# Structural-Based Graph Publishing Under Differential Privacy

Yiping Yin, Qing Liao, Yang Liu, and Ruifeng Xu(✉)

Harbin Institute of Technology (Shenzhen), Shenzhen, China
iron75951@gmail.com, {liaoqing,liu.yang,xuruifeng}@hit.edu.cn

**Abstract.** Mining data from social and communication network have been attracting recent attention from various research fields. However, these data represented by large-scale graphs are often sensitive and private. It is a necessity of developing algorithms to publish large-scale graph while not revealing sensitive information. As a standard for data privacy preservation, differential privacy based algorithm are also widely used in publishing graph-based dataset. However, previous differential privacy based methods often bring huge computational cost and lack the capability of modeling complicated graph structure. To address these challenge, we propose a novel graph publishing algorithm which combines community detection with differential privacy method. By segmenting the graph into several sub-graphs by community detection, differential privacy methods is able to handle large-scale graphs with complex structure. Experimental results on several datasets demonstrates the promising performance of the proposed algorithm compared with original differential privacy methods.

**Keywords:** Graph publishing · Partition · Differential privacy · Structural information

## 1 Introduction

Recent progress in information technology has led to impressive success in a wide range of applications, including recommendation system, medical services, tasks among Neutral Language Processing, etc. Such advances are enabled partly thanks to the availability of open large-scale graph based datasets. The graphs in such datasets are generally large and contain many sensitive information of users. Therefore, we need to explore methods that meet the demand of applications while offering principled and rigorous privacy guarantee.

Current methods in publishing private data could be partitioned into two aspects, including k-anonymity [1] and Differential Privacy [2]. The former is vulnerable to attackers with strong background knowledge, while the latter performs more robust for privacy preserving when facing strong attackers. Considering the specialization of abstract graphs, there are various of ways to convert them. The common procedure is to instantiate abstract graph as specific graph,


R. Xu et al. (Eds.): ICCC 2019, LNCS 11518, pp. 67–78, 2019.
https://doi.org/10.1007/978-3-030-23407-2_6

then to obtain the representation of graph, such as adjacency matrix [3], hierarchical random graph [4], quadtree [3,6], etc. These previous methods are often applicable in small-scale graph datasets. However, real graph are generally complex, which increases the difficulty on representation. Furthermore, the main limitation among the methods is that real graphs are generally large and sparse, leading to high computational expense.

In order to address these challenges, we propose a new algorithmic method for publishing graph dataset. To handle with the problem of high computational expense, we bring in a community detection algorithm to quickly partition the graph into several communities. To handle with the other problem of low utility, we explore a method by adding different degree of noises based on graph structural information. With the combination of Fast-unfolding community detection algorithm and the framework of differential privacy based on hierarchical representation, satisfactory performances are achieved on several large-scaled graph datasets.

## 2 Related Works

Differential privacy has been applied to a wide variety of scenes, including security of data publishing [8], machine learning [10], deep learning [9,10], etc. Data publishing is an essential step along the life circle of data. It can be divided into several branches by differing the type of objects, such as relational data, graph data, etc.

In order to publish graph data with privacy protection, we need to face the challenges on sanitizing graph with high-quality structural information. To address such problem, some proposed methods tried to add noises to edges within a graph, while others build an adjacency matrix first to store the information of a graph [3,5,11], or hierarchical structure [4], or use quadtree [6] to represent the graph, then do perturbation. However, most of the mentioned methods bring in large noises that threaten the utility of sanitized data. For instance, Wang et al. [11] propose to perturb the eigenvalues and eigenvectors of the corresponding adjacency matrix. It imposes noise of magnitude proportional to $\mathcal{O}\sqrt{n}$, where $n$ is the number of nodes in the network. It also causes high computation expense.

Some methods have tried to partition a graph into many subgraphs [7], and then do perturbation. However, it seems to be lack of strict privacy guarantee and some interpretation on why partition them in those way. Meanwhile, same noises are added to edges with no differences, which affects the utility of sanitized data.

