Benchmarking Differentially Private Graph Algorithms

Huiyi Ning Sreeharsha Udayashankar Sara Qunaibi Karl Knopf Xi He
{hy3ning,s2udayas,squnaibi,kknopf,xi.he}@uwaterloo.ca
University of Waterloo

ABSTRACT

Differential privacy is the gold standard when it comes to facilitating data analysis with strong privacy guarantees. The past decade has witnessed the design of numerous algorithms for differentially private graph analysis. These algorithms offer different privacy guarantees, and require different parameters and configurations for their usage. Some of these algorithms also suffer from poor scalability and are not equipped to handle large datasets. The combination of these factors makes determining the optimal algorithmic choice for a given scenario a non-trivial affair.

In this paper, we examine the trade-offs between the accuracy and performance of various classes of differentially private graph analysis algorithms by benchmarking them on real-world datasets. Our evaluation demonstrates that the optimal choice of algorithm is highly dependent on the properties of the underlying dataset as well as the performance and privacy requirements of a given user. This may explain why, despite a wealth of differentially private algorithms being available for graph analysis, their usage is not yet prevalent. We therefore urge the adoption of a standardized benchmarking platform, to facilitate the design and implementation of differentially private graph algorithms.

1 INTRODUCTION

With the advent of the digital era, data collection is on the rise. Graph-like datasets are used to store valuable sensitive data such as email communications, social networking data, financial networks and health networks [2, 3, 8]. These datasets are often used for analysis of the relationships between the entities in the graph, and various aggregate statistics have been designed to facilitate this. For example, graph analytics have been used to improve digital contact tracing efforts [27]. There are increasing volumes of sensitive data that get passed around from telecommunication systems to social networking systems to retailers and so on. Privacy concerns about this data fundamentally limits its potential applications [5]. Traditional security-control methods [1] only provide some degree of protection, and do not meet the stringent privacy requirements needed to prevent the complex attacks facilitated by modern technological advancements. Failure to provide the adequate protection before releasing data in any form can often have disastrous consequences [17].

Originally, privacy-preserving practices for graphs were based on anonymization of the released graphs [10, 16, 20, 31, 38]. However, these techniques fail to offer guarantees in several ways. Two k-anonymous releases can be combined to learn the sensitive locations of an individual [15]. The decisions made by anonymization algorithms can leak information to the attacker as well, jeopardizing privacy [35]. Hence, there is a need for strong privacy guarantees and tools to support the analysis of sensitive graph data.

In recent years, differential privacy (DP) has arisen as a means to achieve these guarantees. It is able to provide strong provable privacy guarantees when used to release aggregate statistics about large datasets [13]. Despite a rich set of differentially private algorithms [12, 19, 23, 29, 34] for graph statistics, they are rarely used in practice due to several challenges. One of the challenges is the difficulty that non-experts face when deciding the optimal algorithm or even privacy guarantee levels. Correctly implementing differentially private algorithms can be tedious [26, 28].

To address this, we implement and benchmark the various algorithms available to handle graph queries by examining the trade-offs between accuracy, privacy guarantees and computational performance. With this evaluation, we demonstrate that the optimal algorithmic choice is highly dependent on a wide-variety of factors, some of which add additional layers of complexity for inexperienced users attempting to use them. To help the broader community, the algorithm implementations and evaluation results presented in this work have been integrated into DPGraph [36], a web-based benchmark platform.

2 BACKGROUND

Graphs considered in prior differential privacy work are mainly undirected and have no additional labels on nodes and edges. Given a graph $G = (V, E)$, where V is the set of nodes and E is a set of edges that connect pairs of nodes in V, prior work focuses on releasing graph statistics, such as degree distribution [12], small subgraph counting [33], cut [18], graphons estimation [7], and generating synthetic graphs [32]. We summarize a subset of the key algorithms in Table 1 and organize them by privacy category and query type. We will first present the privacy guarantees for graphs and then highlight existing approaches from literature.

Variants of DP for graphs The standard differential privacy for tabular data is defined as follows.

