoted as M_{RJ}) is thus given as:

$$\hat{f} = \hat{f}_o M_{RJ}^{-1} = \frac{e^\epsilon + 2r}{e^\epsilon - 1} \hat{f}_o - \frac{1}{e^\epsilon - 1} \tag{6}$$

where $\hat{f}_o$ is the empirical estimate of f_o and

$$M_{RJ}^{-1}(y|d) = \frac{1}{2r + e^\epsilon} \begin{cases} e^\epsilon + 2r - 1, & \text{if } y = d \\ -1, & \text{if } y \neq d \end{cases} \tag{7}$$

Theorem 4: For the private distribution estimation problem under RJ and its empirical estimator as show in Eq. 6, we have

$$\mathbb{E}||\hat{\boldsymbol{f}} - \boldsymbol{f}||_2^2 = \frac{1 - \sum_{i=1}^{2r+1} f_i^2}{n_v} + \frac{2r}{n_v} \left(\frac{2(e^\epsilon + r) - 1}{(e^\epsilon - 1)^2} \right) \tag{8}$$

For a large n_v, we further have

$$\mathbb{E}||\hat{\boldsymbol{f}} - \boldsymbol{f}||_1^1 \approx \sum_{i=1}^{2r+1} \sqrt{\frac{2\left((e^\epsilon - 1)(1 - f_i) + 2r\right)\left((e^\epsilon - 1) f_i + 1\right)}{\pi n_v (e^\epsilon - 1)^2}} \tag{9}$$

Proof: We defer the corresponding proof of Theorem 4 to the Appendix A to improve readability. □

Optimal Jump Radius Deduction: Next, we aim to derive the optimal jump radius that can achieve the best utility-privacy tradeoff. Towards this end, we quantify the utility loss and privacy leakage of RJ and define an objective function, the optimization of which leads to the optimal balance between high utility and good privacy. For the utility metric, we leverage L_1 distance between the original degree d and the perturbed degree d', which is defined as follows.

$$\begin{aligned} UtilityLoss &= E(||d' - d||_1) \\ &= p_{rj}|d - d| + q_{rj} \frac{1}{2r} \sum_{i=1}^{r} |(d+i) - d)| \\ &= \frac{1}{e^\epsilon + 2r} \cdot \frac{1}{2r} \cdot \sum_{i=1}^{r} i = \frac{r+1}{2(e^\epsilon + 2r)} \end{aligned} \tag{10}$$

Inspired by the method of *approximate model counting* [33], [34], which is a technique to assess the extent to which different secret inputs are consistent with different

attacker-controlled inputs and attacker observable outputs, we measure the information leakage of RJ algorithm by leveraging the *Jaccard* similarity coefficient [69]. Let $D = \{d_1, d_2, ..., d_{n_v}\}$ denote the input values to RJ and $D' = \{d'_1, d'_2, ..., d'_{n_v}\}$ represent the set of all possible outputs of RJ. For a given privacy parameter ϵ and the radius of jump domain r in RJ, we define the privacy leakage as:

$$\begin{aligned} PrivacyLeakage &= \min_{\forall D' \in \mathbb{D}} e^{\epsilon} \cdot J(D', D) \\ &= e^{\epsilon} \cdot \frac{|D' \bigcap D|}{|D' \bigcup D|} \geq e^{\epsilon} \cdot \frac{n_v - 2r}{n_v + 2r} \end{aligned} \tag{11}$$

Based on the metrics of utility loss in Eq. 10 and privacy leakage in Eq. 11, we construct the optimization problem for balancing utility and privacy as follows.

$$\begin{aligned} \min F_{RJ}(n_v, r) &= \min UtilityLoss + \beta \cdot PrivacyLeakage \\ &= \min \frac{r+1}{2(e^{\epsilon} + 2r)} + \beta \cdot \frac{e^{\epsilon}(n_v - 2r)}{n_v + 2r} \end{aligned} \tag{12}$$

where β is the parameter to balance utility and privacy.

Our objective is to find the optimal radius r so that the objective function in Eq. 12 is minimized. Therefore, by setting the partial derivative of $F_{RJ}(n_v, r)$ with respect to r to 0, we have:

$$\begin{aligned} \frac{\partial F_{RJ}(n_v, r)}{\partial r} &= \frac{\partial \left[\frac{r+1}{2(e^{\epsilon}+2r)} + \frac{\beta e^{\epsilon}(n_v - 2r)}{n_v + 2r} \right]}{\partial r} \\ &= \frac{e^{\epsilon} - 2}{2(e^{\epsilon} + 2r)^2} - \frac{4\beta e^{\epsilon} n_v}{(n_v + 2r)^2} = 0 \\ \Longrightarrow r &= \frac{\sqrt{\frac{n_v(e^{\epsilon}-2)}{8\beta e^{\epsilon}}} - e^{\epsilon}}{2 - \sqrt{\frac{e^{\epsilon}-2}{2n_v \beta e^{\epsilon}}}} \end{aligned} \tag{13}$$

For $r > 0$ in practice, we have $n_v > \frac{8\ e^{3\epsilon}\beta}{e^{\epsilon}-2}$ and $\beta \propto \frac{8e^{\epsilon}}{e^{\epsilon}-2}$. From Eq. 13, we know that the optimal radius is impacted by both the privacy budge ϵ and the number of participating nodes n_v. Generally, $r \propto \sqrt{n_v}$, the more nodes are getting involved, the larger the optimal radius r. Furthermore, the user (node) can adjust the utility-privacy weight β to regulate the coverage of jump domain.

With the optimal r, the user then perturbs his/her degree with RJ and sends it together with a sanitized attribute list. For example, we suppose node v_1 with degree $d_1 = 10$, node v_2 with degree $d_2 = d_1 + 1 = 11$, and the jump radius $r = 4$. As for node v_2, the jump domain is $JD(d_2, r) = \{7, \cdots, 10, 11, \cdots, 15\}$. If the noisy degree of d_2 is $d'_2 = d_2 + 3 = 14$, $d'_2 \in JD(d_2, r)$, it is difficult for the adversary to distinguish whether the input is d_1 or d_2. This is because: 1) the perturbed output domain of d_1 is $\{6, 7, \cdots, 14\}$; 2) there are $2r + 1$ degrees $\{10, 11, 12, \cdots, 17, 18\}$ that could be perturbed into $d'_2 = 14$.

C. Step 3: Extracting Unbiased Aggregate Information of the Attributed Graph

By combining RAL and RJ perturbation mechanisms, we construct the data collection (DC) approach for decentralized attributed graphs, as shown in Algorithm 2, which consists of user-side perturbation and collector-side calibration. We allocate private budget ϵ_1 and ϵ_2 for RAL and RJ, respectively. For node attributes, each user perturbs his/her own attribute list X_i to X'_i with RAL under privacy budget ϵ_1 and sends the perturbed data to the curator (Line 2). Then, the curator calculates each attribute frequency f_w and calibrates f_w to f^*_w according to Eq. 2 (Line 5-7). As for node degree, each user perturbs his/her degree d_i to d'_i with our proposed RJ method (Line 3). Specifically, each user first determines the jump domain according to the optimal radius r as shown in Eq. 13. Then, the degree is perturbed according to Eq. 3. The curator calculates degree frequency f_d from the noised data and calibrates it to f^*_d to obtain unbiased degree distribution according to Theorem 3 (Line 9-11). Specifically, the expected number of times degree d appears in the perturbed data set, is given by

$$E(T'_d) = p_2\ T_d + (1 - p_2)(2r + 1 - T_d),$$

where T_d is the number of times degree d appears in the original data and $p_2 = \frac{e^{\epsilon 2}}{e^{\epsilon 2}+2r}$. Thus, we can estimate the degree frequency of the output data as

$$f^*_d = \frac{(p_2 - 1)\left((2r+1)/n_v\right) + f_d}{2p_2 - 1}.$$

Theorem 5: Our data collection algorithm in Algorithm 2 satisfies ϵ-LDP.

Proof: As shown in Algorithm 2, each user adds noise to his/her attribute list according to RAL which satisfies ϵ_1-LDP and perturbs node degree according to RJ algorithm which satisfies ϵ_2-LDP. According to the sequential composition property of LDP in Theorem 1, the overall process of DC satisfies ϵ-LDP, since $\epsilon_1 + \epsilon_2 = \epsilon$.

