UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE CIÊNCIAS

DEPARTAMENTO DE COMPUTAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FELIPE TIMBÓ BRITO

DIFFERENTIALLY PRIVATE RELEASE OF COUNT-WEIGHTED GRAPHS

FORTALEZA

2023

FELIPE TIMBÓ BRITO

DIFFERENTIALLY PRIVATE RELEASE OF COUNT-WEIGHTED GRAPHS

Tese apresentada ao Programa de Pós-graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Ciência da Computação. Área de Concentração: Bancos de Dados

Orientador: Prof. Dr. Javam de Castro Machado

FORTALEZA

2023

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

B875d Brito, Felipe Timbó.
Differentially Private Release of Count-Weighted Graphs / Felipe Timbó Brito. – 2023.
102 f. : il. color.

Tese (doutorado) – Universidade Federal do Ceará, Centro de Ciências, Programa de Pós-Graduação em Ciência da Computação , Fortaleza, 2023.
Orientação: Prof. Dr. Javam de Castro Machado.

1. Differential Privacy. 2. Local-DP. 3. Count-Weighted Graphs. I. Título.

CDD 005

FELIPE TIMBÓ BRITO

DIFFERENTIALLY PRIVATE RELEASE OF COUNT-WEIGHTED GRAPHS

Tese apresentada ao Programa de Pós-graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Ciência da Computação. Área de Concentração: Bancos de Dados

Aprovada em: 03/08/2023

To my wife Isabelle and my daughter Sofia, I dedicate this thesis.

ACKNOWLEDGEMENTS

To God and Our Lady, who became my source of hope when the future seemed uncertain and provided me with the strength to overcome my own limitations and persist in the face of adversity.

To my parents, João and Fátima, who did everything they could to help me get into a public university and start my career.

To my wife Isabelle, for your love, your patience, for turning my dream into our dream, and my struggle into our struggle. Your constant support has always kept me on track during the last few years. Thank you for the things you renounced and for being a true pillar of our home.

To my beloved daughter Sofia, whose presence in my life has given new significance to love and served as a constant reminder of the true purpose behind my pursuit of this Ph.D. degree.

To my brother Fábio and my sister Fabricia for the friendship, affection, and love.

To my advisor and mentor, Prof. Javam de Castro Machado, for the numerous contributions and invaluable opportunities provided throughout this ten-year journey. It started in 2013 when I entered the master's program. Thank you for showing me a more pragmatic and straightforward approach to science.

To Divesh Srivastava for the exceptional collaboration during this period. It is more than a pleasure to work with someone who has such great expertise and knowledge. Thank you also for welcoming me to AT&T Labs in the US for one year.

To Prof. Cesar Lincoln for the contributions since my qualifying exam and for accepting to be in my thesis defense.

To Prof. Carlos Eduardo and Profa. Mirella Moro also for accepting to be part of this work.

To my colleagues at AT&T Labs, Cheryl Brooks and Subho Majumdar, for all the contributions to this work.

To Profa. Rosélia Machado, for the teachings and advice shared throughout this journey, which have made me reflect on many important decision-making moments.

To my friend Victor Farias for his companionship during my period in the US, for his contributions to this work, and mainly for listening to me in challenging times.

To my friend Felipe Monteiro for the exceptional working partnership and for the encouraging remarks that helped me to finish this dissertation.

To my childhood friends from the neighborhood where I grew up: Abnadab, Claudinho, Davi, Glaydson, Rafael, and Werley, who in various conversations cheered me up, advised me, and reminded me that life encompasses so much more than just work. Thank you for being true friends.

To my friends and research colleagues at LSBD: Iago C. Chaves, André Luis C. Mendonça, Eduardo Rodrigues D. Neto, Paulo R. P. Amora, Maria de Lourdes M. Silva, and Serafim S. Costa Filho.

To Laboratório de Sistemas e Bancos de Dados - LSBD, for the financial support provided during my PhD journey. I express my deep gratitude!

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance code 88881.189723/2018-01 and finance code 88882.454577/2018-01.

“Carry out the little duty of each moment: do what you ought and concentrate on what you are doing.”

(St. Josemaria Escriva)

RESUMO

Muitos sistemas complexos são comumente modelados como grafos ponderados de contagem, onde os nós representam entidades, as arestas modelam as relações entre eles e os pesos das arestas definem alguma estatística de contagem associada a cada relação. Como dados em formato de grafo geralmente contêm informações confidenciais, a preservação de privacidade no compartilhamento desse tipo de dado torna-se uma questão importante. Nesse contexto, a privacidade diferencial (PD) tornou-se o padrão para o compartilhamento de dados sob fortes garantias matemáticas. Ao lidar com PD para grafos ponderados, a maioria dos trabalhos recentes assumem que a topologia do grafo é conhecida. Entretanto, em diversas aplicações do mundo real, a privacidade da topologia do grafo também precisa ser assegurada. Nesta tese, pretendemos preencher a lacuna entre o compartilhamento de dados de grafos ponderados de contagem e privacidade diferencial, considerando tanto a estrutura do grafo quanto os pesos das arestas como informações privadas. Primeiro adaptamos a definição de PD em grafos ponderados para levar em consideração a privacidade da estrutura do grafo. Em seguida, introduzimos uma técnica escalável para adicionar aleatoriamente ruído aos pesos das arestas e à topologia do grafo. Também aproveitamos a propriedade de pós-processamento da PD para melhorar a utilidade dos dados, considerando restrições do domínio em grafos. Finalmente, essas contribuições combinadas são utilizadas como base para o desenvolvimento de duas novas abordagens para o compartilhamento privado de grafos ponderados de contagem sob as noções de PD global e local. Experimentos utilizando grafos do mundo real demonstram a superioridade das nossas abordagens em relação à utilidade sobre técnicas já existentes, permitindo a computação subsequente de uma variedade de estatísticas no grafo compartilhado com alta utilidade, em alguns casos comparáveis aos resultados originais.

Palavras-chave: privacidade diferencial; privacidade diferencial local; grafos ponderados de contagem.

ABSTRACT

Many complex systems are commonly modeled as count-weighted graphs, where nodes represent entities, edges model relationships between them and edge weights define some counting statistics associated with each relationship. As graph data usually contain sensitive information, preserving privacy when releasing this type of data becomes an important issue. In this context, differential privacy (DP) has become the de facto standard for data release under strong mathematical guarantees. When dealing with DP for weighted graphs, most state-of-the-art works assume that the graph topology is known. However, in several real-world applications, the privacy of the graph topology also needs to be ensured. In this dissertation, we aim to bridge the gap between DP and count-weighted graph data release, considering both graph structure and edge weights as private information. We first adapt the weighted graph DP definition to take into account the privacy of the graph structure. We then introduce a scalable technique to randomly add noise to the edge weights and to the graph topology. We also leverage the post-processing property of DP to improve the data utility, considering graph domain constraints. Finally, these combined contributions are used as the foundation for the development of two novel approaches to privately releasing count-weighted graphs under the notions of global and local DP. Experiments using real-world graph data demonstrate the superiority of our approaches in terms of utility over existing techniques, enabling subsequent computation of a variety of statistics on the released graph with high utility, in some cases comparable to the non-private results.

Keywords: differential privacy; local-DP; count-weighted graphs.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

CONTENTS

1 INTRODUCTION

Graphs, also called networks, are a fundamental tool for understanding the structure and behavior of complex systems (NEWMAN, 2003). Graphs provide a visual representation of the connections between the different components of a system. These connections are represented as nodes (also known as vertices) and edges, which characterize entities and relationships between them, respectively. The study of networks has applications in a wide range of disciplines, including physics, mathematics, computer science, biology, sociology, and economics (AGGARWAL *et al.*, 2010).

In this context, *count-weighted graphs* are a type of graph where each edge is associated with a weight that represents the number of times that edge appears in a dataset. Count-weighted graphs are useful for analyzing data where there are repeated instances of the same edge. By assigning a weight to each edge based on its frequency, count-weighted graphs can provide insights into the importance and frequency of connections between nodes in the graph. They also have been widely adopted to characterize complex systems in the real world, such as targeted marketing and advertising (LESKOVEC *et al.*, 2007), information campaigns through social media (FERRARA *et al.*, 2016; VAROL *et al.*, 2017), analysis of influential people and the interactions between them (CAMACHO *et al.*, 2020), propagation-based fake news detection (MATSUMOTO *et al.*, 2021), spread of epidemic disease (MANRÍQUEZ *et al.*, 2021), detection of social distancing measures violations (SUBAHI, 2021), among others.

For instance, consider a scenario where interactions are recorded by wearable sensors (HEIKENFELD *et al.*, 2018) in Figure 1. Figure 1a shows a table with interactions among users at specified timestamps. A count-weighted graph G (Figure 1b) is derived as the result of groupby/count($*$) in the original table, where nodes correspond to the users and edge weights are the counts of interactions of the users in contact. Specifically, the edge weights in G define the frequency associated with each relationship in the original data.

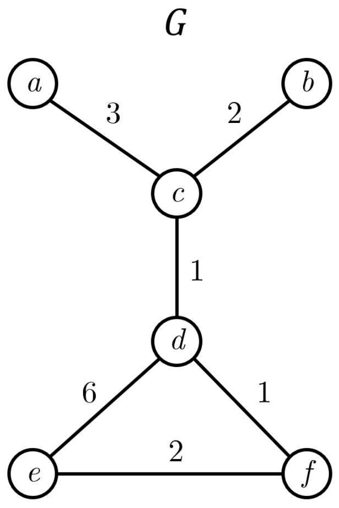
Because count-weighted networks usually contain sensitive information, releasing this type of data for analysis and statistical purposes without sufficient privacy guarantees may seriously jeopardize the individual's privacy. Current laws and regulations on data privacy, such as *General Data Protection Regulation (GPDR)* (European Commission, 2018) and *Lei Geral de Proteção de Dados Pessoais (LGPD)* (Brazil, 2018) require that individuals are no longer re-identifiable from released information.

The most straightforward technique to protect individual's privacy is to naively

Figure 1 – Example of interactions recorded by sensors.

Timestamp	User 1	User 2
2023-04-03 08:10:12	a	c
2023-04-03 09:23:57	b	c
2023-04-03 10:05:31	c	d
...	...	...
2023-04-04 08:34:23	a	c
2023-04-04 09:10:56	b	c
2023-04-04 09:27:44	a	c
...	...	...
2023-04-05 11:27:13	e	f

(a) Original data



(b) Count-weighted graph

Source: Elaborated by the author.

remove the IDs (labels) of users in the dataset. However, it has been shown that this process is vulnerable to linkage attacks (JI *et al.*, 2016). An illustrative example of privacy leakage when releasing count-weighted graphs is presented as follows.

Example 1. *Consider the count-weighted graph G in Figure 1b. Suppose an attacker knows the identity of user "e" in G and he/she is not sure whether "d" (say Doug) is the person he/she knows in the real world. Once the attacker gets the edge weights of this network, he/she can find there are two unknown people ("d" and "f") who interact with user "e" sometimes. Although the labels of all individuals have been removed, such information can still reveal the privacy of user "d" when publishing its edge weights. Let's say the attacker knows in the real world that Doug is the person with the most interaction with the known user "e" in this graph. Then, the attacker can easily infer that "d" is Doug.*

In the past, many privacy techniques were designed with their own requirements to protect individuals' privacy when their data are released. *K-anonymity* (SAMARATI; SWEENEY, 1998; SWEENEY, 2002) is one of the most well-known privacy models that consists of forming classes of k records. In a class, each record is identical to the other $k-1$ records. In other words, each record can not be linked to an individual with a probability lower than 1/k. From k-anonymity, other privacy models were proposed to avoid re-identification of individuals in released datasets, such as *l-diversity* (MACHANAVAJJHALA *et al.*, 2007), *t-closeness* (LI *et al.*, 2007; LI *et al.*, 2009), δ*-presence* (NERGIZ *et al.*, 2007), among others. However, all of these approaches assume that a malicious user has limited background knowledge, which is not true in real-world situations.

1.1 Differential Privacy

Differential privacy (DP) (DWORK, 2006) has emerged as the de facto standard notion of privacy for data release. It has been applied in industry (KENTHAPADI *et al.*, 2019) by companies such as Apple, Google, Uber (CORMODE *et al.*, 2018) and also in the public sector by U.S. agencies, such as the U.S. Census Bureau (HANEY *et al.*, 2017; ABOWD, 2018; GARFINKEL *et al.*, 2018). Differential privacy offers a formal definition of privacy with some interesting properties, such as no computational/informational assumptions about attackers, data type-agnosticism, and composability (MCSHERRY, 2009). The main idea behind DP is that a given query is answered by a *randomized algorithm* that queries the private information and returns a randomized answer sampled from an *output distribution*. A randomized algorithm is also referred to as a mechanism. A mechanism is differentially private if the probability distribution of the outputs does not change significantly based on the presence or absence of an individual.

There are two main types of differential privacy in the literature, global differential privacy (DWORK, 2006), and local differential privacy (LDP) (DUCHI *et al.*, 2013). Under global DP, there is a trusted curator that collects the original weighted graph and globally adds noise to achieve differential privacy, i.e., noise is added only once, at the end of the process before releasing the private graph with third parties. On the other hand, in the LDP setting, each user has a local noise-adding step before sending the perturbed data to an untrusted curator. Local differential privacy provides a higher level of privacy since the random perturbation is performed on the user side, not in the data curator. Whether global or local, differential privacy ensures statistical guarantees against the inference of private information through the use of auxiliary information, unlike the traditional privacy models. Because differential privacy is defined as independent of any auxiliary information assumption, it provides the most rigorous privacy guarantee for releasing count-weighted graphs.

Many efforts have been made towards differentially private publication of graph data. In this context, two main types of DP are particularly relevant: edge differential privacy (edge-DP) (HAY *et al.*, 2009) and node differential privacy (node-DP) (KASIVISWANATHAN *et al.*, 2013). The essential difference between them lies in the definition of neighboring graphs.

In the standard DP model, two databases are neighbors if they differ by at most one record. In the graph context, edge-DP describes two graphs as neighbors if they differ on a single edge. On the other hand, node-DP defines a pair of graphs to be neighbors if they differ by exactly one node and its incident edges. Intuitively, edge differential privacy ensures that the output of a DP algorithm does not reveal the inclusion or removal of a particular edge in the graph, while the node differential privacy hides the inclusion or removal of a node together with all its adjacent edges.

When graphs have weighted edges, neither edge-DP nor node-DP models offer appropriate privacy guarantees. Instead, Sealfon (SEALFON, 2016) introduced DP in the context of neighboring weight functions and proposed the notion of edge weight-DP. Under this model, two graphs are said to be neighbors if they have the same structure (topology) and similar weight functions. In other words, the graph topology is assumed to be public and the private information to protect are only the edge weights. This assumption was recently adopted by several authors in the literature (LI *et al.*, 2017; PINOT *et al.*, 2018; WANG; LONG, 2019; TONG *et al.*, 2019; FAN; LI, 2022).

Wang et al. (WANG *et al.*, 2020) and Ning et al. (NING *et al.*, 2021) proposed new DP techniques that assume the graph topology is unknown. Both works adopt the following strategy: first, they propose a method to protect the privacy of the graph topology and then they superimpose a vector perturbation for the weights on the resulting graph. That is to say, they compose two distinct notions of differential privacy for graphs: edge-DP to perturb the graph structure and edge weight-DP to disturb the values of the weights. However, the DP property satisfied by this composition would need to be established. Thus, no existing DP approach can correctly handle count-weighted graph release when the topology is unknown. It may be desirable in many applications, for example, when protecting the presence or absence of an interaction in a human contact network, or the existence or absence of phone calls, text messages or emails exchanged between people (see, e.g., the FCC regulations in the U.S. around customer proprietary network information (Federal Communications Commission, 2018)). In such cases, only considering the scenario where the graph structure is public is not effective to provide the desired privacy guarantees.

1.2 Problem Statement

In this thesis, we investigate the problem of releasing count-weighted graphs under differential privacy considering both graph topology and edge weights as private information. In particular, let $G = (V, E, \omega)$ be an undirected weighted graph with a vertex set V, an edge set E, and a weight function $\omega : V^2 \to \mathbb{Z}_{\geq 0}$ mapping connections between a pair of vertices (u, v) to integer weights. Because G is undirected, $\omega(u, v) = \omega(v, u)$ for every $(u, v) \in E$. We require consistency between E and ω, such that $\omega(u, v) > 0 \iff (u, v) \in E$. This means that edges in E must characterize non-zero weights, whereas the weight of a non-existing edge is considered to be zero by convention. We assume the set of vertices V are public, i.e., the data holder/curator can access the node labels of the graph. On the other hand, the presence or absence of the set of edges E is private, as well as their weights. Then, the objective of this thesis is to propose techniques to release a count-weighted graph version $\bar{G}$ with high utility under differential privacy guarantees.

1.3 Thesis Hypothesis

The main hypothesis we address in this thesis is stated as follows:

Main Hypothesis. *By adopting differential privacy, it is possible to privately release count-weighted graphs and, at the same time, enable subsequent computation of a variety of statistics on the released graph with high utility.*

We aim to release count-weighted graphs for analysis and statistical purposes with DP while preserving as much as possible the characteristics of the original graph. We investigate our central hypothesis by pursuing three specific goals, driven by the following research questions:
Research Question 1 (RQ1): *How to establish a new definition of neighboring graphs considering both graph topology and edge weights as private information?*

Existing DP techniques assume the graph topology is known. Consequently, the perturbation only occurs in the edge weights. This process may incur excessive perturbation of the weight values since the noise added is proportional to the sensitivity (Section 2.4) and, in this case, the sensitivity is comparable to the maximum edge weight. Additionally, other differentially private methods compose two distinct notions of differential privacy for graphs but do not introduce the DP property satisfied by this composition. Therefore, a new definition of neighboring graphs needs to be introduced to consider both graph topology and edge weights as private information.

Research Question 2 (RQ2): *How can we provide a scalable graph perturbation solution?*

We can represent a weighted graph as a weighted matrix and add noise to both zero and non-zero edge weights in the matrix. The noises added in this strategy have a lower magnitude. However, the computation cost to calculate noisy weights in the entire matrix is computationally expensive $O(|V|^2)$, making this approach efficient only for small to medium-sized graphs. In this way, a sampling technique may be adopted to avoid materializing all the edge weights.

Research Question 3 (RQ3): *How to keep the graph consistent and avoid introducing bias in the edge weights after adopting a DP technique to perturb the graph structure?*

As mentioned before, differential privacy adds noise to query results before releasing their outputs. However, this process may affect some domain constraints. For example, when querying node degrees, a differentially private algorithm must be consistent with the fact that noisy outputs assume positive integer values. The addition of independent noise, contrarily, cannot ensure such domain constraint. In addition, after adding noise to the weights, a dense graph may be produced, i.e., a graph perturbation technique can switch many zero-weighted edges to positive-weighted edges. If we naively clip all noisy weights below a threshold in this dense graph, we may introduce too much bias to make the released graph useful. Hence, adjustment steps may be applied to guarantee the consistency of the graph after perturbation and improve its utility.

1.4 Contributions

To address the mentioned concerns, we propose two main approaches under the notions of global and local differential privacy. In our proposed global approach, first, we privately compute important statistics that provide a good characterization of the original graph, like node degrees and the sum of all edge weights. Such graph statistics are utilized in the next steps to improve the accuracy of the released graph. We then adopt a sampling strategy to perturb the original weighted graph and override the costly operation of materializing all the edge weights, avoiding scalability problems. Finally, we then include two adjustment steps to recover as much as possible of the original graph characteristics before its release. Our local approach works similarly to the global one. The difference is that the perturbation and the adjustment steps are performed by each user. Additional steps to guarantee the consistency of the released graph are included in the curator side.

In particular, the contributions of this thesis are as follows.

1. We introduce a new *neighboring weight graphs* definition to release weighted graphs via differential privacy such that both the graph topology and the edge weights are assumed to be private.
2. We propose a scalable technique using sampling to randomly add noise to the graph topology based on its number of edges and on its edge weights.
3. We improve the accuracy of the perturbed graph by developing post-processing techniques to preserve as much as possible the original node degrees and the sum of all edge weights.
4. We introduce a global DP approach using the mentioned contributions and extend it to propose a new method to release count-weighted graphs under the local differential privacy setting, where the random perturbation is performed on the user side.
5. We conduct extensive experimental analysis on four real-world weighted graphs to evaluate the performance of the proposed approaches. We show that our two techniques outperform state-of-the-art methods and achieve high utility for a variety of statistics on the released graph. In some cases, the results are very close to the non-private version.

The main contributions described in this thesis are also presented in the following published papers:

- Felipe T. Brito, Victor A. E. Farias, Cheryl Flynn, Subhabrata Majumdar, Javam C. Machado, Divesh Srivastava. Global and Local Differentially Private Release of Count-Weighted Graphs. *Proceedings of the ACM on Management of Data* (BRITO *et al.*, 2023).
- André L. C. Mendonça, Felipe T. Brito, Javam C. Machado. Privacy-Preserving Techniques for Social Network Analysis. *Anais Estendidos do XXXVIII Simpósio Brasileiro de Bancos de Dados* (MENDONÇA *et al.*, 2023).
- Victor A. E. Farias, Felipe T. Brito, Cheryl Flynn, Javam C. Machado, Subhabrata Majumdar, Divesh Srivastava. Local Dampening: Differential Privacy for Non-numeric Queries via Local Sensitivity. *The VLDB Journal* (FARIAS *et al.*, 2023).
- Felipe C. Monteiro, Felipe T. Brito, Iago C. Chaves, Javam C. Machado. Compartilhamento de Dados de Tráfego de Rede Utilizando Privacidade Diferencial. *Anais do L Seminário Integrado de Software e Hardware* (MONTEIRO *et al.*, 2023).
- Victor A. E. Farias, Felipe T. Brito, Cheryl Flynn, Javam C. Machado, Subhabrata Majum-

dar, Divesh Srivastava. Local Dampening: Differential Privacy for Non-numeric Queries via Local Sensitivity. *Proceedings of the VLDB Endowment* (FARIAS *et al.*, 2020).
- Bruno C. Leal, Israel C. Vidal, Felipe T. Brito, Juvêncio S. Nobre, Javam C. Machado. δ-DOCA: Achieving Privacy in Data Streams. *International Workshop on Data Privacy Management* (LEAL *et al.*, 2018).
- Eduardo R. D. Neto, André L. C. Mendonça, Felipe T. Brito, Javam C. Machado. PrivLBS: Uma Abordagem para Preservação de Privacidade de Dados em Serviços Baseados em Localização. *Anais do XXXIII Simpósio Brasileiro de Banco de Dados.* (NETO *et al.*, 2018).
- Rôney Reis C. Silva, Bruno C. Leal, Felipe T. Brito, Vânia M. P. Vidal, Javam C. Machado. A Differentially Private Approach for Querying RDF Data of Social Networks. *Proceedings of the 21st International Database Engineering & Applications Symposium.* (SILVA *et al.*, 2017).
- André L. C. Mendonça, Felipe T. Brito, Leonardo S. Linhares, Javam C. Machado. DiPCoDing: a Differentially Private Approach for Correlated Data with Clustering. *Proceedings of the 21st International Database Engineering & Applications Symposium.* (MENDONÇA *et al.*, 2017).
- Felipe T. Brito, Javam C. Machado. Preservação de Privacidade de Dados: Fundamentos, Técnicas e Aplicações. *Jornadas de atualização em informática* (BRITO; MACHADO, 2017).

1.5 Thesis organization

The rest of this thesis is organized as follows. Chapter 2 presents an overview of DP and introduces a new definition to consider graph topology as private information. Chapter 3 summarizes the literature review on DP for graphs. Chapter 4 introduces our new *neighboring weight graphs* definition to release weighted graphs via differential privacy. In this chapter, we also demonstrate how to create a noisy count-weighted graph via differential privacy and present our post-processing techniques that are used to improve the accuracy of the released graph. In Chapter 5 we introduce our global approach to privately release count-weighted graphs and describe our local method for the graph release problem. Experimental results are given in Chapter 6. Finally, Chapter 7 concludes the dissertation and points out future work directions.

2 DIFFERENTIAL PRIVACY

In this chapter, we describe the main concepts of differential privacy that compose this thesis, highlighting its key principles and core components.

2.1 Motivation

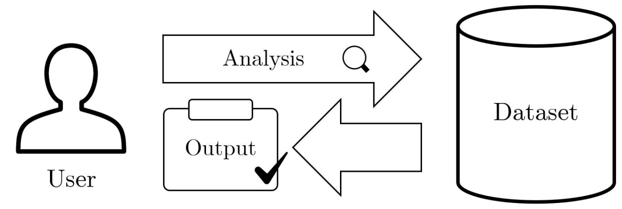
Re-identification attacks, i.e., attacks that aim to infer characteristics of individuals based on information about them in the data, have become sophisticated over time, posing a challenge to the effectiveness of privacy protection techniques. Successful attacks (GANTA *et al.*, 2008; CORMODE *et al.*, 2010; JIN *et al.*, 2010; WONG *et al.*, 2011) on de-identified data have shown that traditional privacy models, such as *k-anonymity*, *l-diversity*, *t-closeness* and *δ-presence*, are vulnerable to several attacks that aim to compromise data privacy.

Because traditional approaches have demonstrated evidence of weaknesses, a new privacy paradigm has emerged: differential privacy. First presented in 2006, differential privacy is a formal privacy model originally designed for use on raw data to provide robust privacy guarantees without depending on an adversary's background knowledge. Differential privacy is not a single tool, but rather a paradigm that quantifies and manages privacy violation risks. DP can be adopted from simple statistical estimations to machine learning (ZHU *et al.*, 2020).

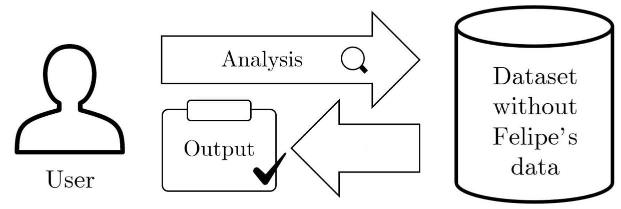
In this context, consider a user who wants to analyze data containing personal information about individuals. A few examples of users can be researchers, data analysts, data scientists, and application developers. The analysis performed by these users can range from simple calculations, such as computing the average age of individuals in the data, to more complex tasks that involve employing sophisticated modeling and inference techniques. Regardless of the complexity of the analysis, the fundamental process involves executing a computation on the input data and generating an output result. This process is illustrated in Figure 2a and referred to as a real-world setting.

In essence, an analysis protects the privacy of individuals within the data when its output avoids disclosing any information regarding any particular individual. This intuition is formalized by differential privacy as a mathematical definition. Consequently, DP can be used to address the scenario where a specific individual is not in the dataset. Consider the *Felipe's opt-out setting* illustrated in Figure 2b, where the analysis is conducted in the dataset but the information about Felipe is omitted. Then, his privacy is guaranteed in this hypothetical setting.

Figure 2 – An example of analysis (computation) on the input data generating an output result.



(a) Real-world setting.



(b) Felipe's opt-out setting.

Source: Elaborated by the author.

Note that the real-world setting involves some potential risk to Felipe's privacy. Since Felipe's information is utilized as input for the analysis, there is a possibility that personal details about him may be exposed in the resulting output. Therefore, differential privacy is designed to protect Felipe's privacy in the real-world setting in a way that emulates the level of privacy protection he would experience in his opt-out setting (WOOD *et al.*, 2018).

2.2 Differential Privacy Definition

In the original definition of differential privacy, private data is viewed as a collection of records, with each record corresponding to an individual. In essence, differential privacy promises privacy protection by injecting noise into these records, i.e., modifying the original data by introducing randomness (DWORK *et al.*, 2006). Let Q be a query function (analysis or computation) to be evaluated on a dataset D, which holds sensitive information about a set of individuals. DP is defined by considering a randomized algorithm $\mathscr{A}$ that operates on D. It ensures that the output of $\mathscr{A}(D)$ should be similar to $Q(D)$, i.e., with a controlled amount of random noise added. In other words, the goal of differential privacy is to make $\mathscr{A}(D)$ close to $Q(D)$ as much as possible to ensure data utility, and at the same time $\mathscr{A}(D)$ should preserve the privacy of all the records in the dataset.

In order for the randomized algorithm $\mathscr{A}$ to preserve the privacy of all the records in the dataset, DP establishes the notion of *neighboring datasets*. It is similar to Felipe's opt-out setting. Two datasets (or databases) D and D' are neighboring if they differ on at most one tuple,

Figure 3 – Example of two neighboring datasets of interactions between two users.