A key limitation to most of the mentioned methods is that the connection strength hidden in the graph is ignored (i.e., all the edges are added to the same noise). Thus, extra noises are brought into the weak connections. Furthermore, the size of real networks is generally so large that causes large computational expense.

## 3 Preliminaries

In this section, we briefly recall the relevant definition of Differential Privacy, introduce both Laplace mechanism and Exponential mechanism, also composition theorems. Furthermore, reveal the logical relationship between hierarchical and community structure.

### 3.1 Differential Privacy

Differential Privacy [2] provides a strong standard with privacy guarantees for algorithms on datasets. It is defined in terms of the concept of adjacent database. In our research, each training dataset is a set of node pairs representing edges. Two sets are adjacent if they differ in a single entry, that is, if one node pair (edge) is present in one set and absent in another.

**Definition 1.** *A randomized function $\mathcal{M}$ gives $\varepsilon$-differential privacy if for all data sets d and d' differing on at most one element, and all $S \subseteq Range(\mathcal{M})$,*

$$P_r[\mathcal{M}(d) \in S] \leq \exp(\varepsilon) \times P_r\left[\mathcal{M}\left(d'\right) \in S\right]$$

A mechanism $\mathcal{M}$ satisfying this definition promises that even if the data of one participant is removed from the dataset, no outputs would become significantly more or less likely. $\varepsilon$ is called privacy budget.

### 3.2 Laplace Mechanism

Before calling Laplace mechanism, we need to recall the definition of Sensitivity first.

**Definition 2.** *The sensitivity of a function $f : D \rightarrow \mathbb{R}^k$, which is a numeric query function that maps datasets to k real numbers, is:*

$$\Delta f = \max_{d,d' \in D} \left|f(d) - f\left(d'\right)\right|, \|d - d'\| = 1$$

Where D is the dataset, $d$ and $d'$ are the arbitrary neighboring subsets belonging to D [14].

The sensitivity (global sensitivity) of a function $f$ captures the magnitude by which a single individual's data can change the function $f$ in the worst case. Therefore, it gives an upper bound on how much we must perturb its output to preserve privacy. Laplace mechanism will simply compute $f$, and perturb each coordinate with noise drawn from the Laplace distribution. The scale of the noise will be calibrated to the sensitivity of $f$ (divided by $\varepsilon$). Thus, given any function $f : D \rightarrow \mathbb{R}^k$, the Laplace mechanism is defined as:

$$\mathcal{M}(d, f(\cdot), \varepsilon) = f(d) + (Y_1, \ldots, Y_k)$$

Where $Y_i$ are *i.i.d.* random variables drawn from $\text{Lap}(\Delta f/\varepsilon)$. The Laplace mechanism preserves $\varepsilon$-differential privacy.

### 3.3 Exponential Mechanism

The exponential mechanism is the natural building block for answering queries with arbitrary utilities, especially non-numeric utilities.

**Definition 3.** *Given arbitrary range $\mathcal{R}$, the exponential mechanism is defined with respect to some utility function $u : \mathcal{D} \times \mathcal{R} \to \mathbb{R}$, which maps output to utility scores. Then the sensitivity of $u$ is:*

$$\Delta u \equiv \max_{r \in R} \max_{d,d':\|d-d'\|\leq 1} |u(d,r) - u(d',r)|$$

The exponential mechanism outputs each possible $r \in \mathcal{R}$ with probability proportional to $\exp\left(\frac{\varepsilon u(d,r)}{\Delta u}\right)$ and so the privacy loss is approximately:

$$\ln\left(\frac{\exp(\varepsilon u(d,r)/\Delta u)}{\exp\left(\varepsilon u\left(d',r\right)/\Delta u\right)}\right) = \varepsilon\left[u(d,r) - u\left(d',r\right)\right]/\Delta u \leq \varepsilon$$

An exponential mechanism is $\varepsilon$-differential privacy when it selects and outputs an $r \in \mathcal{R}$ with probability proportional to $\exp\left(\frac{\varepsilon u(d,r)}{2\Delta u}\right)$.

