Publishing Graph Degree Distribution with Node Differential Privacy

Wei-Yen Day
Dept. of Computer Science
Purdue University
West Lafayette, IN USA
wday@purdue.edu

Ninghui Li
Dept. of Computer Science
Purdue University
West Lafayette, IN USA
ninghui@cs.purdue.edu

Min Lyu*
School of Computer Science and Technology
University of Science and Technology of China
Hefei, China
lvmin05@ustc.edu.cn

ABSTRACT

Graph data publishing under node-differential privacy (node-DP) is challenging due to the huge sensitivity of queries. However, since a node in graph data oftentimes represents a person, node-DP is necessary to achieve personal data protection. In this paper, we investigate the problem of publishing the degree distribution of a graph under node-DP by exploring the projection approach to reduce the sensitivity. We propose two approaches based on aggregation and cumulative histogram to publish the degree distribution. The experiments demonstrate that our approaches greatly reduce the error of approximating the true degree distribution and have significant improvement over existing works. We also present the introspective analysis for understanding the factors of publishing the degree distribution with node-DP.

Categories and Subject Descriptors

K.4.1 [**COMPUTERS AND SOCIETY**]: Privacy

Keywords

Differential privacy; Private graph publishing; Degree distribution

1. INTRODUCTION

Many kinds of interesting graph data exist, e.g., social networks, email networks. However, directly publishing graph data may reveal sensitive information of an individual even if the graph is anonymized [2, 27]. Starting from [12], researchers have been studying the problem of publishing information of a graph under *Differential Privacy* (DP) [7, 8]. An algorithm that satisfies DP guarantees that the output distribution does not change significantly from one dataset to a *neighboring* dataset. When applying DP to graph data, two variants of DP were introduced [12]: in edge-DP, two graphs are neighboring if they differ on a single edge; in node-DP, two graphs are neighboring if by removing one node and all edges incident to the node in one graph, one obtains the other graph. Obviously satisfying node-DP is much harder than satisfying edge-DP, since removing one node may cause, in the worst case, the removal of $|V| - 1$ edges, where V is the set of all nodes. Because of this reason, most existing approaches that apply DP to graphs consider edge-DP, under the rationale that doing so protects relationship between two entities from being disclosed. However, the privacy offered by edge-DP is much weaker than node-DP, and, as we argue in Section 2.3, is insufficient in most settings.

In this paper, we investigate the problem of publishing the degree distribution of a graph under node-DP. Given a graph $G = (V, E)$, the goal is to release a node degree distribution (or, equivalently, a node degree histogram) that approximates the true distribution of G as much as possible while satisfying node-DP. A key technique to satisfy node-DP is that of "graph projection", which transforms a graph to be θ-degree-bounded, i.e., the maximum degree in the graph is no more than θ. A key challenge is to preserve as much information as possible in the projection process, while bounding the sensitivity of the degree histogram constructed from a projected graph. In [18], a graph is truncated so that all nodes with degree above θ are removed; this approach, however, removes significantly more edges than necessary. In [31], projected histograms are constructed using an auxiliary flow graph, and it was shown that this has a sensitivity of 6θ.

We improve the current state of the art on publishing graph node degree distributions under node-DP by introducing two new approaches that have much higher accuracy. Both approaches use a new graph projection method that is based on an edge-addition process. We show that this method, unlike previous projection methods, ensures that the projected graph is *maximal* in the sense that adding any additional edge will make the graph no longer θ-degree-bounded. We also prove that publishing the degree histogram after projection has a sensitivity of $2\theta + 1$, lower than previous approaches. When projecting a graph, a larger θ causes noise with larger magnitude to be added; at the same time, a lower θ means more edges are pruned. The optimal choice of θ thus depends both on the dataset and on the value of ϵ. Further, when publishing noisy histograms, it is sometimes better to merge several adjacent bins based on a configuration Ω and publish the average of the noisy bins [1, 23, 34, 36], reducing the errors caused by noise. We observe that the optimal choices of θ and Ω are mutually dependent, and propose the (θ, Ω)-Histogram approach, which selects the pair (θ, Ω) in one step using the exponential mechanism, with a low-

*The work was done while the author was visiting Purdue University, West Lafayette, IN USA.

SIGMOD'16, June 26-July 01, 2016, San Francisco, CA, USA

DOI: http://dx.doi.org/10.1145/2882903.2926745

sensitivity quality function that captures the resulting error due to the different factors, and then publishes a noisy histogram.

The second approach, which we call θ-CumulativeHistogram, is based on the observation that the edge-addition projection method will result in many nodes changing their degree by 1 between two neighboring datasets, and thus publishing a cumulative degree histogram after such a projection has sensitivity only $\theta + 1$. Furthermore, the monotonicity property of the cumulative degree histogram enables a calibrating step to further reduce the effect of noise. After obtaining a noisy cumulative histogram, one can then reconstruct the degree histogram. In both approaches, we also use an additional post-processing step to improve the accuracy of the histograms. We summarize the contributions of our paper in the following:

1. We have introduced an edge-addition based graph projection method, which preserves more information than previous approaches in the literature. We prove that publishing a degree histogram from the projected graph has sensitivity $2\theta + 1$, and publishing a cumulative degree histogram has sensitivity $\theta + 1$.

2. Based on the sensitivity bounds, we propose two approaches, (θ, Ω)-Histogram and θ-CumulativeHistogram, for publishing degree histograms under node-DP. We develop low-sensitivity quality functions that enable the selection of the parameters for these methods.

3. We have conducted extensive experiments using 8 real world datasets to compare our proposed methods with other existing methods. The experiments shown in Section 4 demonstrate that our proposed mechanisms have significant improvement over the state-of-the-art flow graph approach [31]. We also perform the introspective analysis to investigate the impacts of parameter selection, post-processing strategies for tail distribution, and edge ordering for projection. We demonstrate that our designed quality functions precisely evaluate the θ and Ω parameters with low sensitivities, which enable us to publish the degree distribution for a graph under node-DP.

The rest of this paper is organized as follows. We give the problem definition, discuss ϵ-DP and its application on graphs, and introduce existing approaches in Section 2. Section 3 introduces our proposed approaches. Experimental results are given in Section 4. We discuss related work in Section 5, and conclude in Section 6.

2. PRELIMINARIES

In this section, we introduce the problem definition, the notion of differential privacy and its application on graphs, and existing approaches.

2.1 Problem Definition

We consider undirected graphs with no additional labels on nodes and edges. Given an input graph $G = (V, E)$, we want to release a $|V|$-dimensional vector $\mathbf{d} \in \mathbb{R}^{|V|}$ that represents the probability distribution from degree 0 to $|V|-1$, i.e., $\mathbf{d}_i$ is the probability that a uniformly randomly chosen node in the graph has degree i. Since the degree distribution can be obtained from normalizing the degree histogram, the problem can be equivalently stated as publishing the degree histogram of G, which we use $\mathsf{hist}(G)$ to denote.

Utility Metrics. Following previous works [18, 31], we use the L1 distance between the published distribution and the true distribution (or L1 error) to evaluate different approaches. More formally, the L1 distance between any two distributions $\mathbf{d}$ and $\mathbf{d}'$ with length $|V|$ can be computed by $||\mathbf{d} - \mathbf{d}'||_1 = \sum_{i=0}^{|V|-1} |\mathbf{d}_i - \mathbf{d}'_i|$. Some techniques may publish a distribution with size smaller than $|V|$. We follow the same procedure in [31] to pad $\mathbf{d}$ with 0 if its size is less than $|V|$ for comparison.

In addition to the L1 error, we also use the Kolmogorov-Smirnov distance (KS-distance) between two degree distribution, which was used in [12]. Given two degree distributions $\mathbf{d}$ and $\mathbf{d}'$, the KS-distance between $\mathbf{d}$ and $\mathbf{d}'$ is used to test the closeness between them and is defined as: $KS(\mathbf{d}, \mathbf{d}') = \max_i |CDF_{\mathbf{d}}(i) - CDF_{\mathbf{d}'}(i)|$, where $CDF_{\mathbf{d}}(i)$ is the value of cumulative distribution function on degree i from distribution $\mathbf{d}$.

2.2 ϵ-Differential Privacy

The notion of ϵ-differential privacy (ϵ-DP) [7] is defined based on the concept of *neighboring databases*.

DEFINITION 1. *A randomized algorithm $\mathcal{A}$ satisfies ϵ-differential privacy (ϵ-DP) when for any two neighboring databases, D and D', denoted by $D \simeq D'$,*

$$Pr[\mathcal{A}(D) \in S] \leq \exp(\epsilon) \times Pr[\mathcal{A}(D') \in S]$$

where $S \subseteq Range(\mathcal{A})$ and ϵ is a parameter for privacy level.

Laplace Mechanism. One way to satisfy DP is to add noise to the output of a query. In the Laplace mechanism [8], in order to publish $f(D)$ where $f : \mathbb{D} \rightarrow \mathbb{R}^d$ while satisfying ϵ-DP, one publishes

$$\mathcal{A}(D) = f(D) + \mathsf{Lap}\left(\frac{\Delta \mathsf{f}}{\epsilon}\right)^{\mathsf{d}},$$

$$\text{where} \qquad \Delta f = \max_{D \simeq D'} ||f(D) - f(D')||_1$$

$$\text{and} \quad \mathsf{Pr}\left[\mathsf{Lap}\left(\beta\right) = x\right] = \frac{1}{2\beta} e^{-|x|/\beta}$$

Exponential Mechanism. While the Laplace mechanism provides a solution to handle numeric queries, it cannot be applied to non-numeric valued queries. This motivates the development of the exponential mechanism [26], which can be applied whether a function's output is numerical or not. Given a dataset D, and a quality function $q : (\mathbb{D} \times T) \rightarrow \mathbb{R}$, which assigns a quality score $q(D, t)$ for outputting t on input D, the exponential mechanism outputs t with a probability proportional to $\exp\left(\frac{\epsilon q(D,t)}{2\Delta q}\right)$, where $\Delta q = \max_{\forall t, D, D'} |q(D, t) - q(D', t)|$ is the sensitivity of the quality function. This satisfies ϵ-DP.

Composition Properties. Differential privacy satisfies sequential composition and transformation invariance [25, 20]. In short, sequential composition guarantees that for any sequence of computations $\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_k$, if each $\mathcal{A}_i$ satisfies ϵ_i-DP, then publishing the results of all of $\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_k$ satisfies $\sum_i \epsilon_i$-DP. The transformation invariance property says that given $\mathcal{A}_1$ that satisfies ϵ-DP, and any algorithm $\mathcal{A}_2$, the new algorithm $\mathcal{A}(\cdot) = \mathcal{A}_2(\mathcal{A}_1(\cdot))$ satisfies ϵ-DP. In other words, performing post-processing on the output of an algorithm that satisfies DP does not affect the privacy guarantee.