$D =$

Timestamp	User 1	User 2
2023-04-03 08:10:12	a	c
2023-04-03 09:23:57	b	c
2023-04-03 10:05:31	c	d
2023-04-04 08:34:23	a	c
2023-04-04 09:10:56	b	c
2023-04-04 09:27:44	a	c

$D' =$

Timestamp	User 1	User 2
2023-04-03 08:10:12	a	c
2023-04-03 09:23:57	b	c
2023-04-03 10:05:31	c	d
2023-04-04 08:34:23	a	c
2023-04-04 09:10:56	b	c

Source: Elaborated by the author.

denoted as $D \sim D'$. It is important to mention that the neighboring relation is symmetric, i.e., $D \sim D'$ is equivalent to $D' \sim D$. Figure 3 illustrates an example of two neighboring datasets of interactions between two users.

Differential privacy also requires that whether the input is D or D', the probability of a given output is nearly the same. This is also denoted as the indistinguishability of neighboring databases. Intuitively, it states that any answer to a query occurs with similar probability regardless of the presence or absence of any individual in the database.

More formally, for algorithm $\mathscr{A}$, let $Range(\mathscr{A})$ be the set of possible outputs of $\mathscr{A}$. For instance, if $\mathscr{A}$ computes the number of records in a dataset, then $Range(\mathscr{A})$ is equal to the set of non-negative integers. Then, the definition of differential privacy is stated below:

Definition 1. *(ε-Differential Privacy (DWORK, 2006; DWORK et al., 2006)). A randomized algorithm $\mathscr{A}$ satisfies ε-differential privacy, if for any two neighboring datasets D and D', and for any possible output $O \subseteq Range(\mathscr{A})$,*

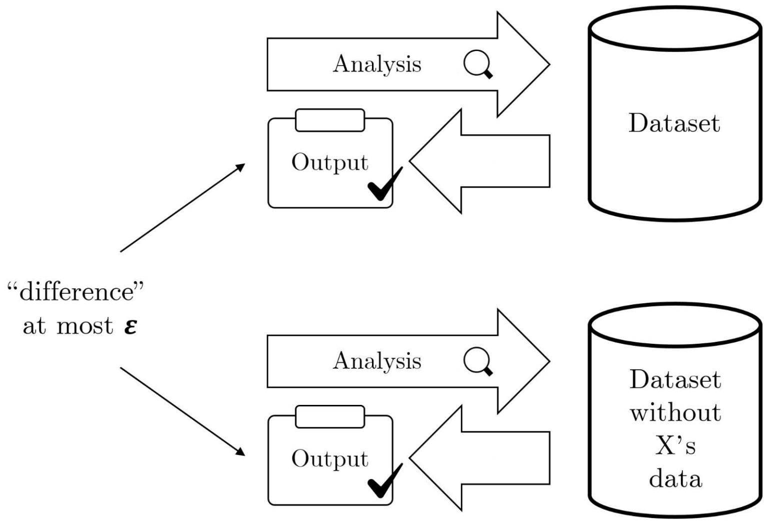
$$Pr[\mathscr{A}(D) = O] \leq \exp(\varepsilon) Pr[\mathscr{A}(D') = O], \tag{2.1}$$

where $Pr[\cdot]$ denotes the probability of an event.

2.3 Privacy Budget

The parameter ε in Definition 1 is denoted as *privacy budget*. As discussed before, differential privacy requires that the output of the analysis remain approximately the same, regardless of the presence or absence of any individual in the dataset. That is, differential privacy permits a minor change between the output of the real-world analysis and that of each individual's opt-out setting. This minor change is controlled by the privacy budget ε, as shown in Figure 4. Note that, in Figure 4, Felipe's data was replaced by X's data to emphasize that DP is made simultaneously to all individuals.

Figure 4 – The privacy budget control.



Source: Elaborated by the author.

The privacy budget is often a positive real number that controls the level of privacy preservation algorithm $\mathscr{A}$ can provide. A smaller ε corresponds to a stronger privacy-preserving guarantee. In contrast, a smaller ε also provides lower data utility, since more noise has to be added. Specifically, when $\varepsilon = 0$, the level of privacy preservation reaches the maximum, i.e., the real-world differentially private analysis replicates the opt-out setting of all individuals perfectly. In this case, the randomized algorithm $\mathscr{A}$ outputs two outcomes with indistinguishable distributions. However, the corresponding outputs do not provide any meaningful information about the dataset.

The setting of ε is a matter of policy, but typically it assumes a "small" value, making the probability "almost the same" whether the input is D or D'. In this context, the problem of choosing ε has been studied in the literature (HSU *et al.*, 2014; LI *et al.*, 2016). It has been argued that setting $\varepsilon = 0.1$ provides quite strong privacy protection, and using $\varepsilon = 1.0$ is also acceptable in many applications (LEE; CLIFTON, 2011; LI *et al.*, 2016). Setting a privacy budget as large as $\varepsilon = 5.0$ is adequate only in some cases, for instance, in location-based personalized services (HONG *et al.*, 2021) or in machine learning algorithms (BAGDASARYAN *et al.*, 2019).

Nevertheless, to define an adequate privacy budget for an application, experts, stakeholders, and data users have to provide extensive feedback to ensure that the privacy of individuals is sufficiently protected while maintaining high levels of accuracy in the released information. That is the case of the U.S. Census Bureau (United States Census Bureau, 2021). After reviewing feedback from the data user community, the U.S. Census Bureau's Data Stewardship Executive Policy Committee (DSEP) selected the settings and parameters for the 2020 Census and approved a total privacy budget of $\varepsilon = 19.61$. Therefore, it is still challenging to properly set the privacy budget in differential privacy.

2.4 Sensitivity

As previously discussed, differential privacy can be achieved by adding an appropriate amount of noise to query (analysis) results. Adding excessive noise may hurt data utility by distorting the accuracy of analysis results, while insufficient noise fails to provide adequate privacy guarantees. The noise added depends on the *global sensitivity* of a query Q (DWORK *et al.*, 2006), as defined next:

Definition 2. *(global sensitivity (DWORK et al., 2006)). The global sensitivity of a query Q is the maximum ℓ_1 distance between the outputs of Q on any two neighboring datasets.*

$$\Delta Q = \max_{D,D'} \parallel Q(D) - Q(D') \parallel_1 \ . \tag{2.2}$$

The global sensitivity, or simply sensitivity, measures the maximum impact on query results resulting from the addition or deletion of any record in the dataset. It serves as a crucial parameter to determine the appropriate magnitude of the added noise. Additionally, it is related only to the query function and is independent of the dataset. For a query function with low global sensitivity, only a minimal amount of noise needs to be added to mask the impact on query results caused by changing one single record. However, when the global sensitivity is high, a significant level of noise must be added to the output to ensure the privacy guarantee, which compromises the data utility.

For some query functions, the global sensitivity is easy to compute. For instance, the global sensitivity for counting queries is 1, since adding or removing one record affects the output of this query by at most 1. On the other hand, sensitivity for summation queries is not as simple as it is for counting queries. Suppose we want to query the sum of the ages of people in a given dataset. The inclusion of a new register in the dataset will increase the result of this query by the age of the additional individual. That implies the sensitivity, in this case, depends on the content (age) of the person added. Then, it is desired to assign a specific value to represent the sensitivity of this query. In this particular domain of age, there exists a reasonable and rational upper limit on the maximum age that an individual can be. It is plausible to assign, for example, the sensitivity is 122 since the oldest person ever lived to be 122 years old (GIBBS; ZAK, 2023). However, this does not serve as definitive evidence that no individual will ever live to the age of 122. Therefore, in some domains, it can be harder to come up with a reasonable global sensitivity (NEAR, JOSEPH P. and ABUAH, CHIKÉ, 2023).

2.5 Differentially Private Mechanisms

A randomized algorithm $\mathscr{A}$ is also referred to as a mechanism. Mechanisms are ways of achieving differential privacy. For numeric queries, DP can be achieved by mechanisms such as the Laplace mechanism (DWORK, 2006), the geometric mechanism (GHOSH *et al.*, 2012), and the log-laplace mechanism (NY; PAPPAS, 2013). They describe what kind of noise to use and how much to add.

2.5.1 Laplace Mechanism

The Laplace mechanism is a widely used algorithm to satisfy differential privacy. As the name of the mechanism suggests, it relies on the Laplace distribution to generate random values that are added to the true query response. Let x be the noise added to the output of a query function Q.

Definition 3. *(Laplace distribution) The Laplace distribution (centered at 0) with scale b is the distribution with probability density function:*

$$Lap(x|b) = \frac{1}{2b} exp\left(-\frac{|x|}{b}\right). \tag{2.3}$$

Denote $Lap(b)$ the Laplace distribution with scale b. Then, the Laplace mechanism (DWORK *et al.*, 2006) simply compute Q, and perturb the result with noise drawn from the Laplace distribution. The scale of the noise is calibrated by the sensitivity ΔQ divided by ε.

Theorem 1. *(Laplace mechanism (DWORK et al., 2006)) The Laplace mechanism that adds noise drawn from Lap($\Delta Q/\varepsilon$) satisfies ε-DP.*

2.5.2 Geometric Mechanism

The geometric mechanism (GHOSH *et al.*, 2012) is the discrete version of the Laplace mechanism. Contrary to the previous one, which adds real-valued noise to query results, the geometric mechanism adds integer noise from the two-sided geometric distribution to achieve differential privacy. Because of that, the output value is guaranteed to be an integer. Another difference is that while the Laplace mechanism provides privacy guarantees for a wide range of query types, the geometric mechanism specifically focuses on protecting count queries. This specialization enables the geometric mechanism to improve performance on count queries compared to the more general Laplace mechanism (GHOSH *et al.*, 2012).

In this thesis, we focus on releasing count-weighted graphs, such as those that are formed by groupby/count($*$) queries, as shown in Figure 1. Consequently, we adopt the geometric mechanism to compute counting queries.

Definition 4. *(two-sided geometric distribution) A random variable X distributed as a two-sided geometric distribution has probability mass function:*

$$P(X = x) = \frac{1-\alpha}{1+\alpha}\alpha^{|x|}\,, \tag{2.4}$$

where $0 \leq \alpha \leq 1$.

In particular, when $\alpha = e^{-\varepsilon/\Delta_Q}$, this mechanism is differentially private.

Theorem 2. *(geometric mechanism (GHOSH et al., 2012)) The geometric mechanism that adds independent noise from the two-sided geometric distribution, with* $\alpha = e^{-\varepsilon/\Delta_Q}$ *satisfies* ε*-DP.*

2.5.3 Log-Laplace Mechanism

The mechanisms presented previously are able to add both positive and negative noise since the distribution of these mechanisms ranges in the interval $[-\infty, \infty]$. However, for some queries, it is desired the output is positive. For instance, when counting the number of interactions in a dataset, as illustrated in Figure 1a, it is not meaningful for the result to be negative. To address this constraint, the log-Laplace mechanism (NY; PAPPAS, 2013) can be useful to maintain sign consistency between an original real-valued query and the result provided by the mechanism.

Let Q be a positive valued query. It is assumed there is a $K > 0$ such that for all $D \sim D'$:

$$\frac{Q(D) - Q(D')}{min\{Q(D) - Q(D')\}} \leq K\,. \tag{2.5}$$

Hence the mechanism $M(D) = log(Q(D))$ + Lap(b), where Lap(b) is a Laplace random variable with mean 0 and $b = \frac{K}{\varepsilon}$ is also differentially private.

Theorem 3. *(log-Laplace mechanism (NY; PAPPAS, 2013)) The log-Laplace mechanism that adds noise to the logarithm of Q(D), drawn from Lap(K/ε), satisfies ε-DP.*

2.5.4 Exponential Mechanism

Not all queries return numerical values as their output. Hence, McSherry and Talwar (MCSHERRY; TALWAR, 2007) have proposed the exponential mechanism (EM) to guarantee differential privacy for categorical queries. Its main idea is to sample an output O from the output space $\mathscr{O}$ according to a utility function u. This function assigns exponentially greater probabilities to outputs of higher utilities. The choice of u is application-dependent and different applications lead to distinct utility functions.

Definition 5. *(sensitivity of the utility function (MCSHERRY; TALWAR, 2007)). The global sensitivity of u is given by:*

$$\Delta u = \max_{O \in \mathscr{O}} \max_{D,D' neighbors} |u(D,O) - u(D',O)| \tag{2.6}$$

Theorem 4. *(exponential mechanism (MCSHERRY; TALWAR, 2007)). Given an utility function $u : (D \times \mathscr{O}) \to \mathbb{Z}$ for a dataset D, the mechanism $\mathscr{A}$ that samples an output $O \in \mathscr{O}$ with probability proportional to $exp(\frac{\varepsilon.u(D,O)}{2\Delta u})$ satisfies ε-DP.*

2.6 Differential Privacy Properties

Differentially private algorithms exhibit useful properties of post-processing and sequential composition. In post-processing, any function applied on the output of a DP algorithm also satisfies DP. Sequential composition relies on the fact that any sequence of randomized algorithms that provides DP in isolation, also provides DP in sequence. Both properties are formally stated as follows:

Theorem 5. *(post-processing (DWORK et al., 2006)) Let $\mathscr{A}$ be any randomized algorithm such that $\mathscr{A}(D)$ is ε-differentially private, and let f be any function. Then, $f(\mathscr{A}(D))$ also satisfies ε-DP.*

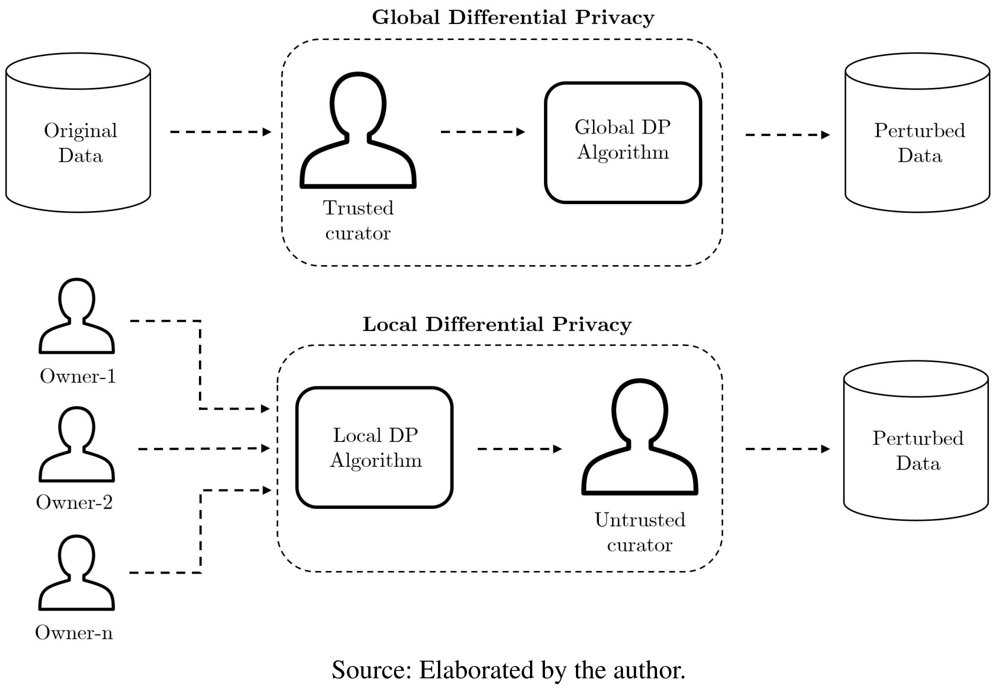
Theorem 6. *(sequential composition (MCSHERRY, 2009)) Let each $\mathscr{A}_i$ provide ε_i-differential privacy. A sequence of differentially private algorithms $\mathscr{A}_i(D)$ provides $\sum \varepsilon_i$-DP.*

When designing a differentially private technique, the aforementioned properties provide the flexibility to easily combine several DP steps into a DP mechanism.

2.7 Local Differential Privacy

The basic (global) setup of differential privacy involves a trusted curator (third party) that has access to the original data and globally adds noise to achieve differential privacy. However, finding a genuinely trusted third party for data collection and processing can be challenging in practical scenarios. The lack of trusted data curators restricts the applicability of the centralized differential privacy approach. To address this concern, local differential privacy (LDP) emerges as an alternative approach that eliminates the need for a trusted data curator (DUCHI *et al.*, 2013). LDP assumes each user does not trust the third party. Consequently, each user locally perturbs its data with a differentially private mechanism before sending it to the data curator. Figure 5 illustrates the difference between global and local DP settings.

Figure 5 – Distinction between global and local configurations for differential privacy.



Source: Elaborated by the author.

Compared to global DP, local differential privacy is a stronger notion of privacy that keeps individuals' sensitive information private even from untrusted data curators. Companies, especially those handling sensitive client information, are often reluctant to store real data due to privacy and security reasons. Storing actual client data poses significant risks in the event of a security breach or unauthorized access. LDP provides a solution by allowing analysis and extraction of valuable insights from the data without exposing the raw, identifiable information (DRECHSLER, 2023).

The formal definition of LDP is stated as:

Definition 6. *(ε-local differential privacy (DUCHI et al., 2013)). A randomized algorithm $\mathscr{A}$ satisfies ε-local differential privacy, if for any pair of input values $v, v' \in D$, and for any possible output $O \subseteq Range(\mathscr{A})$,*

$$Pr[\mathscr{A}(v) = O] \leq \exp(\varepsilon) Pr[\mathscr{A}(v') = O]. \tag{2.7}$$

The key difference between LDP and global DP is that a DP mechanism takes all users' data D as input and requires the output to be indistinguishable, while local DP takes only one user's data v as input and generates noisy responses per user (locally).

2.8 Differential Privacy for Graphs

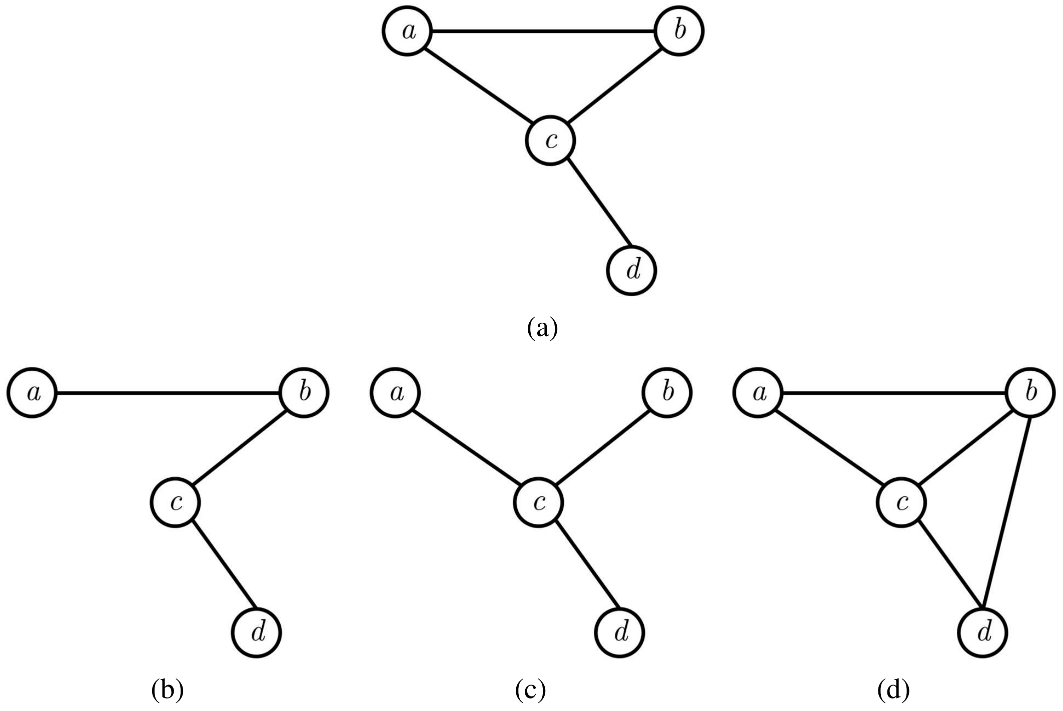
The fundamental concept of differential privacy relies on the definition of neighboring datasets. In previous definitions, a neighboring dataset is defined as a dataset obtained by adding or removing a single record. In the context of graph data, which primarily focuses on the relationships between individuals, the association between private data and dataset records becomes less clear.

To apply differential privacy to graphs, it is necessary to establish a definition for neighboring graphs and understand the privacy semantics associated with each choice. Consider $G = (V, E)$ to be a graph with a vertex set V and an edge set E. Edges are undirected pairs (u, v) such that $u, v \in V$. In this context, there are two main settings of neighboring graphs: edge differential privacy (edge-DP) (HAY *et al.*, 2009) and node differential privacy (node-DP) (KASIVISWANATHAN *et al.*, 2013).

2.8.1 Edge Differential Privacy

The initial adaptation of differential privacy to graphs follows a mathematical formulation similar to the definition used for tabular data. Neighboring graphs are defined as graphs that differ by a single "record". Then, one can produce a neighboring graph G' from G by either adding/removing an edge in E, or by adding/removing an isolated node in V. Figure 6 shows an example of neighboring graphs in the edge differential privacy setting.

Figure 6 – An example of three neighboring graphs in the edge differential privacy setting. (a) original graph. (b) neighboring graph by removing edge (a,c). (c) neighboring graph by removing edge (a,b). (d) neighboring graph by adding a new edge (b,d).



Source: Elaborated by the author.

Definition 7. *(ε-edge differential privacy (HAY et al., 2009)). A randomized algorithm $\mathscr{A}$ satisfies ε-edge differential privacy, if for any pair of graphs $G = (V,E)$ and $G' = (V',E')$, such that $|V \oplus V'| + |E \oplus E'| = 1$ and for any possible output $O \subseteq Range(\mathscr{A})$,*

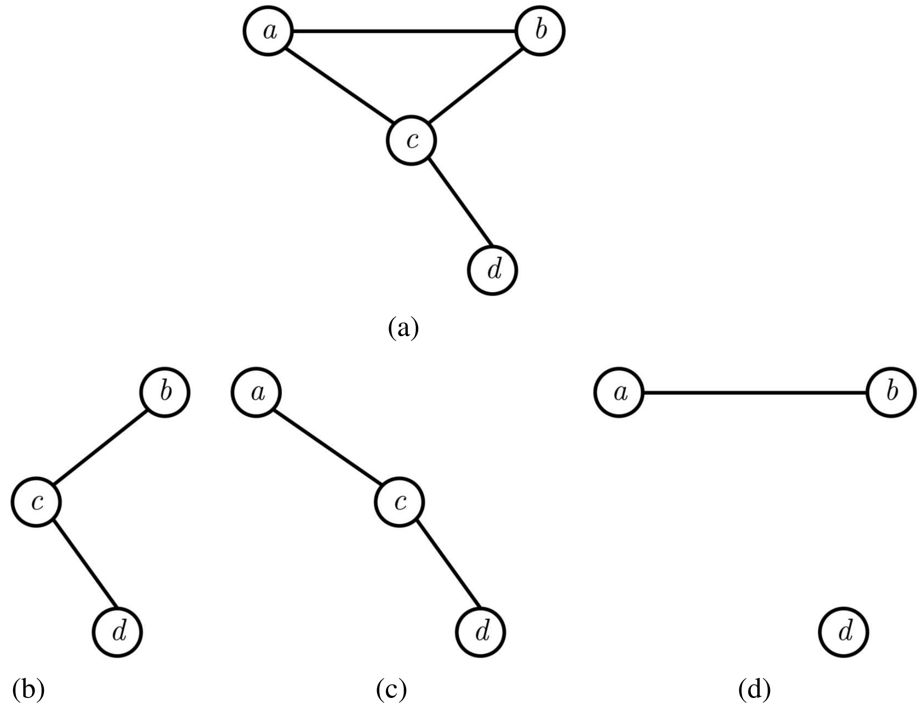
$$Pr[\mathscr{A}(G) = O] \leq \exp(\varepsilon) Pr[\mathscr{A}(G') = O]. \tag{2.8}$$

An algorithm that ensures edge differential privacy offers protection against the disclosure of individual edges. In certain applications, this level of privacy assurance may be reasonable. However, there are scenarios where it becomes desirable to extend privacy protection beyond individual edges, i.e., when the primary focus of the analysis or application is on nodes themselves.

2.8.2 Node Differential Privacy

The node-level differential privacy aims to limit inference about the existence or absence of a node in a graph. Node differential privacy provides protection to the nodes as well as to their adjacent edges. If G and G' differ in only one node and its adjacent edges, we have the concept of node differential privacy. Figure 7 exemplifies neighboring graphs in the node differential privacy setting.

Figure 7 – An example of three neighboring graphs in the node differential privacy setting. (a) original graph. (b) neighboring graph by removing node a. (c) neighboring graph by removing node b. (d) neighboring graph by removing node c.



Source: Elaborated by the author.

Definition 8. *(ε-node differential privacy (KASIVISWANATHAN et al., 2013)). A randomized algorithm $\mathscr{A}$ satisfies ε-node differential privacy, if for any pair of graphs $G = (V,E)$ and $G' = (V',E')$, such that $|V \oplus V'| = 1$ and $E \oplus E' = \{(u,v)|u \in (V \oplus V')$ or $v \in (V \oplus V')\}$, and for any possible output $O \subseteq Range(\mathscr{A})$,*

$$Pr[\mathscr{A}(G) = O] \leq \exp(\varepsilon) Pr[\mathscr{A}(G') = O]. \tag{2.9}$$

Node differential privacy is much harder to achieve than edge differential privacy by definition, since it may be infeasible to design algorithms that are both node-differentially private and that provide accurate graph analysis.

2.9 Summary

In this chapter, we provided the key concepts of differential privacy that compose this thesis. We first motivated the advent of differential privacy and then defined it formally. We discussed the importance of the privacy budget parameter and the sensitivity in differentially private mechanisms. We presented the main mechanisms existing in the literature and those we employed in this thesis. Later, we discussed some theorems that allow us to apply differential privacy in this thesis and we introduced the notion of local differential privacy. Finally, we demonstrated the main notions of neighboring graphs to adopt differential privacy on graph data.

3 LITERATURE REVIEW

In this chapter, we review related efforts made in the last years towards the differentially private release of graph data. Various algorithms that satisfy the DP definition have been developed to release only specific graph statistics. We present them in Section 3.1. Another set of techniques aims to release the entire graph and enable subsequent evaluation of a variety of statistics. Such works are introduced in Section 3.2. Additionally, global DP techniques are most frequent in the literature when compared to the local-DP setting. Consequently, we review general studies on local differential privacy and graphs in Section 3.3. Finally, Section 3.4 details studies in the field of differential privacy applied to weighted graphs.

3.1 DP release of graph statistics

Many works have investigated the problem of querying certain graph statistics, such as degree distribution, subgraph counting, and clustering coefficient, under both edge and node differential privacy (NISSIM *et al.*, 2007; HAY *et al.*, 2009; KARWA *et al.*, 2011; KASIVISWANATHAN *et al.*, 2013; BLOCKI *et al.*, 2013; ZHANG *et al.*, 2015; DAY *et al.*, 2016; CHENG *et al.*, 2018; DING *et al.*, 2018).

3.1.1 Edge differential privacy

Under the edge-DP model, (HAY *et al.*, 2009) proposed a constraint inference-based technique to release degree sequences via DP mechanisms. The authors adapted the definition of differential privacy to graph-structured data and were the first to introduce the notion of edge-DP. The proposed method is based on adding noise to the degree counts of each node. Specifically, for each node, a small amount of Laplacian noise is added to its true degree count. The authors also performed a post-processing step on the noisy answers to infer a more accurate result. An issue with constraint inference is that the degree sequence can not be graphical, i.e., it is not possible to construct a graph without loops or multiple edges between the same pair of vertices with that particular degree sequence. To overcome this, (KARWA; SLAVKOVIĆ, 2012) introduced an optimization step after constraint inference. The authors propose a novel approach to generate synthetic graphs that satisfies DP by using a probabilistic model based on the graphical degree sequence, i.e., a vector that describes the degree distribution of a graph. They also included an additional post-processing step to ensure that the released degree sequence is graphical.

(NISSIM *et al.*, 2007) presented the notion of local sensitivity and studied the problem of privately estimating the number of triangles in a graph. The authors observe that traditional definitions of sensitivity could be overly conservative, leading to algorithms that add more noise than necessary to achieve a given level of privacy. They proposed a new definition of sensitivity that takes into account the smoothness of the function being computed, denoted smooth sensitivity, resulting in algorithms that could achieve the same level of privacy with less added noise. The smooth sensitivity of a function is defined as the maximum change in its output when a single input is changed.