### 3.4 Composition Theorems

**Definition 4.** *Let $\mathcal{M}_1 : \mathcal{D} \to \mathcal{R}_1$ be an $\varepsilon$-differentially private algorithm, and let $\mathcal{M}_2 : \mathcal{D} \to \mathcal{R}_2$ be an $\varepsilon$-differentially private algorithm. Then their combination, defined to be $\mathcal{M}_{1,2} : \mathcal{D} \to \mathcal{R}_1 \times \mathcal{R}_2$ by the mapping: $\mathcal{M}_{1,2} = \mathcal{M}_1(d), \mathcal{M}_2(d)$ is $\varepsilon$-differentially private.*

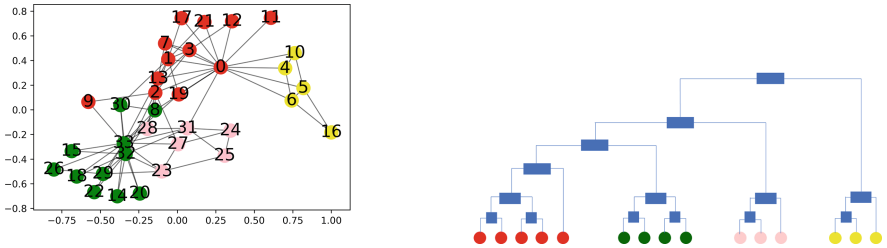
### 3.5 The Relationship Between Hierarchical and Community Structure

Hierarchical structure, which is represented by a binary tree, persistently partitions a network into a set of smaller communities until to the level of single node. The leaves of the hierarchical structure are nodes in the network. The instance drawn on karate club dataset is shown in Fig. 1. We can see that nodes in the same community are more likely to be divided into the same subtree [12].

## 4 Our Method

In this section, we describe four components of our method in turn: Fast-unfolding community detection algorithm, differentially private representation for hierarchical structure, differentially private noise addition, sanitized subgraphs generation. The framework is shown in Fig. 2.

Unlike the mentioned methods in Sect. 2, we explore the fast community detection algorithm to partition an entire graph into two parts, and then to build hierarchical representations of these two parts to make them satisfy Differential Privacy. Finally, we perturb the edges to generate an entire sanitized graph. By setting different privacy budget, we achieved different extend privacy preserving for edges located in different parts of a graph.



(a) Visualization of communities within karate club.

(b) Corresponding hierarchical structure.

**Fig. 1.** The abstract representation of communities within karate club partitioned by Fast-unfolding algorithm. Nodes in the same communities are of the same color, and the number marks id of a node.

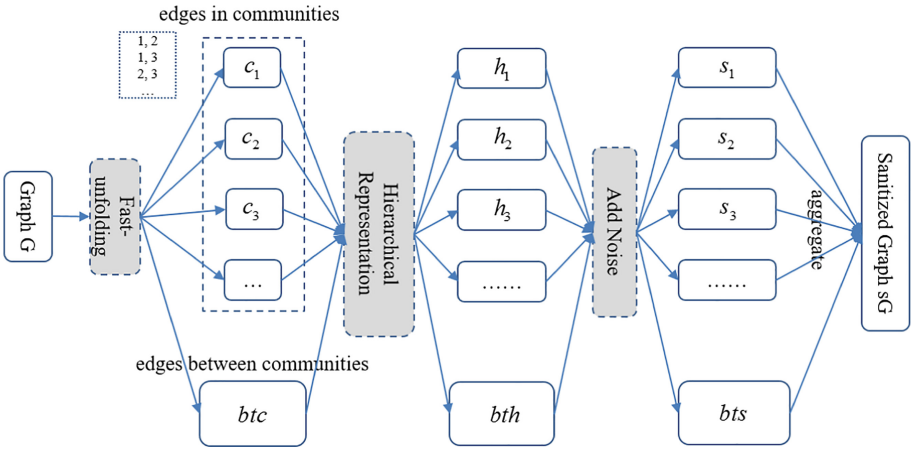



**Fig. 2.** The framework of our method.

### 4.1 Partition the Graph into Subgraphs by Using Fast-Unfolding Algorithm

Fast-Unfolding algorithm [13] is a heuristic community detection method based on modularity optimization, which extract the community structure of large networks. The main goal of the algorithm is to optimize the modularity of the entire network by continuously choosing the new nodes.

