

# Differential Privacy for Network Assortativity

Fei Ma  
School of Computer Science  
Northwestern Polytechnical  
University  
Xi'an, China  
feima@nwpu.edu.cn

Jinzhì Ouyang  
School of Computer Science  
Northwestern Polytechnical  
University  
Xi'an, China  
ouyangjinzhì@mail.nwpu.edu.cn

Xincheng Hu  
School of Computer Science  
Northwestern Polytechnical  
University  
Xi'an, China  
xinchenghu@mail.nwpu.edu.cn

## ABSTRACT

The analysis of network assortativity is of great importance for understanding the structural characteristics of and dynamics upon networks. Often, network assortativity is quantified using the assortativity coefficient that is defined based on the Pearson correlation coefficient between vertex degrees (see Eq.(1) for concrete expression). It is well known that a network may contain sensitive information, such as the number of friends of an individual in a social network (which is abstracted as the degree of vertex.). So, the computation of the assortativity coefficient leads to privacy leakage, which increases the urgent need for privacy-preserving protocol. However, there has been no scheme addressing the concern above.

To bridge this gap, in this work, we are the first to propose approaches based on differential privacy (DP for short). Specifically, we design three DP-based algorithms: **Local<sub>ru</sub>**, **Shuffle<sub>ru</sub>**, and **Decentral<sub>ru</sub>**. The first two algorithms, based on Local DP (LDP) and Shuffle DP respectively, are designed for settings where each individual only knows his/her direct friends. In contrast, the third algorithm, based on Decentralized DP (DDP), targets scenarios where each individual has a broader view, i.e., also knowing his/her friends' friends. Theoretically, we prove that each algorithm enables an unbiased estimation of the assortativity coefficient of the network. We further evaluate the performance of the proposed algorithms using mean squared error (MSE), showing that **Shuffle<sub>ru</sub>** achieves the best performance, followed by **Decentral<sub>ru</sub>**, with **Local<sub>ru</sub>** performing the worst. Note that these three algorithms have different assumptions, so each has its applicability scenario. Lastly, we conduct extensive numerical simulations, which demonstrate that the presented approaches are adequate to achieve the estimation of network assortativity under the demand for privacy protection.

## 1 INTRODUCTION

Nowadays, network analysis plays a crucial role in understanding a great variety of complex systems such as social networks [10], transportation networks [22], biological networks [24], and so on. It is well known that network analysis is often performed using various measures including degree distribution, clustering coefficient, diameter, and assortativity coefficient, to name but a few [3]. Among them, the assortativity coefficient (see subsection 3.2 for more details) [43] is used to reflect the tendency of nodes to connect to other nodes with similar attributes or characteristics in networks under consideration, and has received increasing attention from various science communities [46]. One of the main reasons for this is that the assortativity coefficient, as a fundamental measure, plays a key role in understanding the structural characteristics of and dynamics upon networks. Figure 1 shows an example of assortativity

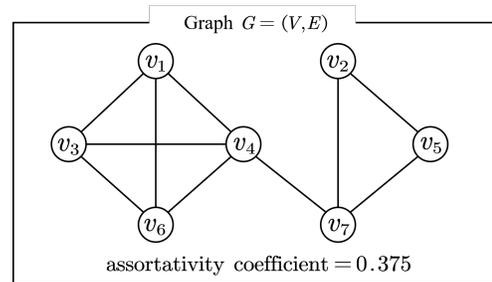


Figure 1: An example graph and its assortativity coefficient.

(see Eq.1 for specific expression). Hereinafter, two terms network and graph are used interchangeably.

Specifically, in the assortative network, nodes with similar attributes tend to connect. Essentially, assortativity can be understood as 'birds of a feather flock together,' where individuals with similar experiences, backgrounds, or knowledge tend to cluster together. The opposite connection between nodes is found in the disassortative network [44]. Besides providing structural information about networks, network assortativity is also related to the dynamic behavior on networks [40]. Assortativity of the network may influence information propagation pathways and speeds, as nodes preferentially connect with others sharing similar attributes, which potentially accelerates information dissemination within homogeneous communities. In addition, network assortativity can impact the stability and resilience of a network [11]. Thus, it is of great interest to analyze network assortativity in order to unravel mixing patterns on the network, predict the evolution behavior of the network, and design proper strategies for information diffusion on the network.