Definition 1 (ϵ-*Differential Privacy* [13, 14]). *A randomized algorithm A satisfies ϵ-differential privacy if for all pairs of neighbouring databases D and D' and for any output set S, we have $Pr[A(D) \in S] \leq e^{\epsilon} \times Pr[A(D') \in S]$.*

The output distribution of a differentially private algorithm is insensitive to small changes to the input database, where databases that only differ by the small change are called neighbours. For tabular data, the small change is defined as "differing by a row". In graphs, there are more options: (i) edge-DP considers pairs of graphs G and G' differing by an edge; (ii) node-DP considers pairs of graphs differing by a node, where two nodes are different if they have different edges. We assume that the total number of nodes are fixed under node-DP.

Node-DP offers a stronger privacy guarantee than Edge-DP but, when using the same algorithm, the queries will be answered more accurately under Edge-DP. We will illustrate the utility difference using the Laplace mechanism [13] for a degree distribution query q that takes in G and outputs a $|V|$-dimensional vector of counts. To

DP Guarantee	Queries	Algorithm and Short Name
Node	DD	Laplace (deg_his_lap) [19] Node truncation (node_trunc) [23] Max-flow (flowgraph) [33] Edge addition (θ-cumulative, θ-constrained, (θ,Ω)-histogram) [12]
Edge	SubG	Laplace [13] Smooth Sensitivity [22, 30] Recursive [9] Ladder [37]

Table 1: DP algorithms for releasing degree distribution (DD) and releasing sub-graph counting (SubG).

achieve ϵ-edge DP (or ϵ-node DP), the Laplace mechanism simply adds a noise vector drawn from the Laplace distribution with a mean value of 0 and standard deviation $\Delta(q)/\epsilon$, where $\Delta(q)$ is known as the global sensitivity of the query. For edge-DP, when adding or removing an edge, at most two nodes will change their degree values and thus four degree counts will be affected, i.e., $\Delta(q) = 4$. For node-DP, changing a node can affect the degree values of $O(|V|)$ number of nodes. Hence, a larger amount of noise will have to be added to $q(G)$ to ensure node-DP as against edge-DP.

Similar to degree distribution, many graph queries have a high sensitivity under node-DP. Subgraph-counting is an important class of join queries involving self-joins i.e. multiple joins on the same record set. Even for edge-DP, some subgraph-counting queries are highly sensitive. Dealing with these queries has been the focus of DP algorithm design on graphs. There are two general approaches to address this problem.

The first approach transforms a given graph G to a θ-bounded graph G^θ, where the maximum degree is bounded by θ, so that the global sensitivity of the query on the new graph G^θ is also bounded. A smaller noise is then added to the query answer on G^θ to ensure DP. Several graph transformations have been proposed in prior work including node truncation [23, 24], edge removal [6], edge addition [12], and Lipschitz extensions [33]. The final noisy answer has two types of errors: the bias in the approximate query answer on G^θ due to graph transformation and the noise added directly to the query answer. When θ is large, few nodes or edges will be removed, but there will be more noise added. When θ is small, the noise will be small but G^θ will be too different from G. The optimal degree bound θ that gives the smallest error needs to be learned privately. Therefore, the optimal end-to-end algorithm differs among datasets even under the same privacy guarantee.

The second approach designs algorithms based on local sensitivity which is the maximum possible change for a given graph instance instead of considering all possible instances. As local sensitivity still leaks information of a graph, a smooth bound is computed for the local sensitivity of all possible databases at a distance from the given database [9, 21, 23, 29, 37]. The computation of a tight smooth bound for arbitrary queries [9, 23, 29] is expensive or requires specialization [37], especially for node-DP. Zhang et al. [37] propose a class of functions called Ladder Functions to determine the level of noise to add to query outputs. A loose upper bound can be derived efficiently [21, 23], but may lead to large noise addition. Such trade-offs can only be examined by benchmarking the performance of these algorithms.