(KARWA *et al.*, 2011) extended the results of smooth sensitivity to privately release other graph statistics, such as k-stars and k-triangles. Specifically, the authors gave an algorithm for computing the smooth sensitivity of the number of k-stars on a given input graph and used a different approach, based on the local sensitivity, to give a differentially private algorithm for releasing k-triangle counts. In the k-triangle counting algorithm, they first find a differentially-private estimate of the local sensitivity for the specified k and release the query answer plus random noise with an expected magnitude proportional to this estimate.

In addition, the ladder function (ZHANG *et al.*, 2015) was used to achieve high accuracy with efficient time complexities. This approach effectively combines the concept of local sensitivity at distance t, from the smooth sensitivity framework (NISSIM *et al.*, 2007) with the exponential mechanism. The local sensitivity of a function at distance t quantifies the maximum local sensitivity among all graphs that can be formed by either adding or deleting up to t edges. In the ladder framework, the local sensitivity at distance t is used to directly derive a quality function that can be used to instantiate the exponential mechanism. They adopt the exponential mechanism to sample the most suitable answer for subgraph queries. The authors focused on counting the following graph statistics using local sensitivity at distance t: triangles, k-stars, k-triangles, and k-cliques.

(CHENG *et al.*, 2018) studied the problem of frequent subgraph mining under differential privacy. They proposed a two-phase differentially private technique called DFG. In the first phase, DFG includes a binary estimation method and a conditional exponential strategy to privately identify frequent subgraphs from the input graphs. In the second phase, in order to calculate the noisy support of each identified frequent subgraph, the authors proposed a noisy support computation approach, which includes a count accumulation method and an error-aware path construction technique.

3.1.2 Node differential privacy

In node-DP setting, (KASIVISWANATHAN *et al.*, 2013) discussed several differentially node-private algorithms for analyzing the accuracy of real networks, that is, algorithms whose output distribution does not change significantly when a node and all its adjacent edges are added to a graph. The main idea behind their techniques is to project the input graph onto the set of graphs with a maximum degree below a certain threshold. However, the challenge of this approach is that the projection operation may be very sensitive to a change of a single node in the original graph. Additionally, this technique suffers from high local sensitivity, especially in dense graphs.

(DAY *et al.*, 2016) propose two approaches based respectively on aggregation and cumulative histogram to publish the degree distribution. Both approaches adopt a new graph projection method that is based on an edge-addition process. The authors proved that publishing a degree histogram from the projected graph has sensitivity $2\theta + 1$, and publishing a cumulative degree histogram has sensitivity $\theta + 1$, where θ is a parameter that bounds the maximum degree in the graph. This process transforms a graph into a θ-degree-bounded graph. The optimal choice of θ depends both on the dataset and on the privacy budget. In both approaches, the authors also use an additional post-processing step to improve the accuracy of the histograms.

(DING *et al.*, 2018) presented a mechanism that can be used for releasing distributions of triangle count, cumulative triangle count, and local clustering coefficient. Usually, large graphs present high sensitivity for queries involving degrees due to the maximum degree of a graph being high. To overcome this issue, the authors propose a novel graph projection method that can be used to obtain an upper bound for sensitivity in different distributions. They proved that when publishing a triangle count distribution with threshold θ, an upper-boundary $4\theta + 1$ of global sensitivity can be achieved. They also extended their method to the local clustering coefficient and publish these coefficients by dividing them into groups.

3.2 DP release of entire graph

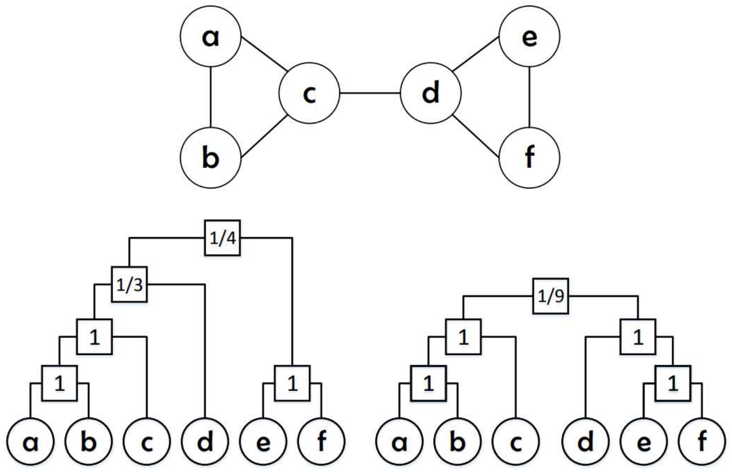
All of the mentioned techniques in Section 3.1 focus on specific subgraph statistics. However, it is important to point out DP techniques to privately release the entire graph structure and enable subsequent evaluation of a variety of statistics. These works are described as follows.

3.2.1 Edge differential privacy

Differentially private release of the entire graph in the centralized DP model has also been studied extensively in recent years. The main advantage of these approaches is that they are agnostic to the analysis in the sense that one can compute any statistics on the released graph. In this scenario, Pygmalion (SALA *et al.*, 2011) aimed to release the graph topology under edge-DP by extracting a graph's detailed structure into private dK-graph (MAHADEVAN *et al.*, 2006) and then generating a synthetic graph. (WANG; WU, 2013) subsequently proposed an improvement in the utility of dK-graph model by calibrating noise based on the smooth sensitivity. The authors first derived from the original graph various parameters (i.e., degree correlations) and used them in the dK-graph model, ensuring edge differential privacy on the learned parameters to generate graphs.

A different approach (XIAO *et al.*, 2014) that adopts the Hierarchical Random Graph (HRG) model (CLAUSET *et al.*, 2006) was introduced to release network data. The authors observed that, by estimating the connection probabilities between nodes, the noise scale enforced by DP could be reduced. Classical graph models are construed by considering the observed edges while HRG uses the connection probabilities between vertices to form dendrograms. An example of dendrograms from a given input graph is illustrated in Figure 8.

Figure 8 – An example of an original graph and two dendrograms.



Source: (XIAO *et al.*, 2014).

Additionally, some approaches focus on perturbing the original adjacency matrix, which allows for representing a graph and adopting matrix perturbation strategies. (CHEN *et al.*, 2014) presented a density-based exploration and reconstruction (DER) mechanism to release the adjacency matrix of a graph. However, both HRG and DER have quadratic time complexity in terms of number of nodes. Top-*m* filter (TmF) (NGUYEN *et al.*, 2015) was proposed to remedy scalability problems in previous work. It adds Laplace noise to each cell in the adjacency matrix and uses an idea similar to High-pass Filter (CORMODE *et al.*, 2012) to avoid the materialization of the noisy adjacency matrix.

Recently, (IFTIKHAR *et al.*, 2020) developed a micro aggregation-based framework for graph anonymization which perturbs graphs by adding noise to the distributions of the original graphs. In particular, the authors propose a distance-constrained algorithm for approximating dK-distributions of a graph via micro aggregation within the proposed framework. PBCN (HUANG *et al.*, 2020) was also developed to release noisy graphs under edge-DP, which is a combination of many techniques: clustering, pre-processing, disturbing degree sequence for edge noise addition, graph reconstruction, and post-processing.

3.2.2 Node differential privacy

Due to the difficulty in obtaining high-utility private mechanisms when releasing the entire graph, there is just one recent work (JIAN *et al.*, 2021) related to node differential privacy when compared to edge-DP. The authors proposed a node perturbation algorithm to achieve node-DP by randomly adding and removing nodes. It first randomly removes each node in the input graph independently with probability p. Then it generates a random number k, which follows a geometric distribution with success probability q. Next, it randomly adds k nodes into the graph, such that each of these k nodes is connected to every existing node with a probability of 0.5. After this step, the number of nodes in the result graphs would be indistinguishable if the input graphs are neighboring graphs. The authors also presented an edge perturbation algorithm that provides privacy guarantees by perturbing edges and adding nodes. It introduces significantly less noise than the previous one for measures such as the global clustering coefficient.

3.3 Graph Privacy and LDP

Under the local DP model, LDPGen (QIN *et al.*, 2017) was initially proposed to privately generate synthetic graphs. The main idea is to capture the structure of the original graph, by incrementally identifying and refining clusters of connected nodes. To do so, LDPGen iteratively partitions nodes into groups, collects information on node-to-group connectivity under local differential privacy, and clusters nodes according to such information. After obtaining such node clusters, LDPGen applies a graph generation model that utilizes such clusters to generate a representative synthetic graph. In this context, (GAO *et al.*, 2018) also proposed a technique to generate synthetic graphs with local-DP. The authors adopted HRG to capture local features and release a synthetic graph. Additionally, the authors grouped the nodes with similar local features by designing two heuristic methods . They also showed the grouping algorithm could enhance the privacy level without the loss of too much information.

(SUN *et al.*, 2019) presented a technique to privately release some graph statistics, such as triangles, three-hop paths, and k-clique counts under LDP. The main idea is that, instead of collecting information directly, which could demand an excessive amount of noise to cover the worst-case scenarios, the data curator first asks each node in the graph about the minimum amount of noise necessary to protect the node's local subgraph.

(YE *et al.*, 2020b) applied randomized response mechanism to the adjacency matrix by local perturbation to guarantee local-DP. However, it introduces high value of bias in the triangle counts. In addition, Ye et al (YE *et al.*, 2020a) presented LF-GDPR, a graph metric estimation framework for general graph analysis. It designs an LDP solution for local perturbation, collector-side aggregation, and calibration. Recently, (IMOLA *et al.*, 2021) presented novel algorithms for counting subgraphs with LDP that use an additional round of interaction between users and the data curator. The authors improved recent results for both triangle and k-star count queries.

3.4 DP Release of Weighted Graphs

When graphs have weighted edges, the aforementioned edge and node differential privacy models may not offer appropriate privacy guarantees. Instead, a more suitable setting is to adapt the differential privacy definition in the context of weighted graphs. This section provides a detailed review of the latest development in DP techniques for weighted graphs. It is divided into two main categories: (1) works that assume the graph topology is known and (2) works that consider graphs with unknown topology.

3.4.1 Public graph topology

The DP setting for weighted graphs was first formally introduced by (SEALFON, 2016). Under this model, the graph topology (V,E) is assumed to be public, and the private information consists only of the edge weights. This new notion is related to the existence of some settings (e.g. road networks) in which the structure of the graph is intrinsic, and the information to be protected is carried by the edge weights. A set of recent works (SEALFON, 2016; LI *et al.*, 2017; PINOT *et al.*, 2018; TONG *et al.*, 2019; WANG; LONG, 2019; FAN; LI, 2022) considered this model for differentially private analysis in weighted graphs. They are described as follows.

3.4.1.1 Shortest paths and distances release with differential privacy

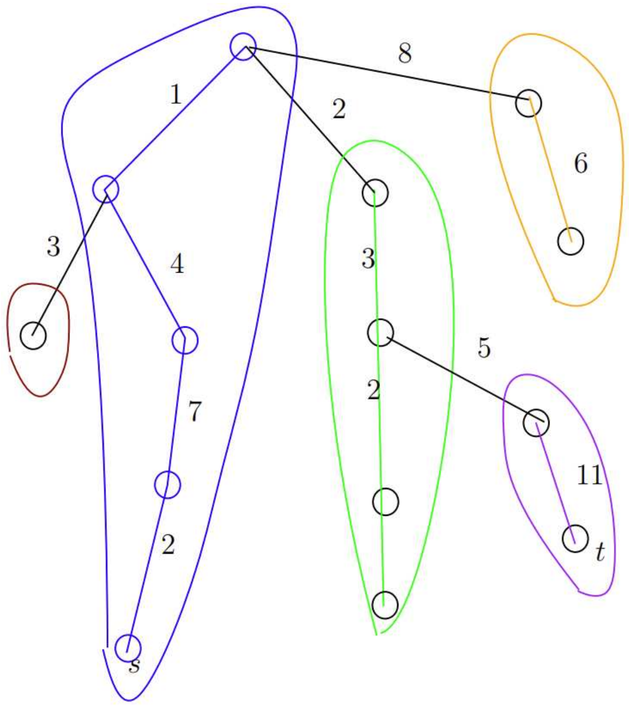
(SEALFON, 2016) aimed to release weighted shortest paths between pairs of nodes and approximate distances between all pairs of nodes without revealing sensitive information about the edge weights.

Initially, he proposed theoretical foundations for considering weight differential privacy on a graph. It assumes that individuals possess influence over the edge weights. As a result, two weight functions are considered neighbors if their l_1 distance is equal to or less than one. A more formalized definition of this DP setting is presented in Section 4.

For the problem of privately releasing shortest paths, the author presented a robust reconstruction-based lower bound, showing that it is not possible to release a short path between a pair of vertices with additive error better than $\Omega(|V|)$, under differential privacy. This lower bound is obtained by reducing the problem of reconstructing many of the rows of a database to the problem of finding a path with low error. The author also showed that an algorithm that utilizes the Laplace mechanism comes close to achieving this bound. Specifically, the author provided a theoretical analysis showing that the weight of the path released by the proposed algorithm is greater by at most $O(k\ log|V|)/\varepsilon$, where the minimum-weight path between a pair of vertices s,t consists of at most k edges. Additionally, since $k < |V|$, the author showed that the weight of the released path is $O(|V|\ log|V|)/\varepsilon$ greater than optimal.

Considering the problem of releasing weighted shortest paths between all pairs of nodes with DP, the author argued that standard techniques outputs error $O(|V|\ log|V|)/\varepsilon$ for each differentially private query. On the other hand, he obtained improved algorithms for two special classes of graphs: (1) trees and (2) graphs with bounded edge weights. Particularly, for trees he proposed a recursive algorithm that releases all-pairs distances with error $O(log^{2.5}|V|)/\varepsilon$. Conversely, for bounded-weight graphs, he demonstrated that a subset of vertices, from the original one, can be obtained such that distances between all pairs of vertices in this subset be sufficient to estimate distances between all pairs of vertices in V. Consequently, the proposed

Figure 9 – An example of a tree partitioned into three heavy paths. A unique color is assigned to every heavy path.



Source: (FAN; LI, 2022).

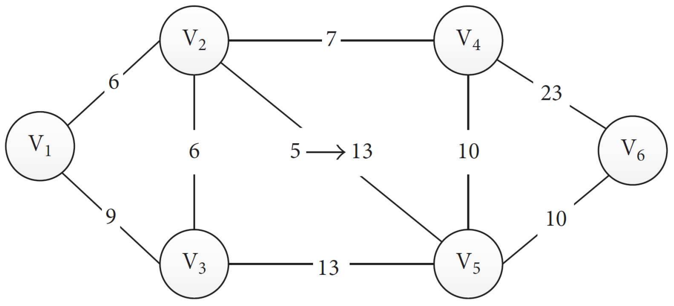
method can release distances between all pairs of vertices with relatively small errors.

(FAN; LI, 2022) revisited the problem of privately releasing approximate distances between all pairs of nodes and recently improved Sealfon's results. The authors first divided a tree into disjoint heavy paths. In a heavy path, each non-leaf node selects one branch, i.e., the edge to the child that has the largest depth. The selected edges form a heavy path. Figure 9 exemplifies three heavy paths in the input tree. Since the tree is decomposed into a set of paths, some edges may not be included in any one of the heavy paths produced during this method.

The authors also argued that the unique path between any pair of vertices intersects at most $log|V|$ heavy paths. They also proved that releasing approximate all-pairs distances is equivalent to releasing several subqueries of heavy paths. For example, in Figure 9, the shortest path between s and t can be decomposed into sub-paths inside disjoint heavy paths. Finally, for trees with depth h, they proposed a new algorithm to release all-pairs distances each with error $O((log^{1.5}\, h).(log^{1.5}\, |V|))/\varepsilon$, compared to the previous additive error $O(log^{2.5}|V|)/\varepsilon$ (SEALFON, 2016).

For bounded-weight graphs, the authors also outperformed Sealfon's results. They discussed that some graphs can be modeled as grid graphs, e.g. the graph representing Manhattan, which is composed of horizontal streets and vertical avenues. This fact enables the division of

Figure 10 – An example of a weighted graph.



Source: (LI *et al.*, 2017).

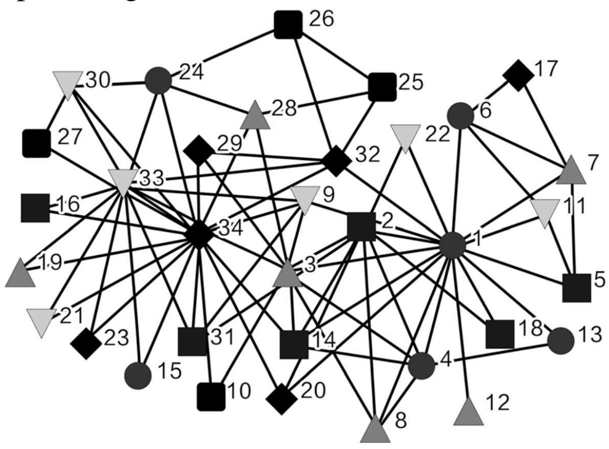
the graph into blocks and then it helps to separate the distances into different types: distances between those vertices in each block; distances between pairs of vertices on the boundary of blocks; and distances not included in the previous cases. Then, the authors present a method to release all-pairs distances on general grid graphs with low error.

3.4.1.2 Differential privacy in weighted social networks based on histograms

(LI *et al.*, 2017) presented a merging barrels and consistency inference (MBCI) approach to releasing weighted graphs under DP guarantees. Initially, they proposed to build a histogram of edge weights, denoted merging barrels, to reduce the noise added to the weights. Consequently, they apply the Laplace mechanism to the groups of edges in the histogram, instead of the weights directly. They argued that simply merging all barrels (edge weights) with the same count into one group may violate differential privacy. Then, the authors proposed a technique to achieve k-indistinguishability to guarantee that these groups require the same amount of noise. Then, groups are said to satisfy k-indistinguishability for an integer $k \geq 1$, if the number of groups with the same amount of barrels is greater than or equal to k. For instance, consider the weighted graph in Figure 10 and the range of weights limited in the range $1-25$. Note that $\omega(V_1,V_2) = \omega(V_2,V_3) = 6$, and $\omega(V_4,V_5) = \omega(V_5,V_6) = 10$. If $k = 2$, $\omega(V_1,V_2)$ and $\omega(V_2,V_3)$ can be merged into one group, as $\omega(V_4,V_5)$ and $\omega(V_5,V_6)$ into another one. Thus, there are two merged groups and the noise added to them is $Lap(\frac{(25-1)/2}{\varepsilon}) = Lap(\frac{12}{\varepsilon})$. If $k = 3$, there are no merged groups and the noise added to each barrel in the histogram is $Lap(\frac{24}{\varepsilon})$. Suppose $\omega(V_2,V_5) = 13$, instead of 5. If $k = 2$ or $k = 3$ three merged groups are present in both cases. For larger k there are no merged groups.

Finally, the authors propose an algorithm to keep most of the shortest paths unchanged. This step is performed according to the original order of the edge-weight sequence. It is important to mention that this process is only based on the known order, without accessing the

Figure 11 – An example of segmentation in a social network with 34 nodes and 78 edges.



Source: (WANG; LONG, 2019).

private dataset. Hence, there is no privacy leakage.

(WANG; LONG, 2019) proposed a modified algorithm to reduce the error introduced by MBCI strategy. They proposed lifted merging barrels and consistency inference (LMBCI) that divides the original weighted graph into several sub-graphs via segmentation. The proposed segmentation algorithm is mainly focused on the network structure and does not involve the change of any edge weight. It is divided into four main steps: (1) a node clustering method based on the number of common neighbors; (2) a grouping approach based on the node clustering results; (3) an establishment of the sub-graphs from the grouping results; and (4) the result of the sub-graph segmentation. An example of this segmentation process is shown in Figure 11. These nodes are divided into the group with the same shape. As a result, there are five sub-graphs after the segmentation.

Finally, the authors adopted the MBCI algorithm for each sub-graph. Theoretical analyses and experimental results showed that LMBCI keeps most of the shortest paths unchanged and improved the accuracy of the published graph. However, the introduced errors in both MBCI and LMBCI techniques are relatively high. Low errors are achieved only for $\varepsilon > 20$, making these approaches inefficient.

3.4.1.3 Differentially-private clustering-based approaches for weighted graphs

(PINOT *et al.*, 2018) presented a differentially private method for node clustering in weighted graphs. Initially, the authors presented theoretical explanations to support the utilization of the Minimum Spanning Tree (MST) algorithm as a clustering algorithm. MST represents a useful summary of the graph, making it a cost-effective and intuitive representation.

For clustering purposes, it has the characteristic of helping the retrieval of non-convex shapes (GRYGORASH *et al.*, 2006). The authors then studied how to incorporate privacy constraints in the DBMSTClu (MORVAN *et al.*, 2017), an MST-based clustering algorithm existing in the literature.

Since the traditional DBMSTClu algorithm only takes weights from (0,1], the authors introduced two normalizing parameters τ and p to ensure lower and upper bounds, respectively, to the weights that fit within DBMSTClu needs. The authors also proposed a weight-release mechanism, that takes normalized weights, transforms them to a new scale s, and adds noise drawn from $Lap(0,s)$. They proved that this weight-release mechanism is ε-differentially private. Then, Pinot et al presented a solution named PTClust that privately outputs clustering partitions. To do so, initially, a differentially private spanning tree topology is produced. Afterward, a randomized version of the edge weights is released using the proposed weight-release mechanism. Finally, the obtained weights are given as input to the DBMSTClu algorithm that performs the clustering partition. Evaluating the accuracy of this method, the authors concluded that it does not deteriorate the final clustering partition.

Another work that discusses the enhancement of privacy in weighted graph clustering is carried out by (CHEN *et al.*, 2022). Specifically, the authors studied the k-median and k-center problems based on the shortest path with weight-differential privacy, which aims to divide the set of vertices in a graph into k clusters in order to minimize both the average and maximum distance between each vertex and its respective cluster center.

For the k-median problem, the authors considered its equivalent form and equivalently reformulated it into the submodular maximization problem (KRAUSE; GOLOVIN, 2014). The authors then presented a greedy differentially private algorithm that provides the best approximation guarantee for this problem. They first calculate the real shortest path distance between any two vertices, then apply the exponential mechanism directly to the objective functions. In this case, the objective functions are indirectly determined by the edge weights. Then, they did not focus on calculating the private edge weights for every edge. Chen et al argued that the larger the number of vertices, the larger the number of evaluations of the objective functions. It makes difficult the adoption of their solution in large-sized graphs. In order to overcome this issue, the authors applied a sampling technique (MIRZASOLEIMAN *et al.*, 2015) to reduce the time complexity from $O(k|V|)$ to $O(ln\,k\ ln\,\frac{k}{\beta}|V|)$, where β is an approximation constant.

For the k-center problem, the authors described a similar process. They provided the best approximation guarantee to this problem using a greedy differentially private algorithm and improved the number of evaluations of the objective functions, also adopting the same sampling technique.

3.4.2 Unknown graph topology

These aforementioned works do not consider the scenario where the graph structure is not known. It leads to incurring excessive noise on the edge weights since the sensitivity can be as large as the maximum edge weight of the graph. To overcome this, Log-laplace mechanism (NY; PAPPAS, 2013) may be applied in the graph context (HANEY *et al.*, 2017). It is suitable to maintain sign consistency between an original query and the result, which is the case of DP release of count-weighted graphs. Truncation techniques (HARDT; ROTH, 2012; KASIVISWANATHAN *et al.*, 2013) can also be adopted to reduce the sensitivity of weighted graphs. The main idea is to limit the value of the maximum edge weight by truncating all the weights below a certain threshold. However, more sophisticated techniques were proposed recently (WANG *et al.*, 2020; NING *et al.*, 2021) to address the private release of weighted graphs problem. They are more related to this thesis and are presented in the next subsections.

3.4.2.1 Differential privacy for weighted network based on probability model

(WANG *et al.*, 2020) were the first to propose a new DP technique that assumes the graph topology is not public. The authors introduced a differentially private stochastic block model algorithm, named SSN (Sufficient Statistic Noisy), to protect the privacy of the network topology. A stochastic block model (HOLLAND *et al.*, 1983) is a generative model for random graphs that has a tendency to generate graphs with communities. Vertices belonging to the same community exhibit comparable structural roles and establish connections with vertices from other groups according to a shared distribution. The main idea of this work is to add noise to the parameters that produce the stochastic block model and, from this model, generate a network that is close (in expectation) to the original one.

During the learning process of the model, an expectation–maximization (EM) algorithm is typically used to find the maximum likelihood estimation. To satisfy differential privacy, the authors argued that a straightforward approach is to add Laplace noise to the parameters directly in each iteration during the learning process. However, this method has the potential to generate a substantial amount of noise and exhibit poor performance. To solve that problem, the authors adopted a variational Bayesian EM (VBEM) technique to compute the model parameters and obtain the differentially private stochastic block model.

Next, the authors developed a differential privacy method for weighted networks denoted VB-WNDP. This approach utilizes the concept of SSN to protect the privacy of the network topology and creates a probability model to privately release the entire weighted network. The authors claimed that the communities have similar distributions of weights. Consequently, the authors computed an approximation to the posterior distribution with differential privacy and obtained a synthetic weighted network.

3.4.2.2 Differential privacy protection on weighted graph in wireless networks

(NING *et al.*, 2021) also proposed a technique to privately release both the edge weights and the graph structure in wireless networks. Initially, the authors presented the notion of a graph dataset, which is basically a set composed of multiple subgraphs. The frequency of an edge is the number of times an edge appears in each subgraph. Then, the authors applied the Laplace mechanism to the edge frequency. In the second step, the authors generated a perturbed graph by taking into account the computed noisy edge frequencies. The graph generation process starts by adding edges with the highest frequencies. The authors considered that there is a certain degree of relationship between nodes and reasonable rules of graph generation were designed.

After getting the perturbed graph sets, the authors designed the edge weight protection algorithm, including a privacy budget division strategy. Because the technique proposed in the previous steps aims to process each graph in a graph set, the protection of the edge weights is viewed as a protection of the edge weights sequence. As a result, the privacy budget needs to be divided. Thus, the privacy budget used in each edge weight sequence is $\frac{|E_i|}{|E|}\varepsilon$, where $|E|$ is the total number of edges in the entire graph and $|E_i|$ is the number of edges in each subgraph.

Finally, the perturbed edge weights are integrated into the releasing process of the graph, and frequent subgraphs are mined. In the mining process, the Laplace mechanism and the exponential mechanism of differential privacy are used to protect the graph structure, and then the data utility is improved. It is worth mentioning that the privacy budget is divided into 4 parts: $\varepsilon_1, \varepsilon_2, \varepsilon_3$ and ε_4. The ε_1 is used to perturb the graph dataset, ε_2 is used to perturb the edge weights, ε_3 and ε_4 are used to perturb the graph structure in the process of frequent subgraph mining. The authors allocated the privacy budget proportion as follows: ε_1:ε_2:ε_3:ε_4 = 2:3:2:3.

3.5 Summary

In this chapter, we reviewed the existing works on DP applied in the graph context. The works presented in Section 3.1 focus on specific queries and are not directly suitable to answer any subsequent graph evaluation under differential privacy guarantees. On the other hand, Section 3.2 introduced works that aim to release the entire graph. The main advantage of these approaches is that they are agnostic to the analysis in the sense that one can compute any statistics on the perturbed graph. However, none of the presented works tackles the problem of directly releasing weighted graphs. Local differentially private approaches, i.e., stronger DP techniques, were discussed in Section 3.3. We presented techniques designed to release either graph statistics or the entire graph. Despite that, no existing local-DP methods are developed to privately release weighted graphs.

Section 3.4 described the latest developments in the context of DP for weighted graphs. Some works assumed that the graph structure is known and considered only the edge weights as private information. This fact is not always true for some real-world situations. Some authors proposed new DP techniques that assume the graph topology is unknown, but they work superimposing a vector perturbation for the weights (using edge weight differential privacy) after perturbing the graph structure (using edge-DP). Consequently, these approaches compose two distinct notions of differential privacy for graphs and the DP property satisfied by this composition would need to be established.

Different from previous studies, we aim to release the entire weighted graph under differential privacy guarantees to enable subsequent computation of a variety of statistics. We introduced a new notion of edge-weight differential privacy and propose two new approaches, based on the two main notions of differential privacy (global-DP and local-DP), to guarantee the privacy of both graph structure and edge weights in the released graph. In particular, we privately compute significant statistics that offer a comprehensive description of the original graph. Subsequently, we employ a sampling technique to perturb the original weighted graph, eliminating the need to materialize all edge weights and mitigating scalability issues. We also add adjustment steps to recover as much as possible of the original graph characteristics before its release.