2.3 Node-DP or Edge-DP

When applying DP to graph data, there are two variants of DP [12]: in edge-DP, two graphs are neighboring if they differ on a

single edge; in node-DP, two graphs are neighboring if by removing one node and all edges incident to it in one graph, one obtains the other graph. They offer different kinds of privacy protection. As it has been extensively argued [7, 9, 21, 24], it is impossible to prevent an adversary with arbitrary background knowledge from learning sensitive information (e.g., by using correlation between data of different individuals) so long as one wants to preserve some utility of the data. What DP achieves is to approximate the effect of *opting out*. Suppose that one wants to publish $\mathcal{A}(D)$, where D consists of data of many individuals; and one individual, whose data is in D, complains that publishing $\mathcal{A}(D)$ violates her privacy. In this case, we can address the privacy concern by letting the individual "opting out", by removing the individual's data from D (or replacing her data with some arbitrary values) to obtain D' and publish $\mathcal{A}(D')$. Note that even after the removal of the individual's data, it may still be possible to learn information about the individual through correlation between this individual's data and others' data [7, 5, 10, 21]; however, such learning (while potentially harmful) is arguably not a privacy violation, since the learning can occur without accessing any personal data of the individual. For example, publishing study results that smoking increases one's chance of having lung cancer arguably leaks information about any smoker, even if the smoker does not participate in the study. However, one cannot say publishing the study results compromises the privacy of these people who are not even in the study. DP tries to approximate the effect of "opting out", by ensuring that any effect due to the inclusion of one's data is just a little more significant compared to the case in which her data is not included. We want to emphasize, however, that DP offers an approximation of "opting-out" **only when the difference of two neighboring datasets includes all the data one individual controls**.

We choose to use node-DP because we believe that edge-DP fails to provide adequate privacy protection. First, attacks on anonymized graph data are in the form of re-identifying nodes, illustrating that this is where the privacy concern lies [2, 27]. Second, edge-DP is often justified by the argument that one's goal is to protect the relationship between two entities encoded as an edge. Even if we accept that, edge-DP falls short of achieving that. DP offers effective protection only when the edges are independent from each other [21, 24]. However, it is hard to make the case that edges are independent. Conceivably, an edge between two high-degree nodes is much more likely than an edge between two low-degree nodes. Also, if two nodes share a lot of common neighboring nodes, then the probability that they are connected is higher too. Last but most importantly, while the above correlation argument can be applied to node-DP as well, node-DP can be justified because when using node-DP, one's goal is not to hide information of one node (which is unachievable), but only to approximate the effect of giving control of personal data to individuals. The same argument cannot be applied to justify edge-DP, since in most graph data, a node and its associated edges represent all personal data of an individual, and an edge does not represent all data controlled by one individual.

2.4 Existing Graph Projection Methods

The main challenge for satisfying node-DP is that the sensitivity is very high. Removing one node may cause, in the worst case, the removal of $|V|-1$ edges, where $|V|$ is the number of nodes in the graph. A natural approach is to apply the *projection* technique [3, 18, 31], which transforms the original graph G into a θ-bounded graph G^θ, such that the maximum degree in G^θ is less than or equal to θ; thus the sensitivity can be restricted. While the idea of projection is natural, how to use it most effectively turns out to be quite nontrivial. One challenge is that the projection operation may result in degree changes in a lot of nodes.

In [18], projection is done by truncation, i.e., removing all nodes with degrees larger than θ; and analyzing the sensitivity of publishing node-degree histograms after truncation is done in two steps. The first step is to compute S_T, the smooth bound on the number of nodes whose degrees may change because of the truncation. In order to keep S_T low, the cutoff threshold θ is randomized. The second step is to observe that changing one node in a θ-bounded graph can change the histogram by at most 2θ. The total sensitivity is the product of the two, i.e., $2\theta S_T$. Hence, it is suggested to add Cauchy noise with parameter $\frac{2\sqrt{2}\theta}{\epsilon}S_T$.

We observe two things. First, truncation removes more edges than necessary for the goal of ensuring a θ-bounded graph. One could keep up to θ edges for these nodes while ensuring θ-boundedness. Second, publishing a histogram after truncation has a global sensitivity of $2\theta+1$. Assume, wlog, that G_2 has one more node v^+ than G_1 such that removing v^+ and all its incident edges from G_2 results in G_1. If $deg(v^+) > \theta$, then G_1 and G_2 become the same after truncation. If $deg(v^+) \leq \theta$, then at most θ nodes other than v^+ will have different degrees in G_1 and G_2 (both before and after truncation), resulting in an L1 distance of no more than $2\theta+1$ between two resulting degree histograms from G_1^θ and G_2^θ.

Another projection method is proposed in [3], in which one chooses an arbitrary order among the edges, traverses the edges in this order, and removes each encountered edge that is connected to a node that has degree $> \theta$. We refer to this method as "Edge-Removal". In [3], the sensitivities for queries such as subgraph counting after projection are shown to be $p(2\theta)^{p-1}$ for node-DP, where p is the number of nodes in the subgraph. However, publishing a degree histogram was not considered in [3]. It is unclear what is the sensitivity of publishing a degree histogram after Edge-Removal.

In a recent work for publishing degree distributions under node-DP [31], another projection method was introduced. Given a graph $G=(V,E)$ and a degree bound θ, the method first constructs a weighted graph as follows. Starting with a source node s and a sink node t, for each node $v \in V$, it creates two nodes, v_ℓ and v_r, and adds edges (s, v_ℓ) and (v_r, t) with capacity θ. For every edge $e=(u,v) \in E$, it then adds a new edge (u_ℓ, v_r) with capacity 1. With the weighted graph, the algorithm computes the maximum flow f from s to t, while minimizing $\sum_{v\in V}\left((\theta - f(s,v_\ell))^2 + (\theta - f(v_r,t))^2\right)$, where $f(u,v)$ is the flow from u to v. Finally, the algorithm removes s from the maximum flow graph, and, for each node $v \in V$, obtains the degrees of v_ℓ, and constructs the degree distribution from these degrees. The analysis given in [31] shows that the sensitivity of releasing the sorted degree list can be bounded by 3θ, and the sensitivity of publishing a histogram after projection is 6θ.

3. PROPOSED APPROACHES

We describe our proposed approaches, starting with our proposed projection approach in Section 3.1. Sections 3.2 and 3.3 describe two ways of this projection to publish degree histograms, and Section 3.4 describes some optimizations that are applicable to both approaches.

3.1 Our Proposed Projection Method

We propose a new projection method, π_θ, which is shown in Algorithm 1. Our algorithm requires a stable ordering of all edges in an input graph G, denoted by $\Lambda(G)$. We say that a graph edge ordering Λ is **stable** if and only if given two neighboring graphs

Algorithm 1 π_θ: projection by edge-addition.

Input: An input graph $G = (V, E)$, a degree bound θ, and a stable edge ordering $\Lambda = \langle e_1, \cdots e_n \rangle$.
Output: An output θ-bounded graph $\pi_\theta(G)$.

```
E^θ ← ∅; d(v) ← 0 for each v ∈ V
for e = (u, v) ∈ Λ do
    if d(u) < θ & d(v) < θ then
        E^θ ← E^θ ∪ {e}; d(u) ← d(u) + 1; d(v) ← d(v) + 1
    end if
end for
return G^θ = (V, E^θ)
```

$G = (V, E)$ and $G' = (V', E')$ that differ by only a node, $\Lambda(G)$ and $\Lambda(G')$ are consistent, in the sense that if two edges appear both in G and G', their relative ordering are the same in $\Lambda(G)$ and $\Lambda(G')$.

Such stabling edge ordering can be easily obtained in practice. For example, if G is abstracted from a data source where each node v has a unique id $id(v)$ (e.g., the account id in social network data), and these node ids can be totally ordered by an ordering $\prec$ (e.g., $\prec$ can be alphabetical ordering), then each edge (u, v) can be represented as a pair $(id(u), id(v))$ if $id(u) \prec id(v)$ and $(id(v), id(u))$ otherwise, and the lexicographical ordering of edges is stable.

The projection π_θ first forms a graph with all nodes in G but no edges, and then keeps inserting edges from Λ following the ordering whenever inserting an edge $e = (u, v)$ will not cause the degree of either u or v larger than θ. This is similar to Edge-Removal, except that π_θ inserts edges while Edge-Removal deletes edges. While this difference appears minor, π_θ can preserve more information. For example, in Figure 1, using π_θ results in two edges remaining; using Edge-Removal, in contrast, results in only one edge remaining.

In fact, **the projection result obtained from π_θ is maximal** in the sense that no additional edge from G can be added without making some node's degree be above θ. To see this, observe that any edge that could be added in the end can also be added when it is encountered while traversing Λ. Therefore, in the end, no additional edge can be added, making the result maximal. However, π_θ does not guarantee that the resulting graph keeps the largest possible number of edges, as the result depends on the insertion order and may be only locally optimal. The example in Figure 1 shows that the Edge-Removal process cannot guarantees maximality. In Section 4.2, we experimentally demonstrate π_θ's advantage over Edge-Removal and other projection methods.

Lemma 1 below shows that Δ_{hist}, the global sensitivity of releasing $\mathsf{hist}(\pi_\theta(G))$, i.e., one first projects G by π_θ and then releases the projected graph's degree histogram, is bounded by $2\theta + 1$.

LEMMA 1. *For any $G \simeq G'$ that differ in one node, we have*

$$||\mathsf{hist}(\pi_\theta(G)) - \mathsf{hist}(\pi_\theta(G'))||_1 \leq 2\theta + 1.$$

PROOF OF LEMMA 1. Assume, without loss of generality, that $G' = (V', E')$ has an additional node v^+ compared to $G = (V, E)$, i.e., $V' = V \cup \{v^+\}$, $E' = E \cup E^+$, and E^+ is the set of all edges incident to v^+ in G'. Let Λ' be the stable orderings for constructing $\pi_\theta(G')$, and t be the number of edges added to $\pi_\theta(G')$ that are incident to v^+. Clearly, $t \leq \theta$. Let $e'_{\ell_1}, \ldots, e'_{\ell_t}$ denote these t edges in their order in Λ'. Let Λ_0 be the sequence obtained by removing from Λ' all edges incident to v^+, and Λ_k, for $1 \leq k \leq t$, be the sequence obtained by removing from Λ' all edges that both are incident to v^+ and come after e'_{ℓ_k} in Λ'. Let $\pi_\theta^{\Lambda_k}(G')$, for $0 \leq k \leq t$, be the graph reconstructed by trying to add edges in Λ_k one by one on nodes in G', and λ_k be the sequence of edges from Λ_k that are actually added in the process. Thus λ_k uniquely determines $\pi_\theta^{\Lambda_k}(G')$; we abuse the notation and use λ_k to also denote $\pi_\theta^{\Lambda_k}(G')$. We have $\lambda_0 = \pi_\theta(G)$, and $\lambda_t = \pi_\theta(G')$.

We now show that $\forall k$ such that $1 \leq k \leq t$, at most 1 node (other than v^+) in λ_k will have a degree that is different from λ_{k-1}, and the difference is 1. This proves the lemma because there are at most t nodes that have a different degree in λ_t when compared with λ_0. Each such node induces a difference of at most 2 in the histograms, and v^+ contributes another difference of 1.

To prove this, consider how the sequence λ_k may differ from λ_{k-1}. The first difference must be that λ_k includes $e'_{\ell_k} = (u_j, v^+)$, and λ_{k-1} does not. (Recall that Λ_k contains e'_{ℓ_k} but Λ_{k-1} does not by construction.) The decisions for all edges coming before e'_{ℓ_k} in Λ' must be the same in both λ_k and λ_{k-1}. After e'_{ℓ_k}, the edges in Λ_k and Λ_{k-1} are exactly the same; thus we can consider each different decision regarding a future edge as one more difference. Assume that there are a total of $s \geq 1$ different decisions.