Graph	$\|V\|$	$\|E\|$	$d_{\max}$	c
Chesapeake	39	170	33	0.284
US airports	500	2980	145	0.6175
Facebook	4039	88234	1045	0.6055
Ca-GrQc	5242	14496	81	0.5296
P2P-Gnutella	6301	20777	95	0.0109
Cit-HepTh	27770	352807	2468	0.3120
DBLP	317080	1049866	343	0.6324

Table 2: Dataset properties, $|V|$, $|E|$, $\mathbf{d_{max}}$, and c denotes the number of nodes, edges, maximum degree, and clustering coefficient of a graph.

3 METHODOLOGY

This section describes the query classes, algorithms, datasets and methods used for evaluation.

Query Classes and Algorithms We evaluate two major query classes in this paper, degree distribution queries and sub-graph counting queries. We compare the algorithms in terms of accuracy and computational performance to determine the optimal choice for each query class.

- **Degree Distribution Queries** We benchmark several node-DP algorithms for answering degree distribution queries. The degree distribution describes the fraction of nodes with degree k in a graph. Table 1 summarizes the algorithms and their short names. The algorithms θ-cumulative and θ-constrained require an input Θ, which is the maximum degree bound that the algorithms will use to select an optimal θ for transforming the graph to be θ-bounded. These two algorithms differ in how they post-process the projected graph. (θ,Ω)-histogram requires an additional input $\mathbf{\Omega}$, a list of partition candidates. A partition is an aggregation structure which is used to group adjacent degree bins to reduce the added noise through averaging. The algorithm selects an optimal pair of θ and Ω values using a quality function. We use $\Theta = 150$ in our experiment and $\mathbf{\Omega}$ is generated using the geometric sequence where $r \in [1.025,1.05,1.1,1.2,1.4,1.8]$. We run these algorithms within the privacy guarantee range $\epsilon \in [0.01, 4]$;
- **Subgraph-Counting Queries** We have evaluated several algorithms for counting triangles, k-star and k-triangle subgraphs. Due to space constraints, we focus on triangle and 7-star (S_7) subgraph counting queries under edge-DP in this paper. Our evaluation results for all other graph-based differential privacy algorithms can be found within the DPGraph repository[36] (https://github.com/DPGraph/DPGraph). Table 1 shows the algorithms which we benchmark in this paper within the privacy guarantee range $\epsilon \in [0.01, 2]$.

Datasets We evaluate several real-world datasets, as shown in Table 2, with varying properties and from different domains. The datasets are unweighted graphs and range from 39 nodes with 170 edges to 317,080 nodes with 1,049,866 edges. We picked datasets of varying sizes and properties to get a better insight into the influence they might have on an algorithm.

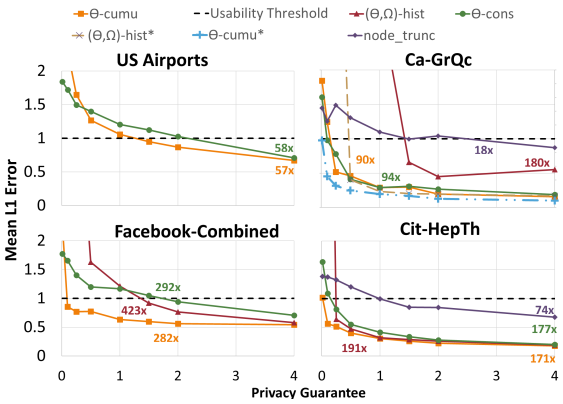

Figure 1: Utility vs Privacy Guarantee - Degree Distribution. Runtime performance is reported beside each algorithm. Refer to Section 4 for details

For example the Facebook dataset has a clustering coefficient of 0.6 which makes it very densely connected as opposed to P2P-Gnutella which is more sparsely connected with a clustering coefficient of 0.01. Smaller datasets like the US Airports dataset [11] are useful in studying algorithms with high computational costs such as the recursive algorithm. Most of our datasets can be obtained from the Stanford Large Network Dataset Collection [25].

