IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 4, 2020, accepted September 14, 2020, date of publication September 24, 2020, date of current version October 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3026008

Protecting Social Network With Differential Privacy Under Novel Graph Model

TIANCHONG GAO[1,2], (Member, IEEE), AND FENG LI[3], (Member, IEEE)

[1]School of Cyber Science and Engineering, Southeast University, Nanjing 210096, China
[2]Purple Mountain Laboratories for Network and Communication Security, Nanjing 210096, China
[3]Department of Electrical Engineering, Indiana University—Purdue University Indianapolis, Indianapolis, IN 46202, USA

Corresponding author: Feng Li (fengli@iupui.edu)

This work was supported in part by the U.S. National Science Foundation under Grant CNS-1852105 and Grant CNS-1560020, in part by the National Natural Science Foundation of China under Grant 62002059, and in part by the Fundamental Research Funds for the Central Universities of China under Grant 224202R10061.

ABSTRACT Online social networks (OSNs) contain sensitive information about individuals, so it's important to anonymize network data before releasing it. Recently, researchers introduced differential privacy to give a strict privacy guarantee. Graph abstraction models are essential to transform graph structural information into numerical type data, and the choice of models may influence the utility preservation of the published graph. In this paper, we propose a comprehensive differentially private graph model which combines the dK-1, dK-2, and dK-3 series together. The dK-1 series stores the degree frequency, the dK-2 series adds the joint degree frequency, and the dK-3 series contains the linking information between edges. In our scheme, low dimensional series data makes the regeneration process more executable and effective, while high dimensional data preserves additional utility of the graph. As the higher dimensional data is more sensitive to the noise, we carefully design the executing sequence and add three levels of rewiring algorithms to further preserve the structural information. The final released graph increases the graph utility under differential privacy. We also experimentally evaluate our approach on real-world OSNs and show that our scheme produces ready-to-be-shared graphs that are closely matched with the originals, while achieving differential privacy.

INDEX TERMS Social network data publishing, anonymization, differential privacy, dK graph abstraction model.

I. INTRODUCTION

Studying online social networks (OSNs) through graph analysis could produce knowledge of human social relationships, help feed advertisements to recommendation targets, and evaluate the effectiveness of applications. Since OSN data contains personal information, any releasing procedure without sufficient anonymization work causes panic to the users of social media. Various anonymization techniques have been proposed. Differential privacy is one of the most remarkable techniques, since it could theoretically achieve a strong privacy guarantee [4].

Differential privacy requires graph abstraction models to convert the graph structure into numerical-like data. Figure 1 gives an example of the dK model. The dK model is separated into different dimensions. The dK-N model captures the degree distribution of connected components of size N. *N. Sala et al.* employed the dK-2 series as the graph abstraction model to achieve differential privacy [22].

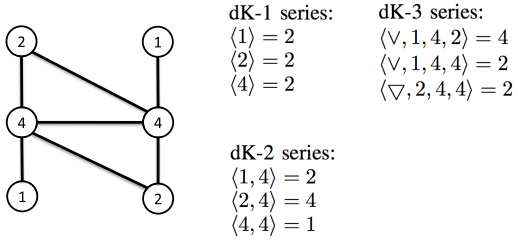

FIGURE 1. An example of the dK model.

However, deploying one abstraction model can only capture some aspects of information, while other utilities are lost in the published graph. For example, because the dK-2 graph model is the record of edges, it may not preserve information involving more than two nodes, e.g., the clustering coefficient. The limitations in the models restrict their

The associate editor coordinating the review of this manuscript and approving it for publication was Vijay Mago.

ability to achieve structural similarity under differential privacy. Therefore, choosing the right abstraction model becomes an important issue. Mahadevan et al. proved that dK models in higher dimensions have more information than the ones in lower dimensions, e.g., the dK-3 model is more precise than the dK-2 model [17].

Since different models have different advantages, choosing the abstraction model becomes an important issue. After studying the differences of abilities between dK-1, dK-2, and dK-3 series, we find that low dimensional models, e.g., dK-1, are less sensitive to noise and can easily regenerate a graph. However, high dimensional models can preserve more structural information. Our initial idea is to preserve differential privacy with the dK-3 model. To the best of our knowledge, there is no systematic regeneration algorithm for the dK-3 model because of its complexity [17], [22]. In our study, we also find that it is hard to reconstruct the graph with only the dK-3 series. However, dK-3 information can still be embedded in the published graph. We find that low dimension dK series, i.e., dK-1 and dK-2, can help the regeneration process. And we can use some rewiring algorithms to inject the dK-3 series in our published graph.

Hence, we absorb the benefits of different models and design a comprehensive model that combines three levels of dK graph models together. To achieve differential privacy, we introduce noise on the dK-2 level, which causes less distortion than with the dK-3 level. Then we use the perturbed dK-2 series to get the corresponding dK-3 and dK-1 series. After that, we use three levels of dK series together in our scheme to construct a new graph.

The impact of noise is the major challenge in the graph regeneration process. Although the three models in our scheme are closely related, they may have conflicts with each other because of noise. Hence, we first use some dK information to regenerate an intermediate graph, then use the remaining information to rewire the edges. In particular, we propose two sub-schemes, namely *consider all together* (CAT) and *lower to higher* (LTH), with different executing sequences in the dK series. The CAT scheme uses all three kinds of dK information in the regeneration phase. It aims to reduce the errors of the dK-2 and dK-3 series. Because the CAT intermediate graph preserves some dK-3 information, it is easy to apply the dK-3 rewiring in the following phases. By contrast, the LTH scheme just focuses on the dK-1 series in the regeneration phase. The intermediate graph fits the dK-1 information extremely well.

In our previous work, we demonstrated the algorithms for building the intermediate graphs [7]. In this paper, our general purpose of graph regeneration is minimizing the error between the target dK series and the one in the published graph under all three levels. Hence, we develop three dK rewiring algorithms to reduce the errors graphically. These rewiring algorithms also help us inject the remaining dK information into the graph.

The major technical contributions are the following: (1) We are the first to build the systematic regeneration algorithm for embedding dK-3 information in graph anonymization, which helps to preserve more utility than existing dK models. (2) We combine the dK-3 model with both dK-1 and dK-2 models in sampling and graph regeneration, which mitigates the high sensitivity and complexity in the dK-3 model and makes the design practical. (3) We design two different routes, CAT and LTH, to generate the graph efficiently and effectively, even under the impact of noise. (4) We use three levels of rewiring algorithms to actively reduce the errors between the desired dK series and the published graph. (5) We reveal the insights and challenges of using different levels of dK abstraction models jointly to enhance the utility under differential privacy.

II. RELATED WORK

Researchers proposed several anonymization techniques to preserve the privacy in OSN data sharing. Naive ID removal, which removes users' identities, is the simplest way but it is vulnerable to structural information attacks [13], [19]. k-anonymity is another kind of anonymization techniques [28], [29]. k-anonymity requires that there are at least k elements in each category, then the attacker is hard to differentiate these K elements. However, k-anonymity is often designed for some specific structure semantics, e.g., neighbors of each node [29], persistent structures [23], and they are overcome by other semantics, e.g., nodes' hierarchies or users' attributes [27].

On the attackers' side, various structural-based de-anonymization attacks have been proposed [9], [14], [16], [21], [25]. They revealed the vulnerability of pervious anonymization techniques in different angles. Some attacks used the seeds in their de-anonymization to have better accuracy and efficiency [9], [14]. In [20], Qian et al. also introduced semantic knowledge to support the structural-based attack.

Hence, researchers utilized differential privacy to provide a strong privacy guarantee [4]. Nowadays, differential privacy has been widely adopted in privacy preservation for research purposes and commercial purposes, e.g., Apple and Google [2], [24]. However, differential privacy was originally proposed for numerical-type data in databases. Researcher need the graph abstraction model to transform OSN from graph-type data into numerical-type data. For example, our work is inspired by existing work about degree sequence model (dK-1) and joint degree model (dK-2) [3], [11]. Researchers also apply other graph abstraction models like the hierarchical random graph model and the adjacency matrix model [1], [8], [10].

In [6], we compared the dK series model with other graph abstraction models, our evaluation results show that the dK series model captures and stores more relative information about degree. As shown in [8], it is hard to combine different aspects of structural information, e.g., degree information and clustering information, into one model. Our scheme introduces different levels of dK series as an attempt.

Recovering the graph-type data after noise injection is another challenge in OSN anonymization. Mahadevan et al. proposed the randomized rewiring concept and they accepted the rewiring procedure if it could reduce the dK-3 error. Gjoka et al. use a semi-active rewiring algorithm which deploys the sequence of dK-2 series to simulate the clustering coefficient [11]. Inspired by these researches, we propose the active rewiring algorithm in our scheme.

III. PRELIMINARIES

A. THE DK GRAPH MODEL

In this paper, the OSN is modeled as an undirected graph $G = (V, E)$, where V is the vertex set and E is the edge set. We denote $|V|$ as the cardinality of the set V, and d_v as the degree of the vertex v. $e_{u,v}$ means that there exists an edge between node u and v.