When $s = 1$, only u_j (other than v^+) has a degree in λ_k that is different from λ_{k-1}, and the difference is by adding 1. When $s > 1$, the second difference must be regarding an edge incident to u_j and is because θ edges incident to u_j are added in λ_{k-1}, and the last one of these, denoted by $(u_j, u_{i\theta})$, cannot be added in λ_k (because u_j's degree has reached θ in λ_k by then, due to the extra edge (u_j, v^+)). In this case, u_j has the same degree (i.e., θ) in both λ_k and λ_{k-1}. If $s = 2$, then only $u_{i\theta}$ (other than v^+) has a different degree, and its degree in λ_k is 1 less than that in λ_{k-1}. If $s > 2$, then the third difference must be about an edge incident to $u_{i\theta}$ and is because the edge is included in λ_k but not included in λ_{k-1}; this can only happen when $u_{i\theta}$ has degree θ in both λ_k and λ_{k-1}, and results in another node's degree change by 1. Since s is finite, this sequence of reasoning will stop, with only the last encountered node has a degree change from λ_{k-1} to λ_k (all nodes earlier must have degree θ in both λ_{k-1} and λ_k). □

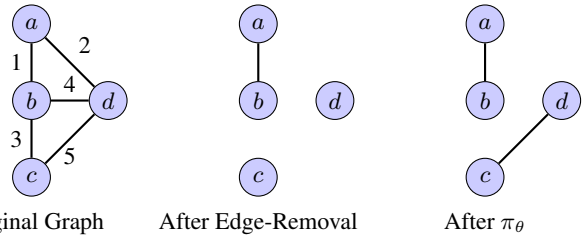


Figure 1: Comparison of Edge-Removal [3] and π_θ ($\theta = 1$). Edge order for removal in Edge-Removal: 5, 4, 3, 2, 1. Edge order for insertion in π_θ: 1, 2, 3, 4, 5.

3.2 Parameter Selection and Aggregation

In projection, the optimal choice of θ depends on both the distribution of the data and the privacy parameter ϵ. A larger θ means that more edges can be preserved in the projection process; however, larger noise needs to be added to the histogram. If most nodes in the graph have small degrees, then a smaller θ is preferred. At the same time, if ϵ is larger, then a larger θ is preferred. In previous work [3, 18], performing the projection step requires the prior knowledge of the projection level θ.

Furthermore, one way to improve the accuracy of publishing his-

tograms after projection is to apply the idea of *aggregation* on histogram bins, which was investigated in the literature [1, 23, 34, 36]. The intuition is to find a structure Ω, in which some bins are grouped together, reducing the noise effect by averaging, with the cost of using an average to estimate each bin within a group. More specifically, $\Omega = \{\mathbf{g}_1, \mathbf{g}_2, \cdots \mathbf{g}_n\}$ is a set of disjoint groups of bins representing the aggregation strategy, where each $\mathbf{g}_k = (k_1, k_2, \cdots, k_{|\mathbf{g}_k|})$ corresponds to the bins of k_1'th, k_2'th, $\cdots$ $k_{|\mathbf{g}_k|}$'th degree in histogram. We use aggregation after the projection with a parameter θ, and thus have $\sum_{\mathbf{g}_k \in \Omega} |\mathbf{g}_k| = \theta$. The optimal aggregating structure Ω and projection level θ are related to each other.

We propose to estimate the optimal aggregation structure Ω and projection level θ simultaneously. To do so, we need to decide what is the candidate pool of (θ_i, Ω_j) pairs. The size of this pool should not be too large, as the running time is linear in the pool size and the quality of selection could deteriorate when the pool includes many poor choices. For the set of all choices for θ_i, we require as input Θ, and consider all integers from 1 to Θ as choices for θ_i. We use $\Theta = 200$ in the experiments. For $\mathbf{\Omega}$, the candidate pool of Ω, adopting a naive approach that considers all combinations results in a very large pool. Let K be the maximum number of groups after aggregation. The number of candidates is approximately on the order of Θ^K, even if we group only adjacent bins. To solve this problem, we exploit the long-tail property of the node degree distributions, and conjecture that a good candidate should group fewer bins on low degrees (usually with larger counts), and more bins on high degrees (long tail part). We then use geometric series, $r^0, r^1, r^2, \cdots$, to determine the size of each group in Ω, such that for each r, $\mathbf{g}_i = \{k : r^{i-1} \leq k < r^i\}$. That is, when $r = 1.5$, $\Omega = \{(1), (2), (3), (4, 5), (6, 7), (8, 9, 10, 11), \cdots\}$. We vary r such that $r \in [1, 2]$ to generate a good coverage of optimal Ω.

We also need to design a quality function to evaluate the desirability of any (θ_i, Ω_j) pair. Our quality function is similar to that used in [23], and has two components. The first component captures the errors introduced in the process of creating the noisy histogram, computed as follows:

$$\ell_{hist}(G, \theta_i, \Omega_j) = \sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} \left(\left| c_d - \frac{\sum_{d \in \mathbf{g}_k} c_d}{|\mathbf{g}_k|} \right| + \frac{2\theta_i + 1}{\epsilon |\mathbf{g}_k|} \right),$$

where c_d denotes the number of nodes with degree d in $\mathsf{hist}(G^{\theta_i})$. The above sums over the error for all bins. For each bin, the first term captures the error due to the aggregation step and the second term captures the error due to added noises.

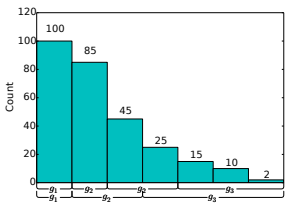
The second component captures the *projection loss*. Intuitively, we want to capture the number of nodes that have a degree above θ and will have degree changes after projection. However, directly using $|\{v | v \in V, deg(v) > \theta_i\}|$ has a high sensitivity; we thus choose to first project the graph G to a maximum degree of Θ, and then compute how many nodes have a degree above θ in the projected graph, i.e.,

$$\ell_{proj}(G, \theta_i) = 2 \cdot \left| \{v | v \in V, deg_{\pi_\Theta(G)}(v) > \theta_i\} \right|,$$

where $deg_{\pi_\Theta(G)}(v)$ gives v's degree in the graph $\pi_\Theta(G)$. This is two times the number of nodes with degree larger than θ_i in $\pi_\Theta(G)$. The intuition is that each of such nodes will have a degree change and result in a change of 2 in the degree histograms before and after projection.

Combining the two components, our proposed quality function is

$$q_\epsilon^h(G, \theta_i, \Omega_j) = -\ell_{hist}(G, \theta_i, \Omega_j) - \ell_{proj}(G, \theta_i). \quad (1)$$



ϵ		Ω_1	Ω_2
$\epsilon = 0.2$	ℓ_{hist}	334	293
	ℓ_{proj}	4	4
	q_ϵ^h	-338	**-297**
$\epsilon = 1$	ℓ_{hist}	94	143
	ℓ_{proj}	4	4
	q_ϵ^h	**-98**	-147

Figure 2: An example of different structures. $\Omega_1 = \{(1), (2), (3, 4), (5, 6, 7)\}$, $\Omega_2 = \{(1), (2, 3), (4, 5, 6, 7)\}$.

A running example to demonstrate the advantage of this quality function is given in Figure 2. Consider two structures Ω_1, Ω_2, where Ω_1 has 4 groups and Ω_2 has 3 groups. Assuming that $\theta = 7$ and only 2 nodes have degree over θ, the projection loss is $\ell_{proj}(G, \theta) = 4$. When ϵ is small, i.e. $\epsilon = 0.2$, the quality function estimates that Ω_2, the coarser aggregation, has more advantage. When ϵ is 1, the quality function prefers Ω_1, the finer aggregation.

To apply the exponential mechanism to select a (θ, Ω) pair, we need to have a bound on the global sensitivity of q_ϵ^h given in Eq. (1); the following lemma establishes this to be $6\Theta + 4$.

LEMMA 2. *For any neighboring G and G', we have*

$$|q_\epsilon^h(G, \theta_i, \Omega_j) - q_\epsilon^h(G', \theta_i, \Omega_j))| \leq 6\Theta + 4$$

The proof is given in Appendix A.

Note that while our method avoids requiring θ as an input, it still requires Θ as an input. The difference is that the selection of Θ is much less sensitive and data-dependent than that of θ. In our experiments, we use $\Theta = 200$ for all datasets, and our method is then able to select different θ values that perform well for different datasets. We show that when using the same θ value for all datasets, invariably the method will perform poorly on some datasets. We also demonstrate that setting $\Theta = 400$ results in only slightly larger errors and is still able to significantly outperform existing methods.

We use (θ, Ω)-Histogram to denote the algorithm of releasing degree histogram from projection and aggregation, which is given in Algorithm 2. (θ, Ω)-Histogram first computes the quality of each candidate (θ_i, Ω_j) and performs the selection process through exponential mechanism. This step, for each θ_i, computes a projection of the graph from Algorithm 1, constructs a histogram, and then computes the quality for each Ω_j, resulting a $O(\Theta \cdot (|E| + |V| + \Theta \cdot |\mathbf{\Omega}|))$ running time. The second step is to perform projection using the selected (θ^*, Ω^*) and to publish the aggregated noisy histogram, which takes time $O(|E| + |V|)$. The final step for post-processing will be introduced in Section 3.4; it takes time at most $O(|V|)$. Thus the total time complexity is dominated by the step of computing quality functions, and is $O(\Theta \cdot |E|)$ for most practical cases.

Finally, we give the privacy guarantee of (θ, Ω)-Histogram as well as the proof in the following.

LEMMA 3. (θ, Ω)-Histogram *in Algorithm 2 satisfies* $(\epsilon_1 + \epsilon_2)$*-node-DP.*

PROOF OF LEMMA 3. In Algorithm 2, the steps of Lines 1-2 use exponential mechanism with privacy budget ϵ_1. The steps of projection (Line 3) and publishing histogram with aggregated counts (Line 4) use Laplace mechanism and satisfy node-DP for ϵ_2, and the post-processing step performs on the private output. By

Algorithm 2 (θ, Ω)-Histogram algorithm.

Input: A graph G, privacy budgets ϵ_1 and ϵ_2, candidates T and O
Output: A noisy degree distribution $\mathbf{d}$.

1: For each $(\theta_i, \Omega_j) \in T \times O$, computes $q_\epsilon^h(G, \theta_i, \Omega_j, \epsilon_2)$
2: Select (θ^*, Ω^*) with prob. $\propto \exp\left(\frac{\epsilon_1 q_\epsilon^h(G,\theta_i,\Omega_j,\epsilon_2)}{2\Delta_{q_\epsilon^h}}\right)$
3: $G^{\theta^*} \leftarrow \pi_{\theta^*}(G)$ by Algorithm 1.
4: $h \leftarrow \mathsf{Aggregate}_{\Omega^*}\left(\mathsf{hist}\left(G^{\theta^*}\right)\right) + \mathsf{Lap}\left(\frac{\Delta_{\mathsf{hist}}}{\epsilon_2}\right)^{|\Omega^*|}$
5: $h' \leftarrow \mathsf{PostProcess}(h)$ by Algorithm 5.
6: **return** $\mathbf{d} = h'/||h'||_1$

the composition theorem and transformation invariance properties, (θ, Ω)-Histogram satisfies $(\epsilon_1 + \epsilon_2)$-node-DP. □

3.3 Cumulative Degree Histogram

Releasing the degree histogram from projection and aggregation technique requires noise level of $\mathsf{Lap}\left(\frac{2\theta+1}{\epsilon}\right)$. One way to lower the noise magnitude is to publish a *cumulative degree histogram*, in which the k'th bin in the histogram is the answer of the query $\mathsf{cumhist}_k = |v : deg(v) \leq k|$, i.e., *"How many nodes in this graph have degree no higher than k?"*. Lemma 4 below shows that $\Delta_{\mathsf{cumhist}}$, the global sensitivity of releasing a cumulative degree histogram, i.e., the vector $\mathsf{cumhist}(\pi_\theta(G))$ where $1 \leq k \leq \theta$, is bounded by $\theta + 1$.