Table 1 summarizes the existing works on DP for graphs. It compares the purpose of release, the differentially private graph model, the way in which the methods preserve the privacy of the graph structure, and the type of setting (global-DP or local-DP).

Table 1 – Summary of existing works on DP for graphs.

Work	**Purpose of Release**	**Graph DP Model**	**Graph Structure**	**Global-DP**	**Local-DP**
(NISSIM *et al.*, 2007) (HAY *et al.*, 2009) (KARWA *et al.*, 2011) (KARWA; SLAVKOVIĆ, 2012) (ZHANG *et al.*, 2015) (CHENG *et al.*, 2018)	Graph statistics	Edge-DP	Protected by Edge-DP	✓	
(KASIVISWANATHAN *et al.*, 2013) (DAY *et al.*, 2016) (DING *et al.*, 2018)	Graph statistics	Node-DP	Protected by Node-DP	✓	
(SALA *et al.*, 2011) (WANG; WU, 2013) (XIAO *et al.*, 2014) (CHEN *et al.*, 2014) (NGUYEN *et al.*, 2015) (IFTIKHAR *et al.*, 2020) (HUANG *et al.*, 2020)	Entire graph	Edge-DP	Protected by Edge-DP	✓	
(JIAN *et al.*, 2021)	Entire graph	Node-DP	Protected by Node-DP	✓	
(SUN *et al.*, 2019) (YE *et al.*, 2020b) (IMOLA *et al.*, 2021)	Graph statistics	Edge-DP	Protected by Edge-DP		✓
(QIN *et al.*, 2017) (GAO *et al.*, 2018) (YE *et al.*, 2020a)	Entire graph	Edge-DP	Protected by Edge-DP		✓
(SEALFON, 2016) (LI *et al.*, 2017) (PINOT *et al.*, 2018) (WANG; LONG, 2019) (CHEN *et al.*, 2022) (FAN; LI, 2022)	Graph statistics	Edge-Weight-DP	Public knowledge	✓	
(WANG *et al.*, 2020) (NING *et al.*, 2021)	Entire graph	Edge-Weight-DP	Protected by the composition of two distinct notions of DP	✓	
This thesis	*Entire graph*	*Edge-Weight-DP*	*Protected by new definition of Edge-weight-DP*	✓	✓

Source: Elaborated by the author.

4 DIFFERENTIAL PRIVACY FOR COUNT-WEIGHTED GRAPHS

In this chapter, we explore the fundamental concepts and provide formal definitions related to differential privacy in the context of weighted graphs. We first present the weighted DP model existing in the literature and present our new notion of neighboring weight graphs that consider both edge weights and graph topology to be private information. Then we show how to efficiently perturb a weighted graph based on the proposed definition and using the geometric mechanism of differential privacy. Finally, we present some post-processing techniques to improve the accuracy of the perturbed graph by preserving as much as possible the original graph characteristics.

4.1 Edge-weight Differential Privacy

The two main alternatives that consider applying differential privacy on graphs (edge and node differential privacy) are not well suited to weighted graphs. In general, it is not possible to release, e.g. shortest paths, with meaningful utility under edge-DP or node-DP, since changing a single edge can significantly change the distances in the graph. For example, removing a single edge between two arbitrary nodes may increase the distance between them from 1 to ∞, when this removal disconnects the graph. Even if the graph remains connected, the removal of an edge may jeopardize distances between nodes. This inadequacy inspired Sealfon (SEALFON, 2016) to propose a new notion of differential privacy for graphs, denoted edge-weight differential privacy.

4.1.1 Edge-weight DP for graphs with known topology

As mentioned before, Sealfon was the first to formally introduce the differential privacy framework for weighted graphs. It starts by defining *neighboring weight functions* in the context of weighted graphs. Let $G = (V, E, \omega)$ be an undirected weighted graph with a vertex set V, an edge set E, and a weight function $\omega : V^2 \to \mathbb{R}^+$ mapping connections between a pair of vertices (u, v) to weights in G.

Definition 9. *(neighboring weight functions (SEALFON, 2016)) Two weight functions ω, ω' : $V^2 \to \mathbb{R}^+$ are neighboring, denoted $\omega \sim \omega'$, if:*

$$||\omega - \omega'||_1 := \sum_{u,v \in V} |\omega(u,v) - \omega'(u,v)| \leq 1. \tag{4.1}$$

Two graphs G and G' are neighbors if they have the same set of vertices and edges, and if their weight functions differ in one unit.

Definition 10. *(neighboring weight graphs (SEALFON, 2016)) Let $G = (V, E, \omega)$ and $G' = (V', E', \omega')$ be two weighted graphs, G and G' are neighbors if $V = V'$, $E = E'$ and $\omega \sim \omega'$.*

As previously stated, several works in the literature utilize these definitions in their studies (LI *et al.*, 2017; PINOT *et al.*, 2018; WANG; LONG, 2019; CHEN *et al.*, 2022; FAN; LI, 2022).

4.1.2 Edge-weight DP for graphs with unknown topology

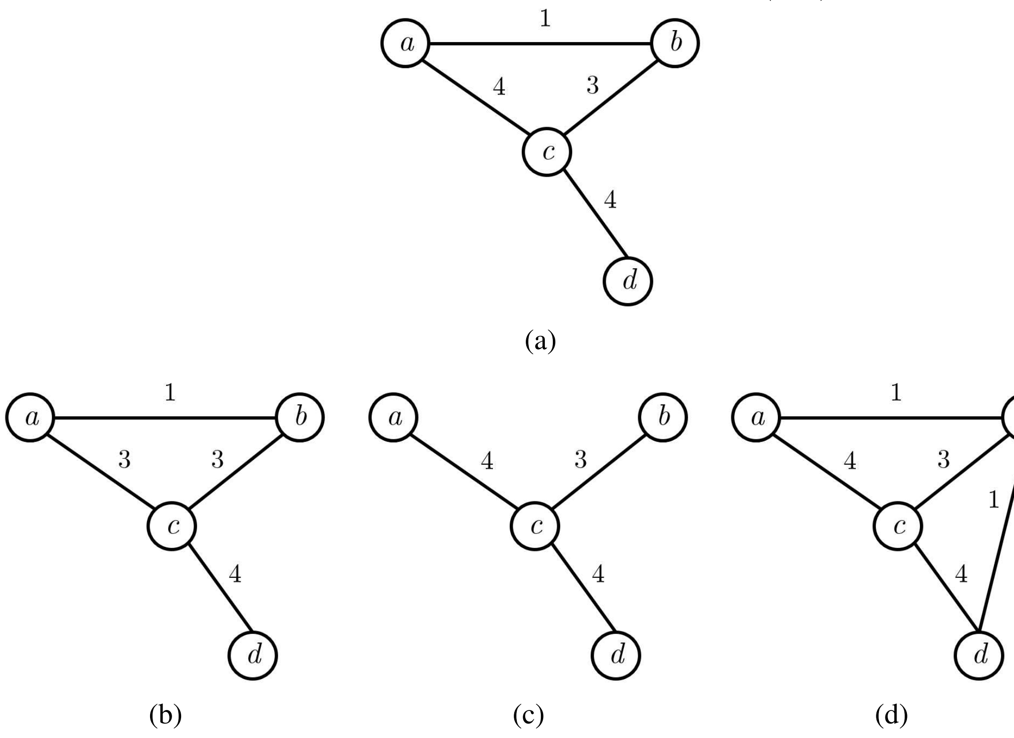
In contrast to the formulation of Sealfon, which considered two graphs to be neighbors if they have the same graph topology and similar weight functions, we consider both the graph structure and the edge weights as private information. To the best of our knowledge, this thesis is the first to formalize a new definition for edge weight differential privacy tailored for graphs with unknown topology.

Our new definition aims to bridge the gap between differential privacy and count-weighted graph data release, based on the mentioned constraints. In addition, it aims to answer our Research Question 1: How to establish a new definition of neighboring graphs considering both graph topology and edge weights as private information?

For some real-world applications, the assumption that the graph topology is public is misleading. For example, when protecting the presence or absence of interactions in a human contact network, or the existence or absence of phone calls, text messages, or the presence or absence of a co-authorship in a paper. These kinds of interactions are not covered by Sealfon's definitions in terms of privacy. Once an edge is already known, any differentially private mechanism will not change the presence or absence of that interaction, i.e., only considering the scenario where the graph topology is publicly known is not effective to provide the desired privacy guarantees.

To address this limitation, we adapt the notion proposed by Sealfon for neighboring weight functions (Definition 9) and provide a new definition for neighboring weight graphs with unknown topology. Let $G = (V, E, \omega)$ be an undirected count-weighted graph with a vertex set V, an edge set E, and a weight function $\omega : V^2 \to \mathbb{Z}_{\geq 0}$ mapping connections (interactions) between a pair of vertices (u, v) to weights in G. The pair $(u, v) \in E$ if vertices u and v share a common

Figure 12 – An example of three neighboring weight graphs. (a) original graph. (b) neighboring graph by changing $\omega(a,c)$ by one unit. (c) neighboring graph by removing edge (a,b) with weight one. (d) neighboring graph by adding a new edge (b,d) with weight one.



Source: Elaborated by the author.

edge, and (u,v) does not belong to E, otherwise. If $(u,v) \notin E$ then $\omega(u,v) = 0$. Because G is undirected, $\omega(u,v) = \omega(v,u)$. We formally define neighboring weight functions with unknown topology next.

Definition 11. *(neighboring weight functions with unknown topology) Two weight functions* ω, $\omega' : V^2 \to \mathbb{Z}_{\geq 0}$ *are neighboring, denoted* $\omega \sim \omega'$*, if:*

$$||\omega - \omega'||_1 := \sum_{u,v \in V} |\omega(u,v) - \omega'(u,v)| = 1. \tag{4.2}$$

Definition 11 differs from Definition 9 in the sense that now weights can assume zero values and, since we are working with count-weighted graphs, weights are also integers. Then, two graphs G and G' are said to be neighboring if they have the same set of vertices and if the weight functions differ in one unit.

Definition 12. *(neighboring weight graphs with unknown topology) Let* $G = (V,E,\omega)$ *and* $G' = (V',E',\omega')$ *be two weighted graphs,* G *and* G' *are neighbors if* $V = V'$ *and* $\omega \sim \omega'$.

Figure 12 shows an example of three neighboring weight graphs for unknown topology from a given weighted graph. Under this new definition, neighboring weight graphs differ in one edge weight unit and can also differ in one edge, as in the edge-DP model (HAY *et al.*, 2009). This is the case where $\omega(u,v) = 0$ in G, and $\omega'(u,v) = 1$ in its neighbor G', or vice versa. In this example, $(u,v) \notin E(G)$, but $(u,v) \in E(G')$. As a consequence, we can see this definition as a relaxation of the traditional edge-DP model. We could establish an extension to the edge-weight differential privacy for unknown topology that allows neighboring graphs to differ by more than a single unit of edge weight, introducing a parameter k. In this *k-edge weight DP*, weighted graphs are neighbors if $||\omega - \omega'||_1 \leq k$. Let ΔW the maximum weight of a graph G. If $k = \Delta W$, then the k-edge weight DP model is equivalent to the traditional edge-DP, since the set of neighboring graphs under the k-edge weight DP is a superset of the neighbors under edge-DP. If $1 < k < \Delta W$, then k-edge weight prevents the disclosure of aggregate properties of any subset of k units of edge weights. It is worth mentioning that, in our setting, we assume both nodes are public knowledge and $k = 1$, as stated in Definition 12.

According to the concepts mentioned previously, the formal definition of differential privacy for count-weighted graphs, considering edges and edge weights as private information, can be described as:

Definition 13. *(ε-edge weight differential privacy. A randomized algorithm $\mathscr{A}$ satisfies ε-edge weight differential privacy, if for any pair of graphs $G = (V,E,\omega)$ and $G' = (V',E',\omega')$, such that G and G′ are neighboring weight graphs and for any possible output $O \subseteq Range(\mathscr{A})$,*

$$Pr[\mathscr{A}(G) = O] \leq \exp(\varepsilon) Pr[\mathscr{A}(G') = O]. \tag{4.3}$$

We can also define LDP in the context of count-weighted graphs when the topology is unknown. Let $\gamma_v = [\omega(v,u_1),...,\omega(v,u_n)]$ be a node v's neighbor weight list. For instance, in Figure 12a, node c's neighbor weight list is $\gamma_c = [4,3,4]$. Note that, if there is no edge (v,u_i) in the graph, $\omega(v,u_i) = 0$. Then, edge weight local differential privacy is defined as follows.

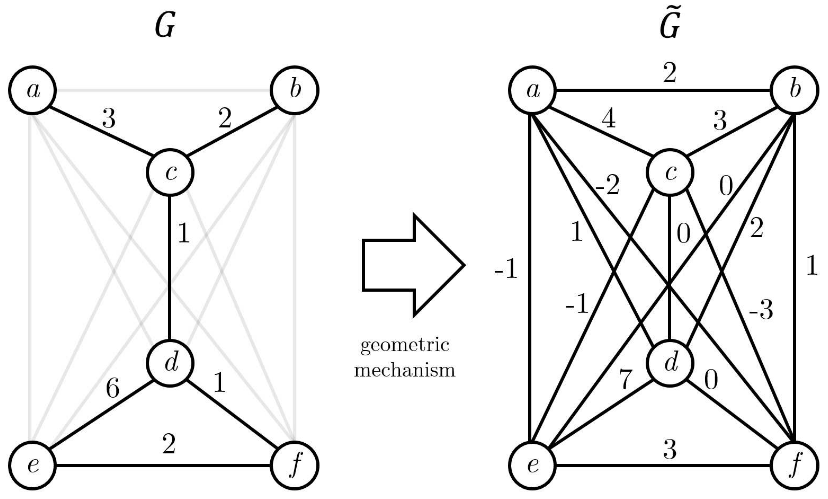
Definition 14. *(ε-edge weight local differential privacy). A randomized algorithm $\mathscr{A}$ satisfies ε-edge weight local differential privacy, if for any two neighboring weight lists γ_v and γ_v', such that γ_v and γ_v' only differ in one unit of weight, and for any possible output O of $\mathscr{A}$,*

$$Pr[\mathscr{A}(\gamma_v) = O] \leq \exp(\varepsilon) Pr[\mathscr{A}(\gamma_v') = O]. \tag{4.4}$$

4.2 Count-Weighted Graph Perturbation

Given that we have established DP to address count-weighted graphs with unknown topology, we can apply this new notion to perturb an input graph G and produce a noisy weighted

Figure 13 – A naive approach example of a count-weighted graph perturbation via the geometric mechanism, where black edges denote non-zero edges and grey edges correspond to edges with weight equal to zero.



Source: Elaborated by the author.

graph version $\tilde{G}$. We consider G as a sparse graph, which implies that non-existing edges have weights equal to 0. A naive approach consists of perturbing all edge weights, i.e., both non-zero and zero values via the geometric mechanism. Figure 13 illustrates an example of this process.

The drawback of this approach is that it incurs quadratic computational cost since the number of perturbed values is equal to $\frac{|V|(|V|-1)}{2}$. In this context, filtering and sampling techniques are commonly used to deal with sparse data (DUFFIELD *et al.*, 2005; DUFFIELD *et al.*, 2007; CORMODE *et al.*, 2012). In short, filters usually prune away parts of the data while sampling randomly selects a subset of elements from the input data. In this thesis, we adopt a sampling strategy to avoid the materialization of all zero edge weights because filter techniques, such as High Pass Filter (CORMODE *et al.*, 2012), tend to introduce additional bias when compared to sampling approaches. Its objective is to provide a response to our Research Question 2: How can we provide a scalable graph perturbation solution?

In particular, we propose a similar idea to the Priority Sampling (PS) method (DUFFIELD *et al.*, 2007) to efficiently compute a noisy graph $\tilde{G}$. This technique is suitable to generate a sample of fixed size with solid accuracy properties. Additionally, in this thesis, we assume the size of the sample is given by the number of expected edges $\bar{m}$ in the noisy weighted graph $\tilde{G}$. More details about how to obtain this number are discussed in Chapter 5.

The intuition of priority sampling is that the weights with high noisy values are most likely to correspond to non-zero edge weights in the original graph. So it is important to include them in the perturbed graph. On the other hand, weights with low noisy values should also be added, but with probability proportional to their noisy values. Cormode et al (CORMODE *et al.*, 2012) also proposed a sampling strategy based on priority sampling to perturb count queries in sparse data. Different from them, we adopt a one-sided priority sampling, i.e., non-positive noisy values have no probability to be in the sample.

Priority sampling is defined as follows: for each noisy edge weight $\tilde{\omega}(u,v)$ is assigned a priority $P_{uv} = \frac{\tilde{\omega}(u,v)}{r_{uv}}$, where r_{uv} is a uniform random variable chosen from the range $(0,1]$. Assuming all priorities are distinct, the priority sample of size $\bar{m}$ consists of the $\bar{m}$ edge weights of highest priority. An associated threshold $\tau > 0$ is the $(\bar{m}+1)^{th}$ priority. Then $\tilde{\omega}(u,v)$ is sampled if:

$$P_{uv} = \frac{\tilde{\omega}(u,v)}{r_{uv}} > \tau \iff r_{uv} < \frac{\tilde{\omega}(u,v)}{\tau}. \tag{4.5}$$

Since r_{uv} is uniform over (0,1], the probability of including the edge (u,v) in $\tilde{G}$ is given by:

$$p_{uv} = min\left(\frac{\tilde{\omega}(u,v)}{\tau}, 1\right)_+, \tag{4.6}$$

where $(y)_+ = max(y,0)$.

Note that the noisy weights with high magnitude ($\tilde{\omega}(u,v) > \tau$) are included in $\tilde{G}$ with probability 1, whereas non-positive noisy weights have probability 0 to be included in $\tilde{G}$. Subsequently, we perturb the non-zero weights separately from the zero ones.

Algorithm 1 describes the proposed perturbation process for weighted graph. Initially, for every existing edge (u,v) in G, we apply geometric noise with $\alpha = e^{-\varepsilon}$ to $\omega(u,v)$ and add it to $\tilde{G}$ with probability p_{uv} (lines 2-5). For the non-existing edges, we first compute the expected number of zero edges, denoted m_0, that are included in $\tilde{G}$ (line 6). The total number of zeros is given by $\left(\frac{n(n-1)}{2} - \bar{m}\right)$ and the probability to be added in $\tilde{G}$ is estimated as $\frac{\alpha(1-\alpha^\tau)}{\tau(1-\alpha^2)}$ (Theorem 7). In line 7, to prevent any bias, we flip a coin to decide whether to round m_0 up or down.

Theorem 7. *Let S be the set of zero edges included in $\tilde{G}$. The probability of a zero edge (u,v) be included in $\tilde{G}$ is given by:*

$$Pr[(u,v) \in S] = \frac{\alpha(1-\alpha^\tau)}{\tau(1-\alpha^2)}. \tag{4.7}$$

The proof of Theorem 7 is deferred to the Appendix A.

Algorithm 1: Graph Perturbation

```
Input  : A graph $G = (V, E, \omega)$, number of expected edges $\bar{m}$ and a privacy budget $\varepsilon$
Output : A perturbed graph $\tilde{G} = (V, \tilde{E}, \tilde{\omega})$
1  $\tilde{G} \leftarrow \emptyset, \alpha \leftarrow e^{-\varepsilon}$
2  foreach $(u,v) \in E$ do
3      $\tilde{\omega}(u,v) \leftarrow \omega(u,v) + Geometric\_Mechanism(\alpha)$ // applying geometric
        mechanism to every existing edge
4      add edge $(u,v)$ to $\tilde{G}$ with prob. $p_{uv} = min(\frac{\tilde{\omega}(u,v)}{\tau}, 1)_+$.
5  end
6  $m_0 \leftarrow \left(\frac{n(n-1)}{2} - \bar{m}\right) \frac{\alpha(1-\alpha^\tau)}{\tau(1-\alpha^2)}$ // computing the expected number of zero
    edges
7  flip a coin to decide $m_0 \leftarrow \lceil m_0 \rceil$ or $\lfloor m_0 \rfloor$ // avoiding introducing bias
8  while $m_0 > 0$ do
9      uniformly at random pick an edge $(u,v) \notin E(G)$
10     if $(u,v) \notin E(\tilde{G})$ then
11         add $(u,v)$ to $\tilde{G}$ and draw the value of $\tilde{\omega}(u,v)$ from the distribution
             $Pr[\tilde{\omega}(u,v) = w \mid (u,v) \in S]$ given by:
12             $\frac{\tau(1-\alpha)^2\alpha^{w-1}}{1-\alpha^\tau}$, if $w \geq \tau$
13             $\frac{w(1-\alpha)^2\alpha^{w-1}}{1-\alpha^\tau}$, if $0 < w < \tau$
14         $m_0 \leftarrow m_0 - 1$
15     end
16 end
17 return $\tilde{G}$
```

In line 9 we uniformly at random pick an edge $(u,v) \notin E(G)$ and verify it was not included yet (line 10). Finally, in line 11 we add (u,v) to $\tilde{G}$ and draw this new edge value from the probability distribution function $Pr[\tilde{\omega}(u,v) = w|(u,v) \in S]$, which is given by:

$$Pr[\tilde{\omega}(u,v) = w \mid (u,v) \in S] = \frac{Pr[(u,v) \in S \mid \tilde{\omega}(u,v) = w] Pr[\tilde{\omega}(u,v) = w]}{Pr[(u,v) \in S]}. \tag{4.8}$$

Substituting the probabilities in the above equation, we have:

$$Pr[\tilde{\omega}(u,v) = w \mid (u,v) \in S] = \frac{min\left(\frac{w}{\tau}, 1\right)_+ \frac{1-\alpha}{(1+\alpha)} \alpha^w}{\frac{\alpha(1-\alpha^\tau)}{\tau(1-\alpha^2)}}. \tag{4.9}$$

If $w \geq \tau$:

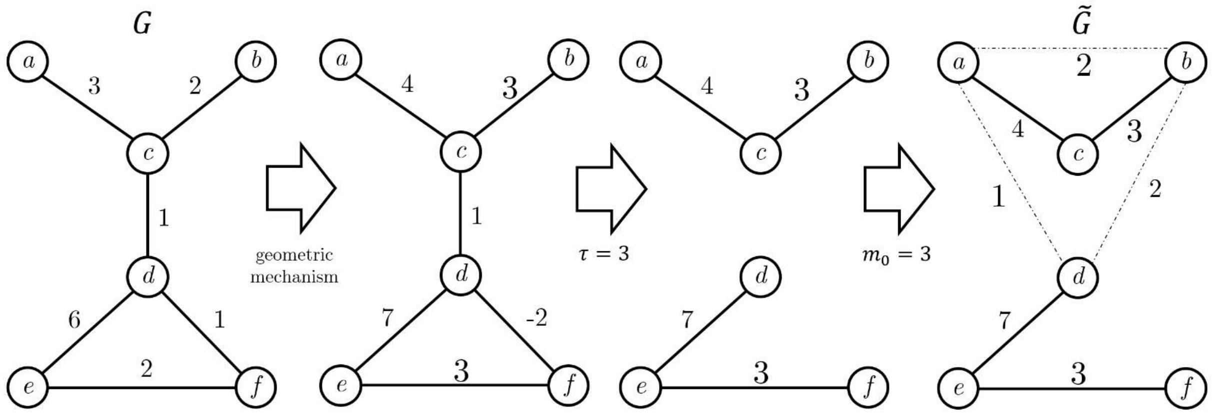
$$Pr[\tilde{\omega}(u,v) = w \mid (u,v) \in S] = \frac{\frac{1-\alpha}{(1+\alpha)} \alpha^w}{\frac{\alpha(1-\alpha^\tau)}{\tau(1-\alpha^2)}} = \frac{\tau(1-\alpha)^2 \alpha^{w-1}}{1-\alpha^\tau}. \tag{4.10}$$

If $0 < w < \tau$:

$$Pr[\tilde{\omega}(u,v) = w \mid (u,v) \in S] = \frac{\frac{w}{\tau}\frac{1-\alpha}{(1+\alpha)}\alpha^w}{\frac{\alpha(1-\alpha^\tau)}{\tau(1-\alpha^2)}} = \frac{w(1-\alpha)^2\alpha^{w-1}}{1-\alpha^\tau}. \tag{4.11}$$

Figure 14 shows a detailed example of the graph perturbation phase. First, the geometric mechanism is applied to the non-zero edges. Then, edges are filtered based on the τ parameter. Finally, m_0 is computed, and three new edges are added to the released perturbed graph.

Figure 14 – A count-weighted graph perturbation example with $\tau = 3$ that produces three new edges: $(a,b),(a,d)$ and (b,d).



Source: Elaborated by the author.

It is important to mention that, in practice, τ is unknown. In order to choose a good value for τ, we first pick a sample of size $m' = \beta\bar{m}$, considering β is small and > 1. We use our initial guess for τ to draw a corresponding priority sampling. We then reduce the sample size to $\bar{m}$ by picking the $\bar{m}$ largest priorities and retaining the $(\bar{m}+1)^{th}$ priority, i.e., the associated threshold τ (CORMODE *et al.*, 2012).

4.3 Post-processing to improve graph utility

Post-processing techniques can be employed to improve the utility of the resulting graph after the perturbation process. This approach is guaranteed to provide differential privacy since any function applied to an output of a DP algorithm also satisfies DP (Theorem 5). The goal in this phase is to answer our Research Question 3: How to keep the graph consistent and avoid introducing bias in the edge weights after adopting a DP technique to perturb the graph structure?

In this thesis, we focus on privately querying two main statistics: (1) node degrees, and (2) sum of all edge weights. The idea is to use them to improve the accuracy of the released weighted graph. In particular, node degrees characterize the structure and behavior of networked systems and help specialists understand how the users interact with each other (NEWMAN, 2003). This metric is also useful for other graph-related studies, such as the number of edges, centrality measures, clustering coefficients, and egocentric analysis.

The second statistic, sum of edge weights, aims to preserve the original number of records since each unit of weight corresponds to one record in the original data (Figure 1a). This metric also avoids introducing bias in the final released set of edge weights. As discussed earlier, to avoid scalability problems, we perturb the entire graph using a sampling strategy. However, it may introduce some bias in the edge weights and consequently reduce the utility of the released weighted graph. Another reason to query the mentioned statistics is that both present low sensitivity, as described as follows.

Theorem 8. *For any neighboring weight graphs with unknown topology G and G' that differ in one unit of weight,*

$$||deg(V(G)) - deg(V(G'))||_1 \leq 2. \tag{4.12}$$

The proof of Theorem 8 is deferred to the Appendix A.

Theorem 9. *For any neighboring weight graphs with unknown topology G and G' that differ in one unit of weight,*

$$\Big|\Big| \sum_{u,v \in E(G)} \omega(u,v) - \sum_{u,v \in E(G')} \omega(u,v) \Big|\Big|_1 = 1. \tag{4.13}$$

The proof of Theorem 9 is also deferred to the Appendix A.

Algorithm 2 shows the steps for differentially private release of the mentioned graph statistics.

Original node degrees D are initially perturbed (line 3) via the geometric mechanism with $\alpha_1 = e^{-\varepsilon_1/2}$. Note that the noisy sum of node degree values may be odd, which makes the subsequent adjustment in Section 4.3.1 infeasible. To remedy this problem, we randomly select a node (line 5), and flip a coin to determine whether to add or subtract 1 to v_i (line 6). This step avoids introducing bias in the result. In line 8, because noisy degrees $\tilde{D}$ may assume negative values, we post-process them to avoid it. We adopt Algorithm 4 with input $\tilde{D}$ and

Algorithm 2: Statistics Extraction

```
   Input   : A graph $G = (V, E, \omega)$, privacy budgets $\varepsilon_1$ and $\varepsilon_2$
   Output : Noisy degrees $\bar{D}$ and noisy sum of all edge weights $\bar{s}$
1  $\alpha_1 \leftarrow e^{-\varepsilon_1/2}$,  $\alpha_2 \leftarrow e^{-\varepsilon_2}$
2  $D \leftarrow deg(V(G))$
3  $\tilde{D} \leftarrow D +$ Geometric_Mechanism($\alpha_1$) // applying geometric mechanism to
     the degrees
4  if $\sum_{i=1}^{n} \tilde{D}_i$ is odd then
5      uniformly at random select a value $v_i$ in $\tilde{D}_i$
6      flip a coin to determine whether to add or subtract 1 to $v_i$ // avoiding
         introducing bias
7  end
8  $\bar{D} \leftarrow$ Post−processing_Projection($\tilde{D}, \sum_{i=1}^{n} \tilde{D}_i$) // preserving the noisy sum
     of all node degrees and guaranteeing all nodes have positive
     degrees
9  $s \leftarrow \sum_{u,v \in E(G)} \omega(u, v)$
10 $\bar{s} \leftarrow s +$ Geometric_Mechanism($\alpha_2$) // applying geometric mechanism to the
     sum of all edge weights
11 return $\bar{D}, \bar{s}$
```

constant $c = \sum_{i=1}^{n} \tilde{D}_i$. This step is discussed in detail in Section 4.3.2, but the main idea of this post-processing step is to preserve the noisy sum of all node degrees and, at the same time, guarantee that all nodes have positive degrees, i.e., they satisfy this domain constraint.