This algorithm is mainly divided into two phases that are repeated iteratively. One where modularity is optimized by allowing only local changed of communities; one where the found communities are aggregated in order to build a new network of communities. The phases are repeated iteratively until no increase of modularity is possible.

The reason why we choose the algorithm is that it performs high efficiency on large networks compared to other community detection algorithm [15].

Parts of the algorithm efficiency results from the fact that gain in modularity obtained by moving a node i into a community C can be easily computed by:

$$Q(G,s) = \frac{1}{4m} \sum_{i \in N} \sum_{j \in N} \left( A_{i,j} - \frac{k_i k_j}{2m} \right) s_i s_j$$

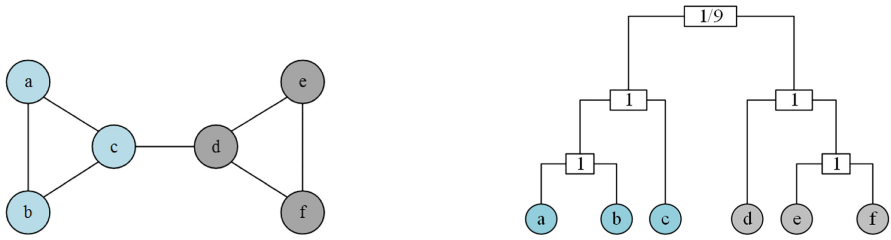
which unfolds a complete hierarchical community structure of a network.

### 4.2 Construct Differentially Private Representation for Hierarchical Structure

Intuitively, there is logical connection between the hierarchical structure and the results of partitioned communities.

A hierarchical structure shown in Fig. 1(b) can represent some structural information of network shown in Fig. 1(a). For instance, graph G consists of nodes and edges between the nodes. We use probability value to label the strength of links between nodes, which ranges from 0 to 1.

A specific form of the above expression is shown in Fig. 3. The strong links will be labeled a value close to 1, while weak links will be labeled a value close to 0.




(a) Graph $G$ (b) Hierarchical graph $h$

**Fig. 3.** The content of hierarchical graph $h$ is corresponding to graph G. For instance, the value of link probability between subset $\{a, b, c, d\}$ and $\{e, f\}$ is close to 0. Thus means that there is a weak connection between them. On the contrary, the value of link probability between subset $\{a, b\}$ and node c is 1, which implies the very strong connection between them.

The leaves in $h$ represents nodes in graph $G$, the internal nodes drawn by rectangle store the value of link probability $P_r$ between nodes or node set.

$$P_r = |e_r| / (|n_{L_r}| \cdot |n_{R_r}|)$$

where $r$ is corresponding to an internal node, $e_r$ is the number of links between left and right node subset that have a common ancient $r$ in $h$, $n_{L_r}$ and $n_{R_r}$ respectively are the number of nodes in the left and right node subsets.

We can construct a large set of $h$ from the different view of a graph. A criterion known as Maximum Likelihood Estimation is used to measure how plausible an $h$ is to represent $G$. The math equation is as follows.

$$L\left(\mathrm{h}, \{\mathrm{P_r}\}\right) = \prod\nolimits_{\mathrm{r}\in h} \mathrm{P_r^{e_r}} \left(1 - \mathrm{P_r}\right)^{n_{L_r} n_{R_r} - e_r}$$

where $h$ is the hierarchical graph, $\{P_r\}$ is the set of link probabilities stored in the internal nodes of $h$.

Intuitively, the logarithmic form of the equation can be converted as follows.

$$\log L = \sum\nolimits_{r\in h} n_{L_r} \cdot n_{R_r} \cdot h\left(\mathrm{P_r}\right)$$

where $\mathrm{h}\left(\mathrm{P_r}\right) = \mathrm{P_r} \log \mathrm{P_r} + \left(1 - \mathrm{P_r}\right) \log\left(1 - \mathrm{P_r}\right)$ is Shannon Entropy. Essentially, an $h$ with higher $logL$ is a better representation of the network structure than those with lower likelihoods.