LEMMA 4. *Given $G \simeq G'$ that differ in one node, we have*

$$||\mathsf{cumhist}(\pi_\theta(G)) - \mathsf{cumhist}(\pi_\theta(G'))||_1 \leq \theta + 1$$

where π_θ is the projection function following Algorithm 1.

PROOF OF LEMMA 4. We follow the idea and notations in the proof of Lemma 1. Let t be the degree of v^+ in $\pi_\theta(G')$. Now consider the effect of v^+ in the cumulative histogram. Since v^+ appears in $\mathsf{cumhist}(\pi_\theta(G'))$ but not in $\mathsf{cumhist}(\pi_\theta(G))$, all $\theta - t + 1$ bins of $\mathsf{cumhist}(\pi_\theta(G'))$ from degree t to θ are increased by 1 compared to those in $\mathsf{cumhist}(\pi_\theta(G))$. Thus, there are at most $\theta - t + 1$ changes due to v^+.

Now consider other changes caused by adding edges incident to v^+. Recall that we use $\lambda_0, \lambda_1, \cdots, \lambda_t$ to denote the sequence of projected graphs resulted from keeping $0, 1, \cdots, t$ of such edges. We have shown that $\forall k$ such that $1 \leq k \leq t$, there is at most 1 node (other than v^+) in λ_k with the degree differs from λ_{k-1}, and the difference in the node's degree is 1. Thus, the cumulative histograms of λ_{k-1} and λ_k (ignoring v^+) have an L1 distance of at most 1. By the Triangular Inequality, the cumulative histograms of λ_0 and λ_t (ignoring v^+) have a distance of at most t. Combining this with the effect of v^+, the cumulative histograms of $\pi_\theta(G) = \lambda_0$ and $\pi_\theta(G') = \lambda_t$ have a distance of at most $\theta + 1$. □

Modeling the selection of the optimal θ requires a low-sensitivity quality function with the ability of estimating the error of cumulative histogram. Considering the property of a cumulative histogram, we point out, as θ is the cutoff point of the cumulative histogram in $\pi_\theta(G)$, the utility that $\mathsf{cumhist}(\pi_\theta(G))$ maintains is "how many nodes with degree equal to or under θ". We propose a quality function as follows:

$$q_\epsilon^c(G, \theta_i) = -2 \cdot \left|\{v : v \in V, deg_{\pi_\Theta(G)}(v) > \theta\}\right| - \sqrt{\theta} \cdot \frac{\theta + 1}{\epsilon}$$

The function q_ϵ^c captures the projection loss for computing histogram, and also considers the noise effect. The projection loss

Algorithm 3 ExtractHistogram algorithm.

Input: A noisy cumulative histogram hc with size θ.
Output: A degree histogram h with size θ.

1: $hc_0 \leftarrow 0, i \leftarrow 1, hc_1 \leftarrow 0$ if $hc_1 < 0$
2: **while** $i \leq \theta$ **do**
3: **if** $hc_i < hc_{i+1}$ **then**
4: $h_i \leftarrow hc_i - hc_{i-1}$
5: $i \leftarrow i + 1$
6: **else**
7: For j from θ to i, find first j such that $hc_{j-1} < hc_i$
8: Assign $h_k \leftarrow \frac{hc_j - hc_{i-1}}{j-i+1}$ for $i \leq k \leq j$
9: $i \leftarrow j + 1$
10: **end if**
11: **end while**
12: **return** h

is the same as ℓ_{proj} for (θ, Ω)-Histogram. In the noise effect term, the coefficient $\sqrt{\theta}$ estimates the noise effect to the histogram extraction from the cumulative histogram, which includes the summation over bins. (Intuitively, the effect of summing θ independent noises have standard deviation $\sqrt{\theta}$ times that of the original noise.) $\Delta_{q_\epsilon^c}$, the sensitivity of q_ϵ^c, is $2\Theta + 2$, as analyzed in the proof of Lemma 2. With all these information on hand, one can perform projection π_θ on graph G and obtain $\mathsf{cumhist}(\pi_\theta(G))$.

In addition to the advantage of requiring less noise, the cumulative degree histogram also has the *monotonicity* property, i.e., the counts from degree 1 to degree θ are non-decreasing. This property provides a way of calibrating the noisy cumulative degree histogram while extracting the histogram from it. We give a histogram extraction algorithm ExtractHistogram based on the calibrating idea in Algorithm 3. Specifically, if the count of degree i is greater than the count of degree $i + 1$, either or both counts are undermined by too much noise injection. It then searches the first non-decreasing bin j from the last bin θ to bin i and assigns the counts uniformly between bin i to bin j (Lines 7-8). After that, it jumps to next non-decreasing bin and extracts the histogram continuously. An additional advantage of our extraction algorithm with calibrating the cumulative histogram is that it averages out the noise effect $\mathsf{Lap}\left(\frac{\theta+1}{\epsilon}\right)$, with the cost of viewing bin counts equally like aggregation in (θ, Ω)-Histogram approach.

Another calibration is based on constrained inference (monotonic regression) proposed in [13]. In Section 4.3, we also report the comparison between our calibration and constrained inference.

The overall algorithm of releasing the degree histogram from the cumulative histogram, named θ-CumulativeHistogram, is shown in Algorithm 4. Similar to Algorithm 2, θ-CumulativeHistogram first performs the computation of qualities and the exponential mechanism in Lines 1-2, and then projects the graph, computes the noisy cumulative histogram, calibrates and extracts, and finally publishes the post-processed histogram. The running time of θ-CumulativeHistogram is $O(|E| \cdot \Theta)$ for any given $G = (V, E)$ and Θ due to Lines 1 and 2, and it requires slightly less time than Algorithm 2 because it does not need to consider different Ω values. We report the experimental results for running time in Section 4.4. We also give the privacy guarantee for θ-CumulativeHistogram algorithm, as well as its proof.

LEMMA 5. *θ-CumulativeHistogram in Algorithm 4 satisfies $(\epsilon_1 + \epsilon_2)$-node-DP.*

PROOF OF LEMMA 5. In Algorithm 4, Lines 1-2 use exponential mechanism with privacy budget ϵ_1. Lines 3 and 4 uses Laplace mechanism with privacy budget ϵ_2. All other steps are indepen-

Algorithm 4 θ-CumulativeHistogram algorithm.

Input: A graph G, privacy budgets ϵ_1 and ϵ_2, candidates T.
Output: A noisy degree distribution $\mathbf{d}$.

1: For each $\theta_i \in T$, computes $q_\epsilon^c(G, \theta_i, \epsilon_2)$
2: Select θ^* from T with probability $\propto \exp\left(\frac{\epsilon_1 q_\epsilon^c(G,\theta_i,\epsilon_2)}{2\Delta_{q_\epsilon^c}}\right)$
3: $G^{\theta^*} \leftarrow \pi_{\theta^*}(G)$ by Algorithm 1.
4: $hc \leftarrow \mathsf{cumhist}\left(G^{\theta^*}\right) + \mathsf{Lap}\left(\frac{\Delta_{\mathsf{cumhist}}}{\epsilon_2}\right)^{\theta^*}$
5: $h \leftarrow \mathsf{ExtractHistogram}(hc)$ by Algorithm 3.
6: $h' \leftarrow \mathsf{PostProcess}(h)$ by Algorithm 5.
7: **return** $\mathbf{d} = h'/||h'||_1$

dent and performed on the noisy output. By the composition theorem and transformation invariance, θ-CumulativeHistogram satisfies $(\epsilon_1 + \epsilon_2)$-node-DP. □

3.4 Post-processing for Tail Distribution

After projection, most nodes that have original degree $> \theta$ will now have degree θ. This results in a high count for the last bin in the histogram. In other words, the count of last bin in histogram, which represents the number of nodes with degree θ in G^θ, actually contains the number of nodes with degree "at least" θ in G, i.e., $|v : deg(v) \geq \theta|$ in G. Releasing a histogram following either (θ, Ω)-Histogram or θ-CumulativeHistogram will preserve such information, given the assumption that the noise is not sufficiently large to destroy it. We can use this property to *reallocate* the counts of degrees larger than θ, making the published distribution be closer to the original distribution and more like a long-tailed distribution.

However, estimating the tail distribution using the noisy histogram is hard, given the perturbed histogram would unlikely to follow the original distribution. We design three simple approaches to allocate counts based on coefficients learned from linear regression, power law distribution, or uniform distribution. Specifically, the post-processing step either learns the slope m and the intercept b from $y = mx + b$ in linear regression, learns a and k from $y = ax^{-k}$ assuming it follows the power law distribution, or allocates the average count to each bin from degree θ to 2θ assuming the tail follows uniform distribution.

Algorithm 5 shows the detail steps of performing linear regression, assuming that from $\theta/2$th bin the counts drop linearly. In particular, Lines 1-2 first compute the average $\bar{c}$ from the second half of histogram H, and Line 3 learns the slope m and intercept b from H with linear regression. Ideally, m should be negative with a small quantity to approximate the long-tail, but m could be positive due to the effect of noise. If so, the algorithm simply assigns each bin with the average $\bar{c}$ (Line 8), otherwise it assigns the estimated count by m and b (Line 6) until the budget is out. For allocating tail distribution based on power law distribution or uniform distribution, we could modify steps in Lines 3 and 6 accordingly. We report the experimental result in Section 4.4 and show that the three approaches do not have significant difference on allocating tail distribution. We use linear regression in our other experiments.

4. EXPERIMENT

In this section, we report experimental results comparing our proposed approaches with approaches with the state of the art, and analyzing how different aspects of our proposed approaches affect the utility.

Algorithm 5 PostProcess algorithm.

Input: A degree histogram h, number of nodes n
Output: A post-processed degree histogram h'.

1: $budget \leftarrow h_\theta$, $H \leftarrow \{h_{\theta/2}, h_{\theta/2+1}, \cdots, h_{\theta-1}\}$
2: $\bar{c} \leftarrow \frac{2}{\theta}\sum_{k=\theta/2}^{\theta-1} h_k$
3: Learn the intercept b and slope m of linear regression on H.
4: **for** k from θ to n **do**
5: **if** $m < 0$ **then**
6: $h_k \leftarrow b + m \times k$
7: **else**
8: $h_k \leftarrow \bar{c}$
9: **end if**
10: $budget \leftarrow budget - h_k$; **break** if $budget < 0$
11: **end for**
12: **return** $h' = h$

4.1 Datasets and Settings

Our experiments use 8 real-world datasets downloaded from [22], as shown in Table 1. The datasets are from several different domains, including social networks, citation networks, and email networks. We pre-processed all graph datasets to be undirected. Table 1 also shows some additional information such as average degree, maximum degree, and the parameter α in α-decay [18] for the distribution (the larger α is, the long-tailed the graph is).

We compare our methods, (θ, Ω)-Histogram and θ-CumulativeHistogram, against two state-of-the-art methods for publishing node degree histograms while satisfying node-DP. The first algorithm is *Truncation* from [18]. The algorithm has two parameters: β and θ. We set $\beta = \epsilon \ln(|V|/\theta)$ as suggested in [18]. Since θ is manually specified in [18], we consider all θ values among $\{1, 2, 4, \cdots, 2^{\lfloor 2\log_2(|V|)\rfloor}\}$ and report the result with the smallest error. The second algorithm is *Flowgraph* [31], which uses the max-flow algorithm with minimizing L2 error of the remaining flows. Following [31], we select $\theta \in \{1, 2, 4, \cdots, 2^{\lfloor 2\log_2(|V|)\rfloor}\}$ by the generalized exponential mechanism.