In order to synthesize the attributed social graph more accurately, we estimate the joint distribution of w-dimension attributes. Inspired by [40], [41], we extend the EM algorithm, which is a common method to approximate maximum likelihood estimates of unknown parameters, to estimate

Algorithm 2 Data Collection (DC)

Input: All user's attribute lists $X = \{X_1, ..., X_{n_v}\}$
All user's degree $D = \{d_1, ..., d_{n_v}\}$
Privacy budgets ϵ_1, ϵ_2 and jump radius r
Output: Frequency estimate for each attribute f^*_w
Degree distribution vector f^*_d

1 **// User-side perturbation**;
2 $X'_i = RAL(X_i, \epsilon_1)$ // Each user perturbs his/her attribute list X_i to X'_i with RAL and sends to data collector;
3 $d'_i = RJ(d_i, \epsilon_2, r)$ // Each user perturbs his/her degree d_i to d'_i with RJ and sends to data collector;
4 **//Collector-side calibration**
5 **for** *each attribute* $w \in W$ **do**
6 Curator calculates frequency f_w
7 Curator calibrates the frequency as:

$$f^*_w = \frac{p - 1 + f_w}{2p - 1}, \text{where } p = \frac{e^{\epsilon 1}}{e^{\epsilon 1} + 1}$$

8 **end**
9 Curator calculates degree frequency f_d in the D'_i:
10 Curator calibrates the degree frequency as:

$$f^*_d = \frac{(p_2 - 1)\left((2r+1)/n_v\right) + f_d}{2p_2 - 1}, \text{where } p_2 = \frac{e^{\epsilon 2}}{e^{\epsilon 2} + 2r}$$

11 return f^*_w and f^*_d

multi-dimension attribute joint distribution. The pseudo-code is provided in Algorithm 5 (Appendix B). We first introduce the following notations. Without loss of generality, we consider w specified independent attributes and their index collection $\mathcal{W} = \{1, 2, ..., w\}$ and the prior probability $P(x_1 = \omega_1, x_2 = \omega_2, ..., x_w = \omega_w)$ as $P(\omega_{\mathcal{W}})$. In AsgLDP, each bit is flipped with probability p_r (recall Section IV-A). Thus, by comparing the bits, the conditional probability $P(X'_i|\omega_{\mathcal{W}})$ can be computed as

$$P(X'_i|\omega_{\mathcal{W}}) = \prod_{k=1}^{w}(1-p_r)^{|X'_i[k]-\omega_{\mathcal{W}}[k]|} \times p_r^{1-|X'_i[k]-\omega_{\mathcal{W}}[k]|}$$

Given all the conditional distribution of one particular combination of attributes, their corresponding posterior probability can be computed according to *Bayes's* Theorem,

$$P_t(\omega_{\mathcal{W}}|X'_i) = \frac{P_t(\omega_{\mathcal{W}}) \cdot P(X'_i|\omega_{\mathcal{W}})}{\sum_{\omega_{\mathcal{W}}} P_t(\omega_{\mathcal{W}}) P(X'_i|\omega_{\mathcal{W}})} \tag{14}$$

After identifying posterior probability of each user, we calculate the mean of the posterior probability from all the users to update the prior probability, which will then be used in the next iteration to update the posterior probability. The procedure is executed iteratively until the difference between consecutive iterations is smaller than a threshold τ.

V. Generating Synthetic Attributed Graph While Optimizing Utility-Privacy Tradeoff

A. Step 1: Seed Graph Creation

Firstly, we describe the method for creating a seed graph based on the aggregated information learned from Phase I. We focus on generating a seed graph $G_s = \langle V_s, E_s, X_s \rangle$ which has similar *degree distribution* and *attribute joint distribution* as the original attributed graph.

Pfeiffer et al. present the attributed graph model (AGM) to jointly model network structure and node attributes [19]. Specifically, AGM learns the attribute correlations in the observed network, exploits a generative graph model and constructs edges with correlated attributes based on the accept-reject sampling method. Motivated by AGM framework [19], we apply the accept-reject sampling method to generate edges between nodes. Accept-Reject sampling is a framework for generating samples from a desired distribution [29]. A typical algorithm for accept-reject sampling is composed of two procedures: *propose* and *accept*. In the *propose* step, the algorithm iteratively draws samples (edges) e_{ij} based on the *propose* mechanism $Pro(M_p)$, where M_p is a generative graph model. Next, with probability $Acc(e_{ij})$, the proposed samples (edges) are accepted.

In AsgLDP, the mechanism M_p is a generative graph model such as Chung Lu Model (CL) [30], Transitive Chung Lu Model (TCL) [17] and Block Two-Level Erdos Renyi Model (BTER) [16]. Let Θ_{M_p} denote a degree distribution over graph configurations with respect to the chosen model M_p. A complete set of edges E_s can be drawn (sampled) using M_p. Every edge is sampled according to $Bernoulli(P(e_{ij} = 1|\Theta_{M_p}))$, i.e., if the draw is a success, the edge e_{ij} will be added to E_s. We consider to adopt the CL model for its simplicity and wide applicability. In CL model, the probability for each edge to be sampled is proportional to the product of the degrees of its end nodes.

$$P_{CL}(e_{ij} = 1|\Theta_{CL}) = \frac{d_i d_j}{\sum_{v_k \in V} d_k}$$

where $\Theta_{CL} = [d_1, d_2, ..., d_{n_v}]$. This formulation guarantees that the expected degree of the sample graph is the same as the degree of the original graph [30].

$$\mathbf{E}_{CL} = \sum_{v_i \in V_s} \frac{d_i d_j}{\sum_{v_k \in V_s} d_k} = \Theta_{d_i} \frac{\sum_{v_j \in V_s} \Theta_{d_i}}{\sum_{v_k \in V_s} \Theta_{d_k}} = \Theta_{d_i}$$

In AsgLDP, we leverage the relationship between attributes X_i and X_j to determine the probability of acceptance. If $Pro(M_p)$ draws an edge e_{ij}, we will accept it with probability $Acc(e_{ij}|X_i, X_j)$. Specifically, we quantify the similarity of attributes between two nodes, and the acceptable probability can be expressed as

$$Acc(e_{ij} = 1|X_i, X_j) = \frac{1}{w}\sum_{k=1}^{w}\delta(x_{ki}, x_{kj}). \tag{15}$$

where $\delta(x_{ki}, x_{kj})$ is the Kronecker delta function [70], i.e., $\delta(x_{ki}, x_{kj}) = 1$ if $x_{ki} = x_{kj}$, and 0 otherwise.

Algorithm 3 Seed Graph Creation (SGC)

Input: Joint distribution of w attributes specified by $\mathcal{W}$, i.e., $P(X_{\mathcal{W}})$
Degree distribution vector $\boldsymbol{f}_d^*$
Output: Seed Graph $G_s = \langle V_s, E_s, X_s \rangle$

1 // Initialize an empty node set
$V_s = \{v_1, ..., v_n\}, E_s = \emptyset, X_s = \emptyset$
2 // Assign attributes to each node according joint distribution $P(X_{\mathcal{W}})$
3 **for** *each* $v_i \in V_s$ **do**
4 $X_s = X_s \bigcup X_s^i$, where $X_s^i = X_i$, w.h.p. $P(X_i)$
5 **end**
6 // Assign pre-degree according $\boldsymbol{f}_d^*$
7 **for** *each* $v_i \in V_s$ **do**
8 $d_i = d$, w.h.p. $P(f_d^*)$
9 **end**
10 $m = \frac{\sum_{v_i \in V_s} d_i}{2}$ // The number of estimated edges
11 // Propose and accept edges
12 **for** $|E_s| \leq m$ **do**
13 $e_{ij} = Pro(\Theta_{CL}, V_s)$
14 **if** $Acc(e_{ij} = 1|X_i, X_j)$ **then**
15 $E_s = E_s \bigcup e_{ij}$
16 **end**
17 **end**
18 return $G_s = \langle V_s, E_s, X_s \rangle$

Algorithm 3 summarizes our method for creating a seed graph. It takes as inputs the joint distribution of w-dimensional attributes $P(X_{\mathcal{W}})$ and the degree distribution $\boldsymbol{f}_d^*$ estimated from the decentralized graph obtained from Phase I, and returns the synthetic seed graph $G_s = \langle V_s, E_s, X_s \rangle$. First, we initialize the seed graph with n_v empty nodes and assign attribute vectors to each node according to $P(X_{\mathcal{W}})$ (Line 1-5). Subsequently, we allocate a degree to each node based on the degree distribution $\boldsymbol{f}_d^*$. Then, we use accept-reject sampling method to generate $m = \frac{1}{2}\sum_{v_i \in V_s} d_i$ edges. Specifically, in our setting, we propose edges based on the CL graph model, and accept the proposed edge according to Eq. 15 (Line 12-17). After obtaining a seed graph, we will continue optimizing the synthetic attributed graph in the next step to improve the utility of the synthetic data.

B. Step 2: Attributed Graph Optimization

In this subsection, we present the method for generating an optimized attributed graph based on the seed graph in Section V-A. Intuitively, a "*good*" synthetic attributed graph should capture as many properties of the original graph as possible such as degree and attribute distribution, attribute correlations and community structure. In AsgLDP, we optimize the generation of the synthetic attributed graph according to *structural and attribute consistency* and *community consistency*. The *structural and attribute consistency* implies that the distribution of degree and node attributes in the synthetic graph should be similar to the original graph. The *community consistency* implies that the synthetic graph has a good community structure (recall *homophily* [38] and *social influence* [28]) and the nodes in the same community are more closely related (e.g., with more similar attributes and more edges connected) than those outside the community.

1) Structural and Attribute Consistency: As discussed in Section IV-C, AsgLDP can calculate unbiased estimation of degree distribution and frequency of each attribute. To quantify the structural and attribute consistency, we propose to leverage similarity between parameters (SimP) in Eq. 16 to quantify the similarity between degree (resp. attribute) distributions of the generated graph and the original graph.

$$SimP = \frac{1}{|w+1|}\left(H(\boldsymbol{f}_d^*, \boldsymbol{f}_d) + \sum_{x_i \in w} H(\boldsymbol{f}_{x_i}^*, \boldsymbol{f}_{x_i})\right) \quad (16)$$

where, $\boldsymbol{f}_d^*$ (resp. $\boldsymbol{f}_{x_i}^*$) is the unbiased degree distribution (resp. the unbiased distribution of i-th attribute) of the original attributed graph, $\boldsymbol{f}_d$ (resp. $\boldsymbol{f}_{x_i}$) is the degree distribution of (resp. the distribution of i-th attribute) in the generated graph, $H(\boldsymbol{f}_d, \boldsymbol{f}_d^*) = \frac{1}{\sqrt{2}}\sqrt{\sum_i \left(\sqrt{\boldsymbol{f}_{di}} - \sqrt{\boldsymbol{f}_{di}^*}\right)^2}$ is the Hellinger distance [71] between two degree distributions. Similarly, $H(\boldsymbol{f}_{x_i}^*, \boldsymbol{f}_{x_i})$ is the Hellinger distance between two attribute distributions. The smaller the Hellinger distance, the closer the two distributions. Thus, a smaller SimP represents higher structural and attribute consistency.