Since the differential privacy is applied on the query result, typically the numerical type data, the dK graph model is chosen as the graph abstraction model to transform the graph structures into a set of structural statistics. The dK graph model is better than most of the other graph abstraction models because the dK series could be used to re-construct a new graph. This graph has similar structural information with the original graph, so it can be used as the released OSN which preserves private information and useful information.

The dK-N model captures the degree distribution of size-N-connected-components in the target graph [17]. For example, the dK-1 model, known as the degree distribution, counts the number of nodes in each degree value. The dK-2 model, known as the joint degree distribution, counts the number of edges in the combination of two degree values. The dK-3 model counts the number of 3-node subgraphs in the combination of three degree values. Specifically, there are two kinds of 3-node subgraphs with different structures, wedges and triangles. In this paper, we also define the dimension of dK information as the subgraph size N, i.e., dK-1 series has lower dimension than dK-2.

1) THE WEDGE dK-3 ENTRY

The dK-3 entry $\langle \vee, d_u, d_v, d_w \rangle = k$ means that there are k 3-node wedges which have the node degree values equal to d_u, d_v and d_w, and each of the two subgraphs have at least one different node. In order to prevent double counting, d_u should be less than or equal to d_w. Assume the combination of node u, v and w forms such a subgraph, then w should not be the neighborhood of u. The node set of the subgraph should be $V = \{u\} \cup \{v|e_{u,v} \in E\} \cup \{w|e_{v,w} \in E \wedge e_{u,w} \notin E\}$.

2) THE TRIANGLE dK-3 ENTRY

The dK-3 entry $\langle \nabla, d_u, d_v, d_w \rangle = k$ means that there are k triangles with node degree d_u, d_v and d_w. To prevent double counting, we have $d_u \leqslant d_v \leqslant d_w$. The node set of the subgraph should be $V = \{u\} \cup \{v|e_{u,v} \in E\} \cup \{w|e_{v,w} \in E \wedge e_{u,w} \in E\}$.

The error between two dK-3 series is defined as the sum of all absolute differences in each corresponding dK-3 entry.

$$err_3 = \sum_{\text{dK-3 } entry} |k_i - k_i'| \tag{1}$$

Similarly, err_1 and err_2 measure the error in dK-1 and dK-2 series. And our work focuses on minimizing the error between the dK series in the published graph and the target dK series calculated under differential privacy.

B. DIFFERENTIAL PRIVACY

Differential privacy is designed to protect the privacy between neighboring databases which differ in only one element. In the model of OSNs, the adversary is not able to be sure if two users are linked in the original network. In this paper, the neighbor database/graph refers to a OSN with one edge added or deleted. Then we have the notion of sensitivity.

Definition 1 (Sensitivity): The sensitivity $\triangle f$ of a function f is the maximum distance of any two neighbor databases D_1 and D_2 in ℓ_1 norm.

$$\Delta f = \max_{D_1, D_2} \|f(D_1) - f(D_2)\| \tag{2}$$

Definition 2 (ϵ-Differential Privacy): A randomized algorithm $\mathcal{A}$ achieves ϵ-differential privacy if for all neighbor datasets and all $S \subseteq Range(\mathcal{A})$

$$\Pr[\mathcal{A}(D_1) \in S] \leq e^{\epsilon} \times \Pr[\mathcal{A}(D_2) \in S] \tag{3}$$

Equation (3) calculates the probability that two neighbor databases have the same result under the same algorithm. Based on the definition, researchers designed the Laplace mechanism to achieve ϵ-differential privacy when the entries have real values. It adds Laplace noise with respect to the sensitivity $\triangle f$ and the desired security parameters ϵ to the result. In particular, the noise is drawn from a Laplace distribution with the density function $p(x|\lambda) = \frac{1}{2\lambda} e^{\frac{-|x|}{\lambda}}$, where $\lambda = \frac{\triangle f}{\epsilon}$.

Theorem 1 (Laplace Mechanism): For a function $f : \mathcal{D} \to \mathbb{R}^d$, the randomized algorithm $\mathcal{A}$

$$\mathcal{A}(G) = f(G) + Lap(\frac{\triangle f}{\epsilon}) \tag{4}$$

achieves ϵ-differential privacy [18].

IV. SCHEME

Given an OSN, our goal is to publish an anonymized network which preserves the structural utility as much as possible while satisfying ϵ-differential privacy. The general idea is to add sufficient noise to the dK model and reconstruct a graph G based on the perturbed dK series.

As mentioned in [17], a model of higher dimension is more precise. Compared with previous research which advocates the dK-2 graph model [11], [22], the dK-3 model could preserve more information under differential privacy. It contains not only the information of nodes and edges, but also the linking information between edges. However the sensitivity of the dK-2 model is lower than the dK-3 model. In the scheme, we inject noise to the dK-2 series, which preserves

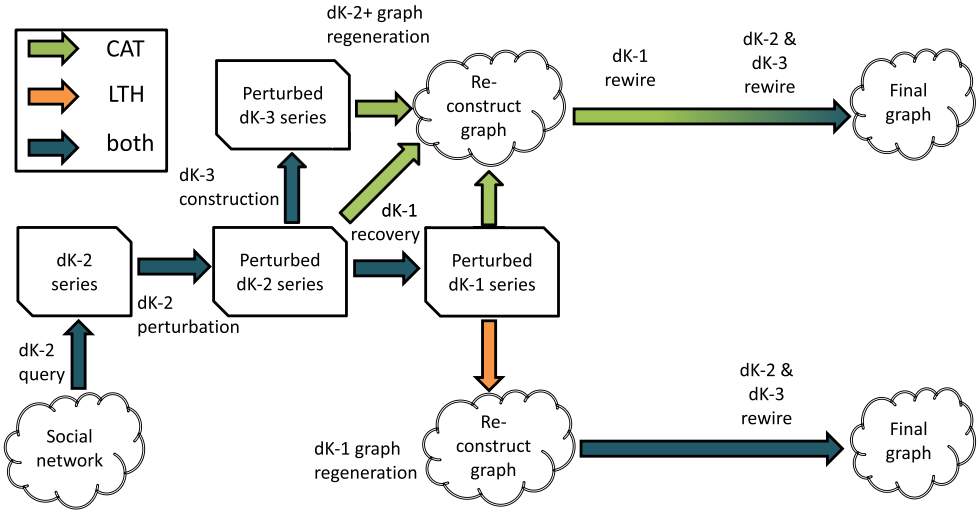

FIGURE 2. Scheme overview.

more utility under the same privacy level. Then we use the perturbed dK-2 series to construct the corresponding dK-3 and dK-1 series. Our purpose of graph regeneration is to publish a graph with similar dK series as the perturbed result.

We propose two sub-schemes, namely *consider all together* (CAT) and *lower to higher* (LTH), in Figure2. The mutual steps are marked in 'both'. The two sub-schemes have the same perturbation process on the dK-2 series. The main differences in the two sub-schemes are in the regeneration algorithms, CAT uses all three kinds of dK series (mainly dK-2 series) while LTH uses dK-1 only. As a result, these two schemes have speciality in reducing dK-1 or dK-2 error. After the regeneration part, both sub-schemes have an active rewiring procedure to mitigate their errors, e.g., the dK-2 and dK-3 series have not been used by LTH.

In the following sections, we discuss these components which are also shown in Figure 2: 1.

1) *dK-2 perturbation:* perturb the dK-2 series under differential privacy,
2) *dK-3 construction:* build the dK-3 model with perturbed dK-2 series,
3) *dK-1 recovery:* recover the dK-1 information,
4) *graph regeneration:* reconstruct the perturbed graph with different combinations of dK series,
5) *target rewiring:* rewire some of the edges according to the dK series.

A. dK-2 PERTURBATION

We find that achieving dK-3 differential privacy needs much more information distortion which largely reduces the benefits of dK-3 model after the analysis in Section V-A,. What's more, as the dK-2 series is the record of edges, we can make it indistinguishable to achieve edge differential privacy. Hence, we choose to inject noise at the dK-2 level. In particular, after counting the dK-2 entries, we add sufficient Laplace noise to achieve differential privacy. According to Equation 4, the noise level is determined by the sensitivity $\triangle f$ and the privacy parameter ϵ.

The sensitivity shows the impact of adding or deleting an edge in the model. For a given entry $\langle d_x, d_y \rangle = k$, the sensitivity is $2 \cdot d_x + 2 \cdot d_y + 1$ where the detail calculating steps are shown in Section V-A. The perturbed dK-2 entry is $\langle d_x, d_y \rangle = k + Lap(\frac{\triangle f}{\epsilon})$.

Example: Figure 1 shows a running example which is also used in the following sections. Figure 3 has the perturbed dK-2 series. If the value of an entry changes, it is marked in red. We can find that some dK-2 series like $\langle 2, 3 \rangle$, although not exist in the original example (have a value of 0), are created. Because of differential privacy request, any entries in the range between $\langle 1, 1 \rangle$ and $\langle d_{max}, d_{max} \rangle$ are modified.