We use L1 error and KS-distance to evaluate the approximation results on privacy budget $\epsilon \in [0.1, 2.0]$ in Section 4.3, where each privacy budget ϵ is divided into $\epsilon_1 = 0.1\epsilon$ and $\epsilon_2 = 0.9\epsilon$ in our proposed approaches. To evaluate the KS-distance on different approaches, we generate the degree sequence using the noisy degree distribution and then compare with the true degree sequence. All results on all approaches are the averages from 30 runs. We also show the complementary CDFs when $\epsilon = 1$ for all graph datasets.

4.2 Evaluating π_θ

We first directly compare our proposed graph projection method, π_θ, with three other existing graph projection methods mentioned in Section 2.4: **T** (Truncation), **FG** (FlowGraph), and **ER** (EdgeRemoval). Table 2 shows the results. We use two metrics: the L1 error of degree histogram after projection, and $\frac{|E'|}{|E|}$, where E' denotes the edges after projection. Since the histogram of $\pi_\theta(G)$ could have lots of counts with degree exactly θ (as all nodes with original degree over θ are *shaved* to θ), we compare π_θ after removing the bin of θ. We consider three θ values, $\theta = 16, 64, 128$.

The results show that Truncation and our π_θ have the lowest L1 error, meaning that these two approaches maintain the shape of distribution after projection. On the other hand, Flowgraph and π_θ preserve the most number of edges. In addition, because Flowgraph has a step to solve a max-flow problem with an objective

Graph	$\|V\|$	$\|E\|$	$d_{\max}$	d_{avg}	α
Facebook	4,039	88,234	1,045	43.69	2.39
Wiki-Vote	7,066	100,736	1065	28.32	1.94
Ca-HepPh	11,204	117,649	491	21.00	1.58
Email-Enron	33,696	180,811	1,383	10.73	1.76
Cit-HepPh	34,401	420,828	846	24.47	2.60
Loc-Brightkite	56,739	212,945	1,134	7.51	1.89
Twitter	81,306	1,342,310	3,383	33.02	2.15
DBLP	317,080	1,049,866	343	6.62	2.31

Table 1: Information about datasets. All graphs are connected (i.e. one connected component). $d_{\max}$ denotes maximum degree, and d_{avg} denotes average degree. α is the parameter for α-decay [18].

function, the running time of Flowgraph projection is roughly 10 to 20 times compared to other three approaches (we omit the result due to space limitation). Our proposed projection approach, π_θ, thus benefits from three sides: preserving well on the shape of distribution, the number of edges, and with better efficiency.

4.3 Evaluating (θ, Ω)-Histogram and θ-Cumulative

L1 and KS of Node Degree Distributions. Figure 3 compares the quality of the resulting node degree distributions of our two proposed method to those of *Truncation* [18] and *Flowgraph* [31]. We also report the result of a variant of our proposed θ-CumulativeHistogram. Instead of using our proposed method to exploit the monotonicity property to extract the resulting histogram from a noisy cumulative histogram, this variant uses the constrained inference (isotonic regression) technique in [13] for calibrating cumulative histogram (θ-*Constrained*). We also plot the standard deviation computed from 30 runs for each data point.

The experimental result shows that *Truncation* generally performs the worst, followed by *Flowgraph*, which has the largest variance. Looking at intermediate results, we note that *Flowgraph* tends to select very different θ values in different runs, and some of the values are very large. This combined with the high sensitivity of the resulting histogram (6θ) causes the high variance.

Our two proposed methods and the variant all perform significantly better than *Truncation* and *Flowgraph*, and result in quite accurate node degree histograms, especially when $\epsilon \geq 0.5$ and for the larger datasets. When looking at the L1 results, the θ-*Constrained* variant performs noticeably worse than the two proposed method. When looking at the KS results, we can see that θ-*Constrained* performs almost identically with θ-CumulativeHistogram, and they both perform noticeably better than (θ, Ω)-Histogram on the three datasets that are relatively small. The reason that θ-*Constrained* performs not as well as our proposed methods in terms of L1 distance is because when encountering an unversed pair in the noisy cumulative histogram, the isotonic regression method tends to result in stepwise cumulative histograms, which when converted to histograms results in many bins having 0 count. Overall, θ-CumulativeHistogram appears to be the best approach for publishing the degree distribution under node-DP.

Approximating the Node Degree Distributions. We now show the complementary cumulative distribution functions (CCDFs) from the distributions with median error out of 30 runs at $\epsilon = 1$ for each approach. As demonstrated in Figure 4, the distributions from (θ, Ω)-Histogram, θ-CumulativeHistogram, and θ-Constrained closely follow the true one, especially in the smaller degree part (less than 100); and θ-CumulativeHistogram performs the best. All approaches have a clear cutoff point, because they are all designed based on the idea of limiting sensitivity. Since these methods all underestimates the distribution for higher degree nodes, there may be ways to correct this systematic bias. This is an interesting future work. *Truncation* tends to perform the worst, often overestimating the number of low-degree nodes and underestimating the number of high-degree nodes. *Flowgraph*'s behavior appears more erratic, likely due to the high variance.

Running time analysis. Table 3 reports the average running time of our proposed approaches on an Intel i7-3770 3.40GHz machine with 12GB memory. As we have discussed in Section 3, the advantage that the time complexity of our algorithms is linear to number of edges ($O(|E| \cdot |\Theta|)$) enables both approaches handle the graphs with very large size. Besides, the computation of the quality on each candidate (which takes most of the time) can be easily parallelized, making our approaches scalable to very large graphs.

4.4 Introspective Analysis

In this section, we perform the introspective analysis, to try to understand how different aspects of our approaches affect the utility.

Analysis on tuning θ and Ω. In this analysis, we would like to know whether our quality functions for selecting the parameters θ, Ω are well-designed, to what extent the usage of aggregation affects the utility of (θ, Ω)-Histogram, and to what extent our proposed approaches are sensitive to the choice of $\Theta = 200$. Figure 5 shows the results.

The left half of Figure 5 shows different variants of (θ, Ω)-Histogram. The original method is dubbed (θ, Ω)-Hist; (θ, Ω)-Q-Best directly selects the value (θ, Ω) that has the best quality function (without using the exponential mechanism, and is thus non-private); (θ, Ω)-Oracle creates histograms using all (θ, Ω) values and select the one that results in noisy histograms with the least L1 error (also non-private); (θ, Ω)-Q-400 uses $\Theta = 400$ instead of $\Theta = 200$; (θ, Ω)-No-group does not use aggregation and is essentially θ-Histogram. From the results, we can see that not using grouping significantly impacts the accuracy. The curves for the original (θ, Ω)-Hist and for (θ, Ω)-Q-400 are very close, and in fact almost overlapping when $\epsilon \geq 0.5$. This suggests that our method is not very sensitive to the choice of $\Theta = 200$. We also see that the results for (θ, Ω)-Q-Best and (θ, Ω)-Oracle are almost overlapping, suggesting that our quality function is well-designed.

The right half of Figure 5 shows different variants of θ-CumulativeHistogram. Similarly, θ-Q-Best uses the θ value that has the best quality function (non-private), θ-Oracle uses the θ value that empirically has the lowest L1 error, and θ-Q-400 uses $\Theta = 400$ instead of $\Theta = 200$. In terms of L1 error, the curves for all methods are quite close. For the KS-error, the curves are also similar, although θ-Oracle sometimes deviates from the other curves slightly. Note that since θ-Oracle selects θ that optimizes L1 error, it may not produce the best KS result. The closeness of the 4 curves suggests the following: (1) the quality function for θ-CumulativeHistogram is well-designed; (2) we are able to select a θ that is quite close to optimal (because the low sensitivity of the quality function); and (3) the method is not sensitive to the choice of parameter Θ.

Analysis on choices of θ. To understand the effects from the choices of θs, we apply $\theta = 25, 50, 100, 200$ on θ-Histogram without aggregation and report the experimental result in Figure 6. The result shows that the utility of the published degree distribution is very sensitive to the choice of θ, i.e. if one chooses a "bad" θ, it easily leads to a noisy degree distribution with poor utility. Our proposed approaches solve the problem by applying the expo-

Graph	θ=16				θ=64				θ=128			
	T	**FG**	**ER**	π_θ	**T**	**FG**	**ER**	π_θ	**T**	**FG**	**ER**	π_θ
Face-	3828	5112	5122	**3228**	1923	2060	1880	**1313**	960	1092	976	**801**
book	0.03	0.30	0.20	0.27	0.27	0.70	0.62	0.66	0.57	0.90	0.86	0.88
Wiki-	5856	8232	4666	**4205**	**3546**	6018	4908	4991	**4622**	5404	5120	5054
Vote	0.00	0.25	0.12	0.19	0.11	0.57	0.45	0.51	0.36	0.76	0.70	0.73
Ca-	**4894**	8570	7198	5728	**4381**	5536	4868	4548	**4533**	4992	4712	**4533**
HepPh	0.11	0.36	0.28	0.33	0.23	0.51	0.43	0.48	0.55	0.81	0.76	0.78
Email-	14866	11200	9749	**8574**	**7361**	11216	10030	9848	**7223**	13602	12635	12577
Enron	0.13	0.37	0.27	0.34	0.33	0.62	0.56	0.60	0.50	0.75	0.72	0.74
Cit-	33657	33112	33373	**23724**	12297	7614	7906	**6401**	4888	4260	4329	**3825**
HepPh	0.05	0.45	0.30	0.39	0.49	0.82	0.77	0.80	0.80	0.93	0.93	0.93
Loc-Bri-	**31510**	43734	37554	36928	**35791**	39720	38629	38455	**37899**	39912	39616	39537
ghtkite	0.22	0.55	0.44	0.51	0.59	0.82	0.78	0.80	0.77	0.91	0.90	0.90
Twitter	78098	74286	75544	**53703**	39932	26358	23754	**20776**	18975	17116	17306	**15293**
	0.02	0.29	0.17	0.24	0.22	0.61	0.52	0.57	0.46	0.77	0.72	0.74
DBLP	**83630**	137112	119932	107484	**108997**	118798	117936	116764	**117225**	118464	118389	118273
	0.43	0.75	0.68	0.72	0.67	0.89	0.86	0.87	0.97	0.99	0.99	0.99

Table 2: L1 error (first row of each dataset) and the number of edges preserved $\frac{|E'|}{|E|}$ (second row of each dataset) of the degree histogram from different projection approaches on 8 datasets. **T** = Truncation, **FG** = Flowgraph, and **ER** = Edge-Removal.

Graph	(θ, Ω)-Histogram		θ-CumulativeHistogram	
	Quality	**Total**	**Quality**	**Total**
Facebook	78.86	79.43	74.38	74.98
Wiki-Vote	104.28	104.99	97.85	98.58
Ca-HepPh	130.84	131.70	119.50	120.39
Email-Enron	287.62	289.45	263.52	265.39
Cit-HepPh	574.06	578.13	548.25	552.41
Loc-Brightkite	407.69	410.29	367.38	370.08
Twitter	1867.66	1880.78	1833.84	1847.35
DBLP	2773.83	2791.14	2566.02	2582.96

Table 3: Running time (in seconds) for our proposed approaches.

nential mechanism with well-designed quality functions, making it only depending on Θ, which is not sensitive as discussed earlier.

Additional analysis. We also compared three different methods for allocating the tail distribution from counts of last bin (i.e., θ) to other bins: linear regression, power law distribution, and simple average. Experimental results are reported in Figure 7 in the appendix, and show that the three strategies produce similar results, suggesting that any of these can be used.

Finally, we analyze the influence from different edge orderings in the projection step. We pick up 10 random-ordered edge sequences, perform the projection and publish the degree distribution, and compare the highest and lowest error among these 10 results with our original approach. The result in Figure 8 in Appendix shows that edge orderings does not have significant impacts on our proposed projection approach and publishing the degree distribution.