Another problem occurs when we quantify "better $h$". There is an infinite space of $h$ for a network with $n$ nodes. Hence, directly computing the logL of each $h$ is unfeasible.

Xiao et al. proposed a method [4] that uses MCMC to sample the best $h$. A detail in her algorithm is to transition operation along the Markov chain, then to use Monte Carlo sampling method to pick some "better $h$". But the network is so large that constructing a set of $h$ leads to high computation expense.

A way to address this problem is to improve the computation efficiency within the algorithm. Changing nodes or subsets by first storing their paths from root to leaves into a list will help a lot. Furthermore, partitioning the entire graph $G$ into two parts (edges stored in communities and between communities) and making different extend differential privacy for them will save the time of computation for very large networks.

### 4.3 Differentially Private Noise Addition

Having constructed the hierarchical structure, a next issue is computing the probabilities stored in internal nodes with applying Laplace noise. The $\varepsilon$-differential privacy allows us to add noise that obeys Laplace distribution with scale $\Delta f/\varepsilon$ to each internal node. Each internal node requires to be updated the new connection probability in a top-down direction, we are one step away from generating a sanitized graph.

### 4.4 Sanitized Subgraphs Generation

Regenerate edges of each subgraph by referring to the connection probabilities. In the end, aggregate these subgraphs to an entire graph for publishing.

## 5 Experimental Results and Discussion

In this section, we evaluate the utility of sanitized graph over three various real networks, which are online social networks, voting networks and collaboration networks. By measuring a group of statistic properties of the original and sanitized graphs, we demonstrate the robustness and generalization of our method.

### 5.1 Datasets

To approximate the real and various networks in the world, we use several different types of networks in this section as shown in Table 1.

**Table 1.** Large network datasets description.

| Name | Type | #Nodes | #Edges | Description |
|---|---|---|---|---|
| ego-Facebook | Undirected, online social network | 4039 | 88234 | Social circles from Facebook |
| wiki-Vote | Directed, voting network | 7115 | 103689 | Wikipedia who-votes-on-whom network |
| ca-HepPh | Undirected, collaboration network | 12008 | 118521 | Collaboration network of Arxiv High Energy Physics |

ego-Facebook is an online social network dataset, where nodes represent users in Facebook, and edges represent the friendship between them. wiki-Vote is a voting network that contain Wikipedia voting information for adminship elections. ca-HepPh is a collaboration network that covers scientific collaborations between authors submitted to High Energy Physics. All of the datasets are available in Stanford Network Analysis Projects.[1]

### 5.2 Evaluation Criteria

A set of graph evaluation criterias of statistic information about degree and path are used to measure the similarity between original graph and sanitized graph. Essentially, we consider an undirected network when compute these statistic properties [16].

**Degree.** Considering an undirected network, the degree of a node is defined as the number of its neighboring edges.

[1] http://snap.stanford.edu/data/index.html.

*Average Degree (AD).* For an undirected network, the average degree is defined as follows.

$$AD = \frac{1}{n_{i=1,n}} \deg(v_i)$$

Where $n$ is the number of nodes in the network, $v_i$ is a node from the set of nodes $v$ of a network.

*Maximum Degree (MD).* The maximum degree of an undirected network is defined as follows.

$$MD = \max(\deg(v_i)), i = 1, \ldots, n$$

*Degree Variance (DV).* For an undirected network, degree variance is used to measure the dispersion of degree, which defined as follows.

$$DV = \frac{1}{n} \sum\nolimits_{i=1,\ldots,n} (v_i - AD)^2$$

*Clustering Coefficient (CC).* Let node $v_i$ has $k_i$ neighbors, then at most $k_i(k_i - 1)/2$ edges can exist between them. CC denotes the fraction of these allowable edges that actually exist.

*Degree Distribution (DD).* The degree distribution denotes the histogram of degree of all nodes.

**Path.** Average path length is one of the three most robust measures of network topology, along with its clustering coefficient and its degree information.