2) Community Consistency: Intuitively, a "*good*" community has many internal edges among its members where they share a set of attributes with similar values. In addition, for a good community, either it has a few edges at its boundary, or its boundary nodes have attributes dissimilar from those of community members. The quality of community consistency of a graph can be evaluated by internal consistency and external separability. To quantify the community consistency, we leverage Normality developed by Perozzi and Akoglu [31], which combines structure and attributes together to consider both internal consistency and external separability.

$$\begin{aligned} Normality = & \sum_{i \in C, j \in C, i \neq j} \left(A_{ij} - \frac{d_i d_j}{2m}\right) sim_{in}\left(X_i, X_j\right) \\ & - \sum_{i \in C, b \in B, e_{ib} \in E} \left(1 - min\left(1, \frac{d_i d_j}{2m}\right)\right) \\ & \times sim_{ex}\left(X_i, X_b\right) \end{aligned} \quad (17)$$

where m is the number of edges in the graph with adjacency matrix A. $sim_{in}(X_i, X_j) = X_i \cdot X_j$ is used to quantify internal consistency and $sim_{ex}(X_i, X_b) = \sum_{k=1}^{w} \delta(x_{ki}, x_{kb})$ is used to quantify external separability for attributed graph [31].

In Eq. 17, the first term presents the consistency of nodes among the same community C. The second term shows the separability between nodes in community C and boundary B (recall Section II-A). Intuitively, the higher the Normality score, the better the communities' quality.

Combining Eq. 16 and Eq. 17, we can quantify the consistency between a synthetic attributed graph and the original graph as follows.

$$Consistency = SimP - \gamma \cdot Normality \quad (18)$$

where γ is a parameter to balance the structure/attribute consistency and the community consistency.

Algorithm 4 Attributed Graph Optimization (AGO)

Input: Seed Graph $G_s = \langle V_s, E_s, X_s \rangle$
Frequency estimate for each attribute $\boldsymbol{f}_w^*$
Degree distribution vector $\boldsymbol{f}_d^*$
Covergence threshold ξ
Output: Attributed Graph $G' = \langle V', E', X' \rangle$

1 **repeat**
2 // Attributed graph community detection
3 $\boldsymbol{C} = \boldsymbol{CESNA}(G_s)$
4 $S = Consistency\left(G_s, \boldsymbol{f}_w^*, \boldsymbol{f}_d^*\right)$
5 // Optimize graph
6 **for** *each* $C_k \in \boldsymbol{C}$ **do**
7 $v_i = \max_{\forall v_i \in B_k} |\sum_{v_j \in C_k, e_{ij} \in E} \frac{d_i d_j}{2m} \cdot \delta(X_i, X_j)|$
8 //delete edge
9 $E'_s = E - e_{ij}$, where $e_{ij} = \max_{\forall v_j \in C_k, e_{ij} \in E} |\frac{d_i d_j}{2m} \cdot \delta(X_i, X_j)|$
10 // change attribute
11 $X'_i = \max\left(\cos(X'_i, X_j) - \cos(X_i, X_j)\right)$, where $X_i \oplus X'_i = 1$
12 $X_e = \min_{\forall v_e \in C_k} (\cos(X_e, X_c))$
13 $X'_e = \max\left(\cos(X'_e, X_c) - \cos(X_e, X_c)\right)$, where $X_e \oplus X'_e = 1$
14 // add edge
15 $e_{ij} = Pro(\Theta_{CL}, C_k)$
16 **if** $Acc(e_{ij} = 1 | X_i, X_j)$ **then**
17 //Accept edge e_{ij}
18 $E' = E \bigcup e_{ij}$
19 **end**
20 **end**
21 $S' = Consistency\left(G', \boldsymbol{f}_w^*, \boldsymbol{f}_d^*\right)$
22 **until** $S' - S \leq \xi$;
23 return $G' = G'_s$

Algorithm 4 shows the process of attributed graph optimization. First, the seed graph G_s is clustered into communities $C = \{C_1, \cdots, C_K\}$ by CESNA [47] which is a probabilistic model that combines community memberships, the network topology and node attributes to cluster attributed graph into communities (Line 3). Then, we calculate Consistency S of the seed graph according to Eq. 18. Next, we optimize the synthetic attributed graph in an iterative manner. In each iteration, we detect the abnormal edges and attributes and modify them to optimize the seed graph. Our optimization consists of two key steps: 1) for each community C_k, and its corresponding boundary B_k (recall Section II-A), we find the boundary node $v_i \in B_k$ which has the most edges and similar attributes to the community C_k (Line 7). Then, we delete the most relevant edge e_{ij} and change the attributes in X_i (Line 8-11). The purpose of this step is to make the boundary

nodes connected less frequently with the community and reduce the similarities between the boundary node attributes and the community node attributes. 2) to enhance the internal consistency of community, we calculate the attribute center X_c of community C_k, locate $v_e \in C_k$ whose attribute vector X_e is the most different from the attribute center in community C_k. We modify X_e (resp. X_i) to X'_e (resp. X'_i) by changing the most abnormal attribute value at a time, where the $\oplus$ operation implies that X_e (resp. X_i) differs from X'_e (resp. X'_i) by one attribute (Line 13). Through multiple iterations, the values of the abnormal attributes becomes more similar to normal attributes. In addition, we add edges based on the method proposed in Algorithm 3 to make the community more tightly connected. At the end of each iteration, we calculate Consistency S' of the optimized attributed graph. The above process is repeated several times until $S' - S \leq \xi$.

Theorem 6: Our AsgLDP algorithm satisfies ϵ-LDP.

Proof: In AsgLDP, the first phase for collecting graph data satisfies ϵ-LDP according to Theorem 5. The second phase in Section V is post processing of the output of the first phase, and thus does not consume any privacy budget [1]. Thus, we have that the overall process of AsgLDP satisfies ϵ-LDP.

3) Summary: AsgLDP is composed of two phases *collecting unbiased aggregate information of attributed graph data under LDP* and *generating synthetic attributed graph while optimizing utility-privacy tradeoff*. In the first phase, the client side perturbs its degree with our proposed RJ method and its attribute list with RAL. The curator side aggregates noisy data and calculates attributed graph parameters (e.g., degree distribution, attribute joint distribution). We prove that data collection method satisfies ϵ-LDP. In the second phase, we generate a seed graph under a generative graph model and optimize the generated attributed graph in order to preserve general graph properties (e.g., community consistency, network structure consistency and attribute consistency). We further prove that AsgLDP rigorously satisfies ϵ-LDP.

VI. Experimental Evaluation

To validate the effectiveness of AsgLDP, we implement AsgLDP using 6 real-world large-scale datasets (details of these datasets are described in Appendix D). Specifically, we first compare our proposed RJ method with 5 state-of-the-art LDP perturbation algorithms in Section VI-B, in order to show the advantage of RJ method in collecting privacy-preserving node degrees. Then, we implement AsgLDP with three graph models BTER [16], TCL [17] and CL [30] to generate AsgLDP-BTER, AsgLDP-TCL and AsgLDP-Cl and show the optimal utility-privacy balance achieved by our approach.

A. Evaluation Metrics

1) Degree Distribution: To evaluate how well a synthetic graph captures the degree distribution of the input attributed social network, we leverage the Kolmogorov-Smirnov (KS) statistic [73] to quantify the degree distribution distance between the original and the synthetic graphs. Let F_G and $F_{G'}$ denote the cumulative distribution functions (CDF) estimated from the degree sequences of the original and the synthetic graph, respectively. Then, we have $KS(G, G') = \max_d |F_G(d) - F_{G'}(d)|$. To further examine the difference between the tails of the two distributions, we also report the Hellinger distance $H(\boldsymbol{f}_d^*, \boldsymbol{f}_d)$ between the two degree distributions. A smaller value of KS statistic and Hellinger distance represents more similar degree distributions between the synthetic and the original graphs.

2) Clustering Coefficient and Modularity: We leverage the graph *clustering coefficient* [42] and *modularity* [59] as metrics to investigate how well the synthetic attributed graph preserves the community structure of the original graph. The clustering coefficient of a node $v_i \in V$ is the fraction of all the possible triangles through that node, $CC(v_i) = \frac{2\ T(v_i)}{d_i(d_i-1)}$ where $T(v_i)$ is the number of triangles through node v_i. The higher the average clustering coefficient is, the more tightly nodes are connected. Modularity [59] is an effective metric to evaluate quality of the detected communities which is defined as $Q = \frac{1}{2m}\sum_{ij}\left[A_{ij} - \frac{d_i d_j}{m}\right]\delta\left(C_i, C_j\right)$. A large value of Q corresponds to a better division of a network into communities.