B. dK-3 CONSTRUCTION

Given the dK-2 model, we construct the dK-3 model to preserve edge linking information. Particularly, if one dK-2 entry is perturbed, its corresponding dK-3 entries is also perturbed, which leaks no edge information beyond differential privacy. Hence, we examine the influence of dK-2 perturbation on dK-3 model in the example of one edge $e_{u,v}$, then do the modification.

First, there is a simple case that all three-node pairs in the graph are wedges. There are $d_u - 1$ edges connected with the node u. Then the edge produces $d_u - 1$ dK-3 entries in the form of $\langle \vee, d_x, d_u, d_v \rangle$ or $\langle \vee, d_v, d_u, d_x \rangle$. Similarly, it also produces $d_v - 1$ dK-3 entries in the form of $\langle \vee, d_y, d_v, d_u \rangle$ or $\langle \vee, d_u, d_v, d_y \rangle$. Hence, there are totally $d_u + d_v - 2$ dK-3 wedges entries produced by the edge $e_{u,v}$.

Second, we improve the case that the graph has some triangles. Adding an edge $e_{u,v}$ between node u and v, if they have an common neighbor x, the original entry $\langle \vee, d_u, d_x, d_v \rangle$ will be changed to $\langle \triangledown, d_u, d_x, d_v \rangle$. However, if they do not have the common neighbor, there will be some new entries added like the case before. Therefore, the total number of

dK-3 entries containing the edge $e_{u,v}$ is also affected by the number of triangles.

Perturbed dK-2 series:
$\langle 1,4\rangle = 2-1 = 1$
$\langle 2,4\rangle = 4-1 = 3$
$\langle 4,4\rangle = 1$
$\langle 2,3\rangle = 0+1 = 1$
$\langle 3,4\rangle = 0+1 = 1$

Recovered dK-1 series:
$\langle 1\rangle = 1$
$\langle 2\rangle = 2$
$\langle 3\rangle = 0.7 \approx 1$
$\langle 4\rangle = 1.75 \approx 2$

Constructed dK-3 series:
$\langle \vee, 1, 4, 2\rangle = 4-3 = 1$
$\langle \vee, 1, 4, 3\rangle = 0+1 = 1$
$\langle \vee, 1, 4, 4\rangle = 2-1 = 1$
$\langle \vee, 2, 3, 4\rangle = 0+1 = 1$
$\langle \vee, 2, 4, 3\rangle = 0+2 = 2$
$\langle \triangledown, 2, 4, 4\rangle = 4-2 = 2$
$\langle \triangledown, 4, 2, 4\rangle = 2-1 = 1$
$\langle \triangledown, 2, 3, 4\rangle = 0+1 = 1$
$\langle \triangledown, 2, 4, 3\rangle = 0+1 = 1$
$\langle \triangledown, 3, 2, 4\rangle = 0+1 = 1$
$\langle \triangledown, 3, 4, 4\rangle = 0+2 = 2$
$\langle \triangledown, 4, 3, 4\rangle = 0+1 = 1$

FIGURE 3. Perturbed dK series.

1) ADJUSTED dK-3 MODEL

We find that if we deploy some specific counting method for triangles, the wedges and triangles can be treated equally. Thus, the adjusted dK-3 model is proposed to simplify the calculation of the dK-3 series. The adjusted model is completely based on the basic dK-3 series. Using adjusted model will not increase or decrease the ability of dK-3 series to present or reconstruct the graph. The new model does not change the wedge entry $\langle \vee, d_u, d_v, d_w\rangle$. But if there is a triangle entry $\langle \triangledown, d_u, d_v, d_w\rangle = k$, it will be replaced by three entries, $\langle \triangledown, d_u, d_v, d_w\rangle = k$, $\langle \triangledown, d_v, d_w, d_u\rangle = k$, and $\langle \triangledown, d_w, d_u, d_v\rangle = k$. In the following sections, all dK-3 series are sampled in the adjusted dK-3 model. After deploying the adjusted dK-3 model, deleting or adding an edge $e_{u,v}$ always changes $d_u + d_v - 2$ dK-3 entries. In the following sections, a wildcard character $*$ is used to match $\vee$ and $\triangledown$. The dK-3 entry is like $\langle *, d_u, d_v, d_w\rangle$.

In the above section, the dK-2 series is perturbed for privacy. Each unit of increment or decrement in dK-2 entries could be viewed as one edge adding or deleting. Then we do corresponding modification on the dK-3 series. Specifically, facing increasing or decreasing, there are three possible changes in dK-3 entries.

First is called replacement. If $\langle d_u, d_v\rangle$ decreased by one and $\langle d_u, d_w\rangle$ increased by one, the graph replaces the edge $e_{u,v}$ by $e_{u,w}$. So we pick $min(d_w, d_v) + d_u - 2$ dK-3 entries and use the number d_w to replace d_v in the dK-3 entries.

Second is subtracting. For each unit of decrement in $\langle d_u, d_v\rangle$, the graph deletes the edge $e_{u,v}$. So we reduce the dK-3 entries containing $\langle d_u, d_v\rangle$ by the total value of $d_u + d_v - 2$.

Third is adding, for each unit of increment in $\langle d_u, d_v\rangle$, the graph adds an edge $e_{u,v}$. The formation part is a little special because there is no original record of the neighbors of u or v. So we randomly pick a structure, wedge or triangle, and a degree number d_x in the range of $[1, d_{max}]$. Then we add the total value of $d_u + d_v - 2$ to the dK-3 entries containing $\langle d_u, d_v, d_x\rangle$.

Example: In the example of Figure 1, the dK-2 entry $\langle 4,4\rangle$ has total $4+4-2 = 6$ corresponding dK-3 entries $\langle \vee, 1, 4, 4\rangle$ and $\langle \triangledown, 2, 4, 4\rangle$ in the adjusted model. In Figure 3, the corresponding dK-3 series is constructed. Taken the dK-2 entry $\langle 1, 4\rangle$ as the example, because the dK-2 perturbation causes 1 unit of decrease, the corresponding dK-3 series has $1+4-2 = 3$ units of decrease. In the constructed dK-3 series, the first three modifications are '-3', '$+1$' and '-1' while the total amount of decrease is 3.

C. dK-1 RECOVERY

The dK-1 series is also important in the generation of the graph. Unlike the dK-3 series, it can be recovered directly from the dK-2 series. It is calculated by the following equation.

$$\langle d_v\rangle = \frac{\sum_{\text{dK-2 } entry}\langle d_u, d_v\rangle + \sum_{\text{dK-2 } entry}\langle d_v, d_u\rangle}{d_v} \tag{5}$$

The recovery process shows that the high dimension data, e.g., dK-2, contains all the information of the low dimension data, e.g. dK-1 .

Example: In the example of Figure 3, as the number '4' total appears $1+3+1*2+1 = 7$ times in the dK-2 series, the dK-1 series should be $\langle 4\rangle = 1.75 \approx 2$. Although the perturbed dK-2 values are integers, the recovery dK-1 values may not be integers. Here we can only round the value to integers because it shows the number of nodes and we have no information besides the dK-2 series. And the round-off error causes the two levels of dK series, dK-1 and dK-2, mismatch. In the rewiring section, we discuss the mismatch problem.

D. GRAPH REGENERATION

Given the target dK-2, dK-3 and dK-1 series, we need to regenerate the corresponding graph. Focusing on different level of dK series, we propose two sub-schemes namely CAT and LTH with different regeneration algorithms.

The LTH scheme starts from the dK-1 series because of the idea that dK-1 series is the base of the graph. If the degree of a node has a error, there will be large distortion on the corresponding dK-2 and dK-3 series. Hence, LTH just needs the dK-1 information and generate a graph with the least err_1. It leaves the task of mitigating err_2 and err_3 to the rewiring procedure.

By contrast, the CAT scheme considers the dK-2 and dK-3 series in regeneration because of the idea that rewiring cannot guarantee to achieve the lowest err_2 and err_3. It aims to reduce the most err_2 while preserving some dK-3 information as well.

In both schemes, we call a node 'saturated' if it has enough neighbors as its label (dK-1 information), and call it 'unsaturated' otherwise. If the value of a dK entry in the graph reaches the target value, we call it 'full'.