As tried in the literature [31], the data involved in networks may contain sensitive information when carrying out the analysis of network assortativity, which potentially leads to privacy breaches. For example, the network in question is a social one, where friendships and the number of friends used to calculate its assortativity are deemed sensitive information for each individual. Therefore, it is necessary to develop a framework that is suitable for the analysis of network assortativity and provides a guarantee for protecting the privacy of individuals within the network simultaneously. Assortativity is structurally sensitive and involves joint statistics over node degrees and their connectivity, making its estimation under strong privacy constraints particularly challenging. To the best of our knowledge, there is no such scheme by far. To bridge this gap, we propose available approaches based on differential privacy in this paper.

Differential Privacy (DP) [14] has become the gold standard for privacy-preserving network analysis, with broad applications in degree distribution estimation [12, 26, 39], triangle and k-star counting [29, 30]. While the traditional Central DP (CDP) model offers strong utility by assuming a trusted data curator, it poses significant risks under server compromise or insider threats [16, 58]. To address this limitation, Local DP (LDP) shifts the trust boundary to individual users by requiring them to perturb their data before submission [18]. On the other hand, this is achieved at the cost of high noise and poor utility, especially for graph statistics relying on edge correlations or higher-order structure [33].

To balance privacy and utility, intermediate models such as Shuffle DP [5, 9] and Decentralized DP (DDP) have been proposed. Shuffle DP enhances LDP by anonymizing locally randomized reports via an intermediate server called shuffler, enabling amplified privacy guarantees and improved estimation accuracy [17]. DDP, firstly introduced by Sun et al. [53], is a decentralized model tailored for network settings where each user holds an extended local view (e.g., 1-hop or 2-hop subgraph). Users apply randomized algorithms to their local views and send only privatized summaries to the server. This allows more expressive and accurate network analysis than standard LDP and, at the same time, preserves strong user-level privacy.

Motivated by this, the goal of this work is to investigate the private estimation of the assortativity coefficient—a critical metric measuring degree correlation between nodes—under LDP, Shuffle DP, and the DDP framework. The main contribution of this work is as follows:

- We are the first to study the estimation of network assortativity under differential privacy.
- We propose two DP-algorithms  $\text{Local}_{ru}$  and  $\text{Shuffle}_{ru}$  for accurate estimation of network assortativity in the setting where each individual is assumed to know only his/her friends. In a mathematically rigorous manner, we prove that the two algorithms output unbiased estimator, and also determine their steadiness by calculating MSE. In addition, we analyze both time and space complexity of the proposed algorithms.
- We also propose another DP-algorithm  $\text{Decentral}_{ru}$  suitable for a common situation in which each individual has a broader view, i.e., also knowing his/her friends' friends. Accordingly, we study the unbiasedness, MSE, time and space complexity related to this algorithm. The result shows that in sparse networks, the consequence produced by  $\text{Decentral}_{ru}$  is overwhelmingly better than that by  $\text{Local}_{ru}$ . This implies that a broader view is helpful to obtain a much steadier estimation.
- We conduct experimental evaluation on both synthetic and real-world datasets, which demonstrates that empirical analysis is in good agreement with theoretical consequences. In the meantime,  $\text{Shuffle}_{ru}$  shows better performance than the other two algorithms due to its own privacy amplification.

The rest of this paper is organized as below. We showcase the related work in Section 2. Section 3 introduces terminologies and

notations including graph, assortativity coefficient, differential privacy and shuffle model. The main results, namely, the proposed algorithms, are shown in Section 4. We conduct extensive experiments, and the results are displayed in Section 5. Some future research directions are listed in Section 6. Lastly, we close this work in Section 7.

## 2 RELATED WORK

Here, we briefly introduce related work. Roundly speaking, it contains two parts, i.e., non-private network assortativity analysis and private graph statistics. Note that two terms network and graph are used interchangeably in the remainder of this work.