Evaluation Methodology We examined the accuracy of each algorithm across a range of privacy guarantee levels. We ran multiple iterations at each privacy guarantee and repeated this for each of the datasets used in our evaluation. For degree distribution queries, we report the L1 difference between the noisy and true distribution. For sub-graph counting queries, we report the mean relative error which is the mean of the random variable $|A(g) - f(g)|/f(g)$ where $A(g)$ is the differentially private output and $f(g)$ is the true answer. We also measure the average running times across these runs and use them to compare algorithmic computational performance.

Evaluation Setup For our experiments, we have used a server with an Intel Xeon Silver CPU consisting of 10 cores and 20 hardware threads. The server had 48GB of memory available for use.

4 EVALUATION

In this section, we present our findings when examining degree distribution, triangle counting and S_7 sub-graph counting queries. In our utility evaluations, we compare all the algorithms against a "usability threshold", which is the performance offered by an algorithm that simply returns a value of zero regardless of privacy guarantee. This represents an error rate of 100%, below which query results are considered to be usable in previous literature [9, 22, 37].

While evaluating the computational performance of these algorithms, their running time is compared against that of an algorithm which reports the true query answers and thus, provides no differential privacy guarantees. We scale the performance of all the other algorithms against this baseline.

Figures 1, 2 and 3 show our results. The accuracies are represented using the plotted lines while the numbers on the chart represent the scaled computational performance of each algorithm.

4.1 Accuracy results

4.1.1 Degree Distribution. Figure 1 shows our evaluation results for degree distribution queries on a subset of graphs; the rest can be found on the DPGraph repository [36]. The Laplace mechanism [19] performs poorly, with an error that is 1.5x worse than just reporting 0 (usability threshold) at the largest $\epsilon = 4$. (θ, Ω)-histogram [12] on the Airport [11] graph has errors of at least 2.94. Hence we omit them from Figure 1. With a very small privacy guarantee, no node-DP algorithm beats the usability threshold; the error is at least 1.3 for all graphs when $\epsilon = 0.01$.

For $\epsilon \geq 0.5$, θ-cumulative [12] performs the best, reporting the lowest error among the algorithms as shown by Figure 1. Its utility depends on the choice of parameter Θ. The difference is clear in Figure 1 on the Ca-GrQc [25] graph, whose maximum degree is 81. At $\epsilon = 0.1$, the error drops from 1.25 to 0.45 when we change the default $\Theta = 150$ to 20. The result with the adjusted Θ is labeled as Θ-cumu* in Figure 1. θ-cumulative's performance also depends on the graph's properties. As $|V|$ increases, the algorithm's accuracy improves. We compare θ-cumulative's performance on the Facebook and DBLP graphs [25] as they possess similar clustering coefficients. For $\epsilon = 2$, the error is 0.56 and 0.025 on Facebook and DBLP respectively (not shown in the figure due to space limitations). In addition, node_trunc [23] becomes useful for larger plots. On Cit-HepTh [25], its error is less than 0.85 for $\epsilon \geq 1.5$ as shown in Figure 1; on the other smaller graphs, its performance does not pass the threshold. We only run the Flowgraph [33] algorithm on the smallest Chesapeake [4] dataset as it is computationally heavy to solve a max-flow problem with an objective function; for example, an iteration of Flowgraph on the 500-node Airport graph is 80000x more expensive than that of node_trunc. Its utility performance is also poor, reporting error of at least 1.5 for $\epsilon \in [0.01, 4]$.

(θ, Ω)-histogram's utility depends on the parameter choices Θ and Ω. On Ca-GrQc under the default setting, its performance does not pass the threshold for $\epsilon < 1.5$; we also observe an increase in error as ϵ increases from 2 to 4, with a high standard deviation. After adjusting Θ to 20 and r to [1.001,1.01,1.1,1.3,1.5] for a finer aggregation, the error decreases to less than 0.5 for all $\epsilon \geq 0.5$. The result with the adjusted parameters is labeled as (Θ, Ω)-hist* in Figure 1. In comparison, we note that θ-cumulative is less sensitive to a poor parameter choice than (θ, Ω)-histogram; as evidence, using the default $\Theta = 150$ on Ca-GrQc, its error is no greater than 0.5 for all $\epsilon \geq 0.1$.