*Average Path Length (APL).* The average path length is defined as the average number of steps along the shortest paths for all possible pairs of network nodes. For an undirected network with a set of vertices V, the APL is defined as follows.

$$APL = \frac{1}{n(n-1)} \cdot \sum\nolimits_{i \neq j} d(v_i, v_j)$$

Where $n$ is the number of vertices, $d(v_i, v_j)$ denotes the shortest distance between $v_i$ and $v_j$, where $v_i, v_j \in v$.

*Diameter (D).* D is defined as the maximum distance among all possible pairs of network nodes.

$$D = \max(d(v_i, v_j))$$

*Effective Diameter (ED).* The effective diameter denotes 90% effective distance among all possible pairs of network nodes.

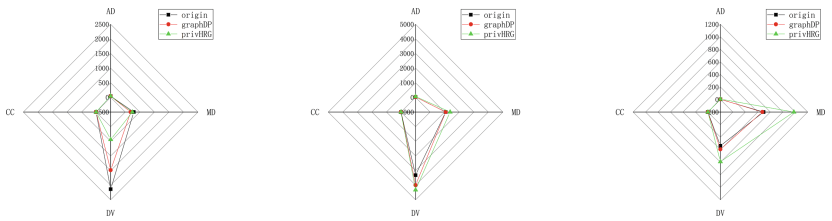
*Connectivity Length (CL).* The connectivity length is defined as the mean size of a smallest edge cut disconnecting $v_i$ from $v_j$, which is an important measure of its resilience as a network.

*Shortest Path Length Distribution (SPLD).* The shortest path length distribution de-notes the histogram of shortest path length among all pairs of nodes.

### 5.3 Experimental Results

This section reports on our evaluation on three various type of datasets: Facebook, wiki-Vote and ca-HepPh. The following results show the utility of sanitized graph compared to the original graph and privHRG [4].

**Statistics Information About Degree.** Figure 4 shows a group of statistic information of degree mentioned in Sect. 4.2. It can be seen that, in all cases, our method better preserves information of original networks on the properties of degree, meaning that it preserves good degree feature within networks.

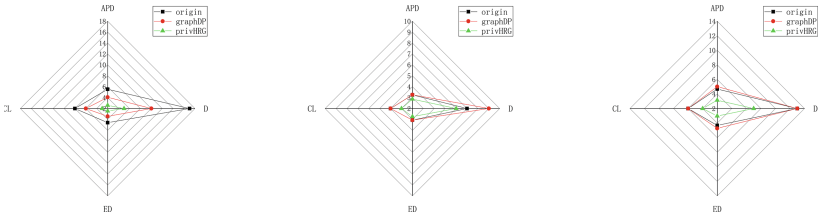



(a) egp-Facebook (b) wiki-Vote (c) ca-HepPh

**Fig. 4.** Results on statistics information about degree, including AD, CC, MD, DV mentioned in Sect. 4.1, on three real large network datasets. Each radar line in these charts corresponds to degree information of a network with different privacy protecting model (i.e. origin, graphDP, privHRG).

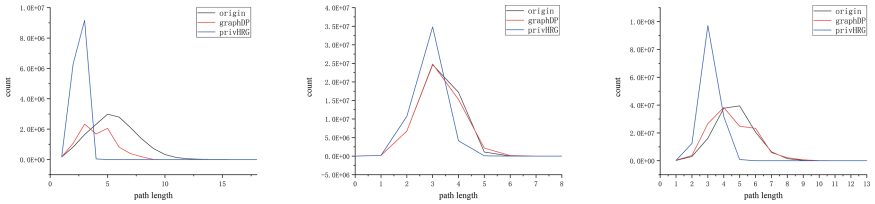
**Statistics Information About Path.** Figure 5 shows some statistic information of path mentioned in Sect. 4.2. It shows that, in all cases, our method preserves better skewness of the original networks than privHRG, meaning that it preserves good path feature within networks.

**Shortest Path Length Distribution.** Figure 6 depicts the shortest path length distribution of each network. We can observe that the sanitized networks preserve the shapes of the distributions with respect to those of the original networks than that of privHRG.