3) Attribute Distribution: To quantify the performance of the attribute distribution in the synthetic graph as compared to the original attributed graph, we report the Hellinger distance $H(\boldsymbol{f}_x^*, \boldsymbol{f}_x)$ between the two attribute distributions. A smaller value of $H(\boldsymbol{f}_x^*, \boldsymbol{f}_x)$ represents that the attribute correlations in the synthetic graph more closely approximates to those of the input graph.

For attributed community search, we use two measures for evaluating the attributes cohesiveness of the communities: community member frequency (CMF) and community pair-wise jaccard (DPJ) [48].

4) Community Member Frequency: CMF [48] uses the occurrence frequencies of attributes in C_i to determine the degree of cohesiveness. Let $f_{i,w}$ be the number of nodes of C_i whose attribute sets contain the w-th attribute of X_i. Then, $\frac{f_{i,w}}{|C_i|}$ is the relative occurrence of this attribute in C_i. The CMF is the average of this value computed over all the attributes and all communities in C:

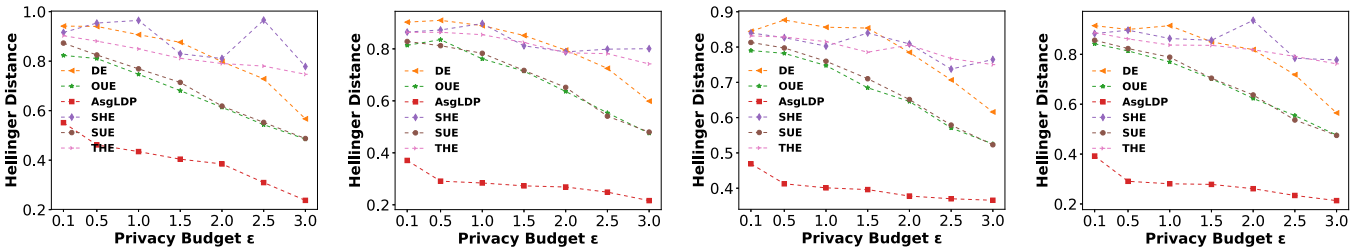
$$CMF(C) = \frac{1}{K \cdot |X_i|}\sum_{i=1}^{K}\sum_{w=1}^{|X_i|}\frac{f_{i,w}}{|C_i|}$$

where CMF $\in [0, 1]$ and $|C_i|$ is the number of nodes in C_i. A higher value of CMF represents that the communities are more cohesiveness.

5) Community Pair-Wise Jaccard: CPJ [48] evaluates the similarity between the attributes of any pair of nodes of community C_i. Let $v_{i,j}$ be the j-th vertex in C_i. The CPJ thus evaluates the average similarity over all the node pairs in C_i and all the communities of C:

$$CPJ(C) = \frac{1}{K}\sum_{i=1}^{K}\left[\frac{1}{|C_i|^2}\sum_{j=1}^{|C_i|}\sum_{k=1}^{|C_i|}\frac{|X_{v_{i,j}} \bigcap X_{v_{i,k}}|}{|X_{v_{i,j}} \bigcup X_{v_{i,k}}|}\right]$$

where $X_{v_{i,j}}$ denotes the attributes of node $v_{i,j}$. A higher value of CPJ$\in [0, 1]$ implies better cohesiveness.

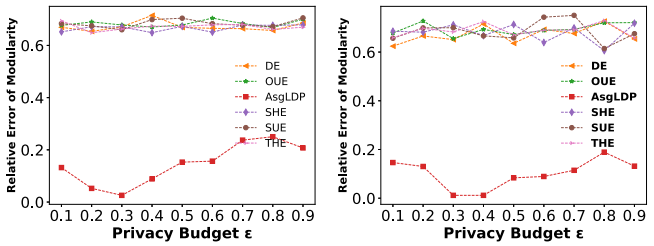


(a) Enron **(b)** Facebook **(c)** Google **(d)** Twitter

Fig. 2. The Hellinger distance between degree distributions of synthetic and original graphs under different methods.

TABLE I
AVERAGE KS STATISTIC ON DEGREE DISTRIBUTION

ϵ	THE	SHE	SUE	OUE	DE	AsgLDP
0.1	0.8227	0.8162	0.7968	0.7737	0.8257	**0.3074**
0.5	0.8038	0.8257	0.7669	0.7716	0.8278	**0.1906**
1.0	0.7947	0.8189	0.7349	0.718	0.8146	**0.1715**
1.5	0.7761	0.7943	0.6755	0.6467	0.8033	**0.1522**
2.0	0.7557	0.7864	0.5753	0.5622	0.7406	**0.1403**
2.5	0.7471	0.7693	0.4584	0.4541	0.6566	**0.0945**
3.0	0.7115	0.7469	0.3699	0.3667	0.4866	**0.0497**



(a) Terrorism **(b)** Facebook

Fig. 3. Relative error of modularity under different methods.

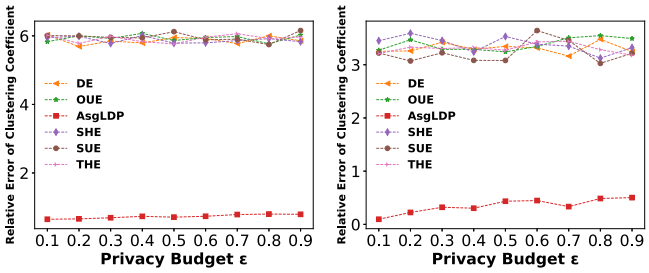
B. Experimental Results

1) Advantages of AsgLDP in Preserving Degree Distribution: We first evaluate the degree distribution, which is an important graph structure statistical property. Specifically, we compare AsgLDP with the other five state-of-the-art LDP methods (direct encoding (DE), thresholding with histogram encoding (THE), summation with histogram encoding(SHE), symmetric unary encoding (SUE) and optimized unary encoding (OUE), as described in [39]). Figure 2 shows the Hellinger distance between degree distributions of the original and the synthetic graphs under different methods. From Figure 2, we can observe that under the same privacy level, the degree distribution generated by our AsgLDP method is the closest to the original data, i.e., AsgLDP method outperforms the five state-of-the-art LDP methods on degree generation. We attribute this result to the jump domain generation mechanism of the RJ method (recall Section IV-B). For example, on *Enron* dataset, the perturbation domain of DE is 1000, while, in RJ method, the jump radius $r = 18$ (for $\epsilon = 3$) and the jump domain is $37 \ll 1000$. Thus, RJ method yields a more similar degree distribution than that of DE. Table I summarizes the experimental result about the average KS statistic of degree distribution. In Table I, we observe that AsgLDP method has the minimum KS distance under different privacy budgets (ranging from 0.0497 to 0.3074), while KS distances obtained by other methods are higher than 0.3667. In addition, in Figure 2, the results of SHE and THE are unstable with the varying ϵ, and the Hellinger distance of DE, SUE and OUE methods decrease rapidly with an increasing ϵ. However, the AsgLDP method performs more stably than others. Thus, from the above experiments, we know that *our AsgLDP method preserves degree distribution more accurately and achieves a significantly better utility-privacy tradeoff as compared to the state-of-the-art LDP methods.*

2) Advantages of AsgLDP in Preserving Community Structure: We apply community discovery algorithms on the synthetic graphs, and evaluate how well the synthetic graphs generated by AsgLDP preserve the community structure of the original graph, by using *modularity* and *clustering coefficient* metrics defined in Section VI-A.

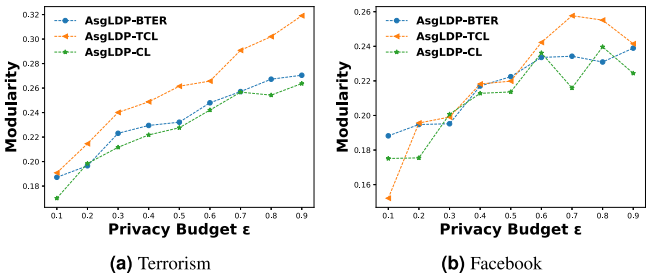
Figure 3 (resp. Figure 4) shows the relative errors of modularity (resp. clustering coefficient) between the generated and the original graphs, where the relative error of modularity = | Modularity of Original Graph −Modularity of Generated Graph| /Modularity of Original Graph (resp. the relative error of clustering coefficient = | Clustering Coefficient of Original Graph− Clustering Coefficient of Generated Graph|/ Clustering Coefficient of Original Graph). We observe that our AsgLDP method obtains lower relative errors in both modularity and clustering coefficient between the synthetic and the original graphs (as compared to previous methods). For example, on *Facebook*, with the increase of privacy budget ϵ, the modularity score approaches that of the original graph, and the relative errors of modularity (resp. clustering coefficient) under AsgLDP are 3 times (resp. 6 times) lower than other five methods. *Therefore, we know that existing methods destroy the graph community structure due to excessive noise injection, and our AsgLDP method outperforms the state-of-the-art methods in preserving community structure.*

We also evaluate the *modularity* and *clustering coefficient* achieved by different graph models (AsgLDP-TCL, AsgLDP-TCL and AsgLDP-BTER), as shown in Figure 5 and Figure 6, respectively. *We find that graph models significantly affect the network structure of the generated graphs*. For example, in Figure 5, we can clearly observe that AsgLDP-TCL model achieves better modularity scores than AsgLDP-CL and AsgLDP-BTER. Figure 6 also validates that AsgLDP-TCL performs better in preserving clustering coefficients than the other two models.