Algorithm 1 dK-1 Graph Regeneration (LTH)

dK-1

Input:

Output: $G_1(V_1, E_1)$: the perturbed graph

1: $V_1 \leftarrow$ dK-1 ▷ add nodes with degree labels
2: $\{d_1, d_2, \ldots, d_{|V|}\} \leftarrow$ dK-1
3: **for** $i = 1, i \leqslant |V|, i++$ **do**
4: pick a node u with degree d_i
5: **while** u is unsaturated **do**
6: **if** all nodes are connected with u **then break**
7: ▷ the dK-1 is non-graphical
8: pick v with the highest degree among all unsaturated nodes unconnected with u
9: E_1 adds edge $e_{u,v}$
10: **end while**
11: **end for**
12: **return** G_1

Algorithm 2 dK-2+ Graph Regeneration (CAT)

dK-1

Input: , dK-2, dK-3

Output: $G_1(V_1, E_1)$: the perturbed graph

1: $V_1 \leftarrow$ dK-1 ▷ add nodes with degree labels
2: dK-2$' \leftarrow 0$, dK-3$' \leftarrow 0$ ▷ initialize the dK-2 and dK-3
3: **while** exists dK-2 entry not full **do**
4: ——————beginning phase——————
5: randomly pick $\langle *, d_u, d_v, d_w\rangle$ not full in dK-3$'$
6: **if** $\langle d_u, d_v\rangle$ not full in dK2$'$ **then**
7: **if** exists u and v unconnected and unsaturated
8: **if** $* = \vee$, add edge $e_{u,v}$
9: **if** $* = \nabla$, add edge $e_{u,v}$, $e_{u,w}$
10: update dK-2 and dK-3 entries
11: **else if** exists u and v connected and unsaturated
12: ▷ adding edge causes multi-edges
13: NeighborSwitch(u, v)
14: **else** mark $\langle d_u, d_v\rangle$ full, **continue**
15: ▷ $\langle d_u, d_v\rangle$ cannot form an edge
16: **else continue**
17: **end if**
18: do Step 6-17, between v and w
19: ——————continuing phase——————
20: pick $\langle *, d_v, d_w, d_x\rangle$, do Step 6-17 between w and x
21: ...
22: **end while**
23: **return** G_1

Algorithm 3 NeighborSwitch(u,v)

1: find unsaturated node v' with degree d_v, $e_{u,v'} \notin E_1$
2: assume z is a neighbor of v', $e_{z,v'} \in E_1$ and $e_{z,v} \notin E_1$
3: E_1 removes edge $e_{z,v'}$, adds edge $e_{z,v}$ and $e_{u,v'}$
4: increase $\langle d_u, d_v\rangle$ in dK-2$'$

1) LTH

Algorithm 1 firstly sorts the degree sequence into a non-increasing order, which means $d_1 \geqslant d_2 \geqslant \ldots \geqslant d_{|V|}$. Each number in the sequence also represents the target degree value of a corresponding node. Then, beginning from the first node with degree d_1, the algorithm links it with d_1 nodes. These nodes are chosen from the node set which are unconnected with the first node, and they have the highest degree values in the set. According to [3], a graph can be reconstructed with the exact dK-1 information if and only if every node v is connected to all d_v nodes in the leftmost part of the degree sequence (having the highest degree values).

2) CAT

In each iteration, Algorithm 2 picks one dK-3 entry and try to add one edge to the graph if it can find two nodes, having corresponding degrees in the dK-3 entry, can pass the edge check. Here, the edge check means there are two unsaturated nodes with the correct degree, the two nodes are not connected, and the corresponding dK-2 entry is not full. When an edge is added in the graph, its corresponding dK-2 and dK-3 entries is updated. The regeneration process stops when there is no node pairs to pass the edge check. Also, in the edge check process, it may happen that the only pair of unsaturated nodes are already connected. Simply connecting them together forms multi-edges in the graph, which is forbidden in OSNs. Algorithm 3 switches one neighbor from a saturated node to an unsaturated node with the same label.

There are two phases for the Algorithm 2 to choose dK-3 entries and add edges. In the beginning phase, it randomly picks a dK-3 entry and add two or three edges into the graph correspondingly if the node pairs could pass the edge check. In the continuing phase, we use the last chosen dK-3 entry, denoted as $\langle *, d_u, d_v, d_w\rangle$, to find a new dK-3 series $\langle *, d_v, d_w, d_x\rangle$. Assuming the node w could pass the edge check with another node x, the algorithm links w which is used in the last step with the new node x. It stops if the newly picked node cannot pass the edge check with any other node, then it jumps to the beginning phase again.

Algorithm 2 make distinctions between wedges and triangles. It builds triangles if the three users link with each other originally, and forces no edge between u and w if the dK-3 entry is in the form of $\langle \vee, d_u, d_v, d_w\rangle$. The dK-3 information used in Algorithm 2 could preserve more structural information on the triangles and wedges, which is helpful to reconstruct a network with similar clustering information.

Example: Figure 4 and Figure 5 give the example of regenerated graph from the perturbed dK series in Figure 3. When the numbers on nodes represent the request degree, the LTH result satisfies the dK-1 series. However, it has no dK-2 information, e.g., $\langle 2, 4\rangle = 4$ in the graph but the desired value is 3. Compared with the given dK series in Figure 3, we have $err_1 = 0$, $err_2 = 4$, $err_3 = 18$.

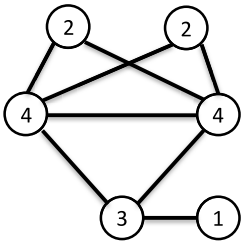

FIGURE 4. LTH results.

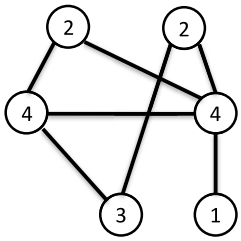

FIGURE 5. CAT result.

By contrast, the CAT result seems to satisfy the dK-2 requirement perfectly. However, one node with mark '3' and one with '4' have not get the required degree, and all the dK-2 series are exhausted. The err_1 happens because of mismatch, and has a impact on err_2 and err_3. Compared with the given dK series, we have $err_1 = 2$, $err_2 = 4$ and $err_3 = 13$. Comparing the two results of the example, each sub-scheme has its own advantage in preserving information.

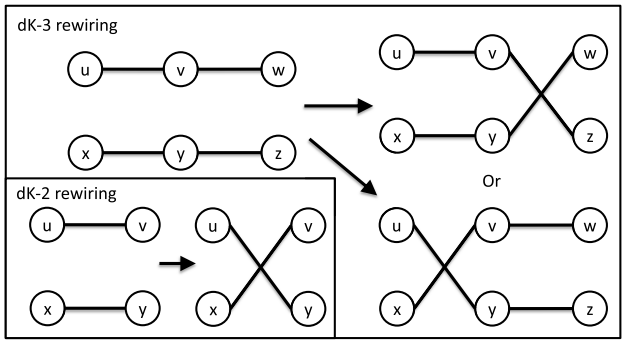

FIGURE 6. Examples of rewiring.

E. TARGET REWIRING

As mentioned in the last section, there is no dK-2 and dK-3 information preserved in the LTH intermediate graphs. LTH needs to compare the graph with the target dK-2 and dK-3 series and apply rewiring. Intuitively, the CAT intermediate graph only needs to apply the dK-3 rewiring because it does not consider the entire dK-3 entries. However, after analyzing the impact of noise in Section V-B, we find that the result which satisfies dK-2 series may have non-trivial error in dK-1 information. As a result, the CAT scheme needs dK-1, dK-2 and dK-3 rewiring, from lower to higher. Here we propose three levels of dK rewiring algorithms, each level of the rewiring preserves the lower dimensional information but may change the higher dimensional information.

Algorithm 4 Target Rewiring

dK-1
Input: , dK-2, dK-3, G_1
Output: $G_2(V_2, E_2)$: a new graph
G_2=G_1
————————dK-1 rewiring————————
u, v unsaturated and unconnected, E_2 adds edge $e_{u,v}$
u, v unsaturated and connected, NeighborSwitch(u, v)
u needs two or more edges, NeighborSwitch(u, u)
————————dK-2 rewiring————————
dK-2$'$ $\leftarrow$ G_2 ▷ count the dK-2 in dK-1 rewired graph
while there exists dK-2 rewiring pairs **do**
    E_2 removes $e_{u,v}$, $e_{x,y}$, adds $e_{u,y}$, $e_{v,x}$
end while
————————dK-3 rewiring————————
dK-3 ^{0} $\leftarrow$ G_2 ▷ count the dK-3 in dK-2 rewired graph
err_3^0 $\leftarrow$dK-3 ^{0} $-$ dK-3 ▷ store the initial error
$i = 0$ ▷ the step number
while there exists dK-3 rewiring pairs **do**
    E_2^{i+1} removes $e_{v,w}$, $e_{y,z}$, adds $e_{v,z}$, $e_{y,w}$
    get new dK-3 $^{i+1}$ and err_3^{i+1}
    if $err_3^{i+1} \geqslant err_3^i$, reject the rewiring, $G_2^{i+1} = G_2^i$
    $i = i + 1$
    do the rewiring check, Step 11-14, between $e_{u,v}$, $e_{x,y}$
end while
return G_2

1) dK-1 REWIRING

Given G_1 as the input, we build edges between pairs of unsaturated nodes. Building each edge reduce the err_1 by two. There are two special cases in the dK-1 rewiring shown in Algorithm 4. First, there are just two nodes unsaturated and they are already linked. A neighbor switch process should be applied on these two nodes. Second, there is just one node unsaturated, but it needs at least two edges. Then the neighbor switch should also be applied on this node. Here the neighbor switch processes in dK-1 rewiring is slightly different from the one in Algorithm. 3. It has no limitation on the degree of v', just needs v' and u to be unconnected.