(a) ego-Facebook (b) wiki-Vote (c) ca-HepPh

**Fig. 5.** Results on statistics information about path, including APD, D, ED, CL mentioned in Sect. 4.1, on three real large network datasets.




(a) ego-Facebook (b) wiki-Vote (c) ca-HepPh

**Fig. 6.** Results on shortest path length distribution on three real large network datasets. Each broken line corresponds to a graph with different privacy protecting model.

### 5.4 Discussion

We demonstrate the large network publishing under differential privacy, sanitizing the entire graph with different degree of protection, achieved by bringing in a community detection algorithm. In our experiments on ego-Facebook, wiki-Vote and ca-HepPh, the results are observed to have better performance on a group of statistic properties of degree and path, which reflect the utility of a network.

It is important to acknowledge that lots of real networks are much larger than those of ego-Facebook or wiki-Vote, and may have much more communities; except taking advantage of fast community detection algorithm, bounding the privacy budget also plays an important parts on sanitizing the large networks. That's will be the immediate focus of our future work.

**Acknowledgement.** This work was partly supported by National Key Research and Development Program of China (2017YFB0802204), National Natural Science Foundation of China U1636103, 61632011, and 61876053, Key Technologies Research and Development Program of Shenzhen JSGG20170817140856618, Shenzhen Foundational Research Funding JCYJ20170307150024907.

## References


1. Sweeney, L.: k-anonymity. Int. J. Uncertainty Fuzziness Knowl. Based Syst. **10**(05), 557–570 (2008)
2. Dwork, C.: Differential privacy. Lect. Notes Comput. Sci. **26**(2), 1–12 (2006)
3. Chen, R., Fung, B.C.M., Yu, P.S., et al.: Correlated network data publication via differential privacy. VLDB J. **23**(4), 653–676 (2014)
4. Xiao, Q., Chen, R., Tan, K.L.: [ACM Press the 20th ACM SIGKDD international conference - New York, New York, USA (2014.08.24–2014.08.27)] Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining - Differentially private network data release via structural inference, pp. 911–920 (2014)
5. Ahmed, F., Jin, R., Liu, A.X.: A random matrix approach to differential privacy and structure preserved social network graph publishing. CoRRabs/1307.0475 (2013)
6. Cormode, G., Procopiuc, M., Shen, E., Srivastava, D., Yu, T.: Differentially private spatial decompositions. CoRRabs/1103.5170 (2011)
7. Bhagat, S., Cormode, G., Krishnamurthy, B., et al.: Class-based graph anonymization for social network data. Proc. VLDB Endow. **2**(1), 766–777 (2009)
8. Zhu, T., Li, G., Zhou, W., et al.: Differentially private data publishing and analysis: a survey. IEEE Trans. Knowl. Data Eng. **29**(8), 1619–1638 (2017)
9. Abadi, M., et al.: Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 308–318 (2016)
10. Song, S., Chaudhuri, K., Sarwate, A.D.: Stochastic gradient descent with differentially private updates. In: 2013 IEEE Global Conference on Signal and Information Processing (Glob-alSIP). IEEE (2013)
11. Wang, Y., Wu, X., Wu, L.: Differential privacy preserving spectral graph analysis. In: Proceedings of 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 329–340 (2013)
12. Perotti, J.I., Tessone, C.J., Caldarelli, G.: Hierarchical mutual information for the comparison of hierarchical community structures in complex networks. Phys. Rev. E **92**(6), 062825 (2015)
13. Blondel, V.D., Guillaume, J.L., Lambiotte, R., et al.: Fast unfolding of communities in large networks. J. Stat. Mech. Theory Exp. **2008**(10), 0 (2008)
14. Dwork, C., Roth, A.: The algorithmic foundations of differential privacy. Found. Trends Theor. Comput. Sci. **9**(3–4), 211–407 (2013)
15. Lancichinetti, A., Fortunato, S.: Community detection algorithms: a comparative analysis. Phys. Rev. E **80**(5), 056117 (2009)
16. Leskovec, J., Faloutsos, C.: Sampling from large graphs. In: ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2006)