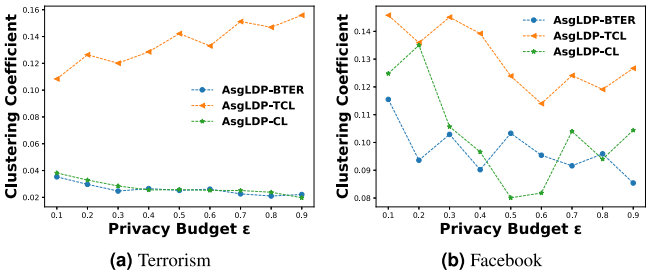
(a) Terrorism **(b)** Facebook

Fig. 4. Relative error of clustering coefficient under different methods.



(a) Terrorism **(b)** Facebook

Fig. 5. Modularity of different graph models.



(a) Terrorism **(b)** Facebook

Fig. 6. Clustering coefficient under different graph models.

TABLE II
AVERAGE HELLINGER DISTANCE ON ATTRIBUTE DISTRIBUTIONS

ϵ	DE	OUE	SHE	SUE	THE	AsgLDP
0.1	0.1000	0.0818	0.0760	0.0883	0.0862	**0.0603**
0.2	0.0623	0.0961	0.0696	0.0938	0.0640	0.0703
0.3	0.0771	0.0907	0.0914	0.0781	0.0912	**0.0567**
0.4	0.0902	0.0724	0.0766	0.1064	0.0771	0.0779
0.5	0.0673	0.0983	0.0858	0.1098	0.0790	0.0759
0.6	0.0784	0.0693	0.0644	0.0970	0.0866	0.0775
0.7	0.0831	0.0851	0.1010	0.0960	0.0916	**0.0444**
0.8	0.0668	0.1075	0.0782	0.0737	0.0744	0.0768
0.9	0.0676	0.0911	0.0966	0.0813	0.0770	0.0863

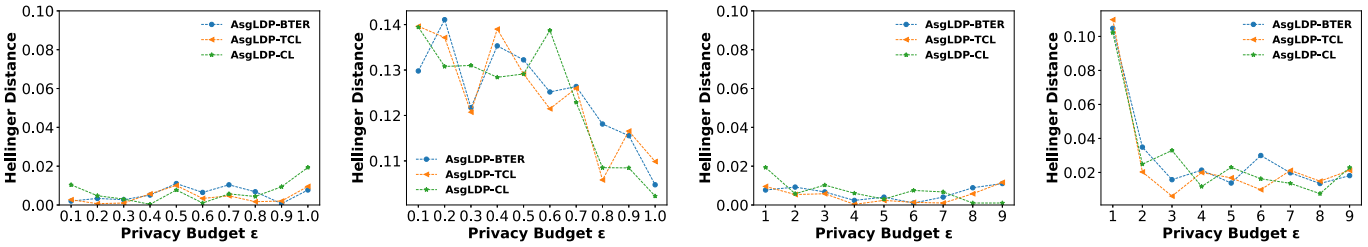
3) Advantages of AsgLDP in Preserving Attribute Distribution: Table II presents the average Hellinger distances on attributes distribution between the synthetic and the original graphs under different LDP methods. Figure 7 shows the Hellinger distances under different graph models (AsgLDP-BTER, AsgLDP-TCL and AsgLDP-CL), using the *Blogs* and *Terrorism* datasets. From Table II and Figure 7, we observe that: 1) *our AsgLDP framework performs better than other methods in preserving node attribute distribution*. For example, in Table II, the Hellinger distances corresponding to our AsgLDP method are less than 0.1, indicating that there are small differences between the synthetic and the original node attribute distributions. In addition, under privacy budget $\epsilon = 0.1$, $\epsilon = 0.3$ and $\epsilon = 0.7$, our AsgLDP method achieves better Hellinger distances in attribute distributions (0.0603, 0.0567 and 0.0444) as compared to other methods; 2) *graph model has negligible effect on attribute distribution*. In Figure 7, with a varying privacy budget, the attribute distributions of different graph models (AsgLDP-BTER, AsgLDP-TCL and AsgLDP-CL) are similar. This is because: 1) AsgLDP can calculate the unbiased attribute distribution by Algorithm 2, and 2) equipped with attributed graph optimization phase (Algorithm 4), although the graph models are different, our AsgLDP method can maintain the attribute consistency with the original graph.

4) Advantages of AsgLDP in Attributed Community Search: We evaluate the effectiveness of AsgLDP in the practical application of attributed community search using the metrics of CMF and CPJ in Section VI-A, which quantify the attributes' cohesiveness of the communities.

Figure 8 (resp. Figure 9) represents the relative error between CMF (CPJ) of the attributed community search in the original graph and the synthetic graph under different methods. We can observe that AsgLDP achieves lower CMF and CPJ relative errors than other five methods. For example, for $\epsilon = 0.1$, the relative error of CMF (resp. CPJ) for our AsgLDP method on Facebook dataset is less than 0.2 (resp. 0.016), while the relative error of other methods are higher than 0.4 (resp. 0.08). *Therefore, our AsgLDP method outperforms other methods in preserving the attributes' cohesiveness of the community.*

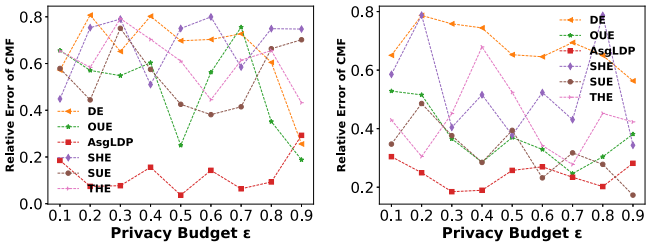
Figure 10 and Figure 11 show the CMF and CPJ results of attributed community search under different graph models. We observe that with the increase of ϵ, CMF and CPJ metrics exhibit no obvious change. In addition, the mean relative errors of CMF and CPJ (as shown in Table III) between the synthetic graph generated by our method and the original graph are small, which implies that our method can perfectly preserve the attributed correlations among nodes within the community. Equipped with the attibuted graph optimization phase (Algorithm 4), regardless of the privacy budget, we can adjust the community structure (edges and attributes) of the synthetic graph to accurately preserve the internal relationships of the original graph.

5) Summary of Experimental Results: AsgLDP is effective in synthesizing attributed graph in a decentralized manner. Specifically, our method is superior to other LDP methods (DE, OUE, SHE, SUE and THE) in preserving degree distribution, community structure, attribute distribution and attributed community search. The AsgLDP framework can be easily combined with existing graph models to generate synthetic attributed graph under LDP guarantees. Furthermore, through optimizing the graph structure, AsgLDP can reduce the utility loss caused by randomization and thus accurately preserve the inherent relationships existing in nodes of the original graphs. In addition, AsgLDP achieves high utility in preserving general graph structure characteristics and attribute properties.



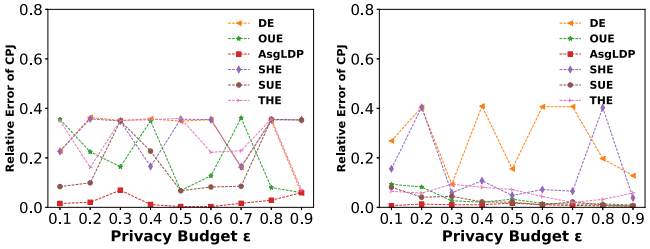
(a) Blogs **(b)** Terrorism **(c)** Blogs **(d)** Terrorism

Fig. 7. The Hellinger distance between attribute distributions of synthetic and original graphs under different graph models.



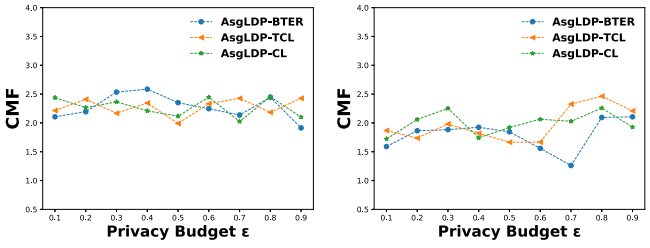
(a) Facebook **(b)** Terrorism

Fig. 8. Relative error of CMF under different methods.



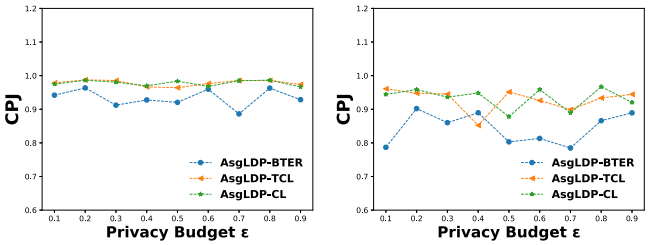
(a) Facebook **(b)** Terrorism

Fig. 9. Relative error of CPJ under different methods.



(a) Terrorism **(b)** Facebook

Fig. 10. CMF of attributed community search under different graph models.



(a) Terrorism **(b)** Facebook

Fig. 11. CPJ of attributed community search under different graph models.