5. RELATED WORK

DP on Graph Data. Applying differential privacy to the graph setting has been studied extensively; most use edge-DP [12, 15, 32, 16, 29, 17, 35], but some consider node-DP [18, 3, 4, 31]. We argue that using node-DP is more meaningful as node-DP gives the personal privacy control on graph data. The work in [12] studies the problem of publishing degree sequence from a graph under edge-DP, which has sensitivity of 2, as removing one edge will influence the degrees of two nodes. The approach in [12] applies the Laplace mechanism directly and performs a constrained inference to synthesize the degree sequence to be monotonic. Several following works have considered publishing the degree sequence or distribution and extended the technique to synthetic generation [32, 16, 29, 15, 16, 35]. These approaches are infeasible when the setting moves to node-DP, where the sensitivity of $\mathsf{hist}(G)$ becomes $2(|V| - 1)$. The techniques of using "projection" to limit the sensitivity are studied to answer degree distribution under node-DP [18, 31]. The truncation approach is proposed in [18], and the Flowgraph approach is proposed in [31]. We have experimentally compared with these approaches. Researchers have proposed several approaches for answering a single-valued query, i.e., a certain statistics value about the graph. For example, subgraph counting queries (e.g., the number of triangles, the number of k-stars) have been studied under edge-DP [15, 35] and node-DP [18, 3, 4], and the computing cost of minimum spanning tree has been studied in [28] using smooth sensitivity.

Histogram publishing under DP. The problem of publishing histogram under differential privacy has been studied extensively [1, 6, 11, 13, 23, 30, 33, 34, 36]. The technique of using aggregation and averaging to reduce errors is studied in [1, 19, 23, 34, 36]. Publishing node degree histograms in graphs has the challenge that the sensitivity is much larger since removing one node may affect every other node; thus one has to consider histogram aggregation together with graph projection. We adopt the idea of aggregating bins for more accurate histogram from this line of work; however we exploit the power-law nature of graph node degree distributions to limit the candidates to those constructed using geometric series. An interesting approach to discover optimal aggregation configuration is proposed in [23], in which one adds the noise to the error estimation of each group of bins, and then uses dynamic programming to select a configuration that minimizes the sum of errors caused by grouping and errors caused by the noise. Fixing each θ value, one can use this method to select an aggregation structure Ω. It is an interesting future work to see whether this technique can be adapted to our problem of selecting (θ, Ω) together without divid-

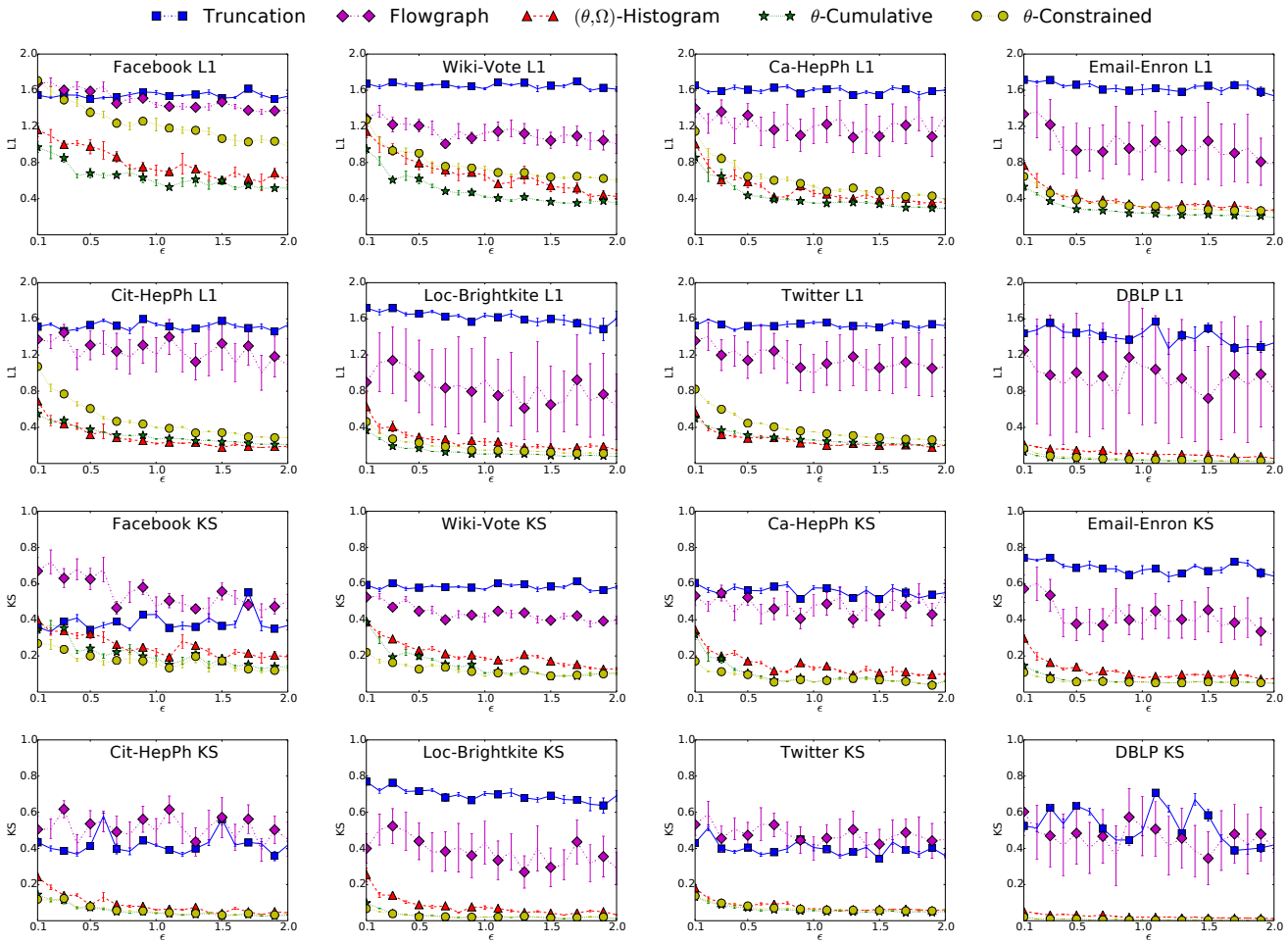


Figure 3: The L1 error (upper half) and KS distance (lower half) of approaches on different datasets.

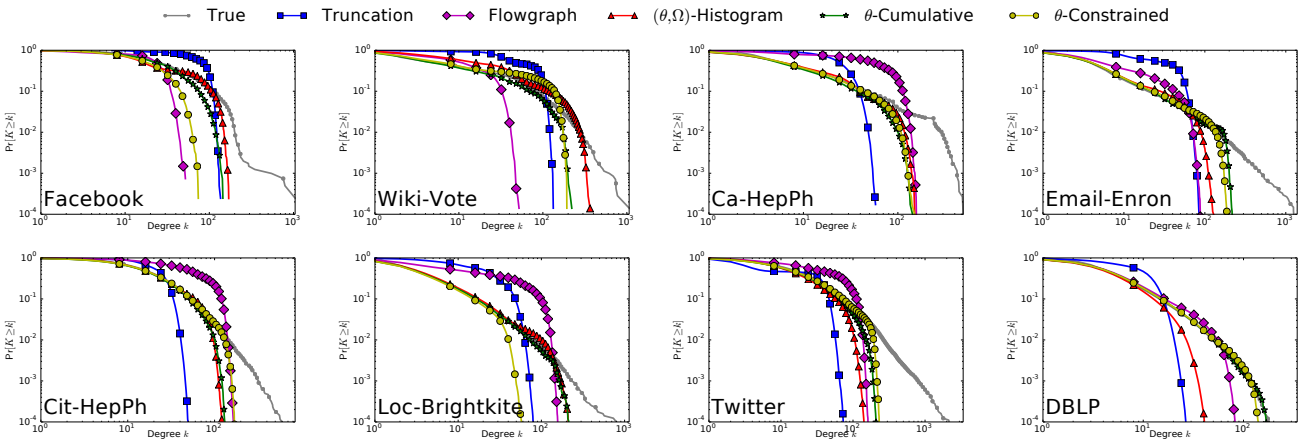


Figure 4: The complementary cumulative distribution function (CCDF) vs. degree of approaches on different datasets at $\epsilon = 1$. For any degree k in x-axis, the value in y-axis denotes the fraction of nodes with degree at least k ($\Pr[K \geq k]$).

ing the privacy budget, and if so, how this method compares with our approach of using geometric series.

It is suggested in [13] that when the number of bins to be published are relatively large, it is better to use hierarchical method to publish histograms under DP. In our work, the number of bins is usually small, and directly publishing the histogram without hierarchical method is preferred based on [30]. Releasing a cumulative histogram is also studied in [14]. While the global sensitivity of releasing a cumulative histogram is the number of bins in the cumulative histogram, the work proposes an ordered mechanism based

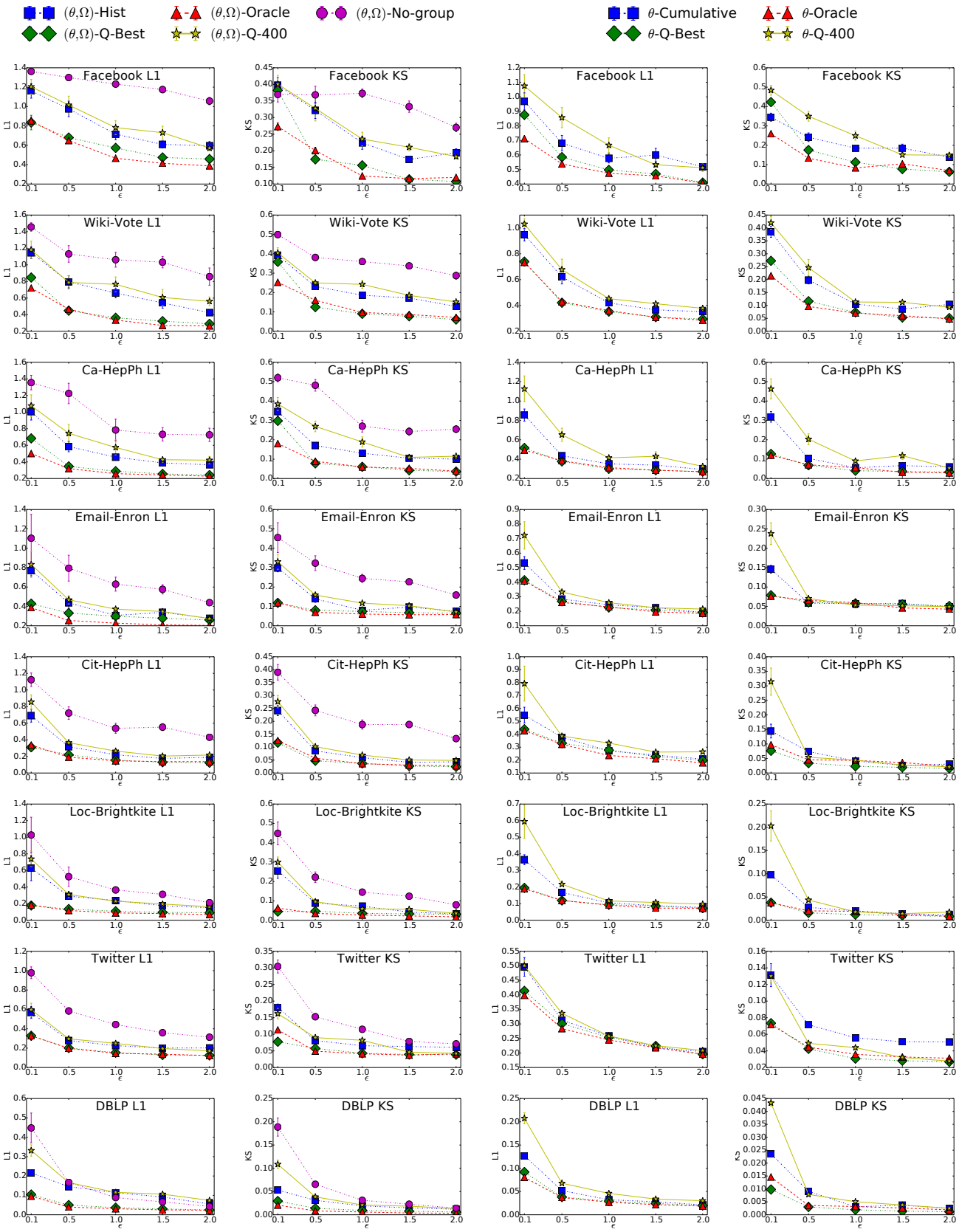


Figure 5: The experimental results on the effects of tuning the parameters θ and Ω for (θ, Ω)-Histogram (left half) and θ-CumulativeHistogram (right half).