2) dK-2 REWIRING

In this step, err_2 is reduced while keeping the result from the first step. Figure 6 shows the dK-2 rewiring process described in Algorithm 4. The dK-2 series in the intermediate graph is compared with the target dK-2. The algorithm applies the rewiring procedure if the prerequisites are satisfied. We define the dK-2 rewiring prerequisites as $\langle d_v, d_w \rangle$ and $\langle d_y, d_z \rangle$ are higher than the target, and $\langle d_v, d_z \rangle$ and $\langle d_y, d_w \rangle$

are lower than the target. When at least three out of four prerequisites are satisfied, we admit a rewiring pair to reduce the err_2 by at least two.

3) dK-3 REWIRING

err_3 is reduced with a similar procedure. Figure 6 shows two kinds of rewiring on the same six nodes. It is notable that the two different solutions lead to the same direct dK-3 changes, which is denoted as the direct impact of dK-3 rewiring. However, the rewiring process may also have indirect impact on dK-3, e.g., some entries involve node u and v are also changed. Hence, in each iteration, Algorithm 4 calculate the dK-3 series of the new graph called dK-3 ^{i} and find the dK-3 rewiring pairs. We admit a step of dK-3 rewiring only if the dK-3 error is decreased.

Numerically, in the example of Figure 6, the rewiring changes the dK-3 series directly but keeps the dK-2 unchanged if and only if $d_u \neq d_x$, $d_v = d_y$ and $d_w \neq d_z$. Hence, we define the dK-3 rewiring prerequisites as $\langle *, d_u, d_v, d_w\rangle$ and $\langle *, d_x, d_v, d_z\rangle$ are higher than the target, and $\langle *, d_u, d_v, d_z\rangle$ and $\langle *, d_x, d_v, d_w\rangle$ are lower than the target. We also admit the pair when at least three requirements are satisfied. Here the rewiring can directly reduce the err_3 by at least two. Structurally, the two types of dK-3 series have additional prerequisites on the existence of edges. For example, if the value of the entry $\langle \nabla, d_u, d_v, d_z\rangle$ are lower than the target value, there should be one edge between u and z before rewiring, then rewiring builds a triangle automatically.

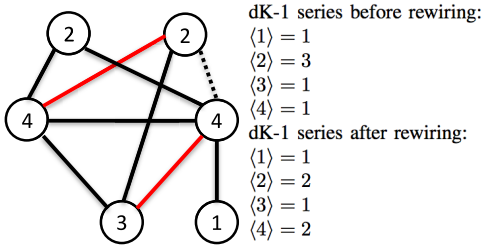

FIGURE 7. dK-1 rewiring.

Example: Figure 7 shows the example of dK-1 rewiring when the original graph is in Figure 5. When the original graph has two unsaturated nodes but the two nodes are linked, a neighbor switch process involving the right node with mark '2' can help all nodes satisfy the dK-1 series.

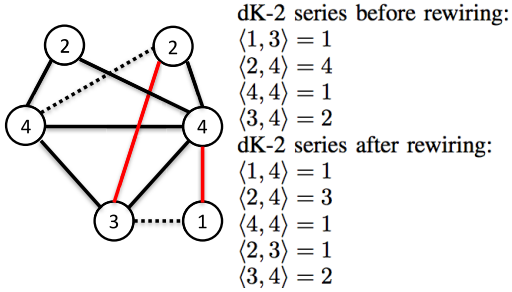

FIGURE 8. dK-2 rewiring.

Figure 8 shows the example of dK-2 rewiring when the original graph is in Figure 4. In the simple example, Figure 7 and Figure 8 are the same graph which shows that the CAT result after dK-1 rewiring can get the same graph as the LTH result after dK-2 rewiring. Both graphs have $err_2 = 1$, $err_3 = 2$, which shows that the rewiring algorithms can significantly reduce the error in dK series.

V. ANALYSIS

A. SENSITIVITY ANALYSIS

The sensitivity shows the impact of adding or deleting an edge in the dK-2 model.

Theorem 2: Given an entry $\langle d_x, d_y\rangle$ in the dK-2 model, the sensitivity $\triangle f$ is upper bounded by $2 \cdot d_x + 2 \cdot d_y + 1$.

Proof: Let $e_{x,y}$ be a new edge added to the graph G between nodes x and y. There is one new dK-2 series $\langle d_x, d_y\rangle$ getting increment by 1. Also, the degrees of x and y increase from d_x and d_y to d_x+1 and d_y+1 respectively. In the original dK-2 model, there are d_x series related with the node x. They are in the form of $\langle d_u, d_x\rangle$ and $\langle d_x, d_u\rangle$. They are deleted and new series $\langle d_u, d_x + 1\rangle$, $\langle d_x + 1, d_u\rangle$ are added. Hence, totally $2 \cdot d_x + 2 \cdot d_y + 1$ dK-2 series are changed when the new edge is added. □

Similarly, given the dK-3 series $\langle *, d_x, d_v, d_z\rangle$, the sensitivity of the adjusted model is $2 \cdot (d_y + d_z) \cdot d_{max} + (d_y + d_z)$, where d_{max} is the max degree value in the graph. And we can compare the noise amount that added on the dK-2 model with the one on the dK-3 model under the same privacy criteria.

Property 1: Under the same level of differential privacy, there is more noise added in the adjusted dK-3 perturbation than the dK-2 perturbation.

Proof: If we denote the total size of dK-2 model as |dK-2|, dK-2 is from $\langle 0, 0\rangle$ to $\langle d_{max}, d_{max}\rangle$, so $|\text{dK-2}| \leqslant d^2_{max}$. Then the expected randomization in the dK-2 model is

$$\begin{aligned}\sum_{|\text{dK-2}|} E[Lap(\frac{\triangle f(\text{dK-2})}{\epsilon})^2] \\ = |\text{dK-2}|Var(Lap(\frac{\triangle f(\text{dK-2})}{\epsilon})) \\ = \mathcal{O}(d^2_{max})\mathcal{O}(\frac{d^2_{max}}{\epsilon^2}) = \mathcal{O}(\frac{d^4_{max}}{\epsilon^2})\end{aligned} \tag{6}$$

where $\triangle f(\text{dK-2})$ is the sensitivity of dK-2 model.

Similarly, $|\text{dK-3}| \leqslant 2 \cdot d^3_{max}$. The expected randomization in the dK-3 model is

$$\begin{aligned}\sum_{|\text{dK-3}|} E[Lap(\frac{\triangle f(\text{dK-3})}{\epsilon})^2] \\ = |\text{dK-3}|Var(Lap(\frac{\triangle f(\text{dK-3})}{\epsilon})) \\ = \mathcal{O}(d^3_{max})\mathcal{O}(\frac{d^4_{max}}{\epsilon^2}) = \mathcal{O}(\frac{d^7_{max}}{\epsilon^2})\end{aligned} \tag{7}$$

Hence, under the same privacy level, the noise in the dK-2 model is much less than the noise in the dK-3 model. If we choose to add noise on the dK-3 level, the information distortion may exceed the benefits of using dK-3 model. □

B. PERFORMANCE ANALYSIS

In this section, we analyze the impact of noise on the dK graph models and then show the ability of our schemes to reduce dK error under noise.

If the dK series is graphical, which means it can build a graph, it must obey the following rules:

1) the values of dK entries being non-negative integers,
2) the dK-1 information, if in non-increasing degree sequence form, following the Erdös-Gallai theorem [5],

$$\sum_{i=0}^{j} d_i \leqslant j(j+1) + \sum_{i=j+1}^{|V|-1} min\{j+1, d_i\} \tag{8}$$

3) the dK-2 entries having $\langle d_x, d_y\rangle \leqslant \langle d_x\rangle \cdot \langle d_y\rangle$, $\langle d_x, d_y\rangle \leqslant d_x \cdot \langle d_x\rangle$, $\langle d_x, d_y\rangle \leqslant d_y \cdot \langle d_y\rangle$ if $d_x = d_y$, $\langle d_x, d_x\rangle \leqslant \langle d_x\rangle^2 - \langle d_x\rangle$
4) the dK-3 entries having $\langle *, d_x, d_y, d_z\rangle \leqslant \langle d_x, d_y\rangle \cdot \langle d_y, d_z\rangle$, $\langle *, d_x, d_y, d_z\rangle \leqslant d_y \cdot \langle d_x, d_y\rangle$, $\langle *, d_x, d_y, d_z\rangle \leqslant d_y \cdot \langle d_y, d_z\rangle$ if $d_x = d_z$, $\langle *, d_x, d_y, d_x\rangle \leqslant \langle d_x, d_y\rangle^2 - \langle d_x, d_y\rangle$

In most of the real cases, the perturbed dK-2 and dK-3 series are non-graphical, so we need to fix the dK-2 and dK-1 values to non-negative integers. However, the approximation makes the dK series have conflicts with each other. For instance, the degree value 3 appears 4 times in the dK-2 series, we can just make $\langle 3\rangle = \frac{4}{3} \approx 1$. Hence, we introduce the rewiring algorithms and apply the approximation graphically from lower to higher.