TABLE III
MEAN RELATIVE ERRORS OF CPJ AND CMF UNDER DIFFERENT GRAPH MODELS

	Facebook			Terrorist		
	AsgLDPBTER	AsgLDP-CL	AsgLDP-TCL	AsgLDP-BTER	AsgLDP-CL	AsgLDP-TCL
CPJ	0.1055	0.2227	0.2169	0.0965	0.1484	0.1488
CMF	0.1790	0.0998	0.1429	0.0788	0.0630	0.0502

VII. Related Work

A. Generative Graph Model

A large number of graph generation methods have been proposed in the literature [16], [17], [22], [24], [30], [49]. The first generative graph model, the Erdos Renyi model [49], constructs every edge with an equal probability. Chuang and LU proposed the CL model [30], which extends ER models and allows edges to exist with different probabilities. However, the structural features of this model failed to match those found in many real world networks, which motivates the development of its variants with increased complexities and more accurate graph modeling, such as TCL graph model [17], BETR graph model [16], kronecker graph model [22], exponential random graph model [24], etc.

The majority of these approaches only attempt to model structure features of the graphs while ignoring the node attributes. One notable exception to this is the multiplicative attributed graph model [25], which leverages latent node attributes in order to match relational structure. Another notable exception is the attributed graph model [19], which generates accurate samples and models the distribution of graphs with similar structures and correlations under an observed set of edges and node attributes.

B. Differentially Private Graph Generation Model

The aforementioned graph generation models fail to prevent information leakage in the process of generating synthetic graphs. To address this issue, recent works have focused on developing graph generation methods while ensuring differential privacy in the centralized setting [18], [20], [36], [50], [51]. There are two concepts developed in the community of differentially private graph data generation: node differential privacy and edge differential privacy [6], [7]. The key steps behind these differentially private graph generation methods

are: 1) extracting model parameters from the original input data, 2) generating noisy model parameters while satisfying differential privacy and 3) constructing synthetic graph data based on graph generation models with these noisy parameters. However, these methods require the data publisher to know the entire input graph. As a consequence, they are not applicable for the decentralized setting as considered in our paper where the data publisher only has limited local view of the entire graph.

C. General LDP Applications

LDP is an important variant of differential privacy in order to protect privacy for distributed users under an untrusted data curator [4], [75]. An appropriate amount of noise is inserted to the local query results computed by each user before being transmitted to the untrusted data curator. The curator then conducts post-processing on the noisy data to obtain the aggregated query outputs, which can be further shared with the public. LDP was first suggested by Evfimievski [75]. Then, Duchi et al. systematically investigate the framework of LDP and show an upper utility bound under LPD from the perspective of information theory [4]. Since then many LDP techniques have been proposed for frequency estimation of numerical or categorical values. Erlingsson et al. propose RAPPOR [11], which is the first LDP technique for frequency estimation in real-world applications. Fanti et al. extend RAPPOR to learn data frequencies without explicit dictionary knowledge [40]. Kairouz et al. further introduce k-random response to protect categorical attributes with an arbitrary number of possible values while guaranteeing LDP [53]. They also present O-random response and O-RAPPOR by combining cohort based hashing with k-random response and RAPPOR, respectively [52]. Hsu et al. use random projection and concentration of measure to estimate heavy hitters while satisfying LDP [54]. Then, Bassily et al. propose an algorithm for succinct histogram estimation with an information-theoretical optimal error while satisfying LDP guarantees [15].

Another thread of research uses frequency estimation as a primitive to protect data privacy in other domains. Qin et al. propose a two-phase LDP mechanism, named as LDPMiner, which aims to provide accurate heavy hitters estimation over set-valued data with LDP [12]. Nguyen et al. propose Harmony, as a practical, accurate and efficient system for collecting and analyzing data from users of smart devices [13]. Harmony can handle multi-dimensional data containing both numerical and categorical attributes, and support basic statistics as well as complex machine learning tasks. Furthermore, Ye et al. propose PrivKV, which aims to provide frequency and mean estimation on key-value data while satisfying LDP [14]. Besides, there also exist works for high-dimensional data publication [41], [57], [58].

D. LDP on Decentralized Attributed Graphs

The closest work to ours is LDPGen [5], which aims to synthesize decentralized social graphs with LDP. However, it is limited to protection of the graph structure which fails to take the attributed information of the graph into consideration. To the best of our knowledge, we are the first to propose an effective method for synthesizing attributed graphs in a decentralized manner while satisfying LDP.

VIII. Conclusion

In this paper, we propose AsgLDP to protect privacy for attributed social graphs in a decentralized manner. By carefully analyzing the structural and attribute characteristics of the social graphs, our framework aims to synthesize decentralized attributed graphs while rigorously satisfying LDP. Both theoretical analysis and extensive experiments confirm the utility, efficiency, and practicality of AsgLDP. We can easily extend AsgLDP to accommodate a variety of data types through modification of step 1 in the first phase. How to further improve the efficiency of AsgLDP in processing different types of attributes would be an interesting direction of future work. In the future, we plan to incorporate node LDP, weights and attributes of edges into the framework of AsgLDP. Furthermore, how to adapt AsgLDP to accommodate complicated machine learning tasks would be another interesting future direction.

Appendix

A. Proof of Theorem 4

Private Distribution Estimation: Fix M_{RJ} and $\hat{\boldsymbol{f}}$ to be the empirical estimator given in (6), we have that

$$
\begin{aligned}
&\mathbb{E}||\hat{\boldsymbol{f}}-\boldsymbol{f}||_2^2 \\
&= \mathbb{E}||\frac{e^\epsilon+2r}{e^\epsilon-1}\hat{\boldsymbol{f}}_{\boldsymbol{o}}-\frac{1}{e^\epsilon-1}-\boldsymbol{f}||_2^2 \\
&= \mathbb{E}||\frac{e^\epsilon+2r}{e^\epsilon-1}\left(\hat{\boldsymbol{f}}_{\boldsymbol{o}}-\boldsymbol{f}_{\boldsymbol{o}}\right)||_2^2 \\
&= \left(\frac{e^\epsilon+2r}{e^\epsilon-1}\right)^2 \mathbf{E}||\hat{\boldsymbol{f}}_{\boldsymbol{o}}-\boldsymbol{f}_{\boldsymbol{o}}||_2^2 \\
&= \left(\frac{e^\epsilon+2r}{e^\epsilon-1}\right)^2 \frac{1-\sum_{i=1}^{2r+1} f_{oi}^2}{n_v} \\
&= \frac{1}{n}\left(\frac{e^\epsilon+2r}{e^\epsilon-1}\right)^2 \\
&\quad \times\left(1-\frac{\sum_{i=1}^{2r+1}\left\{(e^\epsilon-1)^2\ f_i^2+2\left(e^\epsilon-1\right)f_i+1\right\}}{(e^\epsilon+2r)^2}\right) \\
&= \frac{(e^\epsilon+2r)^2-2\left(e^\epsilon-1\right)-(2r+1)-(e^\epsilon-1)^2\sum_{i=1}^{2r+1} f_i^2}{n_v\left(e^\epsilon-1\right)^2} \\
&= \frac{(e^\epsilon+2r)^2-2\left(e^\epsilon-1\right)-(2r+1)-(e^\epsilon-1)^2}{n_v\left(e^\epsilon-1\right)^2} \\
&\quad +\frac{1-\sum_{i=1}^{2r+1} f_i^2}{n_v} \\
&= \frac{4r^2+4re^\epsilon-2r}{n_v\left(e^\epsilon-1\right)^2}+\frac{1-\sum_{i=1}^{2r+1} f_i^2}{n_v} \\
&= \frac{2r}{n_v}\left(\frac{2(e^\epsilon+r)-1}{(e^\epsilon-1)^2}\right)+\frac{1-\sum_{i=1}^{2r+1} f_i^2}{n_v},
\end{aligned}
$$

Algorithm 5 Joint Distribution of Attributes (AJD)

Input: All users' noisied attribute lists $X' = \{X'_1, ..., X'_n\}$
Domain of each attribute Ω_j $(1 \leq j \leq w)$
Flipping probability p_r
Convergency threshold τ
Output: Joint distribution of w attributes specified by $\mathcal{W}$, i.e., $P(X_{\mathcal{W}})$

1 Initialize $P_0(\omega_{\mathcal{W}}) = 1/(\Pi_{j \in \mathcal{W}} |\Omega_j|)$.
2 **repeat**
3 // E Step
4 **for** *each* $i = 1, ..., n$ **do**
5 **for** *each* $(\omega_{\mathcal{W}}) \in \Omega_1 \times \Omega_2 \times \cdots \times \Omega_w$ **do**
6

$$P_t(\omega_{\mathcal{W}}|X'_i) = \frac{P_t(\omega_{\mathcal{W}}) \cdot P(X'_i|\omega_{\mathcal{W}})}{\sum_{\omega_{\mathcal{W}}} P_t(\omega_{\mathcal{W}}) P(X'_i|\omega_{\mathcal{W}})}$$