### 2.1 Non-private network assortativity analysis

Assortativity, an important network property, has been extensively studied since its introduction [43]. In [43], Newman introduced the concept of assortativity in networks in 2002 and proposed the assortativity coefficient as a measure for structure mixing of networks. By applying this metric to various real-world networks, he showed that social networks tend to be assortative, whereas technological and biological networks tend to be disassortative. In 2003, Newman [44] further analyzed mixing patterns in networks, distinguished between assortative and disassortative mixing, and explored their effects on network structure and dynamics. In the follow-up work, Piraveenan et al. [47] investigated the relationship between assortativity and the information content of networks, discovering that assortative and disassortative networks contain more information than neutral networks, which shows neither assortativity nor disassortativity. Chang et al. [7] focused on assortativity in weighted networks by defining the node's strength as the sum of the weights of its connecting edges. Leung et al. [36] also studied the assortativity of weighted networks and introduced the concept of weighted assortativity. Estrada et al. [19] presented a method to determine network assortativity by examining the relationships among the clustering coefficient, modular connectivity, and branching. The results indicate that the clustering coefficient and modular connectivity positively influence assortativity, while branching has a negative impact.

### 2.2 Private graph statistics

Graph analysis under the differential privacy framework has been extensively studied, covering areas such as degree distribution [12, 26, 39], subgraph counting [32, 33], and synthetic graph generation [52, 56]. These studies typically operate within either a centralized or a local model. The centralized model assumes a centralized data curator, posing risks of data leakage. Consequently, the local model (LDP) has become increasingly favored by researchers.

In recent years, there has been a surge in research on Local Differential Privacy (LDP) for graph data, with an increasing number of notable advancements. For example, Sun et al. [53] proposed a subgraph counting algorithm under the assumption that each user knows all friends of his/her friends. Qin et al. [49] devised a method for generating synthetic graphs based on LDP. Ye et al. [59] presented a local one-round algorithm to estimate graph metrics including the clustering coefficient. Zhang et al. [60] developed a software usage analysis algorithm under LDP. Imola et al. [30]

proposed an exact triangle and 4-cycle counting algorithm under LDP by introducing the Shuffle model.

However, to date, there has been no research conducted on assortativity analysis within the context of privacy protection. Hence, the goal of this work is to present assortativity analysis algorithms based on Differential Privacy which is widely recognized as a robust privacy protection framework.

### 3 PRELIMINARY

In this section, we introduce some preliminaries for our work. Subsection 3.1 defines some basic notations related to networks. Subsections 3.2, 3.3 and 3.4 introduce the assortativity coefficient of networks, DP on graph and the shuffle model, respectively. Subsection 3.5 shows the utility metrics used in this work.

#### 3.1 Notations

Let  $\mathbb{N}$ ,  $\mathbb{R}$  and  $\mathbb{R}_{\geq 0}$  be the sets of natural numbers, real numbers and non-negative real numbers, respectively. We consider an undirected, non-weighted graph  $G = (V, E)$ , where  $V$  represents a set of nodes (users) and  $E \subseteq V \times V$  represents a set of edges. Let  $n \in \mathbb{N}$  be the number of nodes (users) in  $G$ , and  $M \in \mathbb{N}$  be the number of edges in  $G$ . We use  $v_i$  to represent the  $i$ -th node (user), and then have  $V = \{v_1, v_2, \dots, v_n\}$ . Let  $d_i \in \mathbb{N}$  be the degree of node  $v_i$  (i.e. the number of edges connected to  $v_i$ ), and  $d_{max} \in \mathbb{N}$  be the maximum degree of nodes in  $G$ , i.e.,  $d_{max} = \max \{d_1, \dots, d_n\}$ . We denote by  $\mathcal{G}$  the set of graphs with  $n$  nodes. It is a convention to represent a graph  $G$  using the adjacency matrix  $A = (a_{ij}) \in \{0, 1\}^{n \times n}$  in which if  $(v_i, v_j) \in E$ , then  $a_{ij} = 1$ ; and  $a_{ij} = 0$  otherwise. Accordingly, the  $i$ -th row of  $A$  is  $\mathbf{a}_i = (a_{i1}, a_{i2}, \dots, a_{in})$ . Table 1 shows some basic notations used in this paper.