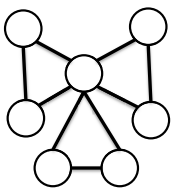

FIGURE 9. dK-1 rewiring target graph.

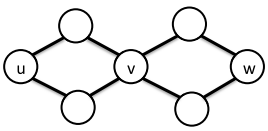

FIGURE 10. dK-1 rewiring regenerated result.

When considering the ability of reducing err_1, our LTH scheme has such property.

Property 2: The dK-1 regeneration result (LTH) has less err_1 than the dK-2+ regeneration result (CAT), although CAT has the dK-1 rewiring algorithm.

Proof: Because the dK-1 regeneration algorithm is directed by the Erdös-Gallai theorem [5], it always builds a graph when the dK-1 information is graphical. If the dK-1 information is not graphical, the dK-1 regeneration algorithm adds possible links to high degree nodes as much as possible, which does not form a forbidden link enlarging err_1 [3]. We also show the shortage of the CAT scheme in an example. Assuming the target dK-1 information is $\langle 6\rangle = 1$, $\langle 2\rangle = 6$, Figure 4 is a correct solution of dK-1 regeneration algorithm. Although Figure 5 violates the dK-2 series of the graph, it is still a possible intermediate graph published by the dK-2+ regeneration algorithm when noise is injected. Then the dK-1 rewiring algorithm tries to add neighbors to unsaturated node v. Node u and w are the possible candidates. The two nodes are saturated which means the dK-1 rewiring needs to apply the neighbor switch process. However, all neighbors of them are linked with v, which is the only unsaturated node in the graph. Hence, it is impossible to switch a neighbor of u (or w) to v. Finally, the dK-1 rewiring cannot get the correct graph while the LTH scheme can. □

As shown in the example, when the dK series do not match with each other, the ability of dK-1 rewiring algorithm is limited. Also, we analyze the errors in the dK-2 and dK-3 rewiring algorithms. Here we define the term of local minimum as there is no neighbor graph (with one edge changing) having lower error than the rewiring result.

Property 3: Given a graph G with fixed dK-1 series, the dK-2 rewiring algorithm can achieve the local minimum in err_2.

Proof: By contradiction, assuming there is a neighbor graph G' having lower err_2 than the rewired graph G with edge $e_{u,w}$ deleted. In order to preserve the dK-1 series, node u and w should each be linked with a new node, the two new nodes should also delete one existing edge. As a result, deleting $e_{u,w}$ makes $\langle d_u, d_w\rangle$ lower, $\langle d_u, d_z\rangle$ and $\langle d_y, d_w\rangle$ higher and at least one other change in the dK-2 entries. Because the edge changing violates the prerequisites of dK-2 rewiring, which means at least two of four prerequisites are not satisfied, the err_2 is not reduced by the 'illegal' edge changing. So the assumption must be wrong, and the rewiring result have the minimum err_2 among all its neighboring graphs. □

The dK-3 rewiring algorithm also has the similar property. If the indirect impact of dK-3 rewiring is ignored, it can also achieves the local minimum in err_3. The rewiring pairs reduce the error by two or four in dK-2 and dK-3 rewiring algorithms. Then, considering some particular pairs may trap the error in the local area. Deploying a weighted mapping algorithm can help us design the sequence of picking rewiring pairs and solve the problem at the cost of efficiency.

All three kinds of rewiring algorithms have possibility to be trapped in local area when searching global minimum, so it is significant to choose a start graph before rewiring. LTH starts from a graph with the best dK-1, the most basic information. CAT starts from a graph with some dK-3 information, which restricts the level of err_3. Our two schemes use two routes to deal with the noise and the conflict problem. Each of them has its own advantages in reducing the error.

VI. EVALUATION

A. EXPERIMENT SETTINGS

In this section, we evaluate our anonymization scheme over three real-world datasets, namely ca-HepTh, Facebook and

TABLE 1. Network dataset statistics.

Dataset	# of nodes	# of edges
ca-HepTh	9877	25998
Facebook	4039	88234
Enron	2977	7198

Enron [15]. The statistic of the three OSNs are shown in Table 1.

ϵ is a privacy parameter to measure the ability of hiding existence edges. Smaller ϵ means more strict privacy guarantee as well as more noise injected in the model. We generate ϵ-private graphs with $\epsilon \in [5, 100]$ to evaluate the performance under different privacy level. For comparison purposes, we implement one state-of-the-art technique as the reference method, which is the differential privacy algorithm with only the dK-2 model [22]. In the following figures, results of this scheme is marked as 'reference', while two of our sub-schemes are marked as 'CAT' and 'LTH', respectively.

To evaluate the three different anonymization schemes, we compare their published graphs under the same privacy level, i.e., the same ϵ. The similarity between the published graphs and the original graph is compared under four graph topological utility metrics: the average shortest path length, the clustering coefficient, the average degree and the betweenness centrality. We also evaluate the three levels of errors and one application utility metric, the influence maximization.

B. TOPOLOGICAL UTILITY METRICS

1) CLUSTERING COEFFICIENT

Clustering coefficient is a measure of how nodes in a graph tend to cluster together. While the dK-2 model may break the features of cluster, a scheme with the dK-3 series is believed to preserve partial clustering information because structural information like the triangles and wedges are included. Figure 11 shows the clustering coefficient distribution under different ϵ. Figure12 shows the distribution in different datasets. In the original ca-HepTh graph, there are 28% nodes with the median clustering coefficient (0.2 to 0.8). However, when $\epsilon = 20$, this kind of nodes only occupy 9% of total nodes in the reference result, 12% in the CAT graph and 13% in the LTH graph. Three dK anonymization methods all lose some clustering information.

The original ca-HepTh dataset has an average clustering coefficient of 0.47. When $\epsilon = 5$, the average clustering coefficient of the reference result is 0.21, and 0.24 for CAT and 0.26 for LTH. When $\epsilon = 100$, the average clustering coefficient is 0.12, 0.25 and 0.27 for the reference, CAT and LTH result correspondingly. The figures show that the clustering coefficient distribution of our two schemes are always closer to the original distribution than the reference result. The dK-3 series in our scheme preserves the structure

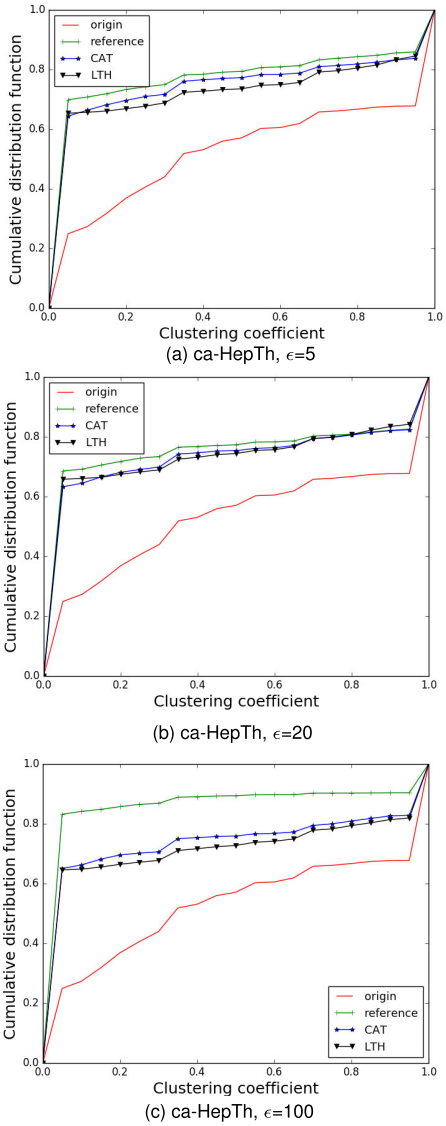

FIGURE 11. Clustering coefficient distribution under different ϵ.

information of triangles and wedges, which determines the clustering coefficient. Hence, the reference scheme shows more randomness while our schemes can preserve more clustering information.

In the Facebook dataset, the average clustering coefficient is 0.55, 0.17, 0.34 and 0.69 for the origin, reference, CAT and LTH data correspondingly. In the Enron dataset, it is 0.36, 0.13, 0.16 and 0.37. It shows that CAT and LTH consistently outperform the reference method in terms of clustering information preservation in all three cases.