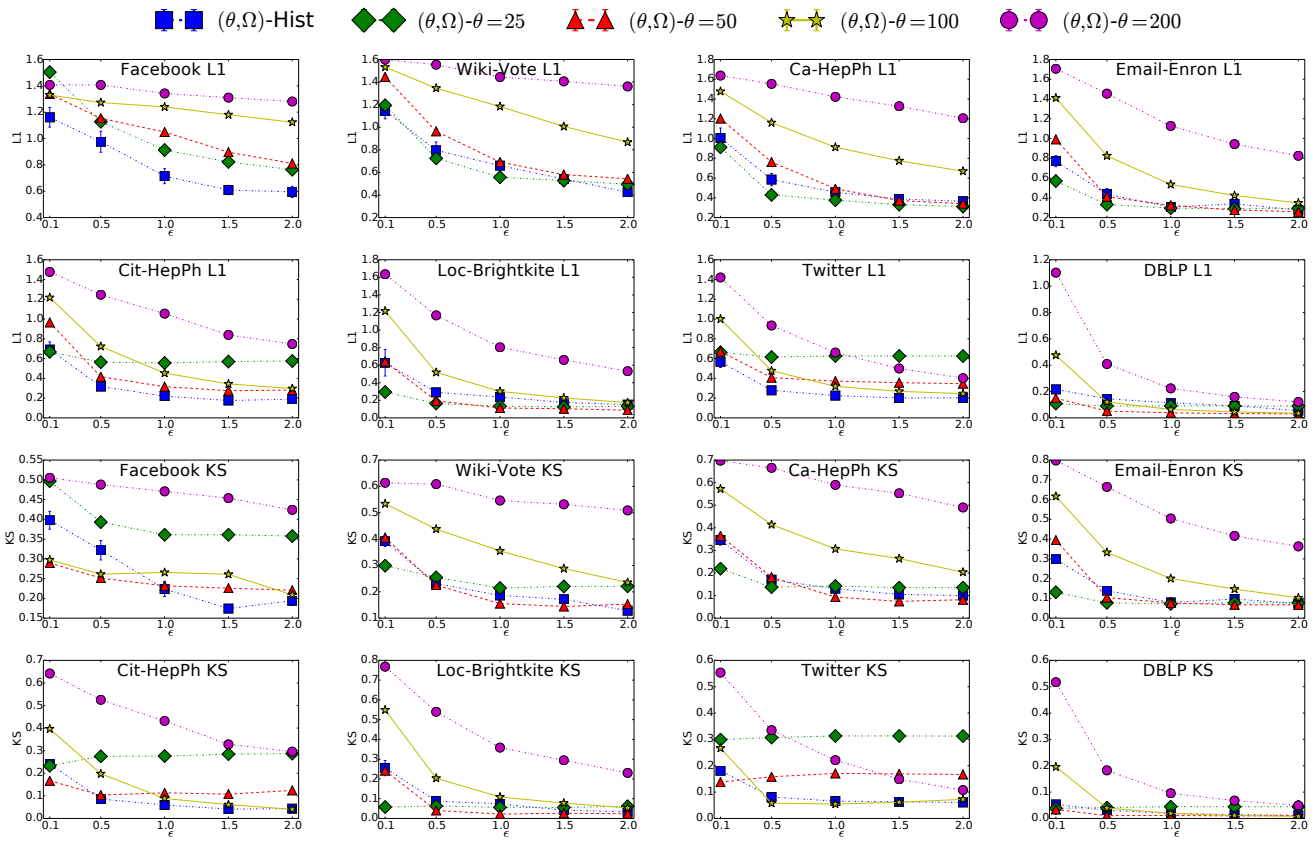


Figure 6: The L1 error (upper half) and KS distance (lower half) of (θ, Ω)-Histogram with manually setting $\theta \in \{25, 50, 100, 200\}$ on different datasets.

on a relaxed privacy notion named Blowfish privacy, in which the sensitivity becomes 1 for some predefined policy. In our work, we study the sensitivity of releasing a cumulative histogram after projection while satisfying node-DP.

6. CONCLUSION

In this paper, we discuss the necessity of using node-DP in private analysis of graph data, and study how to publish node degree histograms while satisfying node-DP. We propose a new projection method to preserve as much information of a graph as possible and limit the sensitivity of publishing the degree histograms on the projected graph. Based on the projection, we propose two approaches, (θ, Ω)-Histogram and θ-CumulativeHistogram, and experimentally compare with existing works for publishing degree distribution. The experimental results show that our proposed approaches have significant improvement over the state of the art. While these methods accurately capture the distribution of low degree nodes, future research is needed to better predict the distribution of higher-degree nodes using the preserved lower-degree information.

7. ACKNOWLEDGMENTS

This work reported in this paper is supported by National Science Foundation under grant CNS-1116991. We would like to thank Rui Chen and the reviewers for their useful comments.

8. REFERENCES

[1] G. Acs, C. Castelluccia, and R. Chen. Differentially private histogram publishing through lossy compression. In *ICDM*, pages 1–10, 2012.

[2] L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *WWW*, pages 181–190, 2007.

[3] J. Blocki, A. Blum, A. Datta, and O. Sheffet. Differentially private data analysis of social networks via restricted sensitivity. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science*, ITCS '13, pages 87–96, New York, NY, USA, 2013. ACM.

[4] S. Chen and S. Zhou. Recursive mechanism: Towards node differential privacy and unrestricted joins. In *SIGMOD '13*, pages 653–664, 2013.

[5] G. Cormode. Personal privacy vs population privacy: Learning to attack anonymization. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 1253–1261, 2011.

[6] G. Cormode, C. Procopiuc, D. Srivastava, and T. T. L. Tran. Differentially private summaries for sparse data. In *ICDT '12*, pages 299–311, 2012.

[7] C. Dwork. Differential privacy. In *ICALP*, pages 1–12, 2006.

[8] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[9] C. Dwork and M. Naor. On the difficulties of disclosure prevention in statistical databases or the case for differential privacy. *Journal of Privacy and Confidentiality*, 2, 2008.

[10] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014.*, pages 17–32, 2014.

[11] M. Hardt, K. Ligett, and F. Mcsherry. A simple and practical algorithm for differentially private data release. In *Advances in Neural Information Processing Systems (NIPS) 25*, pages 2339–2347, 2012.

[12] M. Hay, C. Li, G. Miklau, and D. Jensen. Accurate Estimation of the Degree Distribution of Private Networks. In *ICDM*, pages 169–178, 2009.

[13] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private histograms through consistency. *PVLDB*, 3(1):1021–1032, 2010.

[14] X. He, A. Machanavajjhala, and B. Ding. Blowfish privacy: Tuning privacy-utility trade-offs using policies. In *SIGMOD*, pages 1447–1458, 2014.

[15] V. Karwa, S. Raskhodnikova, A. Smith, and G. Yaroslavtsev. Private analysis of graph structure. In *VLDB*, 2011.

[16] V. Karwa and A. B. Slavković. Differentially private graphical degree sequences and synthetic graphs. In *Proceedings of the 2012 International Conference on Privacy in Statistical Databases*, PSD'12, pages 273–285, Berlin, Heidelberg, 2012. Springer-Verlag.

[17] V. Karwa, A. B. Slavkovic, and P. N. Krivitsky. Differentially private exponential random graphs. In *Privacy in Statistical Databases - UNESCO Chair in Data Privacy, International Conference, PSD 2014, Ibiza, Spain, September 17-19, 2014. Proceedings*, pages 143–155, 2014.

[18] S. P. Kasiviswanathan, K. Nissim, S. Raskhodnikova, and A. Smith. Analyzing graphs with node differential privacy. In *Proceedings of the 10th Theory of Cryptography Conference on Theory of Cryptography*, TCC'13, pages 457–476, Berlin, Heidelberg, 2013. Springer-Verlag.

[19] G. Kellaris and S. Papadopoulos. Practical differential privacy via grouping and smoothing. *PVLDB*, 6(5):301–312, 2013.

[20] D. Kifer and B.-R. Lin. Towards an axiomatization of statistical privacy and utility. In *PODS*, pages 147–158, 2010.

[21] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *SIGMOD*, pages 193–204, 2011.

[22] J. Leskovec. Stanford large network dataset collection. http://snap.stanford.edu/data/.

[23] C. Li, M. Hay, G. Miklau, and Y. Wang. A data-and workload-aware algorithm for range queries under differential privacy. *PVLDB*, 7(5), 2014.

[24] N. Li, W. H. Qardaji, D. Su, Y. Wu, and W. Yang. Membership privacy: a unifying framework for privacy definitions. In *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*, pages 889–900, 2013.

[25] F. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *SIGMOD*, pages 19–30, 2009.

[26] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007.

[27] A. Narayanan and V. Shmatikov. De-anonymizing social networks. In *IEEE Symposium on Security and Privacy*, pages 173–187. IEEE Computer Society, 2009.

[28] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *STOC*, pages 75–84, 2007.

[29] D. Proserpio, S. Goldberg, and F. McSherry. A workflow for differentially-private graph synthesis. In *Proceedings of the 2012 ACM Workshop on Workshop on Online Social Networks*, WOSN '12, pages 13–18, New York, NY, USA, 2012. ACM.

[30] W. Qardaji, W. Yang, and N. Li. Understanding hierarchical methods for differentially private histograms. *PVLDB*, 6(14):1954–1965, 2013.

[31] S. Raskhodnikova and A. D. Smith. Efficient lipschitz extensions for high-dimensional graph statistics and node private degree distributions. *CoRR*, abs/1504.07912, 2015.

[32] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Y. Zhao. Sharing graphs using differentially private graph models. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 81–98, New York, NY, USA, 2011. ACM.

[33] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. *IEEE Trans. Knowl. Data Eng.*, 23(8):1200–1214, 2011.

[34] J. Xu, Z. Zhang, X. Xiao, Y. Yang, and G. Yu. Differentially private histogram publication. In *KDD*, 2012.

[35] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. Private release of graph statistics using ladder functions. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, pages 731–745, New York, NY, USA, 2015. ACM.

[36] X. Zhang, R. Chen, J. Xu, X. Meng, and Y. Xie. Towards accurate histogram publication under differential privacy. In *Proceedings of the 2014 SIAM International Conference on Data Mining, Philadelphia, Pennsylvania, USA, April 24-26, 2014*, pages 587–595, 2014.