The next statistic privately obtained is the sum of the weights of all edges in the graph. In line 10, a noise is added to this statistic using the geometric mechanism with $\alpha_2 = e^{-\varepsilon_2}$. Finally, the algorithm returns the noisy degrees $\bar{D}$ and the noisy sum of all edge weights $\bar{s}$. Both values $\bar{D}$ and $\bar{s}$ are used in the degrees adjustment and weights adjustment steps, respectively, to improve graph utility.

4.3.1 Degrees Adjustment

The post-processing step for degrees adjustment aims to modify the degree values, obtained from the graph perturbation phase, as close as possible to the original node degrees. To this end, we use the privately queried estimate of original values $\bar{D}$. We propose a two-step algorithm in order to create a graph G^* with adjusted node degrees from $\tilde{G}$. It is described in Algorithm 3.

The core idea of the first algorithm step is to swap the edges such that the node degrees of G^* are as close as possible of $\tilde{G}$ while preserving the edges with higher weights value. Initially, in order to adjust the node degrees according to $\bar{D}$, we create an empty graph G^*

Algorithm 3: Degrees Adjustment

```
Input   : A graph $\tilde{G} = (V, E, \omega)$, an array of node degrees $\bar{D}$
Output  : Degrees adjusted graph $G^*$
1  $G^* \leftarrow \emptyset$
2  sort $E(\tilde{G})$ in descending order of weight
3  foreach $(u,v) \in E(\tilde{G})$ do
4      if $\bar{D}(u) > 0$ and $\bar{D}(v) > 0$ then
5          add edge $(u,v)$ to $G^*$ with weight $\omega(u,v)$ // preserving the edges with
            higher weights value
6          $\bar{D}(u) \leftarrow \bar{D}(u) - 1$
7          $\bar{D}(v) \leftarrow \bar{D}(v) - 1$
8  end
9  realize the graph $G^*$ following the remaining $\bar{D}$ // swapping edges such that the
    node degrees of $G^*$ are as close as possible of $\tilde{G}$
10 return $G^*$
```

(line 1). Then, for each edge (u, v) in $\tilde{G}$, following the descending order of weight, we verify if the degrees in $\bar{D}$, related to both nodes u and v, are greater than 0 (line 4). If this condition is satisfied, we add (u, v) to G^* and decrease one unit of degree in $\bar{D}$ for each corresponding node u and v (lines 5-7). It means the edge (u, v) does not violate the insertion in G^*, because according to the expected degrees $\bar{D}$, there is still some space for the edge addition. Note that the first algorithm step is maximal in the sense that no more edges with higher weights can be included in G^* without decreasing any node degree below 0.

The second step (line 9) consists of realizing the graph G^*, i.e., generate a random graph with a given degree distribution. In our case, the degree distribution is set as the remaining $\bar{D}$. We adopt an algorithm that implements an efficient Markov chain based on edge swaps, with a mixing time which depends on the degree distribution (HASTINGS, 1970; KARRER; NEWMAN, 2011). This algorithm iterates through all the edges in the network and tries to swap its target or source with the target or source of another edge. We used the implementation available in graph-tool package (PEIXOTO, TIAGO P., 2014) which runs in $O(|E| + |V|)$ time.

4.3.2 Weights Adjustment

Recall the main advantage of priority sampling is that it addresses edge weights with high values and does not neglect small ones. At the same time, PS algorithm generates a sample of fixed size $\bar{m}$. However, priority sampling introduces bias in the weights when the data distribution is skewed. Another case of bias introduction occurs when querying node degrees.

The addition of independent noise from DP mechanisms cannot ensure that all node degrees are perturbed to positive values. A remarkably simple solution to project query results to the positive integer space is to clip values that lie outside the allowed region $\mathbb{Z}_{>0}$. However, it clearly introduces bias to the post-processed output. Therefore, this step aims to avoid introducing bias to the weights in the graph perturbation phase. Additionally, it is also designed to guarantee that all nodes have positive degrees when this type of query is executed, e.g. Algorithm 2 line 8.

Denote a query by $q \in Q \subseteq \mathbb{R}^n$, and $\mathscr{K}$ be a set of constraints which hold among the true answers. The proposed post-processing technique takes the randomized output of the query ($\tilde{q}$) and aims to find a feasible solution $\bar{q}$ that (i) minimizes squared l_2-distance to the noisy answer $\tilde{q}$, and (ii) satisfies the constraints in $\mathscr{K}$. We focus on two constraints in particular: (1) positive integer outputs, since the studied count-weighted graphs are drawn in this domain; (2) the entire output should be summed up to a constant c, which aims to avoid introducing bias and preserve original data characteristics.

We now formally discuss the underlying optimization problem. Our goal is to find the closest solution to $\tilde{q}$ that satisfies the constraints in $\mathscr{K}$:

$$\min_{\bar{q} \in \mathscr{K}} ||\bar{q} - \tilde{q}||_2^2 \quad s.t. \ \mathscr{K} = \left\{ \bar{q} \in \mathbb{Z}_{>0} \middle| \sum_{i=1}^{n} \bar{q}_i = c \right\}, \tag{P_A}$$

where $c \in \mathbb{Z}_{>0}$ is a constant.

Problem P_A refers to "closest" as measured in squared l_2 norm. To solve this optimization problem, we use projected gradient descent (PGD), rather than a commercial solver that would not scale to large graphs. Due to its efficiency and simplicity, PGD is one of the most popular approaches for solving constrained optimization. Each iteration in PGD consists of a descent step from the traditional gradient descent method followed by a projection onto the feasible set $\mathscr{K}$. Particularly, when the least-squares objective is strongly convex, PGD is known to converge very quickly (VU; RAICH, 2021).

Algorithm 4 describes the PGD algorithm to solve problem P_A.

In line 1, we start from the initial solution $x^{(0)} = \tilde{q}$ (the closest solution at this point). The gradient step performs the update $g^{(i+1)} \leftarrow x^{(i)} - 2\eta \left(x^{(i)} - x^{(0)}\right)$ since the gradient of squared l_2 norm is given by

$$\nabla_{\bar{q}} ||\bar{q} - \tilde{q}||_2^2 = 2(\bar{q} - \tilde{q}). \tag{4.14}$$

Algorithm 4: Post-processing Projection

```
   Input  : Array $\tilde{q} = (v_1, ..., v_n)$, target sum $c$, step size $\eta$, number of steps $t$
   Output : Minimum squared $l_2$ solution $x^{(t)}$
1  $x^{(0)} \leftarrow \tilde{q}$
2  for $i \in \{0, 1, ..., t-1\}$ do
3      $g^{(i+1)} \leftarrow x^{(i)} - 2\eta\left(x^{(i)} - x^{(0)}\right)$    // gradient step
4      $x^{(i+1)} \leftarrow \Pi_c\left(g^{(i+1)}, x^{(i)}\right)$           // projection
5  end
6  return $x^{(t)}$
```

The projection Π_c takes into account the computed gradient $g^{(i+1)}$, the result from the step before $x^{(i)}$ and the constant c to project the output back to

$$\mathscr{K}\,\Pi_c(g^{(i+1)}, x^{(i)}) = max(g^{(i+1)} - \lambda, 1). \tag{4.15}$$

Intuitively the projection works as follows: we subtract the same scalar

$$\lambda = \frac{\sum_{j=1}^{n} x_j^{(i)} - c}{n} \tag{4.16}$$

to every entry in $g^{(i+1)}$ and clip the values outside the region $\mathbb{Z}_{>0}$. This operation aims to find the number of values missing in $\sum_{j=1}^{n} x_j^{(i)}$ to achieve the desired sum c. It is distributed uniformly among all n entries in $g^{(i+1)}$. Both gradient and projection operations are performed for each iteration until the given number of steps t (lines 2-5). Finally, we output the minimum squared l_2 solution $x^{(t)}$.

4.4 Computational Cost

The computational cost of the graph perturbation phase (Algorithm 1) consists of the time to process the edges in E in the original graph and the time to upgrade the m_0 new edges, which implies in expected time $O(|E|)$, since $m_0 \leq |E|$. The running time complexity of Algorithm 2 is $O(|V|.t)$ since it includes the post-processing projection to adjust node degrees. Constant t is the number of steps in the post-processing projection. Algorithm 3 runs in $O(|E|.log(|E|))$ because it sorts the edges in descending order of weights. As discussed earlier, the graph realization step also in Algorithm 3 has complexity $O(|E| + |V|)$. Finally, the complexity of the weights adjustment in Algorithm 4 is given by the post-processing projection on edge weights, which is $O(|E|.t)$. Therefore, the overall complexity of our approach is $O(|E|.t + |E|.log(|E|))$.

4.5 Summary

This chapter discussed the fundamental concepts and formal definitions related to differential privacy within count-weighted graphs. Initially, we motivated why traditional graph DP models, such as node and edge differential privacy, do not offer appropriate privacy guarantees, and why a new model is required. Then, we introduced the existing weighted DP model found in the literature and we discussed its limitations. We also presented our definition of neighboring weight graphs with unknown topology, in order to release weighted graphs via differential privacy considering that both the graph topology and the edge weights as private information.

Since we have established a new definition to address count-weighted graphs with unknown topology, we demonstrated an efficient graph perturbation technique based on a sampling strategy that does not materialize zero edge weights when the geometric mechanism is performed. Our method keeps weights in the perturbed graph with high noisy values, since they are most likely to correspond to non-zero edge weights in the original one, and also adds low noisy weight values with probability proportional to their weights.

Additionally, we improved the utility of the perturbed graph by developing post-processing techniques to preserve as much as possible the original node degrees and the sum of all edge weights. In particular, for node degrees adjustment, we proposed an algorithm to swap edges such that the node degrees of the adjusted graph is as close as possible to the perturbed one. In the weights adjustment step, we proposed and solved an optimization problem to avoid introducing bias in the weights and to preserve some original graph characteristics. Finally, we proved that our methods, when run in sequence, have an overall complexity $O(|E|.t + |E|.log(|E|))$.

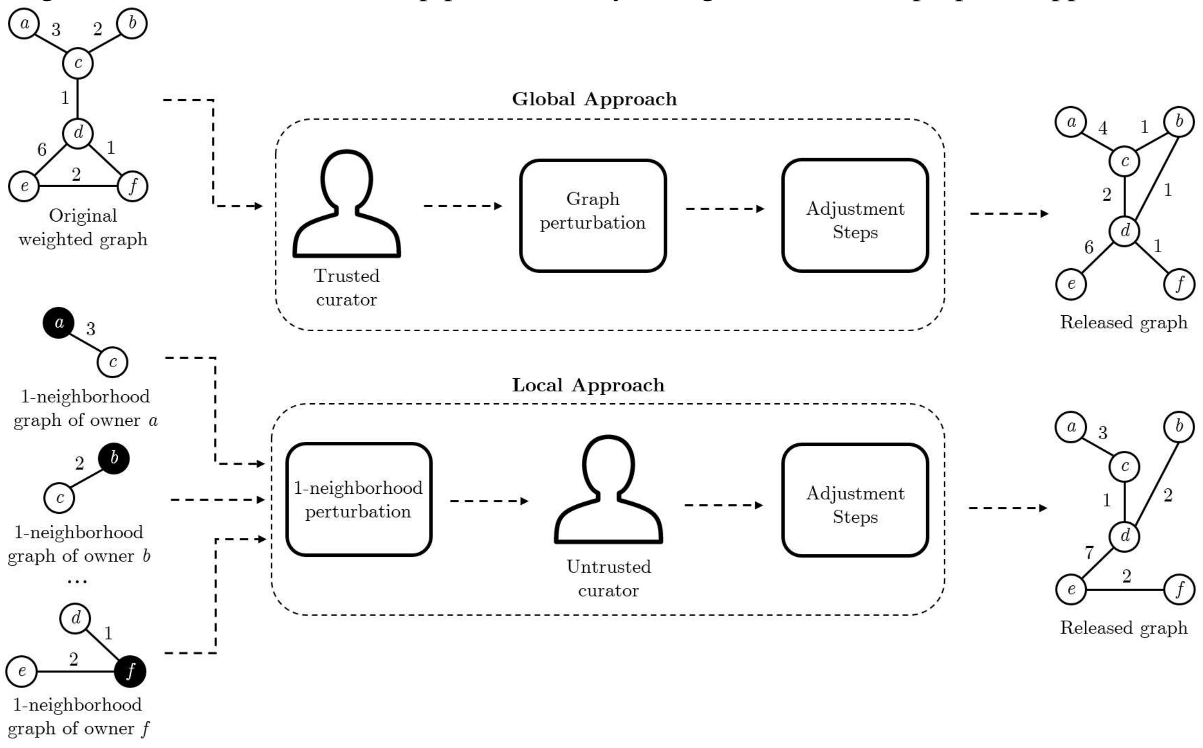
5 GLOBAL AND LOCAL DIFFERENTIALLY PRIVATE APPROACHES

In this chapter, we initially introduce our global differential privacy approach using the contributions presented previously in the entire dataset, i..e., in a global view of the data. We extend our results to propose a new method to release count-weighted graphs under the local differential privacy setting, where the random perturbation is performed on the user side. This chapter also provides running examples to effectively demonstrate and exemplify the utilization of both approaches.

In global DP, there is a trusted curator who has access to the entire graph and applies privacy-preserving mechanisms to ensure the privacy of the overall dataset during analysis or processing. The curator ensures that any information released from the dataset adheres to the principles of differential privacy. In particular, the trusted curator obtains the original weighted graph, applies the graph perturbation step, and subsequently, adjusts the node degrees and the edge weights before releasing its perturbed version. The necessity of a trusted curator makes the existing global-DP methods unsuitable in some scenarios, such as decentralized systems where data is distributed across multiple, autonomous entities (ERLINGSSON *et al.*, 2014). In such cases, the reliance on a single trusted curator contradicts the principles of decentralization and poses a potential single point of failure. Additionally, in scenarios involving highly sensitive or competitive data, entrusting a third party with the role of curator may not be feasible due to concerns about data leakage or misuse. Furthermore, in environments with limited trust or where establishing a trusted curator is logistically challenging, alternative privacy-preserving techniques that do not rely on a central authority become more desirable.

Local DP (LDP) has risen in popularity due to its adoption in both academia and industry. LDP provides much stronger privacy protection when compared to the traditional global DP, since the perturbation is performed by the users (nodes), not by the data curator. LDP has been originally adopted to protect each user's data independently from the other users. However, in the graph context, it should consider that an edge contains information about two users. Edge weight LDP (Definition 14) assumes that the edge connecting the user v to user u and the edge connecting the user u to user v are different pieces of information. Particularly, user u must trust user v to not leak information about their shared edge. Figure 15 presents an overview of the pipeline used by both global and local proposed approaches. Detailed information regarding both approaches is presented in this chapter.

Figure 15 – An overview of the pipeline used by both global and local proposed approaches.



Source: Elaborated by the author.

5.1 The Global Approach

In this section, we describe our global approach to releasing weighted networks under DP. The main idea is to first privately query important graph statistics, then add noise to both graph structure and edge weights. In the end, we use the queried statistics to post-process the perturbed graph and recover the original graph characteristics before releasing it.

The global approach consists of four main general phases: (1) statistics extraction; (2) graph perturbation; (3) degrees adjustment; and (4) weights adjustment. These phases are presented in details in Chapter 4. Algorithm 5 presents the global approach and their respective phases.

Algorithm 5: Global DP Releasing of Weighted Graphs

```
   Input  : A graph $G = (V, E, \omega)$, privacy budgets $\varepsilon_1$, $\varepsilon_2$, $\varepsilon_3$
   Output : Sanitized graph $\bar{G}$
1  $\bar{D}, \bar{s} \leftarrow$ Statistics_Extraction$(G, \varepsilon_1, \varepsilon_2)$
2  $\bar{m} \leftarrow \sum_{i=1}^{n} \bar{D}_i / 2$ // obtaining the number of expected edges
3  $\tilde{G} \leftarrow$ Graph_Perturbation$(G, \bar{m}, \varepsilon_3)$
4  $G^* \leftarrow$ Degrees_Adjustment$(\tilde{G}, \bar{D})$
5  $\bar{G} \leftarrow$ Post-processing_Projection$(G^*, \bar{s})$
6  return $\bar{G}$
```

The global approach starts privately querying the node degrees and the total sum of edge weights (line 1) using privacy budgets ε_1 and ε_2, respectively. Recall that neighboring weight graphs (Definition 12) can be seen as a relaxation of the traditional edge-DP model. Under edge-DP, the number of edges is not publicly known (XIAO *et al.*, 2014). Thus, we assume the total number of edges in our thesis is also private. To keep the expected number of edges, we obtain it by the sum of the post-processed node degrees divided by 2 (line 2), since the sum of the degree values is twice the number of edges. It is plausible to make this assumption because, in expectation, the sum of all node degrees in the original graph is similar to the sum of all node degrees in the perturbed graph, i.e., $\mathbb{E}[\sum_{i=1}^{n} D_i] = \mathbb{E}[\sum_{i=1}^{n} \tilde{D}_i]$. It occurs because noises introduced by the geometric mechanism are independently generated from a zero-mean geometric distribution.

In line 3, the algorithm efficiently perturbs the input graph using a sampling strategy. Finally, the adjustments are made in the degrees and in the weights (lines 4 and 5). Figure 16 illustrates a running example of the proposed global approach.

The running example of the global approach starts as follows.

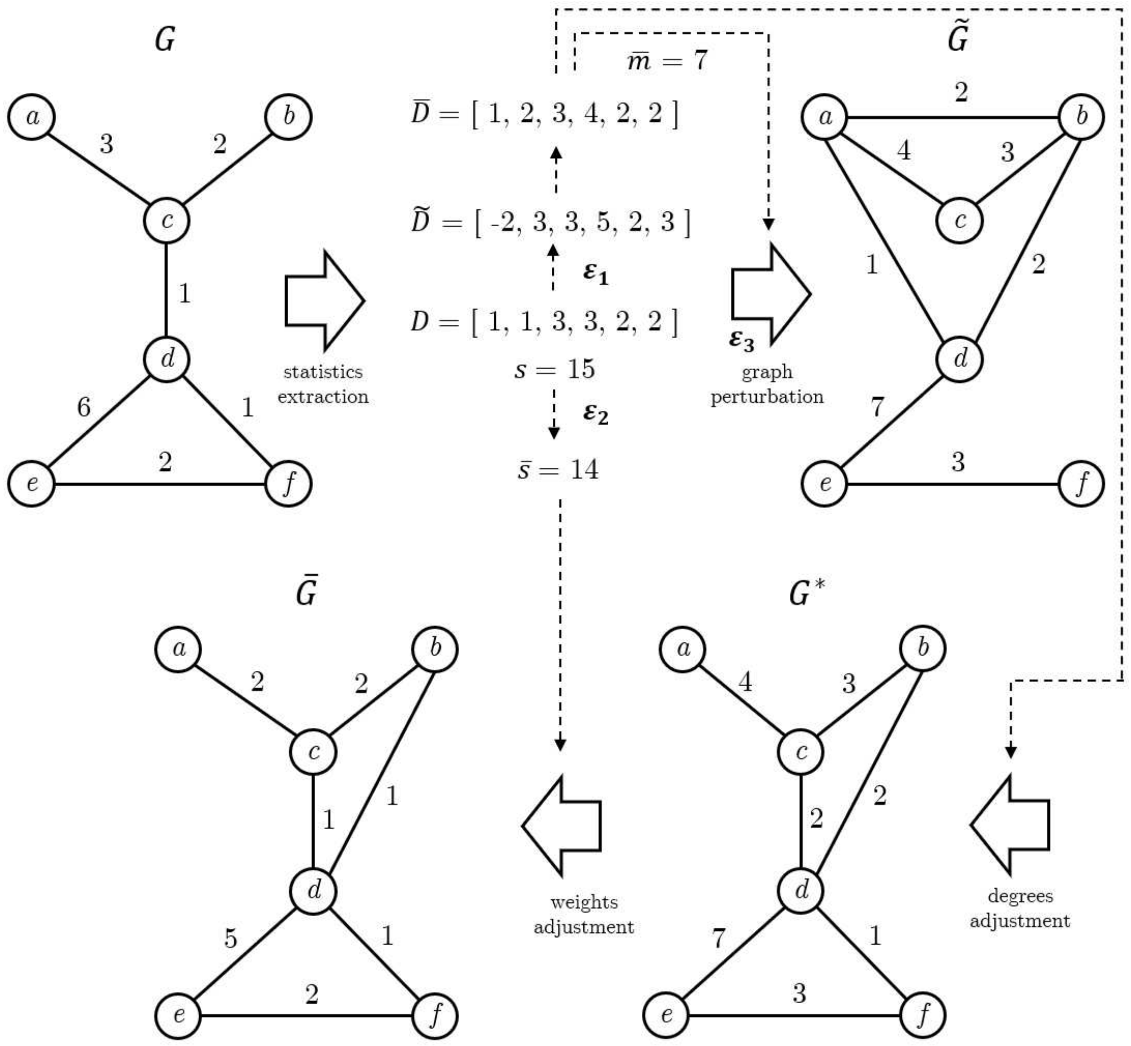
Example 2. *Consider the original weighted graph G in Figure 16. It has original degrees D = [1, 1, 3, 3, 2, 2] and the total sum of edge weights (constant s) equal to 15. We first apply the geometric mechanism to D with* ε_1 *and get* $\tilde{D}$ *= [-2, 3, 3, 5, 2, 3]. Next, we post-process the node degrees to satisfy domain constraints and obtain* $\bar{D}$ *= [1, 2, 3, 4, 2, 2]. In addition, the noisy number of edges* $\bar{m}$ *= 7 is established from the consistent degrees* $\bar{D}$*, by summing up all their values and dividing it by 2. Finally, we apply the geometric mechanism to the sum of edge weights with* ε_2 *and get* $\bar{s}$ *= 14.*

The graph perturbation phase is also exemplified as follows.

Example 3. *Consider the graph G in Figure 16. The number of necessary edges* $\bar{m}$ *in* $\tilde{G}$ *was found in the previous steps. Consider the threshold* $\tau = 3$ *for the desired number of edges* $\bar{m} = 7$*. For every existing edge* (u,v) *in G, we add geometric noise with* ε_3 *to* $\omega(u,v)$ *to get* $\omega'(u,v)$ *and add it to* $\tilde{G}$ *with probability* $p_{uv} = min(\frac{\tilde{\omega}(u,v)}{3}, 1)_{+}$*. Four edges* (a,c)*,* (b,c)*,* (d,e) *and* (e,f) *remain in* $\tilde{G}$ *since their noisy weights are greater than or equal to* τ*. For the non-existing edges, we uniformly at random select three edges:* (a,b)*,* (a,d) *and* (b,d)*. We draw their noisy weights from the probability distribution function mentioned previously.*

We use the Example 4 to illustrate the degrees adjustment step.

Figure 16 – A running example of the proposed global approach.



Source: Elaborated by the author.

Example 4. *Consider the graph $\tilde{G}$ in Figure 16. Note that some node degrees do not correspond to the expected values in $\bar{D}$. For instance, node "a" has degree equal to 3 in $\tilde{G}$, while it should be equal to 1. We first add edge (d,e) to G^* following the descending order of weight. The remaining degree sequence after such addition is $\bar{D}$ = [1, 2, 3, 3, 1, 2]. Next, we add edges (a,c), (b,c) and (e,f) to G^* and $\bar{D}$ is converted to [0, 2, 2, 3, 1, 2], [0, 1, 1, 3, 1, 2] and [0, 1, 1, 3, 0, 1], respectively. The next edge to be added should be (a,b). However, node a has already reached the number of allowed edges, since its value in $\bar{D}$ is now equal to 0. Thus, its insertion is skipped. Finally, (b,d) is added to G^* and (a,d) is also skipped. The resulting $\bar{D}$ is now [0, 0, 1, 2, 0, 1]. Following the remaining $\bar{D}$, the final graph is realized. Edge (a,b) in $\tilde{G}$ becomes (c,d) in G^* and edge (a,d) in $\tilde{G}$ becomes (d,f) in G^*. In the end, $\bar{D}$=[0, 0, 0, 0, 0, 0] and node degrees in G^* are, in expectation, close to the original ones.*

The final step of the proposed global approach is to adjust the edge weights.

Example 5. *Consider the graph G^* in Figure 16. The total sum of edge weights is equal to 22. However, the expected sum $\bar{s}$ should be equal to 14. In order to achieve this goal, we post-process all the weights in G^* and we find a solution that preserves this sum and that also guarantees the weights are positive. Finally, the graph $\bar{G}$ is released, including these new edge weights.*

5.2 Local Approach

Our local approach adopts the same ideas as its global counterpart, but in two main phases: (i) a local neighborhood perturbation in each user and (ii) a graph data aggregation in the curator side.

5.2.1 Local Neighborhood Perturbation

In the LDP setting, each node has its own local neighborhood. In this work, we consider the neighborhood of a node v to be composed of all nodes connected to v along with their edges and their respective weights. Algortithm 6 provides a detailed description of the local neighborhood perturbation phase.