7 **end**
8 **end**
9 // M step
10 set $P_{t+1}(\omega_{\mathcal{W}}) = \frac{1}{n} \sum_{i=1}^{n} P_t(\omega_{\mathcal{W}}|X'_i)$
11 **until** $\max_{\mathcal{W}} |P_{t+1}(\mathcal{W}) - P_t(\mathcal{W})| \leq \tau$;
12 return $P(X_{\mathcal{W}}) = P_{t+1}(\omega_{\mathcal{W}})$

and

$$\begin{aligned}
&\mathbb{E}||\hat{\boldsymbol{f}} - \boldsymbol{f}||_1^1 \\
&= \mathbb{E}||\frac{e^\epsilon + 2r}{e^\epsilon - 1} \hat{\boldsymbol{f}}_{\boldsymbol{o}} - \frac{1}{e^\epsilon - 1} - \boldsymbol{f}||_1^1 \\
&= \mathbb{E}||\frac{e^\epsilon + 2r}{e^\epsilon - 1} \left(\hat{\boldsymbol{f}}_{\boldsymbol{o}} - \boldsymbol{f}_{\boldsymbol{o}}\right)||_1^1 \\
&\approx \frac{e^\epsilon + 2r}{e^\epsilon - 1} \sum_{i=1}^{2r+1} \sqrt{\frac{2 f_i (1 - f_i)}{\pi n_v}} \\
&= \frac{1}{e^\epsilon - 1} \sum_{i=1}^{2r+1} \sqrt{\frac{2 ((e^\epsilon - 1) + 1) (2r + (e^\epsilon - 1)(1 - f_i))}{\pi n_v}} \\
&= \sum_{i=1}^{2r+1} \sqrt{\frac{2 ((e^\epsilon - 1)(1 - f_i) + 2r) ((e^\epsilon - 1) f_i + 1)}{\pi n_v (e^\epsilon - 1)^2}}
\end{aligned}$$

Observe that for $\boldsymbol{f}_U = \left(\frac{1}{2r+1}, ..., \frac{1}{2r+1}\right)$, we have that

$$\begin{aligned}
\mathbb{E}||\hat{\boldsymbol{f}} - \boldsymbol{f}||_2^2 &\leq \mathbb{E}||\hat{\boldsymbol{f}} - \boldsymbol{f}_U||_2^2 \\
&= \left(1 + \frac{(2(e^\epsilon + r) - 1)(2r + 1)}{(e^\epsilon - 1)^2}\right) \frac{2r}{n_v (2r + 1)} \\
&= \frac{2r}{n_v (2r + 1)} + \frac{4r (e^\epsilon + r) - 2r}{(e^\epsilon - 1)^2}
\end{aligned}$$

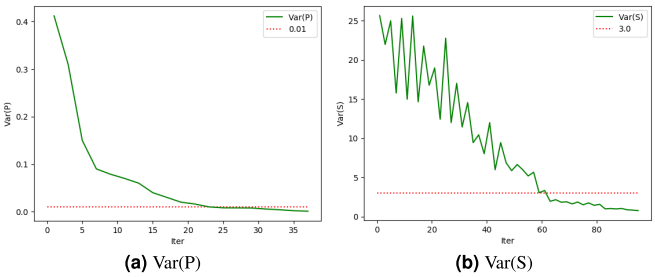
and

$$\begin{aligned}
\mathbb{E}||\hat{\boldsymbol{f}} - \boldsymbol{f}||_1^1 &\leq \mathbb{E}||\hat{\boldsymbol{f}} - \boldsymbol{f}_U||_1^1 \\
&\approx 2\frac{(e^\epsilon + 2r)}{(e^\epsilon - 1)} \sqrt{\frac{r}{\pi n_v}}
\end{aligned}$$

B. Algorithm for Estimation of Joint Distribution of Attributes

C. Investigating Proper Values of τ and ϵ

We denote the difference between two posterior probabilities in Algorithm 5 (Line 11) as $Var(P) = \max_{\mathcal{W}} |P_{t+1}(\mathcal{W}) - P_t(\mathcal{W})|$. Figure 12(a) presents the value of $Var(P)$ in each iteration on Facebook dataset, from which we observe that our method can learn the joint distribution of attributes and converge quickly. In order to balance computation time and accuracy, we set the threshold $\tau = 0.01$ in our experiments.



(a) Var(P) **(b)** Var(S)

Fig. 12. The experiments on τ and ξ parameters, where $Var(P) = \max_{\mathcal{W}} |P_{t+1}(\mathcal{W}) - P_t(\mathcal{W})|$ and $Var(S) = S' - S$.

We denote the difference between S' and S as $Var(S) = S' - S$ (Algorithm 4 Line 22). Figure 12(b) shows the value of $Var(S)$ in each iteration on Facebook dataset. We can observe that the $Var(S)$ fluctuates largely at the initial stages and then coverges quickly in later stages (i.e., the graph structure tends to be stable). In order to balance computation time and accuracy, we set the threshold $\xi = 3.0$ in our experiments.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments and input to improve this article. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Combat Capabilities Development Command Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

REFERENCES

[1] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, 2013.
[2] C. Dwork, F. Mc, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. Theory Cryptogr. Conf.* Berlin, Germany: Springer, 2006, pp. 265–284.
[3] C. Dwork, G. N. Rothblum, and S. Vadhan, "Boosting and differential privacy," in *Proc. IEEE 51st Annu. Symp. Found. Comput. Sci.*, Oct. 2010, pp. 51–60.
[4] 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.*, Oct. 2013, pp. 429–438.
[5] 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. (CCS)*, 2017, pp. 425–438.
[6] 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.*, Oct. 2012, pp. 410–419.
[7] S. P. Kasiviswanathan, K. Nissim, S. Raskhodnikova, and A. Smith, "Analyzing graphs with node differential privacy," in *Theory of Cryptography*. Berlin, Germany: Springer, 2013, pp. 457–476.
[8] F. D. McSherry, "Privacy integrated queries: An extensible platform for privacy-preserving data analysis," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2009, pp. 19–30.

[9] C. Bothorel, J. D. Cruz, M. Magnani, and B. Micenková, "Clustering attributed graphs: Models, measures and methods," *Netw. Sci.*, vol. 3, no. 3, pp. 408–444, Sep. 2015.

[10] B. Perozzi, L. Akoglu, P. Iglesias Sánchez, and E. Müller, "Focused clustering and outlier detection in large attributed graphs," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2014, pp. 1346–1355.

[11] 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.

[12] 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. (CCS)*, 2016, pp. 192–203.

[13] T. T. Nguyên, X. Xiao, Y. Yang, S. Cheung Hui, H. Shin, and J. Shin, "Collecting and analyzing data from smart device users with local differential privacy," 2016, *arXiv:1606.05053*. [Online]. Available: http://arxiv.org/abs/1606.05053

[14] Q. Ye, H. Hu, X. Meng, and H. Zheng, "PrivKV: key-value data collection with local differential privacy," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 317–331.

[15] R. Bassily and A. Smith, "Local, private, efficient protocols for succinct histograms," in *Proc. 47th Annu. ACM Symp. Theory Comput. (STOC)*, 2015, pp. 127–135.

[16] C. Seshadhri, T. G. Kolda, and A. Pinar, "Community structure and scale-free collections of Erdős-Rényi graphs," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 85, no. 5, 2012, Art. no. 056109.

[17] J. J. Pfeiffer, T. La Fond, S. Moreno, and J. Neville, "Fast generation of large scale social networks while incorporating transitive closures," in *Proc. Int. Conf. Privacy, Secur., Risk Trust Int. Conf. Social Comput.*, Sep. 2012, pp. 154–165.

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

[19] J. J. Pfeiffer III, S. Moreno, T. La Fond, J. Neville, and B. Gallagher, "Attributed graph models: Modeling network structure with correlated attributes," in *Proc. 23rd Int. Conf. World Wide Web*, 2014, pp. 831–842.

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

[21] D. J. Mir and R. N. Wright, "A differentially private graph estimator," in *Proc. IEEE Int. Conf. Data Mining Workshops*, Dec. 2009, pp. 122–129.

[22] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani, "Kronecker graphs: An approach to modeling networks," *J. Mach. Learn. Res.*, vol. 11, no. 3, pp. 985–1042, 2010.

[23] F. Chung and L. Lu, "The average distance in a random graph with given expected degrees," *Internet Math.*, vol. 1, no. 1, pp. 91–113, Jan. 2004.

[24] G. Robins, P. Pattison, Y. Kalish, and D. Lusher, "An introduction to exponential random graph (p*) models for social networks," *Social Netw.*, vol. 29, no. 2, pp. 173–191, May 2007.

[25] M. Kim and J. Leskovec, "Multiplicative attribute graph model of real-world networks," *Internet Math.*, vol. 8, nos. 1–2, pp. 113–160, Mar. 2012.

[26] D. K. Sewell and Y. Chen, "Latent space approaches to community detection in dynamic networks," *Bayesian Anal.*, vol. 12, no. 2, pp. 351–377, Jun. 2017.

[27] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing, "Mixed membership stochastic blockmodels," *J. Mach. Learn. Res.*, vol. 9, pp. 1981–2014, Sep. 2008.

[28] T. La Fond and J. Neville, "Randomization tests for distinguishing social influence and homophily effects," in *Proc. 19th Int. Conf. World Wide Web (WWW)*, 2010, pp. 601–610.

[29] F. Liang, C. Liu, and R. Carroll, *Advanced Markov Chain Monte Carlo Methods: Learning From Past Samples*, vol. 714. Hoboken, NJ, USA: Wiley, 2011.

[30] F. Chung and L. Lu, "The average distances in random graphs with given expected degrees," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 25, pp. 15879–15882, Dec. 2002.

[31] B. Perozzi and L. Akoglu, "Discovering communities and anomalies in attributed graphs: Interactive visual exploration and summarization," *ACM Trans. Knowl. Discovery Data*, vol. 12, no. 2, p. 24, 2018.

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

[33] Z. Zhou, Z. Qian, M. K. Reiter, and Y. Zhang, "Static evaluation of noninterference using approximate model counting," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2018, pp. 514–528.