2) AVERAGE SHORTEST PATH LENGTH

The average shortest path length measures the average length of the shortest path from one node to every other node. Figure 13 shows the average shortest length distribution in three datasets when $\epsilon = 20$. Taken the result of Enron dataset as example, the overall average shortest path length

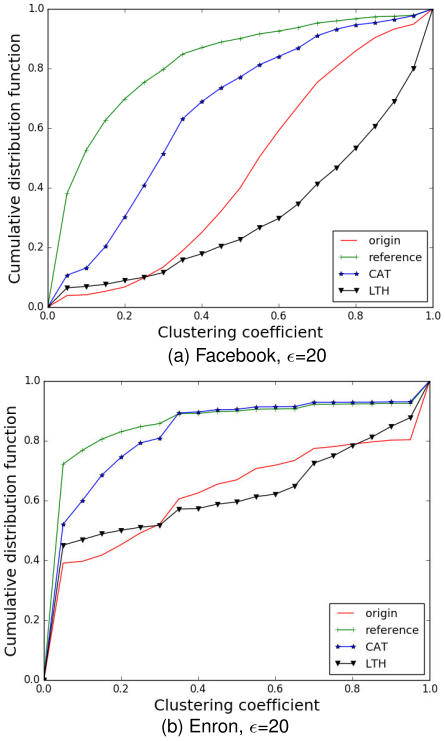

FIGURE 12. Clustering coefficient distribution in different datasets.

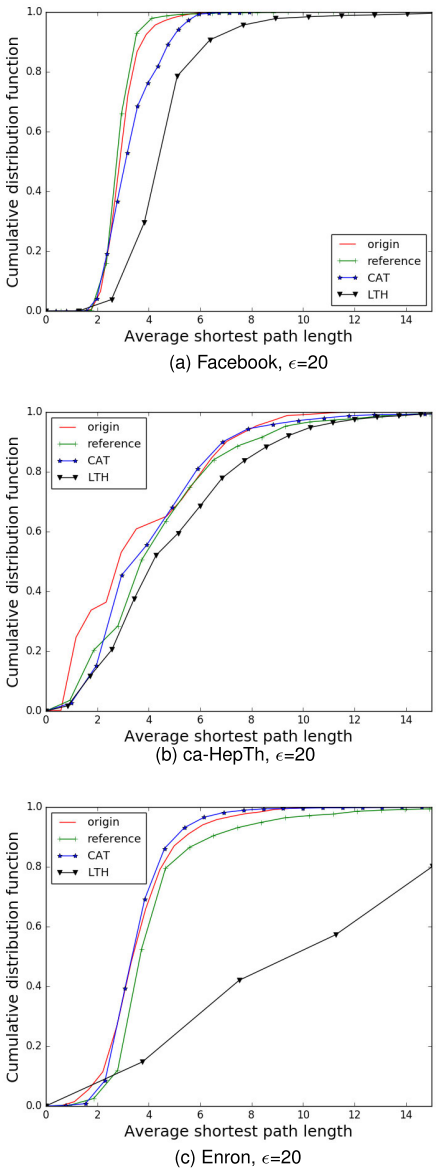

FIGURE 13. Shortest path length distribution.

of the original data is 3.61, and the reference, CAT and LTH schemes have the result 4.20, 3.54 and 11.35 correspondingly. The figure shows that the reference scheme and the CAT scheme can preserve the shortest path length information well in all three datasets. However, the nodes in the LTH anonymized graph have a longer distance between each other than the origin.

As shown in the analysis in Section V-B, the dK-2 rewiring algorithm cannot help LTH to minimize err_2. The LTH scheme just use the dK-1 information in graph regeneration, and the probability of high degree nodes linking with each other is much higher than the original. We can make an observation that the shortest path length is closely related with the dK-2 series.

3) BETWEENNESS CENTRALITY

Betweenness centrality measures the number of shortest paths that pass through each node. The betweenness centrality of the original graph and each released graph is shown in Figure 14. The experiment result shows that all three anonymization schemes can preserve some of the shortest paths from the original OSN. Our two sub-schemes outperform the reference scheme on some OSNs. For example, the average betweenness is 0.0041 for the original graph, 0.0014 for the reference result, 0.0019 for the CAT result, and 0.0030 for the LTH result on the Facebook dataset. The average betweenness is 0.0119 for the original graph, 0.0100 for the reference result, 0.0120 for the CAT result, and 0.0202 for the LTH result on the Enron dataset.

4) DEGREE

The degree of a node in the network is the number of edges incident to the node. Figure 15 shows the average degree in the ca-HepPh dataset. The published graph is more similar to the origin when noise level is lower in all three schemes. When $\epsilon = 5$, the average degree difference of the reference result is 0.37 while CAT has difference 0.24, LTH has difference 0.19. Compared with the reference method, the CAT scheme has an additional dK-1 rewiring algorithm which effectively reduces the err_1, so the CAT result is better than the reference result. The LTH scheme achieves the best result in reducing err_1. In the regeneration part,

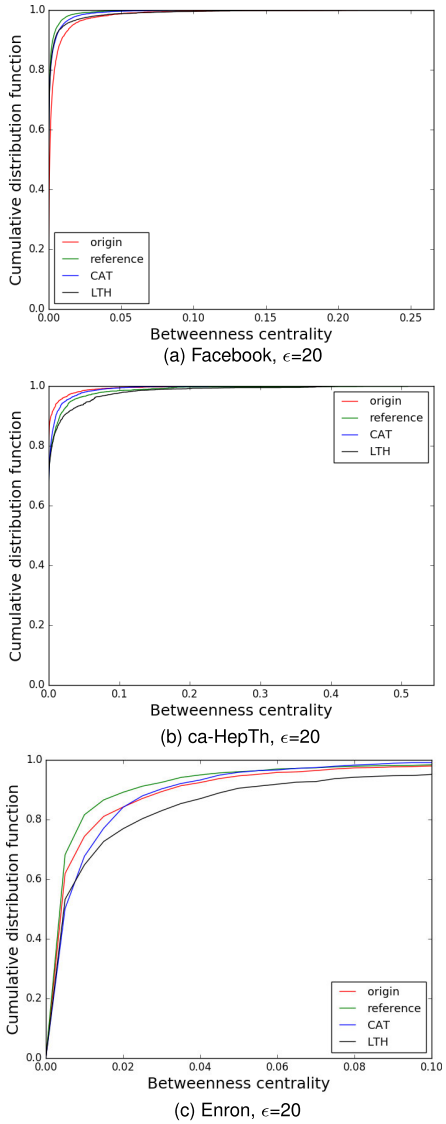

FIGURE 14. Betweenness centrality distribution.

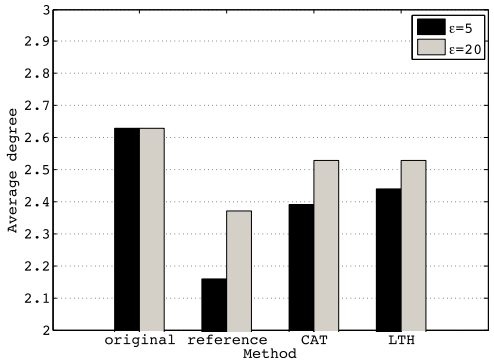

FIGURE 15. Average degree, ca-HepTh.

it just uses the dK-1 information while other schemes begin with the dK-2 series. Here we can make the observation that LTH has better performance than CAT even though CAT has the dK-1 rewiring algorithm, which is consistent with Property 3.

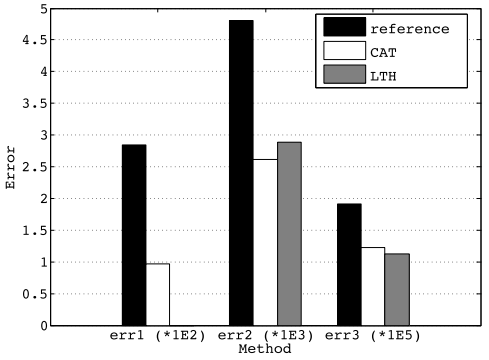

FIGURE 16. Error, Facebook.

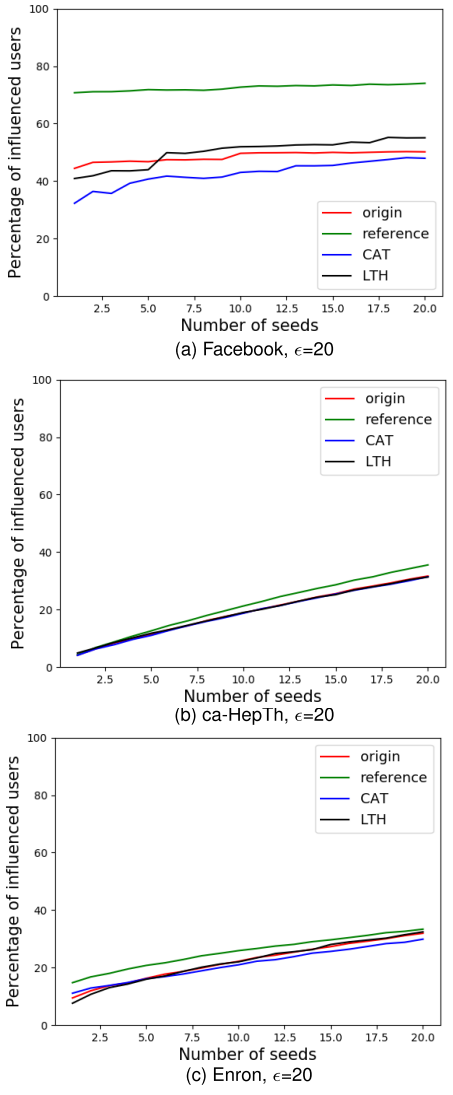

FIGURE 17. Percentage of influenced users in information maximization.