Algorithm 6: Local Neighborhood Perturbation

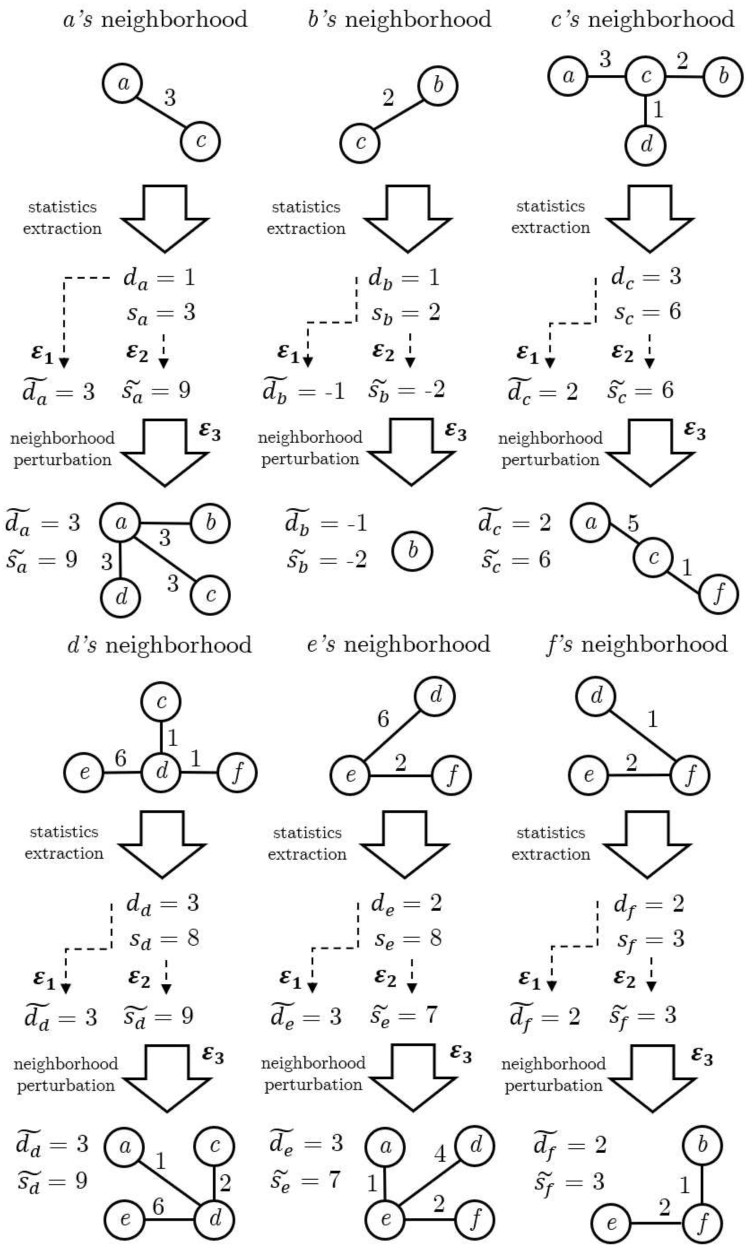
```
Input  : All user's neighboring weight lists $\gamma = (\gamma_{v_1}, ..., \gamma_{v_n})$, privacy budgets $\varepsilon_1$, $\varepsilon_2$, $\varepsilon_3$
Output : All noisy neighboring weight lists $\tilde{\gamma}$, noisy degrees $\tilde{D}$ and noisy sum of
         weights $\tilde{S}$
1  $\tilde{D}, \tilde{S}, \tilde{\gamma} \leftarrow \emptyset$
2  $\alpha_1 \leftarrow e^{-\varepsilon_1/2}, \ \alpha_2 \leftarrow e^{-\varepsilon_2}, \ \alpha_3 \leftarrow e^{-\varepsilon_3}$
3  foreach $\gamma_{v_i} \in \gamma$ do
4      $d_{v_i} \leftarrow |\gamma_{v_i}|$ // counting of non-zero edge weights, i.e., the $v_i$ degree
5      $s_{v_i} \leftarrow \sum \gamma_{v_i}$ // sum of incident edge weights
6      $\tilde{d_{v_i}} \leftarrow d_{v_i} +$ Geometric_Mechanism($\alpha_1$)
7      $\tilde{s_{v_i}} \leftarrow s_{v_i} +$ Geometric_Mechanism($\alpha_2$)
8      $\gamma'_{v_i} \leftarrow$ Graph_Perturbation($\gamma_{v_i}, \tilde{d_{v_i}}, \varepsilon_3$) // perturbing each neighborhood
        centered at node $v_i$
9      $\tilde{\gamma_{v_i}} \leftarrow$ Post-processing_Projection($\gamma'_{v_i}, \tilde{s_{v_i}}$) // adjusting the sum of
        edge weights based on $\tilde{s_{v_i}}$
10     add perturbed local neighborhood $\tilde{\gamma_{v_i}}$ to $\tilde{\gamma}$
11     add noisy node degree $\tilde{d_{v_i}}$ to $\tilde{D}$
12     add noisy sum of incident edge weights $\tilde{s_{v_i}}$ to $\tilde{S}$
13 end
14 return $\tilde{\gamma}$, $\tilde{D}$, $\tilde{S}$
```

Initially, for each node v_i neighborhood of the graph (line 3), denoted γ_{v_i}, we locally extract both the degree (line 4) and the sum of incident edge weights (line 5), as well as the global approach. In lines 6 and 7 we add geometric noise to these values with privacy budgets ε_1 and ε_2, respectively, to get their noisy versions. Then, we apply the priority sampling strategy to perturb each neighborhood centered at a given node v_i with ε_3 in line 8. The number of expected edges in the local approach is given by the noisy queried degree, denoted $\tilde{d_{v_i}}$. As previously mentioned in Chapter 5.1, edge weights with high noisy values are most likely to correspond to non-zero edge weights in the neighborhood of the original v_i. However, some zero-edge weights may be included in the neighborhood. In this case, these edges must be incident to the central node. Note that, if the noisy degree or the noisy sum of edge weights assumes negative values, only the central node remains after the neighborhood perturbation phase, i.e., without any incident edge. In line 9, we post-process the weights of v_i's neighborhood considering the noisy sum of incident edge weights, denoted $\tilde{s_{v_i}}$, to preserve the number of records in the original dataset. Finally, each noisy neighborhood is sent to the data curator, along with its noisy degree and noisy sum of weights. The example in Figure 17 illustrates this phase.

The running example of the local neighborhood perturbation approach is given as follows.

Example 6. *Our goal is to release the weighted graph G in Figure 16 under the local approach. Consider the node a's neighborhood in Figure 17. Originally, it has degree equal to one ($d_a = 1$) and the sum of edge weights equal to three ($s_a = 3$). We first add geometric noise to both d_a and s_a to get their noisy versions $\tilde{d_a}$ and $\tilde{s_a}$. The expected noisy a's neighborhood has degree equal to 3 and sum of edge weights equal to 9. Let the threshold $\tau = 2$ for the desired number of edges $\tilde{d_a} = 3$. In this case, the original edge (a,c) remains in the noisy a's neighborhood, since its noisy weight is greater than τ. Additionally, edges (a,b) and (a,d) are uniformly at random selected and their noisy weights are drawn from the distribution mentioned in global approach. Next, the weights of a's neighborhood are post-processed considering the noisy sum of incident edge weights $\tilde{s_a} = 9$. The whole process is repeated for each node's neighborhood in G. Note that, in node b's neighborhood, only the central node remains after neighborhood perturbation step, since its noisy degree is negative. In the end, each noisy neighborhood, noisy degree and noisy sum of weights are sent to the data curator.*

Figure 17 – Local neighborhood perturbation example.



Source: Elaborated by the author.

5.2.2 *Graph Data Aggregation and Release*

This phase aims to guarantee the graph consistency after each local neighborhood perturbation. Algorithm 7 shows the details of this phase. Initially, the data curator aggregates all noisy degrees (in $\tilde{D}$) and post-processes them to satisfy domain constraints to obtain $\bar{D}$ (line 1). This step guarantees that node degrees assume positive values. The expected number of edges $\bar{m}$ is also obtained, by summing up all values in $\bar{D}$ and dividing it by 2 (line 2). Next, the curator aggregates all the sum of incident edge weights in $\tilde{S}$ to compute the expected total sum of edge weights in the released graph. It is obtained by summing up all values in $\tilde{S}$ and then dividing it by 2 (line 3).

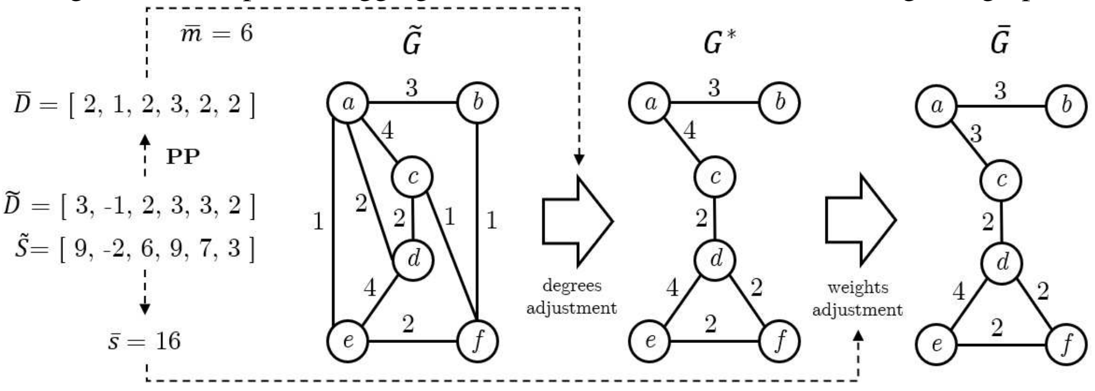
Algorithm 7: Data Aggregation and Release

```
Input  : All noisy neighboring weight lists $\tilde{\gamma}$, noisy degrees $\tilde{D}$ and noisy sum of
         weights $\tilde{S}$
Output : Sanitized graph $\bar{G}$
1 $\bar{D} \leftarrow$ Post−processing_Projection($\tilde{D}, \sum_{i=1}^{n} \tilde{D}_i$) // preserving the noisy sum
    of all node degrees and guaranteeing all nodes have positive
    degrees
2 $\bar{m} \leftarrow \sum_{i=1}^{n} \bar{D}_i/2$ // obtaining the number of expected edges
3 $\bar{s} \leftarrow \frac{\sum_{i=1}^{n} \tilde{S}_i}{2}$ // computing the expected total sum of edge weights
4 $\tilde{G} \leftarrow$ Merge_Neighborhoods($\tilde{\gamma}, \bar{m}$) // merging each weight list to a single
    graph, keeping the expected number of edges
5 $G^* \leftarrow$ Degrees_Adjustment($\tilde{G}, \bar{D}$)
6 $\bar{G} \leftarrow$ Post-processing_Projection($G^*, \bar{s}$)
7 return $\bar{G}$
```

In line 4, the merged graph $\tilde{G}$ is built by adding all the edges in each noisy neighborhood, including their weights. For edges that are present in two neighborhoods, the curator computes the average for the two weight values. The next step is to post-process the merged graph $\tilde{G}$ to adjust the degrees based on the expected $\bar{D}$ and $\bar{m}$. The curator creates an empty graph G^* and starts adding the edges to G^* following the descending order of weight in $\tilde{G}$ (line 5). Finally, in line 6 the expected total sum of edge weights is adjusted similarly to the global strategy and the data curator releases the graph $\bar{G}$. Example 7 illustrates how this phase guarantees graph consistency.

Example 7. *In the example of Figure 18, the data curator first aggregates the noisy degrees,* $\tilde{D} = [3,-1,2,3,3,2]$, *post-processes them to satisfy domain constraints and obtain* $\bar{D} = [2,1,2,3,2,2]$. *The curator also gets the expected number of edges* $\bar{m} = 6$. *Additionally, the noisy total sum of edge weights* $\bar{s} = 16$ *is obtained. The merged graph* $\tilde{G}$ *is built by adding all the edges in each noisy neighborhood. Note that, the edge* (a,c) *has noisy weight equal to 3 in node a's neighborhood, while it has noisy weight equal to 5 in node c's neighborhood. Then, edge* (a,c) *assumes weight equal to 4 in* $\tilde{G}$ *(Figure 18), i.e., the average of the values. The degrees of the merged graph* $\tilde{G}$ *are then adjusted based on the expected* $\bar{D}$ *and* $\bar{m}$*. The curator creates an empty graph* G^* *and initially the edges* (a,c), (d,e), (a,b), (c,d) *and* (e,f) *are added. The remaining* $\bar{D}$ *at this point is equal to [0, 0, 0, 1, 0, 1]. Then, the edge* (d,f) *is added to* G^* *with weight equal to 2, since it follows the descending order of weights. The edges with lowest weights* (a,e), (b,f) *and* (c,f) *in* $\tilde{G}$ *do not remain in* G^**. Finally, the curator adjusts the expected total sum of edge weights, from 17 in* G^* *to 16 in* $\bar{G}$ *and releases it.*

Figure 18 – Graph data aggregation and release for local DP on weighted graphs.


$\bar{m} = 6$
$\bar{D} = [\,2, 1, 2, 3, 2, 2\,]$
PP
$\tilde{D} = [\,3, -1, 2, 3, 3, 2\,]$
$\tilde{S} = [\,9, -2, 6, 9, 7, 3\,]$
$\bar{s} = 16$
$\tilde{G}$
G^*
$\bar{G}$
degrees adjustment
weights adjustment

Source: Elaborated by the author.

It is worth mentioning that the final output of the global approach may be different from that of the local one, since the graph perturbation and adjustments are performed in different phases.

5.3 Summary

In this chapter, we presented our approach to releasing weighted graphs under global-DP. First, we privately computed important statistics that provided a good balance of the original graph. The statistics computed were (1) the node degrees and, (2) the sum of all edge weights. We then adopted a sampling strategy to perturb the original weighted graph, since filtering techniques introduce additional bias when compared to sampling approaches. Our strategy overrode the costly operation of materializing all the edge weights. Additionally, we included two adjustment steps to recover as much as possible of the original graph characteristics before its release: degrees and weights adjustments. Finally, we provided details about the post-processing step, applied during the mentioned phases, to guarantee the consistency of the released graph.

We also presented in this chapter our approach to releasing weighted graphs under local-DP. We argued that our local approach works similarly to the global one. The difference is that the perturbation and the adjustment steps are performed by each user. On the user side, we privately computed the node degrees and the sum of all edge weights. Then we applied the priority sampling strategy to perturb each neighborhood centered on a given user. We included the adjustment steps and recovered as much as possible of the original neighborhood characteristics before the noisy neighborhood was sent to the data curator. On the curator side, additional steps were added to guarantee the consistency of the released graph.

6 EXPERIMENTAL EVALUATION

In this chapter, we empirically evaluate the effectiveness of our global and local approaches on a variety of real-world weighted graphs. Experiments were carried out in Linux 64-bit, Intel(R) Core(TM) i7-7820X CPU @ 3.60GHz CPU and 128GB RAM. We implemented both approaches in Python with the graph-tool package (PEIXOTO, TIAGO P., 2014). For each dataset, we repeat each experiment 10 times and report the average results. The number of steps t and the step size η in `PostProcessing_Projection` are set to 10 and 0.1, respectively. In our experiments, we vary the privacy budget ε from 0.1 to 1.0. As discussed in Section 2.3, to set a proper privacy budget for an application, experts, stakeholders, and data users have to provide extensive feedback to ensure that the privacy of individuals is sufficiently protected while maintaining high levels of accuracy in the released information (United States Census Bureau, 2021). Therefore, how to properly set the privacy budget is outside the scope of this paper, and we use values of the privacy budget commonly used by other works in this area.

6.1 Datasets

We conducted experiments over four real-world network datasets from different domains and characteristics. The statistics of these datasets are summarized in Table 2 and detailed as follows.

Table 2 – Statistics of graph datasets.

	HS contact	Reality call	Enron	DBLP
Nodes	327	6,809	37.9k	1.9M
Edges	5,818	87,680	250k	4M
deg_{avg}	35.5	25.7	9.2	4.18
deg_{max}	87	283	1,718	1,088
ω_{avg}	32.4	1.5	15.9	1.7
ω_{max}	2,949	966	9,080	325

– *High-school contact*: This is a very small dense network with high weight values where nodes correspond to students and edge weights are the number of interactions of two students in contact (MASTRANDREA *et al.*, 2015). In light of our proposed notion of neighboring weight graphs, in this dataset we aim to protect the presence or absence of an interaction.

– *Reality call*: This is a small dense network where a node represents a person, an edge indicates a phone call, and edges weights are the number of phone calls between two people. Based on our definition of neighboring weight graphs, the object of privacy protection is the existence of a phone call between two people.

– *Enron*: This is a medium-sized sparse network with high weight values. Each node is an email address and an edge connects a pair of addresses. Edge weights represent the number of emails exchanged. In this dataset, our model aims to preserve the presence or absence of an exchanged email.

– *DBLP*: This is a sparse and large network with low edge weight values. Nodes are authors and edges represent the co-authorships among them. Different from the traditional DBLP dataset, where two authors are connected if they published at least one paper together, in this dataset we consider two authors to be connected if one of them is the first author. In other words, instead of forming a clique of co-authors for each paper, we form a star graph centered at the first author for each paper, with each co-author connected with the first author through an edge. Edge weights represent the number of such co-authorships for each pair of authors. This definition aligns with our proposed notion of privacy in neighboring weight graphs, i.e., preserving the privacy of the presence or absence of a co-authorship in a paper.

Figure 19 exhibits the main characteristics of the tested weighted graphs, in terms of sparsity and heaviness.

Figure 19 – Main characteristics of the tested weighted graphs, in terms of sparsity and heaviness.

	Strongly connected	Weakly connected
Heavy	**HS Contact**	**Enron**
Non-heavy	**Reality Call**	**DBLP**

Source: Elaborated by the author.

6.2 Baselines

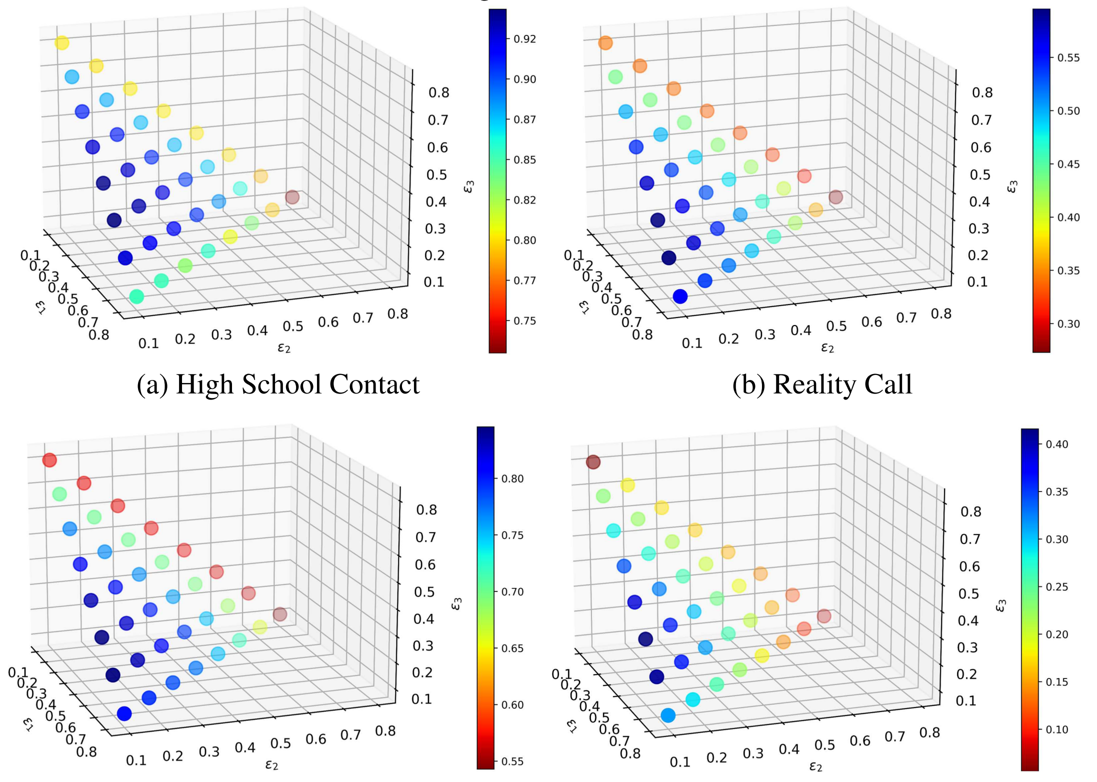
We compare our two approaches with the naive geometric mechanism (GHOSH *et al.*, 2012), exponential mechanism (MCSHERRY; TALWAR, 2007), log-laplace (NY; PAPPAS, 2013), truncation (HARDT; ROTH, 2012), high-pass filter (HPF) (CORMODE *et al.*, 2012) and priority sampling (PS) (DUFFIELD *et al.*, 2007). Of these, geometric, log-laplace, and truncation techniques consider the graph structure to be public and provide a good baseline for understanding the quality of our results, since we propose going beyond the scope of the works that assume the graph structure is public. Specifically, for some experiments, we use three different values in the truncation approach to limit the maximum edge weights: $\theta = \Delta W^{1/2}, \theta = \Delta W^{1/3}$ and $\theta = \Delta W^{1/4}$, where ΔW denotes the maximum weight.

The baseline exponential mechanism implemented in this thesis employs a utility function u that measures how far a candidate graph g is from the original graph G, where g and G share the same set of nodes (which are publicly known). Thus, we adopt a utility function $u(G,g) = -d(G,g)$, where $d(G,g)$ denotes the number of unit edge weight modifications on G to obtain g. All graphs that can be obtained from G through modification of one edge weight by one unit have $u = -1$. Similarly, all graphs that can be minimally obtained from G through edge weight modifications of two units (whether by modifying one edge by two units, or two distinct edges by one unit each) have $u = -2$, and so on. This general approach of a utility function based on distances has achieved good results in DP literature (ZHANG *et al.*, 2015; MEDINA; GILLENWATER, 2020), since it has low sensitivity and is easy to compute. Based on our definition of neighboring weight graphs (Definition 12), the sensitivity Δu of our utility function is equal to 1, because the maximum possible change in the mechanism output, when we modify one arbitrary edge weight in the input, is 1.

A naive implementation of the proposed approach is to iterate over all output weighted graphs and assign probability proportional to $\exp(\frac{\varepsilon.u}{2})$. However, since the number of output graphs grows exponentially with the number of nodes, this implementation is not practical. To overcome this limitation, we group output graphs with the same utility from G and apply the exponential mechanism.

First, we compute the probability mass function of each group L_i, which is measured by the probability mass of a graph at a specific group i times the number of graphs at that group, i.e., $P(L_i) \propto \exp(\frac{\varepsilon.(-i)}{2}).|L_i|$. Next, we pick a group i based on its probability $P(L_i)$. Finally, we randomly sample an output graph from the picked group. It is worth mentioning that the baseline exponential mechanism usually outputs graphs with many positive- weighted edges that originally did not exist, i.e., dense graphs. Consequently, we do not present an experimental evaluation of the baseline EM for the DBLP dataset due to its sparsity. In this thesis, we do not perform any type of adjustment step in the baselines.

Figure 20 – Similarity between the original and the perturbed graphs with different privacy budget allocations.



(a) High School Contact (b) Reality Call

(c) Enron (d) DBLP

Source: Elaborated by the author.

6.3 Privacy Budget Allocation

The use of the total privacy budget ε needs to be carefully allocated in each phase. Recall that our proposed approaches divide the entire budget into three parts: ε_1 to query node degrees, ε_2 to query the total sum of edge weights and ε_3 to perturb the graph ($\varepsilon_1 + \varepsilon_2 + \varepsilon_3 = \varepsilon$). In order to determine which is the best privacy budget allocation, we empirically measure how the original graph G and the perturbed one $\bar{G}$ are similar for different privacy budget combinations (the similarity definition is presented in Chapter 6.4). This analysis is shown in Figure 20. In this experiment, $\varepsilon = 1$. Note that, for all studied datasets, the similarity values are higher when $\varepsilon_1 \approx 0.6, \varepsilon_2 \approx 0.1$ and $\varepsilon_3 \approx 0.3$. It makes sense to use this combination of budgets. Intuitively, when querying the total sum of edge weights with ε_2, the noise magnitude is low, since this sum is usually quite large. So, we do not have to allocate a high budget. One can assume the distributions of weights and degrees are similar. Then we could split the remaining budget in two equal parts for ε_1 and ε_3. However, the sensitivity of querying degrees is twice the sensitivity of querying weights (Theorem 8). Therefore, it makes sense to assign $\varepsilon_1 = 2\varepsilon_3$.

6.4 Utility Evaluation

In this section, we first define some metrics to evaluate the utility of the released graphs. Then, we compare our approaches with the mentioned baselines.

6.4.1 Graph Statistics

We adopt the following statistics to measure the utility of a graph $G = (V, E, \omega)$:

– Graph similarity (*Sim*): it is a general metric that allows for comparing graphs with different edges and different edge weights. Since some baseline approaches only perturb edge weights, not the edges themselves, the similarity measure can take that into account. In this thesis we adopt an iterative method to measure the similarity between two graphs, based on the fact that two graphs are similar if their both neighbourhoods and weights are similar (KOUTRA *et al.*, 2011). Then, the graph similarity is computed by:

$$Sim(G_1, G_2) = \frac{\left(\sum_{i \leq j} |A_{ij}^{(1)}| + |A_{ij}^{(2)}|\right) - \left(\sum_{i \leq j} |A_{ij}^{(1)} - A_{ij}^{(2)}|\right)}{\left(\sum_{i \leq j} |A_{ij}^{(1)}| + |A_{ij}^{(2)}|\right)}, \tag{6.1}$$

where $A^{(1)}$ and $A^{(2)}$ are weighted matrices of two graphs G_1 and G_2, respectively. The higher the score is, the higher the similarity between G_1 and G_2.

– Sum of all edge weights (*SEW*): it is the total weight of all the edges in the graph. It also corresponds to the original number of records in a dataset of interactions, since each unit of weight corresponds to one record in the original data, as exemplified in Figure 1a. This metric is computed by:

$$SEW = \sum_{(u,v) \in E} \omega(u, v) \tag{6.2}$$

– Average weighted shortest path length (AWSP): it indicates the level of integration of the weighted graph. It has a strong effect on various dynamics networks statistics, such as synchronization (NISHIKAWA *et al.*, 2003), random walks (CONDAMIN *et al.*, 2007), among others. Let $d(u, v)$, where $u, v \in V$ denote the shortest weighted distance between u and v. Assume that $d(u, v) = 0$ if v cannot be reached from u. Then, the average weighted shortest path length is given by:

$$AWSP = \frac{1}{n(n-1)} \sum_{u \neq v} d(u,v) \tag{6.3}$$

where n is the number of nodes in the graph.

– Global clustering coefficient: is a measure of the density of triangles in the graph. It is computed as the sum of the weights of all triangles in the graph, divided by the sum of the weights of all possible triangles (triples). The global clustering coefficient takes into account both the existence of triangles and the strengths of the connections between the nodes that form them. A higher value of the global clustering coefficient indicates a higher degree of clustering in the graph. The global clustering coefficient of a weighted graph is given by:

$$C = \frac{A^3}{\sum_{i \neq j} [A^2]_{ij}} \tag{6.4}$$

where A is the weighted matrix and it is assumed that the weights are normalized, i.e., $A_{ij} \leq 1$.

– Node strength (NS): it is a measure of the importance of a node based on the weighted connections it has with other nodes in the graph. More specifically, the node strength of node v is defined as the sum of the weights of all the edges incident on node v, which is given as follows:

$$NS(v) = \sum_{u \in N(v)} \omega(u,v) \tag{6.5}$$

where $N(v)$ is the set of neighbors of node v.

– Sum of neighboring node strength (SNNS): it refers to the sum of the node strengths that are directly connected to a particular node v in the graph. This measure provides useful information about the influence of a node in a network, as nodes with a higher sum of neighboring node strengths are generally more connected and have a greater impact on the flow of information or resources in the network. It is defined as:

$$SNNS(v) = \sum_{u \in N(v)} NS(u) \tag{6.6}$$

– Weighted PageRank (WPR): the traditional PageRank is a widely used algorithm for ranking web pages in search engine results. The weighted PageRank is an extension of the PageRank algorithm that takes into account the edge weights. It assigns a score to each node in the graph based on the number and weights of incoming edges to that node. The higher the weight of an incoming edge, the higher the score assigned to the node. The weighted PageRank formula for a particular node v is:

$$WPR(v) = \frac{1-d}{n} + d\sum_{i=1}^{n} \frac{\omega(v,i)\cdot WPR(i)}{deg(i)} \tag{6.7}$$

where d is the damping factor (typically set to 0.85 in practice), n is the total number of nodes in the graph and $deg(i)$ is the degree of node i.

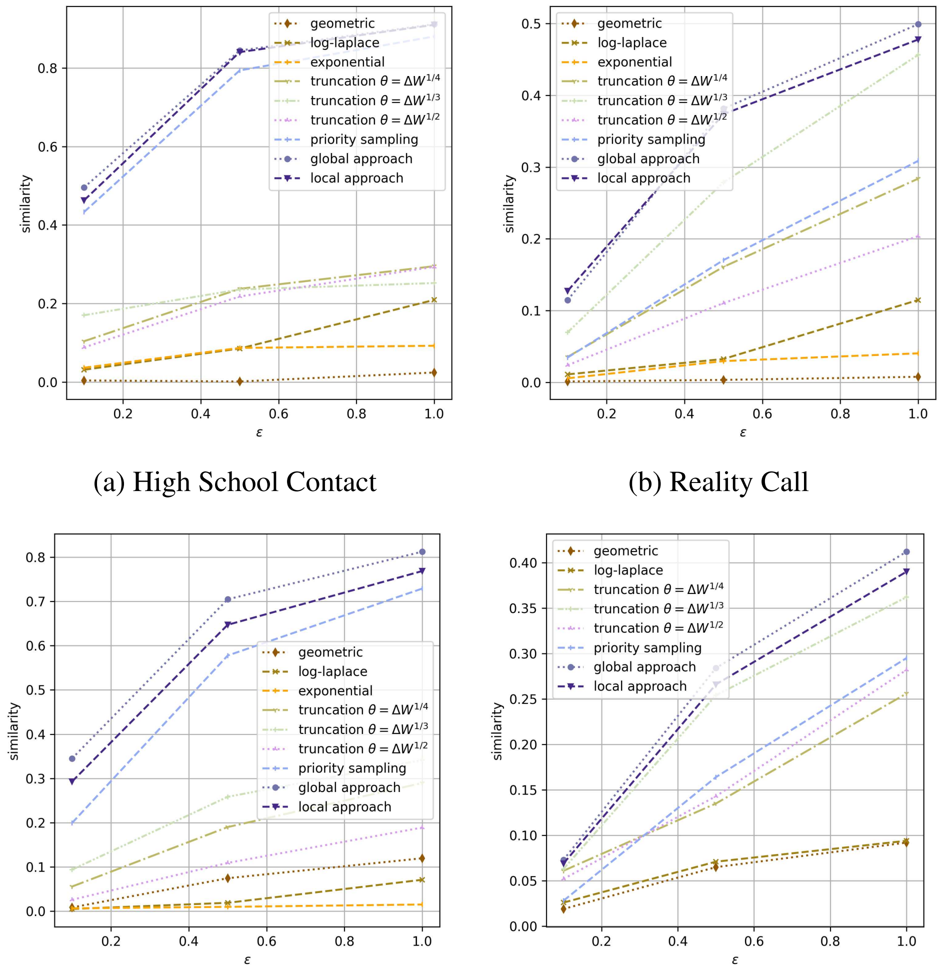
6.4.2 Utility Analysis

We evaluate the results in four different groups of utility metrics: graph similarity, distributions of degrees and weights, general graph statistics, and node centrality measures. For graph similarity and weight distribution, we compare our two approaches with the geometric mechanism, exponential mechanism, log-laplace, truncation, and priority sampling techniques. For the distribution of degrees, we compare with exponential mechanism, high-pass filter, and priority sampling because other techniques assume the graph topology is public and consequently preserve their original node degrees. For general graph statistics and node centrality measures, we compare with exponential mechanism, log-laplace, truncation, and priority sampling. In this work, we do not compare our approaches against the works proposed by Wang et al (WANG *et al.*, 2020) and Ning et al (NING *et al.*, 2021), since these works compose two distinct notions of differential privacy for graphs and they do not establish a DP property satisfied by this composition.