APPENDIX

A. PROOF OF LEMMA 2

PROOF OF LEMMA 2. Lemma 2 says that for any neighboring G, G', we have $|q_\epsilon^h(G,\theta_i,\Omega_j) - q_\epsilon^h(G',\theta_i,\Omega_j))| \leq 6\Theta+4$. Since $q_\epsilon^h(G,\theta_i,\Omega_j) = -\ell_{hist}(G,\theta_i,\Omega_j) - \ell_{proj}(G,\theta_i)$, it suffices to show that (A) $\ell_{hist}(G,\theta_i,\Omega_j) - \ell_{hist}(G',\theta_i,\Omega_j) \leq 2(2\Theta+1)$ and that (B) $\ell_{proj}(G,\theta_i) - \ell_{proj}(G',\theta_i) \leq 2(\Theta+1)$.

The latter (B) holds because $\ell_{proj}(G,\theta_i) = 2 \cdot \big|\{v | v \in V, deg_{\pi_\Theta(G)}(v) > \theta_i\}\big|$, and from the proof of Lemma 1, at most $\Theta+1$ nodes will have different degrees in $\pi_\Theta(G)$ and $\pi_\Theta(G')$.

We now show that the former (A) holds. Note that $\ell_{hist}(G,\theta_i,\Omega_j) = \sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} \left(\left| c_d - \frac{\sum_{d \in \mathbf{g}_k} c_d}{|\mathbf{g}_k|} \right| + \frac{2\theta_i+1}{\epsilon|\mathbf{g}_k|} \right)$,

The second term in ℓ_{hist}, i.e., $\frac{2\theta_i+1}{\epsilon|\mathbf{g}_k|}$ is independent of the input dataset and thus does not change between two neighboring graphs, so we only need to analyze the first term. We use $avg_{\mathbf{g}_k} = \frac{\sum_{d \in \mathbf{g}_k} c_d}{|\mathbf{g}_k|}$ to denote the average count within group $\mathbf{g}_k$ on $\pi_\theta(G)$,

and similarly $avg'_{\mathbf{g}_k}$ on $\pi_\theta(G')$. We have

$$\begin{aligned}
&\sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} \left| avg_{\mathbf{g}_k} - avg'_{\mathbf{g}_k} \right| \\
&= \sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} \left| \frac{\sum_{d \in \mathbf{g}_k} c_d}{|\mathbf{g}_k|} - \frac{\sum_{d \in \mathbf{g}_k} c'_d}{|\mathbf{g}_k|} \right| \\
&= \sum_{\mathbf{g}_k \in \Omega_j} \left| \sum_{d \in \mathbf{g}_k} c_d - \sum_{d \in \mathbf{g}_k} c'_d \right| \\
&\leq \sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} \left| c_d - c'_d \right| = 2\theta_i + 1
\end{aligned}$$

Thus, we have

$$\begin{aligned}
&\left| \sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} |c_d - avg_{\mathbf{g}_k}| - \sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} \left| c'_d - avg'_{\mathbf{g}_k} \right| \right| \\
&\leq \sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} \left| \left(|c_d - avg_{\mathbf{g}_k}| - \left| c'_d - avg'_{\mathbf{g}_k} \right| \right) \right| \\
&\leq \sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} \left| (c_d - avg_{\mathbf{g}_k}) - (c'_d - avg'_{\mathbf{g}_k}) \right| \\
&\leq \sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} \left| (c_d - c'_d) - (avg_{\mathbf{g}_k} - avg'_{\mathbf{g}_k}) \right| \\
&\leq \sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} \left| c_d - c'_d \right| + \sum_{\mathbf{g}_k \in \Omega_j} \sum_{d \in \mathbf{g}_k} \left| avg_{\mathbf{g}_k} - avg'_{\mathbf{g}_k} \right| \\
&\leq (2\theta_i + 1) + (2\theta_i + 1) \leq 2(2\Theta + 1).
\end{aligned}$$

□

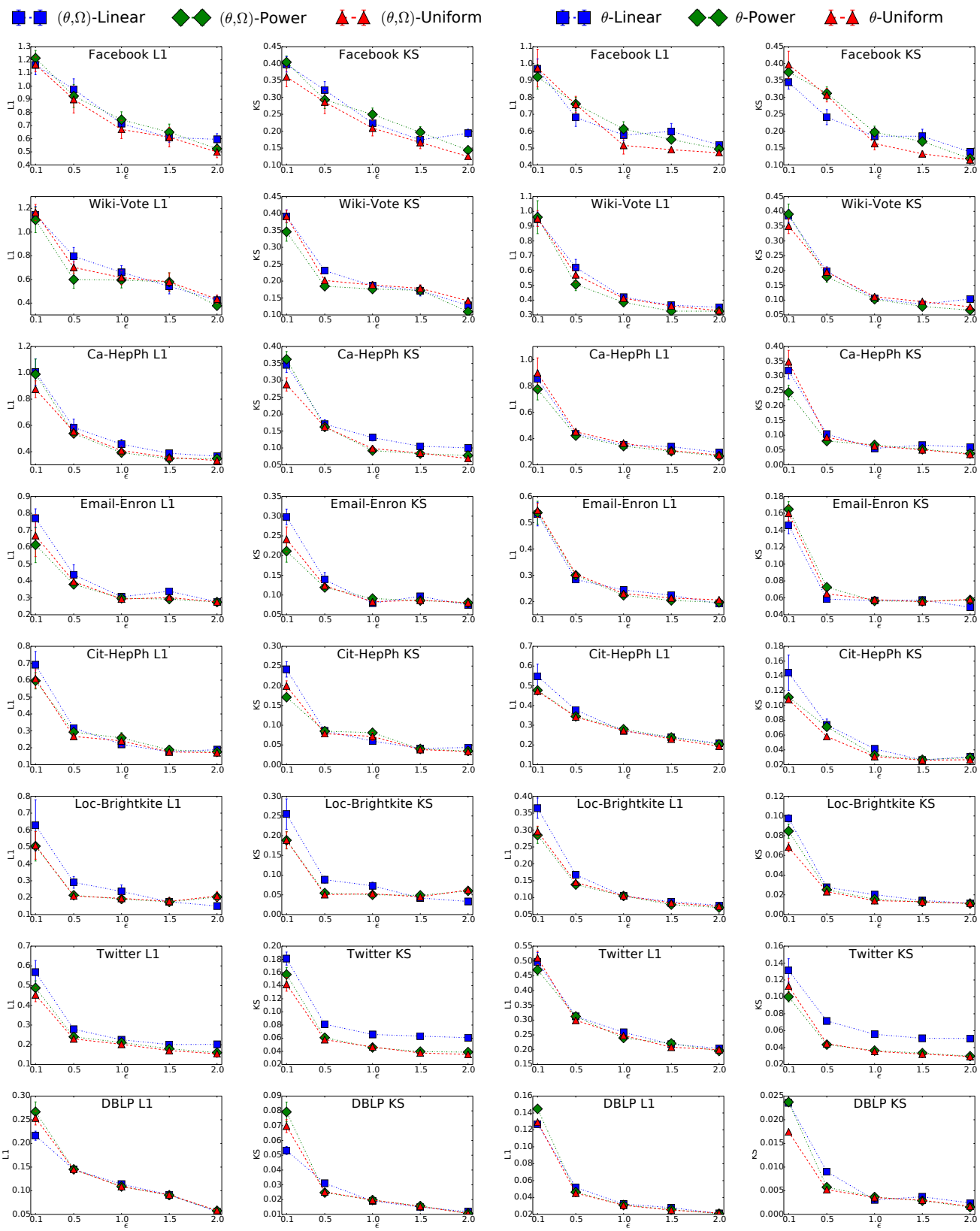


Figure 7: The experimental results on the effects of different allocation strategies for tail distribution for (θ, Ω)-Histogram (left half) and θ-CumulativeHistogram (right half).

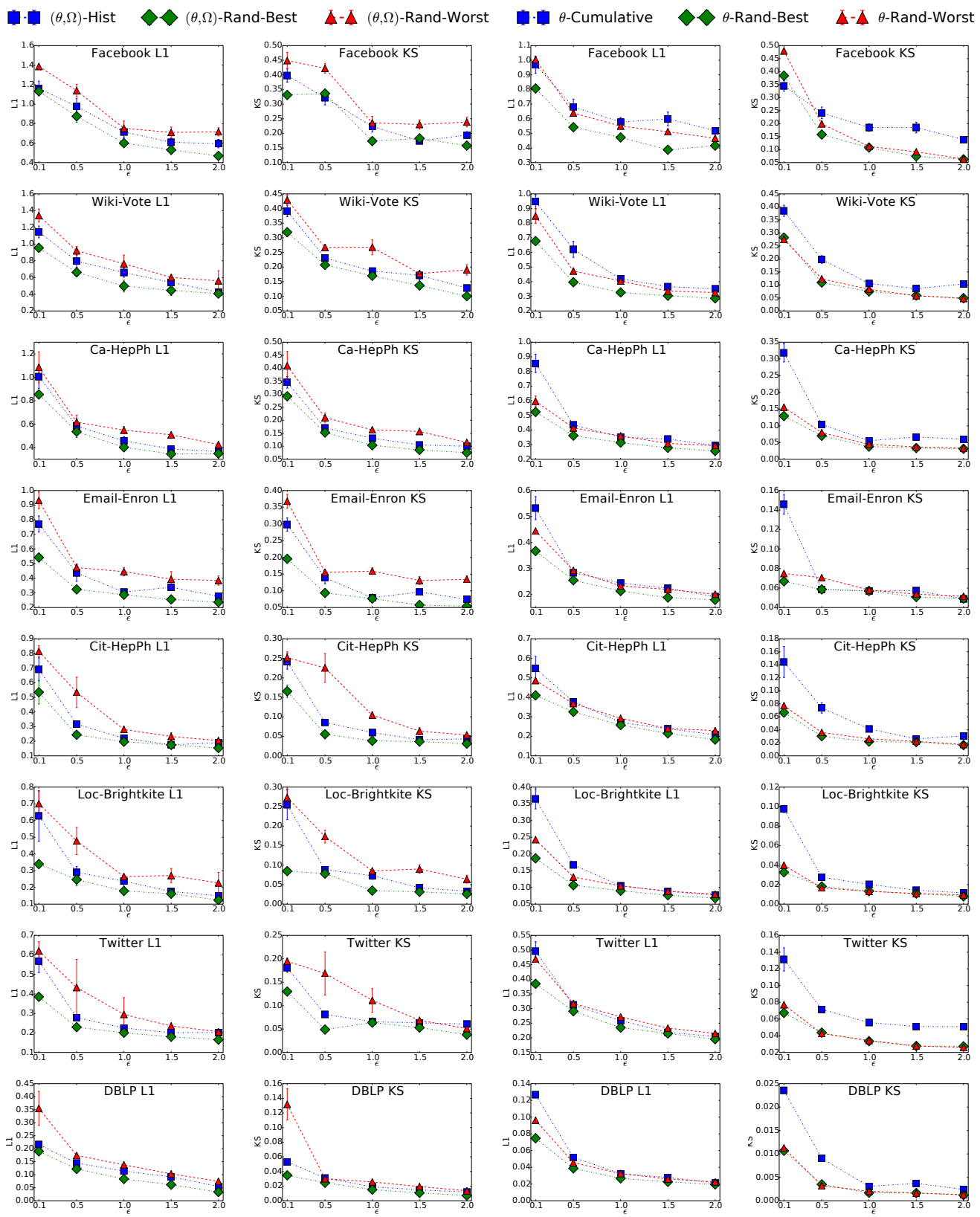


Figure 8: The experimental results on the effects of permutation on the edge orderings in projection for (θ, Ω)-Histogram (left half) and θ-CumulativeHistogram (right half).