C. APPLICATION UTILITY METRIC

Information Maximization: Information maximization measures the proportion of users influenced by other users [12]. At the very beginning, we set a part of users as seeds, i.e., the persons who hold the information. Then we measure how

many users in the OSN are affected, i.e., received information from other users, after certain rounds. In the evaluation, we choose the greedy algorithm to choose the seed users and we implement the independent cascade model to propagate information [26]. The propagation probability is 0.02 in this experiment and this experiment terminates after 20 rounds.

The result is shown in Figure 17. Our two sub-schemes have more similar number of influenced users with the original graph than the reference result. For example, on the ca-HepPh network, when comparing with the percentages on the original graph, the Root Mean Square Error is 3.52 for the reference result, 1.41 for the CAT result and 0.63 for the LTH result. The information maximization evaluation shows that our two sub-schemes outperform the reference dK-2 scheme in simulating the information transmission result. The published graph of our anonymization schemes are more useful when analyzing the information propagation properties of OSNs.

D. dK SERIES ERRORS

We also compare the dK series in the original graph with different regeneration results. Figure 16 gives the three levels of error in the Facebook dataset. The reference result has 284 unit of err_1 while the CAT result has 96 and LTH result has no err_1. It shows that the dK-1 rewiring algorithm can reduce large amount of err_1, but using dK-1 in graph regeneration is better. Reducing err_1 also helps the CAT result, which has smaller err_2 (2.6K) than the reference result (4.8K). Because of the cumulative error in dK series and lacking a dK-3 rewiring algorithm, the reference result has 0.19M err_3 while CAT has 0.12M and LTH has 0.11M. The closer dK-3 distance between our results and the original graph is a reason that our schemes can preserve more structural information in the published graph.

E. EVALUATION SUMMARY

From the above experiments, we can conclude that the CAT and LTH schemes perform better in most measurements than the reference scheme. The LTH scheme can better preserve degree information but it lacks the ability to preserve average shortest path information. The CAT generally produce better result in all other graph metrics we evaluated. Using the comprehensive dK graph model helps us achieve better graph utility than the reference scheme which solely depends on the dK-2 model.

VII. CONCLUSION

In this paper, we propose a uniform scheme which combines three levels of dK graph models to publish a perturbed social network. We design two different sub-schemes, CAT and LTH, and three levels of rewiring algorithms to regenerate the graph and reduce the error under the differential privacy noise. The empirical study indicates that our two schemes have different merits in preserving graph utility. The design, analysis and comparison also reveal more insights and challenges of using multiple levels of graph abstraction models together in differential private graph releasing for OSNs.

REFERENCES

[1] 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.
[2] J. Cortes, G. E. Dullerud, S. Han, J. Le Ny, S. Mitra, and G. J. Pappas, "Differential privacy in control and network systems," in *Proc. IEEE 55th Conf. Decis. Control (CDC)*, Dec. 2016, pp. 4252–4272.
[3] C. I. Del Genio, H. Kim, Z. Toroczkai, and K. E. Bassler, "Efficient and exact sampling of simple graphs with given arbitrary degree sequence," *PLoS ONE*, vol. 5, no. 4, Apr. 2010, Art. no. e10012.
[4] C. Dwork, "Differential privacy: A survey of results," in *Proc. Int. Conf. Theory Appl. Models Comput.* Springer, 2008, pp. 1–19.
[5] P. E.-T. Gallai, "Graphs with prescribed degree of vertices (hungarian)," *Matematikai Lapok*, vol. 11, no. 1, pp. 264–274, 1960.
[6] T. Gao, "Privacy preserving in online social network data sharing and publication," Ph.D. dissertation, Dept. Elect. Comput. Eng., Purdue Univ. Graduate School, West Lafayette, IN, USA, 2019.
[7] T. Gao and F. Li, "Sharing social networks using a novel differentially private graph model," in *Proc. 16th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2019, pp. 1–4.
[8] T. Gao and F. Li, "PHDP: Preserving persistent homology in differentially private graph publications," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2019, pp. 1–9.
[9] T. Gao and F. Li, "De-anonymization of dynamic online social networks via persistent structures," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.
[10] 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, Dec. 2018.
[11] M. Gjoka, B. Tillman, and A. Markopoulou, "Construction of simple graphs with a target joint degree matrix and beyond," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2015, pp. 1553–1561.
[12] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Sep. 2019, pp. 2–25. [Online]. Available: https://openreview.net/forum?id=Bklr3j0cKX
[13] 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.
[14] S. Ji, W. Li, S. Yang, P. Mittal, and R. Beyah, "On the relative de-anonymizability of graph data: Quantification and evaluation," in *Proc. IEEE INFOCOM 35th Annu. IEEE Int. Conf. Comput. Commun.*, Apr. 2016, pp. 1–9.
[15] J. Leskovec and R. Sosič, "SNAP: A general-purpose network analysis and graph-mining library," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 1, p. 1, 2016.
[16] H. Li, Q. Chen, H. Zhu, D. Ma, H. Wen, and X. Shen, "Privacy leakage via de-anonymization and aggregation in heterogeneous social networks," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 2, pp. 350–362, Mar. 2020.
[17] P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat, "Systematic topology analysis and generation using degree correlations," in *Proc. Conf. Appl., Technol., Archit., Protocols Comput. Commun. SIGCOMM*, 2006, pp. 135–146.
[18] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *Proc. 48th Annu. IEEE Symp. Found. Comput. Sci. (FOCS)*, Oct. 2007, pp. 94–103.
[19] A. Narayanan and V. Shmatikov, "De-anonymizing social networks," in *Proc. 30th IEEE Symp. Secur. Privacy*, May 2009, pp. 173–187.
[20] J. Qian, X.-Y. Li, C. Zhang, and L. Chen, "De-anonymizing social networks and inferring private attributes using knowledge graphs," in *Proc. IEEE INFOCOM 35th Annu. IEEE Int. Conf. Comput. Commun.*, Apr. 2016, pp. 1–9.

[21] J. Qian, X.-Y. Li, T. Jung, Y. Fan, Y. Wang, and S. Tang, "Social network de-anonymization: More adversarial knowledge, more users re-identified?" *ACM Trans. Internet Technol.*, vol. 19, no. 3, pp. 1–22, Nov. 2019.
[22] 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.
[23] A. Speranzon and S. D. Bopardikar, "An algebraic topological perspective to privacy," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2016, pp. 2086–2091.
[24] J. Tang, A. Korolova, X. Bai, X. Wang, and X. Wang, "Privacy loss in Apple's implementation of differential privacy on MacOS 10.12," 2017, *arXiv:1709.02753*. [Online]. Available: http://arxiv.org/abs/1709.02753
[25] B. K. Tripathy, *De-Anonymization Techniques for Social Networks*. London, U.K.: Academic Press an imprint of Elsevier, 2019.
[26] J. M. Tylianakis, L. B. Martínez-García, S. J. Richardson, D. A. Peltzer, and I. A. Dickie, "Symmetric assembly and disassembly processes in an ecological network," *Ecol. Lett.*, vol. 21, no. 6, pp. 896–904, Jun. 2018.
[27] C. Zhang, S. Wu, H. Jiang, Y. Wang, J. Yu, and X. Cheng, "Attribute-enhanced de-anonymization of online social networks," in *Proc. Int. Conf. Comput. Data Social Netw.* Ho Chi Minh City, Vietnam: Springer, 2019, pp. 256–267.
[28] B. Zhou and J. Pei, "Preserving privacy in social networks against neighborhood attacks," in *Proc. IEEE 24th Int. Conf. Data Eng.*, Apr. 2008, pp. 506–515.
[29] L. Zou, L. Chen, and M. T. Özsu, "K-automorphism: A general framework for privacy preserving network publication," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 946–957, Aug. 2009.

TIANCHONG GAO (Member, IEEE) received the Ph.D. degree in computer engineering from Purdue University in December 2019. His Ph.D. Advisor was Dr. F. Li. He is a Lecturer with the School of Cyber Science and Engineering, Southeast University, China. He joined the School of Cyber Science and Engineering in April 2020. He has worked on problems on security, privacy, and social networks. He has published research articles in top-tier conferences and journals. His research vision is to explore the privacy issues in computing and networking.

FENG LI (Member, IEEE) received the Ph.D. degree in computer science from Florida Atlantic University in August 2009. His Ph.D. Advisor was Dr. J. Wu. He is an Associate Professor of Computer and Information Technology with Indiana University–Purdue University Indianapolis (IUPUI). He has published more than 50 papers in top conferences, including the INFOCOM and ICDCS. His research interests include the areas of cybersecurity and trust issues, cloud, and mobile computing.