4.1.2 Subgraph Counting. In this section, we present our evaluation results for triangle and S_7 counting queries. It is also important to note that we evaluate the Smooth Sensitivity mechanism by Nissim et al. [30] for triangle counting queries and use the extensions to this mechanism by Vishesh et al. [22] for S_7 queries.

Triangle Counting Queries. The mean relative errors for each algorithm across a range of privacy guarantees have been shown in Figure 2. The usability threshold represents an error of 100% above which an algorithm is considered to be unusable [22, 37].

From Figure 2, it is evident that Ladder Functions [37] provide the best accuracy across all datasets when compared to their counterparts. The Laplace [13] and Smooth mechanisms [30] may or may not be usable depending on the properties of the underlying data. For instance, in the Facebook dataset, both the Laplace [13]

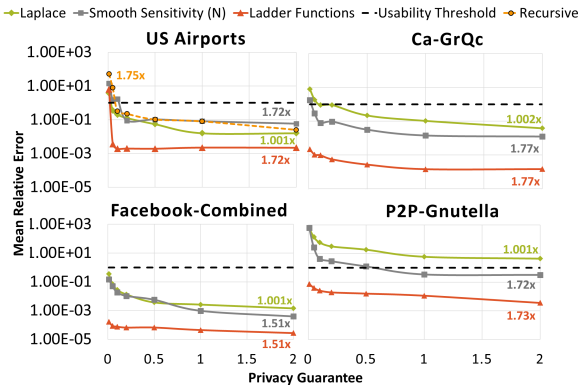

Figure 2: Utility vs Privacy Guarantee - Triangle Counting. Note that the Y-axis is plotted in log scale. Runtime performance is reported beside each algorithm. Refer to Section 4 for details.

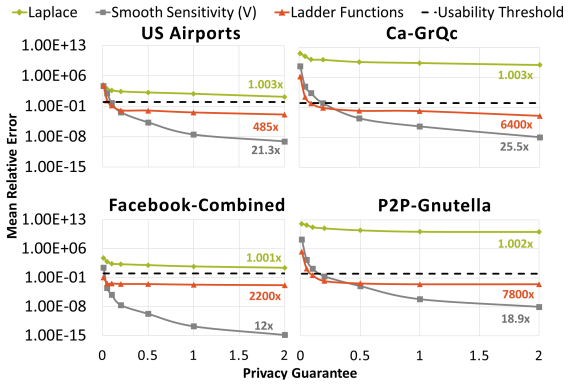

Figure 3: Utility vs Privacy Guarantee - S_7 Counting. Note that the Y-axis is plotted in log scale. Runtime performance is reported beside each algorithm. Refer to Section 4 for details.

and Smooth [30] mechanisms maintain a relative error far below the usability threshold at all privacy guarantees. However, when using the P2P-Gnutella [25] dataset, the Laplace [13] mechanism is unusable at all privacy levels while the Smooth [30] mechanism is only usable under relatively relaxed privacy guarantees.

The Recursive mechanism [9] has only been run on the smaller US Airports [11] dataset due to its performance complexity. The Recursive mechanism is outperformed by its counterparts on this dataset. This surprisingly includes the naive Laplace [13] mechanism (Figure 2). However, the small nature of the dataset may explain these results, as even the Smooth [30] mechanism performs significantly better when we move to larger datasets.

S_7 Counting Queries. Figure 3 demonstrates that the Smooth Sensitivity mechanism [22] outperforms Ladder Functions [37] at most privacy guarantee levels when running S_7 counting queries across a wide variety of datasets. Unlike in triangle counting queries, the Laplace mechanism [13] reports larger errors than the usability threshold for S_7 queries regardless of the amount of relaxation afforded by the privacy guarantee levels.

4.2 Performance Results

Algorithm running time is largely independent of privacy guarantee level. Hence, we report the performance at a single fixed privacy guarantee across all algorithms. All the algorithms' performances have been scaled against a baseline function which returns the true answer and hence does not provide any privacy guarantees. The performance of each algorithm is shown in Figures 1, 2 and 3 beside each algorithm.