6.4.2.1 Graph similarity

Figure 21 shows the results for similarity between the original and the perturbed graphs. These results indicate a superiority of both global and local approaches in all datasets as ε increases. Note that for non-heavy graphs, such as DBLP and reality call, the achieved similarity is below 50% for all investigated approaches. It occurs because the number of edges which have the same source and target originally decreases significantly in the perturbed graphs.

Figure 21 – Similarity between the original and the perturbed graphs.



(a) High School Contact (b) Reality Call

(c) Enron (d) DBLP

Source: Elaborated by the author.

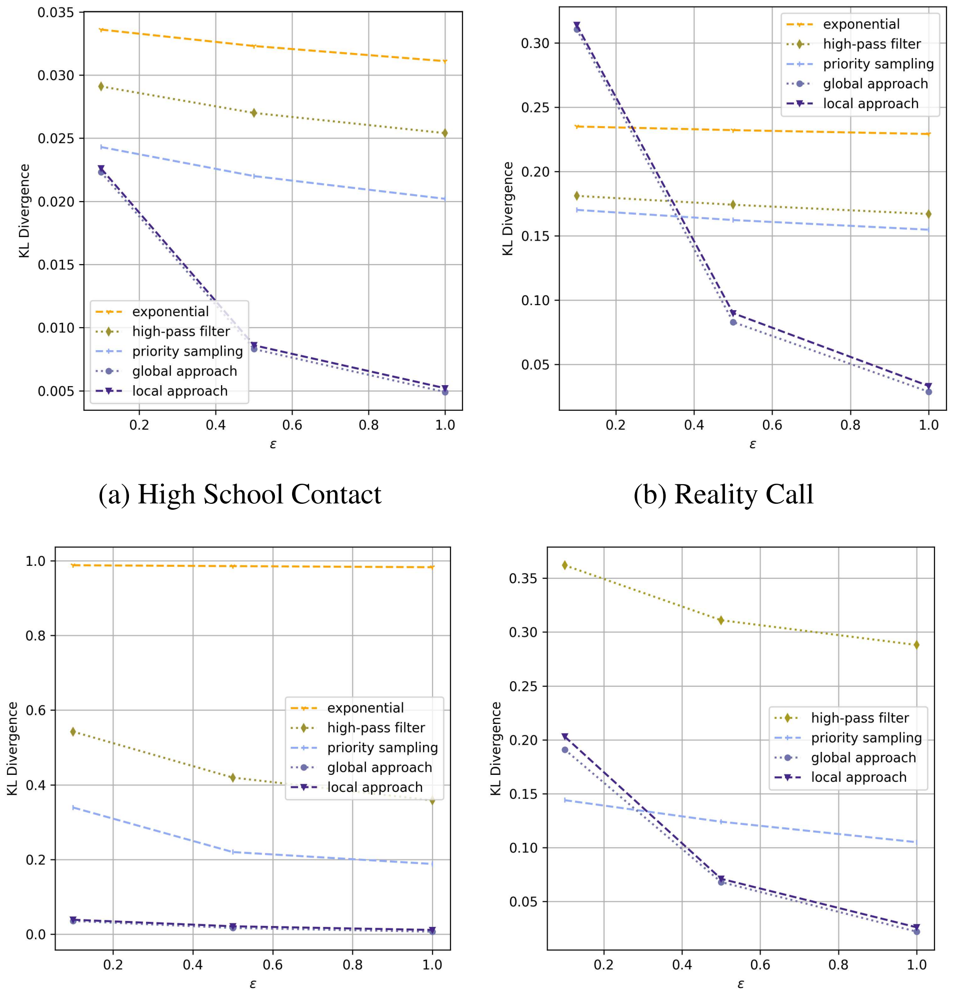
6.4.2.2 *Distributions Comparison*

We use the Kullback-Leibler (KL) divergence to evaluate the utility of both degree and edge weights distributions. It is defined as

$$D_{KL}(P||Q) = \sum_{x \in X} P(x) \log\left(\frac{P(x)}{Q(x)}\right), \tag{6.8}$$

where $P(x)$ and $Q(x)$ are probability distributions of some statistic obtained using a differentially private approach and the original graph, respectively. The smaller the KL divergence is, the higher the utility. In Figure 22, we compare the degree distributions obtained from global and local approaches with that of exponential mechanism, high-pass filter (HPF), and priority sampling (PS) as ε increases. Both approaches outperform PS and HPF when $\varepsilon > 0.5$ and present very similar results in most cases. However, for the non-heavy graphs, when ε is too low, other approaches could be better suited. Particularly, the degree is a local metric centered at a specific

Figure 22 – KL Divergence between degree distributions.



(a) High School Contact (b) Reality Call

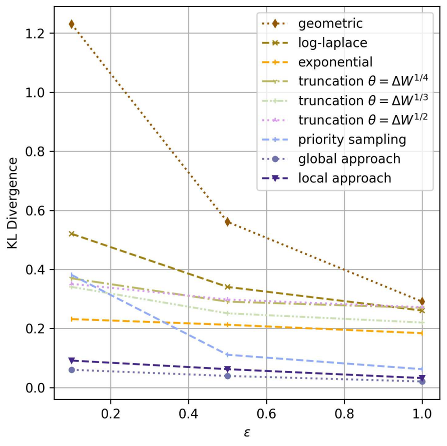
(c) Enron (d) DBLP

Source: Elaborated by the author.

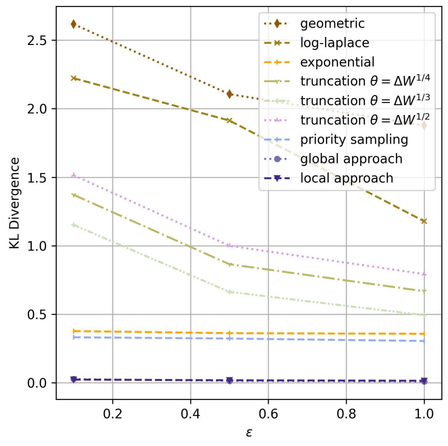
node, and consequently, the local approach performs as well as the global one. As mentioned in Section 4.2, filtering techniques tend to introduce additional bias to the weights when compared to sampling approaches. For this reason, priority sampling outperforms high-pass filter.

We also evaluate the KL divergence of edge weight distributions in Figure 23. Since our weights adjustment phase alleviates the bias introduced to them previously, both global and local approaches achieve lower KL divergence when compared to the competitors, even those that consider the graph topology to be known. Although the exponential mechanism produces dense graphs, such new edges generally have low edge weights, it reflects positively on its KL divergence of edge weight distributions.

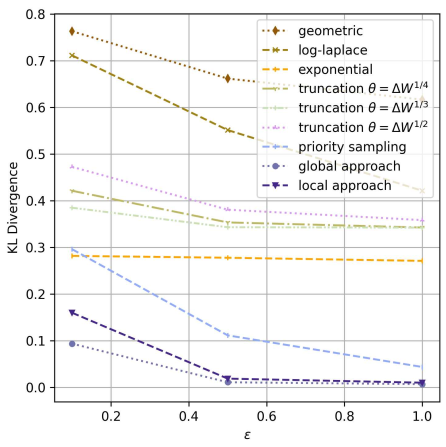
Figure 23 – KL Divergence between edge weights distributions.



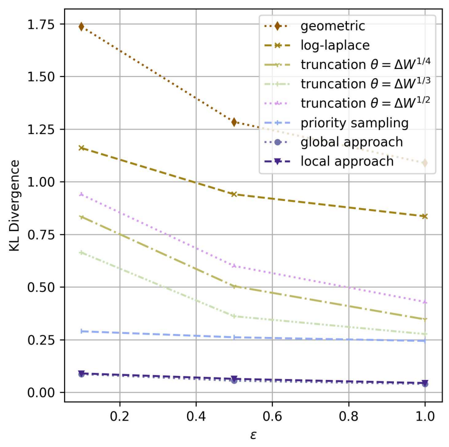
(a) High School Contact



(b) Reality Call



(c) Enron



(d) DBLP

Source: Elaborated by the author.

6.4.2.3 General Graph Statistics

Table 3 summarizes three graph statistics obtained using our global and local DP approaches, as well as through log-laplace, truncation, and PS, and compares with the non-private statistic values. The global approach achieves the best results for such metrics, even comparing it with the methods that consider the graph topology is known. In particular, the sum of edge weights, i.e. the total number of records in the original tabular dataset, is preserved in the global approach. In the local approach, due to some edges being removed in the neighborhood perturbation step, this statistic presents lower values when compared to the others. For the global clustering coefficient statistic (Clustering coef.), the global approach slightly outperforms the other methods. In the particular case of the sparse *reality call* dataset, global, local, and PS techniques have basically the same results as ε varies.

Table 3 – Graph statistics of the original and differentially private networks.

		exponential			log-laplace			truncation $\theta = \Delta W^{1/3}$		
Statistics	non-private	*ε=0.1*	*ε=0.5*	*ε=1.0*	*ε=0.1*	*ε=0.5*	*ε=1.0*	*ε=0.1*	*ε=0.5*	*ε=1.0*
High School Contacts										
Sum of edge weig.	188,507	253.4k	231.1k	216.1k	6.6M	2.5M	1.0M	430.0k	104.5k	65.3k
Avg. Weighted SP	2.72	1.66	1.69	1.71	4.19	3.73	3.65	3.61	3.43	3.25
Clustering coef.	0.61	0.85	0.80	0.77	0.35	0.39	0.42	0.41	0.44	0.48
Reality Call										
Sum of edge weig.	135,425	31.6M	20.8M	13.5M	3.1M	1.0M	2.9M	4.4M	972.6k	536.5k
Avg. Weighted SP	3.03	1.63	1.65	1.67	**3.62**	3.57	3.51	3.63	3.62	3.61
Clustering coef.	0.02	0.07	0.06	0.05	**0.01**	0.01	0.01	**0.01**	0.01	0.01
Enron										
Sum of edge weig.	3,237,456	983.7M	646.M	421.6	813.2M	229.2M	58.5M	25.9M	5.9M	3.5M
Avg. Weighted SP	8.84	2.63	2.67	2.71	92.86	60.26	43.04	38.42	26.08	16.27
Clustering coef.	0.79	1.61	1.47	1.35	0.05	0.16	0.26	0.11	0.25	0.34
DBLP										
Sum of edge weig.	7,286,515	-	-	-	662.5M	431.0M	257.9M	148.9M	34.2M	20.0M
Avg. Weighted SP	8.31	-	-	-	61.39	34.89	20.50	56.00	18.65	13.59
Clustering coef.	0.11	-	-	-	0.82	0.45	0.29	0.33	0.28	0.24

		priority sampling			global approach			local approach		
Statistics	non-private	*ε=0.1*	*ε=0.5*	*ε=1.0*	*ε=0.1*	*ε=0.5*	*ε=1.0*	*ε=0.1*	*ε=0.5*	*ε=1.0*
High School Contacts										
Sum of edge weig.	188,507	492.7k	238.6k	209.5k	**187.5k**	**188.6k**	**188.4k**	185.2k	186.4k	187.2k
Avg. Weighted SP	2.72	11.49	5.97	3.92	**3.22**	**3.10**	**3.01**	10.64	8.02	6.54
Clustering coef.	0.61	0.13	0.32	0.46	**0.41**	**0.47**	**0.51**	**0.41**	0.46	0.50
Reality Call										
Sum of edge weig.	135,425	5.8M	1.2M	644.8M	**133.5k**	**135.0k**	**135.4k**	133.2k	134.0k	134.9k
Avg. Weighted SP	3.03	8.41	6.71	4.34	5.02	**3.37**	**3.07**	6.08	4.13	3.27
Clustering coef.	0.02	**0.01**	**0.02**	**0.02**	**0.03**	**0.02**	**0.02**	0.04	0.03	**0.02**
Enron										
Sum of edge weig.	3,237,456	18.6M	5.7M	4.2M	**3.24M**	**3.23M**	**3.23M**	3.04M	3.12M	3.20M
Avg. Weighted SP	8.84	**36.85**	21.81	12.51	40.73	**19.25**	**12.02**	49.56	21.28	16.01
Clustering coef.	0.79	0.07	0.27	0.34	**0.15**	**0.38**	**0.49**	0.09	0.22	0.37
DBLP										
Sum of edge weig.	7,286,515	83.4M	56.5M	20.0M	**7.23M**	**7.29M**	**7.28M**	6.92M	7.04M	7.15M
Avg. Weighted SP	8.31	38.60	26.20	18.10	**10.58**	**8.78**	**8.51**	11.84	9.41	8.70
Clustering coef.	0.11	0.83	0.31	0.15	**0.02**	**0.05**	**0.06**	0.01	0.02	0.04

Source: Elaborated by the author.

6.4.2.4 *Node Centrality Measures*

Finally, we evaluate the average relative error for node strength (NS), sum of neighboring NS, and weighted PageRank in Table 4. The mean relative error is a statistical measure used to quantify the average difference between perturbed values (y'_i) and original ones (y_i), expressed as a percentage. MRE is defined as:

$$MRE(y,y') = \frac{\sum_{i=1}^{n} |y'_i - y_i|}{\sum_{i=1}^{n} y_i}, \tag{6.9}$$

where n is the number of elements in both y_i and y'_i. In our results, a relative error equal to 1.00 indicates 100%.

Table 4 – Average relative errors for the evaluated approaches.

	exponential			log-laplace			truncation $\theta = \Delta W^{1/3}$		
Centrality measure	*ε=0.1*	*ε=0.5*	*ε=1.0*	*ε=0.1*	*ε=0.5*	*ε=1.0*	*ε=0.1*	*ε=0.5*	*ε=1.0*
High School Contacts									
Node strength	1.18	0.77	0.50	9.20	5.17	4.13	3.05	0.60	0.59
Sum of Neighboring NS	10.33	7.47	5.19	15.74	8.14	4.60	1.39	0.63	0.42
Weighted PageRank	**0.18**	0.14	0.11	0.46	0.45	0.43	0.48	0.44	0.42
Reality Call									
Node strength	7.16	5.31	4.43	12.93	10.82	7.04	12.82	8.50	4.27
Sum of Neighboring NS	27.8	14.52	7.08	17.86	15.86	12.98	12.54	4.20	1.89
Weighted PageRank	**0.29**	0.28	0.27	0.60	0.42	0.29	0.43	0.41	0.39
Enron									
Node strength	38.61	32.21	27.94	25.08	19.05	10.52	20.08	10.06	4.98
Sum of Neighboring NS	90.87	47.49	23.09	32.12	24.62	17.98	**29.14**	**5.54**	**2.67**
Weighted PageRank	1.65	1.64	1.64	1.73	1.32	1.09	0.59	0.54	0.50
DBLP									
Node strength	-	-	-	15.04	12.27	11.35	9.26	5.83	2.91
Sum of Neighboring NS	-	-	-	8.15	6.39	5.93	12.08	4.48	2.17
Weighted PageRank	-	-	-	0.55	0.47	0.47	**0.54**	0.46	0.40

	priority sampling			global approach			local approach		
Centrality measure	*ε=0.1*	*ε=0.5*	*ε=1.0*	*ε=0.1*	*ε=0.5*	*ε=1.0*	*ε=0.1*	*ε=0.5*	*ε=1.0*
High School Contacts									
Node strength	3.49	0.59	0.25	1.08	0.20	0.10	**1.04**	**0.18**	**0.07**
Sum of Neighboring NS	**0.88**	0.39	0.29	0.93	**0.30**	**0.20**	1.12	0.34	0.22
Weighted PageRank	0.39	0.14	0.08	0.45	0.12	0.08	0.32	**0.11**	**0.06**
Reality Call									
Node strength	7.69	5.58	4.16	1.11	0.44	0.27	**1.05**	**0.33**	**0.17**
Sum of Neighboring NS	9.71	4.74	2.33	**1.23**	**0.46**	**0.40**	1.67	0.58	0.49
Weighted PageRank	0.31	0.29	0.27	0.45	0.33	0.21	**0.29**	**0.24**	**0.13**
Enron									
Node strength	19.80	12.86	7.81	24.7	11.65	5.87	**11.48**	**5.90**	**3.12**
Sum of Neighboring NS	36.71	12.91	8.87	33.52	9.77	6.25	37.90	22.77	15.92
Weighted PageRank	1.12	0.85	0.71	1.20	0.76	0.56	**0.54**	**0.46**	**0.39**
DBLP									
Node strength	7.97	4.85	2.08	3.60	2.59	1.17	**2.05**	**1.24**	**0.98**
Sum of Neighboring NS	**3.38**	**3.24**	2.99	8.73	4.48	**2.78**	10.11	5.20	4.87
Weighted PageRank	0.79	0.76	0.73	0.95	0.81	0.66	**0.54**	**0.42**	**0.37**

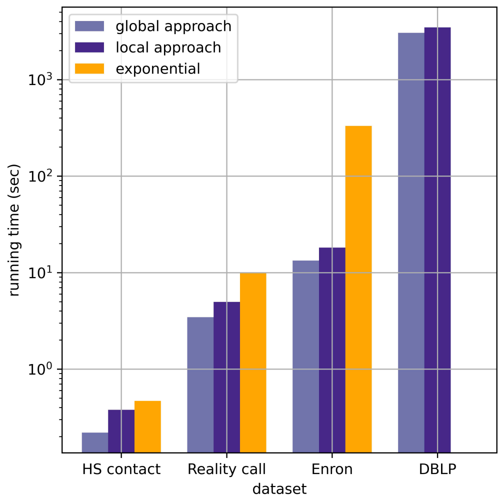
Source: Elaborated by the author.

For NS, the local approach outperforms the other methods, including the global approach. The reason is that node strength is a local metric and it is directly queried in the statistics extraction phase of the local approach. In terms of sum of neighboring NS, the approaches do not present a clear winner. It depends on the distribution of the weights in the original graph. On the other hand, for the weighted pagerank centrality, the local approach also outperforms the competitors, since the value of pagerank of a node is proportional to the importance of its neighbors. Note that not all reported values for average relative errors are useful in practice. For instance, even with $\varepsilon = 1.0$, sum of neighboring NS and pagerank centrality still present values of relative errors, especially in larger graphs.

6.5 Running time analysis

Figure 24 shows the average running time of global and local approaches and exponential mechanism. We set $\varepsilon = 1.0$ and report the average of 10 runs. The low-time complexities of our global and local approaches enable them to handle large-size graphs. In particular, the global approach runs faster than the local since the latter presents adjustments on the local neighborhood perturbation and on the curator side. Due to the fact that the exponential mechanism materializes a large amount of originally non-existing edges, it takes more time to run than the other approaches, even grouping output graphs with the same utility function. The results for the exponential mechanism in the DBLP dataset were not included in Figure 24 due to the extended duration of the experiments, which prevented them from reaching completion within the scope of this study.

Figure 24 – Average running time of global and local approaches and exponential mechanism.



Source: Elaborated by the author.

6.6 Summary

In this chapter, we evaluated the results obtained from our global and local approaches and we compared them with existing baselines. We tested four real-world weighted graphs with different sizes, sparsities, and heaviness: high-school contact, reality call, Enron, and DBLP.

In terms of utility, we started evaluating the graph similarity between the original weighted graph and the perturbed one. In summary, the global approach presented better results. We also observed that all the approaches presented low similarity values for the non-heavy graphs since the number of edges that have the same source and target originally changes substantially.

We also evaluated the results in terms of distributions of degrees and weights. Since we propose adjustment steps for both statistics directly, it is expected that these approaches exhibit satisfactory results. Particularly, for the distributions of degrees, we did not consider geometric, log-Laplace, and truncation as baselines since the graph topology is not perturbed by them in their approaches, and consequently, the node degrees are not modified.

In terms of general graph statistics and node centrality measures, the global approach presented better results for general graph statistics, while the local approach worked well for node centrality measures. The results for global and local approaches outperformed the baselines mainly when the tested datasets have heavy edge weights. That was the case of high school contact and Enron datasets.

7 CONCLUSIONS AND FUTURE WORK

In this chapter, we present a comprehensive overview of the findings obtained through our research. We have studied the problem of releasing count-weighted graphs for analysis and statistical purposes with differential privacy while preserving as much as possible the characteristics of the original graph. Particularly, Section 7.1 summarizes the achieved results based on the main research questions introduced in Chapter 1. The impact of this research in real-world applications is also presented in Section 7.2. Finally, open problems and future work directions are provided in Section 7.3.

7.1 Summary of Results

As discussed in Chapter 1, the main research questions that guided our work were:

RQ1: *How to establish a new definition of neighboring graphs considering both graph topology and edge weights as private information?*

RQ2: *How can we provide a scalable graph perturbation solution?*

RQ3: *How to keep the graph consistent and avoid introducing bias in the edge weights after adopting a DP technique to perturb the graph structure?*

In the following, we discuss the achieved results for each research goal.

7.1.1 How to establish a new definition of neighboring graphs considering both graph topology and edge weights as private information?

In this thesis, we introduced a new definition of neighboring weight graphs with unknown topology that considers two graphs to be neighbors if the set of vertices is the same and if the weight functions differ in one unit, i.e., both the graph topology and the edge weights are assumed to be private. As a result, we applied this new notion to perturb an input graph G and produce a noisy weighted version $\tilde{G}$ considering both edges and edge weights as private information. The results presented in Section 6.4 demonstrated the effectiveness of our approaches in terms of utility when compared to existing techniques. This allowed for the subsequent computation of various statistics on the released graph while maintaining high utility.

7.1.2 How can we provide a scalable graph perturbation solution?

Initially, we adopted a sampling strategy to avoid the materialization of all zero-edge weights in the graph perturbation process. Then we proposed an algorithm, based on the priority sampling method, to efficiently perturb a count-weighted graph and output a noisy graph with a pre-defined number of edges. In particular, Section 6.5 showed that the low-time complexity obtained by our perturbation technique enabled our global and local approaches to handle large-size graphs.

7.1.3 How to keep the graph consistent and avoid introducing bias in the edge weights after adopting a DP technique to perturb the graph structure?

To maintain the node degrees consistent and avoid introducing bias in the perturbed graph, we proposed two post-processing techniques to adjust both degrees and the sum of all edge weights after the graph perturbation process. In particular, our degrees adjustment algorithm is maximal in the sense that no more edges with higher weights can be included in the adjusted graph without decreasing any node degree below 0. This fact enables swapping edges such that the node degrees of the adjusted graph are as close as possible to the original one and, at the same time, it preserves the edges with higher weight values. Finally, in the weights adjustment phase, we proposed and solved an optimization problem to avoid introducing bias in the weights after the perturbation process. When comparing the results obtained without the adjustment steps, our global and local approaches outperformed the baselines that did not apply any kind of post-processing, as shown in Section 6.4.

7.2 Impact of the work

This thesis has the potential to create a positive impact on data privacy research across a variety of domains that require count-weighted graphs. Systems that enable the exchange of information between multiple devices or locations, facilitating efficient and reliable communication, can benefit from our two approaches. As stated in Chapter 6, the four tested real-world datasets were based on communication networks, such as calls, messages, and e-mails exchanged between entities. This fact provides an opportunity to examine influential people and the interactions between them, along with the analysis of information campaigns through social media in a private way.

This research can also find applications within the domain of information among entities within a propagation network, such as the private detection of misinformation. In this context, edges are connected when news sharing occurs between two users and the edge weights are defined as the intensity of propagation from one node to another. While accessing news via social media platforms offers a convenient and straightforward experience, there exists a susceptibility to being exposed to fake news that contains unchecked or intentionally false information. Additionally, the privacy of the connections in this type of graph can be violated if they are released to machine learning practitioners or researchers without sufficient privacy guarantees. Then, our local and global approaches can be adopted to protect the presence or absence of such connections and also of the propagation level.

In the domain of healthcare, our research has the potential for application in conducting privacy-preserving analyses related to the dissemination of epidemic diseases, such as COVID-19. In this field, edges represent interpersonal interactions, and the edge weights quantify the frequency of contact between two individuals. In a similar context, our work can explore the private detection of violations of social distancing measures, where edge weights indicate the physical distance between nodes. Consequently, researchers can conduct statistical analysis on such count-weighted graphs, by adopting our both local and global approaches, while preserving the privacy of both interactions and the distances between individuals.

7.3 Open Problems and Future Work

By conducting a thorough investigation into the difficulties and resolutions concerning the preservation of data privacy, while enhancing data analysis for weighted graphs, we have uncovered some unresolved issues. First, our both solutions are designed to handle integer-weighted values. It is important to introduce a weight function that associates real values with the edges. It can provide a more precise and flexible representation of the relationships associated with the edges in various applications, such as the strength of the association between a user and an item in a recommendation system, the cost of transferring funds in a financial network, the risk in portfolio management, among others. Additionally, edges may also convey directional information (directed edges), offering a more detailed representation of information flow, causality, or impact within a network.

In future work, we aim to investigate how the results perform when training machine learning models with differentially private weighted graphs, comparing them to non-private versions and assessing their robustness and accuracy across various tasks. We plan to test graph neural networks (GNNs) with differential privacy, capturing the dependencies and relationships among nodes, edges, and weights, in order to generate weighted graphs with DP guarantees.

In addition, we intend to examine hybrid differential privacy, where certain relationships may be accessible to one group of users, while others can be accessed by a different group. It is an important field of study since some weighted networks may include data from individuals who make choices to either grant (opt-in) or deny (opt-out) access to specific private information.

Moreover, we aim to propose new utility functions to the Exponential mechanism that are more efficient in terms of running time and utility, since querying graphs are a closely related field to categorical analysis. Thus, we can introduce novel differentially private techniques to address attribute edges, enabling private analysis in emerging applications like fraud detection within e-commerce networks or determining top rating levels in recommendation systems.

Finally, we aim to investigate fairness in artificial intelligence decisions over weighted graphs. It involves ensuring that no specific interactions among nodes receive unfair advantages or disadvantages in machine learning algorithms or analyses based on the graph's count-weighted structure. This concept is essential in various applications, including network optimization, data analysis, and resource allocation, where fairness considerations play a crucial role in achieving balanced and just outcomes.

BIBLIOGRAPHY

ABOWD, J. M. The us census bureau adopts differential privacy. In: ACM. **Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**. [*s.l.*], 2018. p. 2867–2867.

AGGARWAL, C. C.; WANG, H. *et al.* **Managing and mining graph data**. [*s.l.*]: Springer, 2010. v. 40.

BAGDASARYAN, E.; POURSAEED, O.; SHMATIKOV, V. Differential privacy has disparate impact on model accuracy. **Advances in neural information processing systems**, v. 32, 2019.

BLOCKI, J.; BLUM, A.; DATTA, A.; SHEFFET, O. Differentially private data analysis of social networks via restricted sensitivity. In: **Proceedings of the 4th conference on Innovations in Theoretical Computer Science**. [*s.l.*: *s.n.*], 2013. p. 87–96.

Brazil. **Lei Geral de Proteção de Dados Pessoais**. 2018. http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/L13709.htm. Online; accessed 15 May 2023.

BRITO, F. T.; FARIAS, V. A.; FLYNN, C.; MAJUMDAR, S.; MACHADO, J. C.; SRIVASTAVA, D. Global and local differentially private release of count-weighted graphs. **Proceedings of the ACM on Management of Data**, ACM New York, NY, USA, v. 1, n. 2, p. 1–25, 2023.