[34] S. Chakraborty, K. S. Meel, and M. Y. Vardi, "A scalable approximate model counter," in *Proc. Int. Conf. Princ. Pract. Constraint Program.* Berlin, Germany: Springer, 2013, pp. 200–216.

[35] 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. (IMC)*, 2011, pp. 81–98.

[36] Y. Wang and X. Wu, "Preserving differential privacy in degree-correlation based graph generation," *Trans. Data Privacy*, vol. 6, no. 2, p. 127, 2013.

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

[38] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annu. Rev. Sociol.*, vol. 27, no. 1, pp. 415–444, Aug. 2001.

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

[40] G. Fanti, V. Pihur, and Ú. Erlingsson, "Building a RAPPOR with the unknown: privacy-preserving learning of associations and data dictionaries," *Proc. Privacy Enhancing Technol.*, vol. 2016, no. 3, pp. 41–61, Jul. 2016.

[41] X. Ren *et al.*, "LoPub: High-dimensional crowdsourced data publication with local differential privacy," 2016, *arXiv:1612.04350*. [Online]. Available: http://arxiv.org/abs/1612.04350

[42] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'smallworld' networks," *Nature*, vol. 393, no. 6684, p. 440, 1998.

[43] J. Leskovec and J. J. Mcauley, "Learning to discover social circles in ego networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 539–547.

[44] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 US election: Divided they Blog," in *Proc. 3rd Int. workshop Link Discovery*, 2005, pp. 36–43.

[45] B. Zhao, P. Sen, and L. Getoor, "Event classification and relationship labeling in affiliation networks," in *Proc. Workshop Stat. Netw. Anal. (SNA) 23rd Int. Conf. Mach. Learn. (ICML)*, 2006, pp. 271–280.

[46] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowl. Inf. Syst.*, vol. 42, no. 1, pp. 181–213, Jan. 2015.

[47] J. Yang, J. McAuley, and J. Leskovec, "Community detection in networks with node attributes," in *Proc. IEEE 13th Int. Conf. Data Mining*, Dec. 2013, pp. 1151–1156.

[48] Y. Fang, R. Cheng, S. Luo, and J. Hu, "Effective community search for large attributed graphs," *Proc. VLDB Endowment*, vol. 9, no. 12, pp. 1233–1244, Aug. 2016.

[49] P. Erdős and A. Rényi, "On the evolution of random graphs" *Publ. Math. Inst. Hung. Acad. Sci*, vol. 5, no. 1, pp. 17–60, 1960.

[50] W.-Y. Day, N. Li, and M. Lyu, "Publishing graph degree distribution with node differential privacy," in *Proc. Int. Conf. Manage. Data (SIGMOD)*, 2016, pp. 123–138.

[51] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "Private release of graph statistics using ladder functions," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2015, pp. 731–745.

[52] P. Kairouz, K. Bonawitz, and D. Ramage, "Discrete distribution estimation under local privacy," 2016, *arXiv:1602.07387*. [Online]. Available: http://arxiv.org/abs/1602.07387

[53] P. Kairouz, S. Oh, and P. Viswanath, "Extremal mechanisms for local differential privacy," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2879–2887.

[54] J. Hsu, S. Khanna, and A. Roth, "Distributed private heavy hitters," in *Proc. Int. Colloq. Automata, Lang., Program.* Springer, 2012, pp. 461–472.

[55] S. Ji, P. Mittal, and R. Beyah, "Graph data anonymization, deanonymization attacks, and de-anonymizability quantification: A survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 1305–1326, 2nd Quart., 2016.

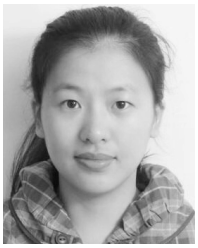
[56] S. Ji, W. Li, M. Srivatsa, and R. Beyah, "Structural data deanonymization: Theory and practice," *IEEE/ACM Trans. Netw.*, vol. 24, no. 6, pp. 3523–3536, Dec. 2016.

[57] 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.*, Jan. 2018, pp. 212–229.

[58] G. Cormode, T. Kulkarni, and D. Srivastava, "Marginal release under local differential privacy," in *Proc. Int. Conf. Manage. Data (SIGMOD)*, 2018, pp. 131–146.
[59] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 70, no. 6, Dec. 2004, Art. no. 066111.
[60] W. Assaad and J. M. Gómez, "Social network in marketing (social media marketing) opportunities and risks," *Int. J. Manag. Public Sector Inf. Commun. Technol.*, vol. 2, no. 1, p. 13, 2011.
[61] S. Ji, W. Li, P. Mittal, X. Hu, and R. Beyah, "Secgraph: A uniform and open-source evaluation system for graph data anonymization and de-anonymization," in *Proc. 24th USENIX Secur. Symp.*, 2015, pp. 303–318.
[62] S. Ji, W. Li, N. Z. Gong, P. Mittal, and R. Beyah, "On your social network de-anonymizablity: Quantification and large scale evaluation with seed knowledge," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2015, pp. 1–15.
[63] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2010, pp. 1029–1038.
[64] H. Gao, J. Tang, X. Hu, and H. Liu, "Content-aware point of interest recommendation on location-based social networks," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 1721–1727.
[65] Z. Wang, J. Liao, Q. Cao, H. Qi, and Z. Wang, "Friendbook: A semantic-based friend recommendation system for social networks," *IEEE Trans. Mobile Comput.*, vol. 14, no. 3, pp. 538–551, Mar. 2015.
[66] S. Ji, W. Li, M. Srivatsa, and R. Beyah, "Structural data de-anonymization: Quantification, practice, and implications," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2014, pp. 1040–1053.
[67] M. A. Al-garadi, M. S. Khan, K. D. Varathan, G. Mujtaba, and A. M. Al-Kabsi, "Using online social networks to track a pandemic: A systematic review," *J. Biomed. Informat.*, vol. 62, pp. 1–11, Aug. 2016.
[68] R. G. Rodrigues, R. M. das Dores, C. G. Camilo-Junior, and T. C. Rosa, "SentiHealth-cancer: A sentiment analysis tool to help detecting mood of patients in online social networks," *Int. J. Med. Informat.*, vol. 85, no. 1, pp. 80–95, Jan. 2016.
[69] P. Jaccard, "Distribution de la flore alpine dans le Bassin des Dranses et dans quelques regions voisines," *Bull. Soc. Vaudoise Sci. Nat.*, vol. 37, pp. 241–272, Jan. 1901.
[70] S. Hassani, *Mathematical Methods: For Students of Physics and Related Fields*, vol. 720. Berlin, Germany: Springer, 2008.
[71] E. Hellinger, "Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen," *J. für die reine und angewandte Mathematik*, vol. 136, pp. 210–271, 1909.
[72] D. Silver, H. van Hasselt, M. Hessel, T. Schaul, A. Guez, T. Harley, G. Dulac-Arnold, D. Reichert, N. Rabinowitz, A. Barreto, and T. Degris, "The predictron: End-to-end learning and planning," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 3191–3199.
[73] A. Kolmogorov, "Sulla determinazione empirica di una legge di distribuzione," *Selected Works AN Kolmogorov*, vol. 2, pp. 139–147, 1933.
[74] T. Gao, F. Li, Y. Chen, and X. Zou, "Local differential privately anonymizing online social networks under HRG-based model," *IEEE Trans. Comput. Social Syst.*, vol. 5, no. 4, pp. 1009–1020, 2018.
[75] A. Evfimievski, "Randomization in privacy preserving data mining," *ACM SIGKDD Explor. Newslett.*, vol. 4, no. 2, pp. 43–48, Dec. 2002.

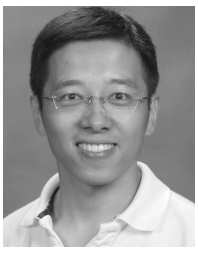
Chengkun Wei received the M.S. degree in computer science and technology from the Second Institute of China Aerospace Science. He is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Zhejiang University. His current research interests include federated learning, big data privacy, and security.

Shouling Ji (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology and the second Ph.D. degree in computer science from Georgia State University. He is currently a ZJU 100-Young Professor with the College of Computer Science and Technology, Zhejiang University, and a Research Faculty Member of the School of Electrical and Computer Engineering, Georgia Institute of Technology. His current research interests include AI security, data-driven security, privacy, and data analytics. He was the Membership Chair of the IEEE Student Branch at Georgia State from 2012 to 2013. He is a member of ACM.



Changchang Liu received the Ph.D. degree in electrical engineering from Princeton University. She is currently a Research Staff Member of the Department of Distributed AI, IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA. Her current research interests include federated learning, big data privacy, and security.

Wenzhi Chen (Member, IEEE) received the Ph.D. degree from the College of Computer Science and Engineering, Zhejiang University. He is currently a Professor with the College of Computer Science and Technology, Zhejiang University, and the Director of the Information Technology Center, Zhejiang University. He used to be the Vice Dean of the College of Computer Science and Technology. His current research interests include embedded systems and its application, computer architecture, computer system software, and information security. He is a member of ACM and the ACM Education Council.



Ting Wang received the Ph.D. degree from the Georgia Institute of Technology. He is currently an Assistant Professor with the College of Information Science and Technology, Pennsylvania State University. He conducts research at the intersection of machine learning and privacy and security. His ongoing work focuses on making machine learning systems more practically usable through mitigating security vulnerabilities, enhancing privacy awareness, and increasing decision-making transparency.