4.2.1 Degree Distribution. We present the computational performance of the degree distribution algorithms including node_trunc, θ-cumulative, θ-constrained and (θ, Ω)-histogram. Among the algorithms that beat our utility threshold, node_trunc is the most efficient algorithm. It is a reasonable candidate for larger graphs under relatively relaxed privacy guarantees. In our experiment, θ-cumulative and θ-constrained rank second in run-time. Their runtime depends on the input Θ, as they need to compute the quality function for each of the θ candidates. Lastly, (θ, Ω)-histogram is most computationally heavy among the 4 useful algorithms in our experiment. Its performance depends on the input parameters Ω and Θ; as a result, it is slower than θ-cumulative and θ-constrained because it must choose the optimal pair of θ and Ω values.

4.2.2 Triangle counting queries. For triangle counting queries, we see that all the algorithms (except the Laplace Mechanism) have a 50-70% performance overhead when compared to a baseline without differential privacy. This adds another layer of complexity when determining the optimal algorithm for a given dataset. For example, while Ladder Functions [37] possess the best accuracy, a performance conscious analyst may opt to use the Laplace [13] mechanism if their data has properties similar to the Facebook [25] dataset. The reasonable error rates, coupled with the low performance overheads, make this algorithm an attractive option in such cases.

4.2.3 S_7 queries. When running S_7 counting queries, Figure 3 shows that the Laplace [13] mechanism still continues to maintain a performance similar to the baseline without differential privacy. However, there is a large performance difference between Ladder Functions [37] and Smooth Sensitivity [22]. When coupled with the accuracy, we observe that Smooth Sensitivity is the optimal algorithm when it comes to handling S_7 sub-graph counting queries.

5 CONCLUSION

In this work, we show that the choice of an optimal algorithm is dependent on the query class, privacy guarantee, algorithmic performance and properties of the underlying graph. This partially explains the lack of differentially private algorithms in real-world deployments despite a wealth of available choices.

Thus, there is a need for a standardized platform such as DP-Graph [36] to enable users to compare the utility, time complexity and privacy guarantee trade-offs of the various available algorithms, interactively run standardized implementations on a variety of datasets, and make informed choices tailored to their requirements. This work is a step towards that goal and we hope that it encourages the prevalent deployment of these algorithms.

REFERENCES

[1] N. R. Adam and J. C. Worthmann. Security-control methods for statistical databases: A comparative study. *ACM Comput. Surv.*, page 515–556, Dec. 1989.

[2] J. G. Anderson. Evaluation in health informatics: social network analysis. *Computers in biology and medicine*, 32(3):179—193, May 2002.

[3] M. Baglioni, S. Pieroni, F. Geraci, F. Mariani, S. Molinaro, M. Pellegrini, and E. Lastres. A new framework for distilling higher quality information from health data via social network analysis. In *2013 IEEE 13th International Conference on Data Mining Workshops*, pages 48–55, 2013.

[4] D. Baird and R. E. Ulanowicz. The seasonal dynamics of the chesapeake bay ecosystem. *Ecological monographs*, 59(4):329–364, 1989.

[5] M. Barbaro and T. Z. Jr. A face is exposed for aol searcher no. 4417749. *The New York Times*, 2006.

[6] J. Blocki, A. Blum, A. Datta, and O. Sheffet. Differentially private data analysis of social networks via restricted sensitivity. In *ITCS*, 2013.

[7] C. Borgs, J. T. Chayes, A. D. Smith, and I. Zadik. Revealing network structure, confidentially: Improved rates for node-private graphon estimation. In *FOCS*, 2018.

[8] P. J. Carrington, J. Scott, and S. Wasserman. *Models and Methods in Social Network Analysis*. Structural Analysis in the Social Sciences. Cambridge University Press, 2005.

[9] S. Chen and S. Zhou. Recursive mechanism: Towards node differential privacy and unrestricted joins. In *SIGMOD*, 2013.

[10] J. Cheng, A. W.-c. Fu, and J. Liu. K-isomorphism: Privacy preserving network publication against structural attacks. In *SIGMOD*, 2010.