BRITO, F. T.; MACHADO, J. C. Preservação de privacidade de dados: Fundamentos, técnicas e aplicações. **Jornadas de atualização em informática**, p. 91–130, 2017.

CAMACHO, D.; PANIZO-LLEDOT, A.; BELLO-ORGAZ, G.; GONZALEZ-PARDO, A.; CAMBRIA, E. The four dimensions of social network analysis: An overview of research methods, applications, and software tools. **Information Fusion**, Elsevier, v. 63, p. 88–120, 2020.

CHEN, L.; HAN, K.; XIU, Q.; GAO, D. Graph clustering under weight-differential privacy. In: IEEE. **2022 IEEE 24th Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)**. [*s.l.*], 2022. p. 1457–1464.

CHEN, R.; FUNG, B.; YU, P. S.; DESAI, B. C. Correlated network data publication via differential privacy. **The VLDB Journal**, Springer, v. 23, n. 4, p. 653–676, 2014.

CHENG, X.; SU, S.; XU, S.; XIONG, L.; XIAO, K.; ZHAO, M. A two-phase algorithm for differentially private frequent subgraph mining. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 30, n. 8, p. 1411–1425, 2018.

CLAUSET, A.; MOORE, C.; NEWMAN, M. E. Structural inference of hierarchies in networks. In: SPRINGER. **ICML Workshop on Statistical Network Analysis**. [*s.l.*], 2006. p. 1–13.

CONDAMIN, S.; BÉNICHOU, O.; TEJEDOR, V.; VOITURIEZ, R.; KLAFTER, J. First-passage times in complex scale-invariant media. **Nature**, Nature Publishing Group UK London, v. 450, n. 7166, p. 77–80, 2007.

CORMODE, G.; JHA, S.; KULKARNI, T.; LI, N.; SRIVASTAVA, D.; WANG, T. Privacy at scale: Local differential privacy in practice. In: **Proceedings of the 2018 International Conference on Management of Data**. [*s.l.*: *s.n.*], 2018. p. 1655–1658.

CORMODE, G.; PROCOPIUC, C.; SRIVASTAVA, D.; TRAN, T. T. Differentially private summaries for sparse data. In: **Proceedings of the 15th International Conference on Database Theory**. [*s.l.*: *s.n.*], 2012. p. 299–311.

CORMODE, G.; SRIVASTAVA, D.; LI, N.; LI, T. Minimizing minimality and maximizing utility: analyzing method-based attacks on anonymized data. **Proceedings of the VLDB Endowment**, VLDB Endowment, v. 3, n. 1-2, p. 1045–1056, 2010.

DAY, W.-Y.; LI, N.; LYU, M. Publishing graph degree distribution with node differential privacy. In: **Proceedings of the 2016 International Conference on Management of Data**. [*s.l.*: *s.n.*], 2016. p. 123–138.

DING, X.; ZHANG, X.; BAO, Z.; JIN, H. Privacy-preserving triangle counting in large graphs. In: **Proceedings of the 27th ACM International Conference on Information and Knowledge Management**. [*s.l.*: *s.n.*], 2018. p. 1283–1292.

DRECHSLER, J. Differential privacy for government agencies—are we there yet? **Journal of the American Statistical Association**, Taylor & Francis, v. 118, n. 541, p. 761–773, 2023.

DUCHI, J. C.; JORDAN, M. I.; WAINWRIGHT, M. J. Local privacy and statistical minimax rates. In: IEEE. **2013 IEEE 54th Annual Symposium on Foundations of Computer Science**. [*s.l.*], 2013. p. 429–438.

DUFFIELD, N.; LUND, C.; THORUP, M. Learn more, sample less: control of volume and variance in network measurement. **IEEE Transactions on Information Theory**, IEEE, v. 51, n. 5, p. 1756–1775, 2005.

DUFFIELD, N.; LUND, C.; THORUP, M. Priority sampling for estimation of arbitrary subset sums. **Journal of the ACM (JACM)**, ACM New York, NY, USA, v. 54, n. 6, p. 32–es, 2007.

DWORK, C. Differential privacy. In: SPRINGER. **International Colloquium on Automata, Languages, and Programming**. [*s.l.*], 2006. p. 1–12.

DWORK, C.; MCSHERRY, F.; NISSIM, K.; SMITH, A. Calibrating noise to sensitivity in private data analysis. In: SPRINGER. **Theory of cryptography conference**. [*s.l.*], 2006. p. 265–284.

ERLINGSSON, Ú.; PIHUR, V.; KOROLOVA, A. Rappor: Randomized aggregatable privacy-preserving ordinal response. In: ACM. **Proceedings of the 2014 ACM SIGSAC conference on computer and communications security**. [*s.l.*], 2014. p. 1054–1067.

European Commission. **2018 reform of EU data protection rules**. 2018. https://ec.europa.eu/commission/sites/beta-political/files/data-protection-factsheet-changes_en.pdf. Online; accessed 15 May 2023.

FAN, C.; LI, P. Distances release with differential privacy in tree and grid graph. In: IEEE. **2022 IEEE International Symposium on Information Theory (ISIT)**. [*s.l.*], 2022. p. 2190–2195.

FARIAS, V. A.; BRITO, F. T.; FLYNN, C.; MACHADO, J. C.; MAJUMDAR, S.; SRIVASTAVA, D. Local dampening: differential privacy for non-numeric queries via local sensitivity. **Proceedings of the VLDB Endowment**, VLDB Endowment, v. 14, n. 4, p. 521–533, 2020.

FARIAS, V. A.; BRITO, F. T.; FLYNN, C.; MACHADO, J. C.; MAJUMDAR, S.; SRIVASTAVA, D. Local dampening: Differential privacy for non-numeric queries via local sensitivity. **The VLDB Journal**, Springer, p. 1–24, 2023.

Federal Communications Commission. **Customer privacy**. 2018. https://www.fcc.gov/general/customer-privacy. Online; accessed 13 October 2022.

FERRARA, E.; VAROL, O.; MENCZER, F.; FLAMMINI, A. Detection of promoted social media campaigns. In: **Proceedings of the International AAAI Conference on Web and Social Media**. [*s.l.*: *s.n.*], 2016. v. 10, n. 1, p. 563–566.

GANTA, S. R.; KASIVISWANATHAN, S. P.; SMITH, A. Composition attacks and auxiliary information in data privacy. In: **Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. [*s.l.*: *s.n.*], 2008. p. 265–273.

GAO, T.; LI, F.; CHEN, Y.; ZOU, X. Local differential privately anonymizing online social networks under hrg-based model. **IEEE Transactions on Computational Social Systems**, IEEE, v. 5, n. 4, p. 1009–1020, 2018.

GARFINKEL, S. L.; ABOWD, J. M.; POWAZEK, S. Issues encountered deploying differential privacy. In: ACM. **Proceedings of the 2018 Workshop on Privacy in the Electronic Society**. [*s.l.*], 2018. p. 133–137.

GHOSH, A.; ROUGHGARDEN, T.; SUNDARARAJAN, M. Universally utility-maximizing privacy mechanisms. **SIAM Journal on Computing**, SIAM, v. 41, n. 6, p. 1673–1693, 2012.

GIBBS, P.; ZAK, N. A review of longevity validations up to may 2023. SocArXiv, 2023.

GRYGORASH, O.; ZHOU, Y.; JORGENSEN, Z. Minimum spanning tree based clustering algorithms. In: IEEE. **2006 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)**. [*s.l.*], 2006. p. 73–81.

HANEY, S.; MACHANAVAJJHALA, A.; ABOWD, J. M.; GRAHAM, M.; KUTZBACH, M.; VILHUBER, L. Utility cost of formal privacy for releasing national employer-employee statistics. In: **Proceedings of the 2017 ACM International Conference on Management of Data**. [*s.l.*: *s.n.*], 2017. p. 1339–1354.

HARDT, M.; ROTH, A. Beating randomized response on incoherent matrices. In: **Proceedings of the forty-fourth annual ACM symposium on Theory of computing**. [*s.l.*: *s.n.*], 2012. p. 1255–1268.

HASTINGS, W. K. Monte carlo sampling methods using markov chains and their applications. Oxford University Press, 1970.

HAY, M.; LI, C.; MIKLAU, G.; JENSEN, D. Accurate estimation of the degree distribution of private networks. In: IEEE. **2009 Ninth IEEE International Conference on Data Mining**. [*s.l.*], 2009. p. 169–178.

HEIKENFELD, J.; JAJACK, A.; ROGERS, J.; GUTRUF, P.; TIAN, L.; PAN, T.; LI, R.; KHINE, M.; KIM, J.; WANG, J. Wearable sensors: modalities, challenges, and prospects. **Lab on a Chip**, Royal Society of Chemistry, v. 18, n. 2, p. 217–248, 2018.

HOLLAND, P. W.; LASKEY, K. B.; LEINHARDT, S. Stochastic blockmodels: First steps. **Social networks**, Elsevier, v. 5, n. 2, p. 109–137, 1983.

HONG, D.; JUNG, W.; SHIM, K. Collecting geospatial data with local differential privacy for personalized services. In: IEEE. **2021 IEEE 37th International Conference on Data Engineering (ICDE)**. [*s.l.*], 2021. p. 2237–2242.

HSU, J.; GABOARDI, M.; HAEBERLEN, A.; KHANNA, S.; NARAYAN, A.; PIERCE, B. C.; ROTH, A. Differential privacy: An economic method for choosing epsilon. In: IEEE. **2014 IEEE 27th Computer Security Foundations Symposium**. [*s.l.*], 2014. p. 398–410.

HUANG, H.; ZHANG, D.; XIAO, F.; WANG, K.; GU, J.; WANG, R. Privacy-preserving approach pbcn in social network with differential privacy. **IEEE Transactions on Network and Service Management**, IEEE, v. 17, n. 2, p. 931–945, 2020.

IFTIKHAR, M.; WANG, Q.; LIN, Y. dk-microaggregation: Anonymizing graphs with differential privacy guarantees. In: SPRINGER. **Pacific-Asia Conference on Knowledge Discovery and Data Mining**. [*s.l.*], 2020. p. 191–203.

IMOLA, J.; MURAKAMI, T.; CHAUDHURI, K. Locally differentially private analysis of graph statistics. In: **30th USENIX Security Symposium (USENIX Security 21)**. [*s.l.*: *s.n.*], 2021. p. 983–1000.

JI, S.; MITTAL, P.; BEYAH, R. Graph data anonymization, de-anonymization attacks, and de-anonymizability quantification: A survey. **IEEE Communications Surveys & Tutorials**, IEEE, v. 19, n. 2, p. 1305–1326, 2016.

JIAN, X.; WANG, Y.; CHEN, L. Publishing graphs under node differential privacy. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, 2021.

JIN, X.; ZHANG, N.; DAS, G. Algorithm-safe privacy-preserving data publishing. In: **Proceedings of the 13th International Conference on Extending Database Technology**. [*s.l.*: *s.n.*], 2010. p. 633–644.

KARRER, B.; NEWMAN, M. E. Stochastic blockmodels and community structure in networks. **Physical review E**, APS, v. 83, n. 1, p. 016107, 2011.

KARWA, V.; RASKHODNIKOVA, S.; SMITH, A.; YAROSLAVTSEV, G. Private analysis of graph structure. **PVLDB**, VLDB Endowment, v. 4, n. 11, p. 1146–1157, 2011.

KARWA, V.; SLAVKOVIĆ, A. B. Differentially private graphical degree sequences and synthetic graphs. In: SPRINGER. **International Conference on Privacy in Statistical Databases**. [*s.l.*], 2012. p. 273–285.

KASIVISWANATHAN, S. P.; NISSIM, K.; RASKHODNIKOVA, S.; SMITH, A. Analyzing graphs with node differential privacy. In: SPRINGER. **Theory of Cryptography Conference**. [*s.l.*], 2013. p. 457–476.

KENTHAPADI, K.; MIRONOV, I.; THAKURTA, A. Privacy-preserving data mining in industry. In: ACM. **Companion Proceedings of The 2019 World Wide Web Conference**. [*s.l.*], 2019. p. 1308–1310.

KOUTRA, D.; PARIKH, A.; RAMDAS, A.; XIANG, J. Algorithms for graph similarity and subgraph matching. In: CITESEER. **Proc. Ecol. inference conf**. [*s.l.*], 2011. v. 17.

KRAUSE, A.; GOLOVIN, D. Submodular function maximization. **Tractability**, v. 3, n. 71-104, p. 3, 2014.

LEAL, B. C.; VIDAL, I. C.; BRITO, F. T.; NOBRE, J. S.; MACHADO, J. C. -doca: Achieving privacy in data streams. In: SPRINGER. **International Workshop on Data Privacy Management**. [*s.l.*], 2018. p. 279–295.

LEE, J.; CLIFTON, C. How much is enough? choosing ε for differential privacy. In: SPRINGER. **Information Security: 14th International Conference, ISC 2011, Xi'an, China, October 26-29, 2011. Proceedings 14**. [*s.l.*], 2011. p. 325–340.

LESKOVEC, J.; ADAMIC, L. A.; HUBERMAN, B. A. The dynamics of viral marketing. **ACM Transactions on the Web (TWEB)**, ACM New York, NY, USA, v. 1, n. 1, p. 5–es, 2007.

LI, N.; LI, T.; VENKATASUBRAMANIAN, S. t-closeness: Privacy beyond k-anonymity and l-diversity. In: IEEE. **2007 IEEE 23rd International Conference on Data Engineering**. [*s.l.*], 2007. p. 106–115.

LI, N.; LI, T.; VENKATASUBRAMANIAN, S. Closeness: A new privacy measure for data publishing. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 22, n. 7, p. 943–956, 2009.

LI, N.; LYU, M.; SU, D.; YANG, W. Differential privacy: From theory to practice. **Synthesis Lectures on Information Security, Privacy, & Trust**, Morgan & Claypool Publishers, v. 8, n. 4, p. 1–138, 2016.

LI, X.; YANG, J.; SUN, Z.; ZHANG, J. Differential privacy for edge weights in social networks. **Security and Communication Networks**, Hindawi, v. 2017, 2017.

MACHANAVAJJHALA, A.; KIFER, D.; GEHRKE, J.; VENKITASUBRAMANIAM, M. l-diversity: Privacy beyond k-anonymity. **ACM Transactions on Knowledge Discovery from Data (TKDD)**, ACM New York, NY, USA, v. 1, n. 1, p. 3–es, 2007.

MAHADEVAN, P.; KRIOUKOV, D.; FALL, K.; VAHDAT, A. Systematic topology analysis and generation using degree correlations. **ACM SIGCOMM Computer Communication Review**, ACM New York, NY, USA, v. 36, n. 4, p. 135–146, 2006.

MANRÍQUEZ, R.; GUERRERO-NANCUANTE, C.; MARTÍNEZ, F.; TARAMASCO, C. Spread of epidemic disease on edge-weighted graphs from a database: A case study of covid-19. **International Journal of Environmental Research and Public Health**, MDPI, v. 18, n. 9, p. 4432, 2021.

MASTRANDREA, R.; FOURNET, J.; BARRAT, A. Contact patterns in a high school: a comparison between data collected using wearable sensors, contact diaries and friendship surveys. **PloS one**, Public Library of Science San Francisco, CA USA, v. 10, n. 9, p. e0136497, 2015.

MATSUMOTO, H.; YOSHIDA, S.; MUNEYASU, M. Propagation-based fake news detection using graph neural networks with transformer. In: IEEE. **2021 IEEE 10th Global Conference on Consumer Electronics (GCCE)**. [*s.l.*], 2021. p. 19–20.

MCSHERRY, F.; TALWAR, K. Mechanism design via differential privacy. In: IEEE. **48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)**. [*s.l.*], 2007. p. 94–103.

MCSHERRY, F. D. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In: **Proceedings of the 2009 ACM SIGMOD International Conference on Management of data**. [*s.l.*: *s.n.*], 2009. p. 19–30.

MEDINA, A. M.; GILLENWATER, J. Duff: A dataset-distance-based utility function family for the exponential mechanism. **arXiv preprint arXiv:2010.04235**, 2020.

MENDONÇA, A. L.; BRITO, F. T.; LINHARES, L. S.; MACHADO, J. C. Dipcoding: a differentially private approach for correlated data with clustering. In: **Proceedings of the 21st International Database Engineering & Applications Symposium**. [*s.l.*: *s.n.*], 2017. p. 291–297.

MENDONÇA, A. L.; BRITO, F. T.; MACHADO, J. C. Privacy-preserving techniques for social network analysis. In: SBC. **Anais Estendidos do XXXVIII Simpósio Brasileiro de Bancos de Dados**. [*s.l.*], 2023. p. 174–178.

MIRZASOLEIMAN, B.; BADANIDIYURU, A.; KARBASI, A.; VONDRÁK, J.; KRAUSE, A. Lazier than lazy greedy. In: **Proceedings of the AAAI Conference on Artificial Intelligence**. [*s.l.*: *s.n.*], 2015. v. 29, n. 1.

MONTEIRO, F. C.; BRITO, F. T.; CHAVES, I. C.; MACHADO, J. C. Compartilhamento de dados de tráfego de rede utilizando privacidade diferencial. In: SBC. **Anais do L Seminário Integrado de Software e Hardware**. [*s.l.*], 2023. p. 296–307.

MORVAN, A.; CHOROMANSKI, K.; GOUY-PAILLER, C.; ATIF, J. Graph sketching-based massive data clustering. **arXiv preprint arXiv:1703.02375**, 2017.

NEAR, JOSEPH P. and ABUAH, CHIKÉ. **Programming Differential Privacy**. 2023. https://uvm-plaid.github.io/programming-dp/. Online; accessed 15 May 2023.

NERGIZ, M. E.; ATZORI, M.; CLIFTON, C. Hiding the presence of individuals from shared databases. In: **Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data**. [*s.l.*: *s.n.*], 2007. p. 665–676.

NETO, E. R.; MENDONÇA, A. L.; BRITO, F. T.; MACHADO, J. C. Privlbs: uma abordagem para preservação de privacidade de dados em serviços baseados em localização. In: SBC. **Anais do XXXIII Simpósio Brasileiro de Banco de Dados**. [*s.l.*], 2018. p. 109–120.

NEWMAN, M. E. The structure and function of complex networks. **SIAM review**, SIAM, v. 45, n. 2, p. 167–256, 2003.

NGUYEN, H. H.; IMINE, A.; RUSINOWITCH, M. Differentially private publication of social graphs at linear cost. In: IEEE. **2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)**. [*s.l.*], 2015. p. 596–599.

NING, B.; SUN, Y.; TAO, X.; LI, G. Differential privacy protection on weighted graph in wireless networks. **Ad Hoc Networks**, Elsevier, v. 110, p. 102303, 2021.

NISHIKAWA, T.; MOTTER, A. E.; LAI, Y.-C.; HOPPENSTEADT, F. C. Heterogeneity in oscillator networks: Are smaller worlds easier to synchronize? **Physical review letters**, APS, v. 91, n. 1, p. 014101, 2003.

NISSIM, K.; RASKHODNIKOVA, S.; SMITH, A. Smooth sensitivity and sampling in private data analysis. In: **Proceedings of the thirty-ninth annual ACM symposium on Theory of computing**. [*s.l.*: *s.n.*], 2007. p. 75–84.

NY, J. L.; PAPPAS, G. J. Privacy-preserving release of aggregate dynamic models. In: **Proceedings of the 2nd ACM international conference on High confidence networked systems**. [*s.l.*: *s.n.*], 2013. p. 49–56.

PEIXOTO, TIAGO P. **The graph-tool python library**. 2014. http://figshare.com/articles/graph_tool/1164194. Online; accessed 15 May 2023.

PINOT, R.; MORVAN, A.; YGER, F.; GOUY-PAILLER, C.; ATIF, J. Graph-based clustering under differential privacy. **arXiv preprint arXiv:1803.03831**, 2018.

QIN, Z.; YU, T.; YANG, Y.; KHALIL, I.; XIAO, X.; REN, K. Generating synthetic decentralized social graphs with local differential privacy. In: **Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security**. [*s.l.*: *s.n.*], 2017. p. 425–438.

SALA, A.; ZHAO, X.; WILSON, C.; ZHENG, H.; ZHAO, B. Y. Sharing graphs using differentially private graph models. In: **Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference**. [*s.l.*: *s.n.*], 2011. p. 81–98.

SAMARATI, P.; SWEENEY, L. Generalizing data to provide anonymity when disclosing information (abstract). In: MENDELZON, A. O.; PAREDAENS, J. (Ed.). **Proceedings of the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 1-3, 1998, Seattle, Washington, USA**. [*s.l.*]: ACM Press, 1998. p. 188.

SEALFON, A. Shortest paths and distances with differential privacy. In: **Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems**. [*s.l.*: *s.n.*], 2016. p. 29–41.

SILVA, R. R. C.; LEAL, B. C.; BRITO, F. T.; VIDAL, V. M.; MACHADO, J. C. A differentially private approach for querying rdf data of social networks. In: **Proceedings of the 21st International Database Engineering & Applications Symposium**. [*s.l.*: *s.n.*], 2017. p. 74–81.

SUBAHI, A. F. A model transformation approach for detecting distancing violations in weighted graphs. **Computer Systems Science & Engineering**, v. 36, n. 1, 2021.

SUN, H.; XIAO, X.; KHALIL, I.; YANG, Y.; QIN, Z.; WANG, H.; YU, T. Analyzing subgraph statistics from extended local views with decentralized differential privacy. In: **Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security**. [*s.l.*: *s.n.*], 2019. p. 703–717.

SWEENEY, L. k-anonymity: A model for protecting privacy. **International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems**, World Scientific, v. 10, n. 05, p. 557–570, 2002.

TONG, D.; YONG, Z.; MENGLI, L.; ZHIHONG, L.; JIANFENG, M.; XIAOYAN, Z. A topology based differential privacy scheme for average path length query. In: IEEE. **2019 International Conference on Networking and Network Applications (NaNA)**. [*s.l.*], 2019. p. 350–355.

United States Census Bureau. **Census Bureau Sets Key Parameters to Protect Privacy in 2020 Census Results**. 2021. https://www.census.gov/newsroom/press-releases/2021/2020-census-key-parameters.html. Online; accessed 10 January 2023.

VAROL, O.; FERRARA, E.; MENCZER, F.; FLAMMINI, A. Early detection of promoted campaigns on social media. **EPJ data science**, Springer, v. 6, p. 1–19, 2017.

VU, T.; RAICH, R. On asymptotic linear convergence of projected gradient descent for constrained least squares. **arXiv preprint arXiv:2112.11760**, 2021.

WANG, D.; LONG, S. Boosting the accuracy of differentially private in weighted social networks. **Multimedia Tools and Applications**, Springer, v. 78, n. 24, p. 34801–34817, 2019.

WANG, Y.; WU, X. Preserving differential privacy in degree-correlation based graph generation. **Transactions on data privacy**, NIH Public Access, v. 6, n. 2, p. 127, 2013.

WANG, Y.; YANG, J.; ZHANG, J. Differential privacy for weighted network based on probability model. **IEEE Access**, IEEE, v. 8, p. 80792–80800, 2020.

WONG, R. C.-W.; FU, A. W.-C.; WANG, K.; YU, P. S.; PEI, J. Can the utility of anonymized data be used for privacy breaches? **ACM Transactions on Knowledge Discovery from Data (TKDD)**, ACM New York, NY, USA, v. 5, n. 3, p. 1–24, 2011.

WOOD, A.; ALTMAN, M.; BEMBENEK, A.; BUN, M.; GABOARDI, M.; HONAKER, J.; NISSIM, K.; O'BRIEN, D. R.; STEINKE, T.; VADHAN, S. Differential privacy: A primer for a non-technical audience. **Vand. J. Ent. & Tech. L.**, HeinOnline, v. 21, p. 209, 2018.

XIAO, Q.; CHEN, R.; TAN, K.-L. Differentially private network data release via structural inference. In: **Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining**. [*s.l.*: *s.n.*], 2014. p. 911–920.

YE, Q.; HU, H.; AU, M. H.; MENG, X.; XIAO, X. Lf-gdpr: A framework for estimating graph metrics with local differential privacy. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, 2020.

YE, Q.; HU, H.; AU, M. H.; MENG, X.; XIAO, X. Towards locally differentially private generic graph metric estimation. In: IEEE. **2020 IEEE 36th International Conference on Data Engineering (ICDE)**. [*s.l.*], 2020. p. 1922–1925.

ZHANG, J.; CORMODE, G.; PROCOPIUC, C. M.; SRIVASTAVA, D.; XIAO, X. Private release of graph statistics using ladder functions. In: **Proceedings of the 2015 ACM SIGMOD international conference on management of data**. [*s.l.*: *s.n.*], 2015. p. 731–745.

ZHU, T.; YE, D.; WANG, W.; ZHOU, W.; PHILIP, S. Y. More than privacy: Applying differential privacy in key areas of artificial intelligence. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 34, n. 6, p. 2824–2843, 2020.

APPENDIX A – PROOFS

A.1 Proof of Theorem 7

Theorem 7. *Let S be the set of zero edges included in $\tilde{G}$. The probability of a zero edge (u,v) be included in $\tilde{G}$ is given by:*

$$Pr[(u,v) \in S] = \frac{\alpha(1-\alpha^{\tau})}{\tau(1-\alpha^2)}. \tag{4.7}$$

Proof. A zero edge (u,v) is included in the perturbed graph with probability $p_{uv} = min\left(\frac{\tilde{\omega}(u,v)}{\tau}, 1\right)_+$. Then:

$$Pr[(u,v) \in S \mid \tilde{\omega}(u,v) = w] = min\left(\frac{w}{\tau}, 1\right)_+,$$

The probability of a zero edge (u,v) be added in $\tilde{G}$ is given by:

$$\begin{aligned}
Pr[(u,v) \in S] &= \sum_{w} Pr[(u,v) \in S \mid \tilde{\omega}(u,v) = w] Pr[\tilde{\omega}(u,v) = w] \\
&= \sum_{w \leq \tau} \frac{w}{\tau} \frac{1-\alpha}{1+\alpha} \alpha^{|w|} + \sum_{w > \tau} \frac{1-\alpha}{1+\alpha} \alpha^{|w|} \\
&= \frac{1-\alpha}{\tau(1+\alpha)} \sum_{w=0}^{\tau} w\alpha^{w} + \frac{1-\alpha}{1+\alpha} \sum_{w=\tau+1}^{\infty} \alpha^{w} \\
&= \frac{\alpha}{\tau(1-\alpha^2)}(1-(\tau+1)\alpha^{\tau} + \tau\alpha^{\tau+1}) + \frac{\alpha^{\tau+1}}{1+\alpha} \\
&= \frac{(1+\alpha)\alpha(1-(\tau+1)\alpha^{\tau} + \tau\alpha^{\tau+1}) + \tau(1+\alpha)(1-\alpha)\alpha^{\tau+1}}{\tau(1-\alpha^2)(1+\alpha)} \\
&= \frac{\alpha - \tau\alpha^{\tau+1} - \alpha^{\tau+1} + \tau\alpha^{\tau+2} + \tau\alpha^{\tau+1} - \tau\alpha^{\tau+2}}{\tau(1-\alpha^2)} \\
&= \frac{\alpha(1-\alpha^{\tau})}{\tau(1-\alpha^2)}
\end{aligned}$$

□

A.2 Proof of Theorem 8

Theorem 8. *For any neighboring weight graphs with unknown topology G and G' that differ in one unit of weight,*

$$||deg(V(G)) - deg(V(G'))||_1 \leq 2. \tag{4.12}$$

Proof. Assume, without loss of generality, that G' has an additional edge (u,v) with weight equal to 1. Consequently, it increases the degree of both nodes u and v in one unit. Similarly, suppose that G' has a missing edge (u,v) with weight equal to 1. It implies both nodes u and v lose one unit of degree. Therefore, it induces a difference of at most 2 in all node degrees for both cases.

□

A.3 Proof of Theorem 9

Theorem 9. *For any neighboring weight graphs with unknown topology G and G' that differ in one unit of weight,*

$$\Big|\Big| \sum_{u,v \in E(G)} \omega(u,v) - \sum_{u,v \in E(G')} \omega(u,v) \Big|\Big|_1 = 1. \tag{4.13}$$

Proof. Assume, without loss of generality, that G' has one unit of edge weight higher than G. As a result, it increases the total sum of edge weights in one unit. Similarly, suppose that G' has one unit of edge weight lower than G. Consequently, it decreases the total sum of edge weights in one unit. Therefore, both cases induce a difference of at most 1 in the sum of all edge weights.

□