[11] V. Colizza, R. Pastor-Satorras, and A. Vespignani. Reaction–diffusion processes and metapopulation models in heterogeneous networks. *Nature Physics*, 3(4):276–282, 2007.

[12] W.-Y. Day, N. Li, and M. Lyu. Publishing graph degree distribution with node differential privacy. In *SIGMOD*, 2016.

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

[14] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 2014.

[15] S. R. Ganta, S. P. Kasiviswanathan, and A. Smith. Composition attacks and auxiliary information in data privacy. In *KDD*, 2008.

[16] J. Gao, B. Song, Z. Chen, W. Ke, W. Ding, and X. Hu. Counter deanonymization query: H-index based k-anonymization privacy protection for social networks. In *SIGIR*, 2017.

[17] S. Garfinkel, J. M. Abowd, and C. Martindale. Understanding database reconstruction attacks on public data: These attacks on statistical databases are no longer a theoretical danger. *Queue*, page 28–53, Oct. 2018.

[18] A. Gupta, A. Roth, and J. Ullman. Iterative constructions and private data release. *TCC*, 2012.

[19] M. Hay, C. Li, G. Miklau, and D. Jensen. Accurate estimation of the degree distribution of private networks. In *ICDM*, 2009.

[20] M. Hay, G. Miklau, D. Jensen, D. Towsley, and P. Weis. Resisting structural re-identification in anonymized social networks. *VLDB*, 2008.

[21] N. Johnson, J. P. Near, and D. Song. Towards practical differential privacy for sql queries. *VLDB*, 2018.

[22] V. Karwa, S. Raskhodnikova, A. Smith, and G. Yaroslavtsev. Private analysis of graph structure. *PVLDB*, 4:1146–1157, 08 2011.

[23] S. P. Kasiviswanathan, K. Nissim, S. Raskhodnikova, and A. Smith. Analyzing graphs with node differential privacy. In *TCC*, 2013.

[24] I. Kotsogiannis, Y. Tao, X. He, M. Fanaeepour, A. Machanavajjhala, M. Hay, and G. Miklau. Privatesql: a differentially private sql query engine. *VLDB*, 2019.

[25] J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[26] M. Lyu, D. Su, and N. Li. Understanding the sparse vector technique for differential privacy. *PVLDB*, 2017.

[27] Z. Mao, H. Yao, Q. Zou, W. Zhang, and Y. Dong. Digital contact tracing based on a graph database algorithm for emergency management during the covid-19 epidemic: Case study. *JMIR mHealth and uHealth*, 9(1):e26836, 2021.

[28] I. Mironov. On significance of the least significant bits for differential privacy. In *CCS*, 2012.

[29] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *STOC*, 2007.

[30] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. page 75–84, 2007.

[31] K. P. Puttaswamy, A. Sala, and B. Y. Zhao. Starclique: Guaranteeing user privacy in social networks against intersection attacks. In *CoNEXT*, 2009.

[32] Z. Qin, T. Yu, Y. Yang, I. Khalil, X. Xiao, and K. Ren. Generating synthetic decentralized social graphs with local differential privacy. In *CCS*, 2017.

[33] S. Raskhodnikova and A. Smith. Lipschitz extensions for node-private graph statistics and the generalized exponential mechanism. In *FOCS*, 2016.

[34] V. Rastogi, M. Hay, G. Miklau, and D. Suciu. Relationship privacy: output perturbation for queries with joins. In *PODS*, 2009.

[35] R. C.-W. Wong, A. W.-C. Fu, K. Wang, and J. Pei. Minimality attack in privacy preserving data publishing. In *VLDB*, 2007.

[36] S. Xia, B. Chang, K. Knopf, Y. He, Y. Tao, and X. He. DPGraph: A Benchmark Platform for Differentially Private Graph Analysis. In *Proceedings of the 2021 International Conference on Management of Data*, pages 2808–2812, 2021.

[37] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. Private release of graph statistics using ladder functions. In *SIGMOD*, 2015.

[38] E. Zheleva and L. Getoor. Preserving the privacy of sensitive relationships in graph data. In *PinKDD*, 2008.