**Table 1: Basic notations**

Symbol	Description
$G$	Undirected, non-weighted graph with $n$ nodes.
$M$	Number of edges in $G$ .
$A$	Adjacency matrix corresponding to graph $G$ .
$v_i$	$i$ -th node (user) in $G$ .
$\mathbf{a}_i$	Neighbor list of $v_i$ (i.e., the $i$ -th row of $A$ ).
$d_i$	Degree of $v_i$ .
$d_{max}$	Maximum degree of nodes in $G$ .
$d_{avg}$	Average degree of nodes in $G$ .
$q_{ru}$	Assortativity factor query of $G$ .

We adhere to standard notation conventions in our study. Concretely speaking, we employ the normal font  $\theta$  to represent the true value of a statistic (e.g.,  $d_i$  for the true degree of node  $v_i$ ) and use the tilde symbol  $\tilde{\theta}$  to indicate the noisy version of the quantity (e.g.,  $\tilde{d}_i$  is the noisy degree of  $v_i$  after applying some operation, for example, the Laplace mechanism). Additionally, the hat symbol  $\hat{\theta}$  is used to signify the estimated value (e.g.,  $\hat{d}_i$  is the estimation of degree of node  $v_i$  in  $G$ ).

#### 3.2 Assortativity Coefficient

Assortativity is a significant property of networks, measuring the preference of nodes to attach to others that are alike in some way. To quantitatively study network assortativity, Newman proposed an index  $r$  called assortativity coefficient [44]. Specifically, the concept of assortativity coefficient is based on Pearson correlation coefficient, which is defined as

$$r = \frac{M^{-1} \sum_{e_{ij} \in E} d_i d_j - \left[ M^{-1} \sum_{e_{ij} \in E} \frac{1}{2} (d_i + d_j) \right]^2}{M^{-1} \sum_{e_{ij} \in E} \frac{1}{2} (d_i^2 + d_j^2) - \left[ M^{-1} \sum_{e_{ij} \in E} \frac{1}{2} (d_i + d_j) \right]^2}, \quad (1)$$

where  $e_{ij}$  represents an edge between vertices  $v_i$  and  $v_j$ . If  $r > 0$ , the network in question is referred to as assortative. In this setting, it is believed that on average, nodes with similar degrees tend to connect to each other. If  $r < 0$ , the network is considered disassortative, where nodes with different degrees are more likely to connect to one another. If  $r = 0$ , it indicates that nodes in the network are connecting randomly, without any preference for nodes with similar or dissimilar degrees.

Ma et al. have proven that the denominator in Eq.(1) consistently remains non-negative [38]. This implies that whether a network exhibits assortativity or disassortativity depends solely on the value of the numerator, denoted by  $r_u$  for convenience, in Eq.(1). Namely, we write

$$r_u = M^{-1} \sum_{e_{ij} \in E} d_i d_j - \left[ M^{-1} \sum_{e_{ij} \in E} \frac{1}{2} (d_i + d_j) \right]^2. \quad (2)$$

Also, we refer to  $r_u$  as the assortativity factor. As shall be shown below, we focus on how to compute the accurate estimate of  $r_u$  under DP. Let  $q_{ru} : \mathcal{G} \rightarrow [-1, 1]$  be a assortativity factor query that takes  $G \in \mathcal{G}$  as input and outputs the assortativity factor  $q_{ru}(G)$  of  $G$ .

#### 3.3 Differential Privacy

**Local Differential Privacy.** Local Differential Privacy (LDP) [13] is a privacy measure that protects the personal information of each user locally. Due to its ability to protect sensitive information without relying on trusted third-party servers, LDP has garnered widespread attention in the field of network analysis [28, 49, 55]. In LDP, each user first obfuscates his/her personal data by himself/herself and sends the obfuscated data to a data collector.

LDP on graphs includes edge LDP and node LDP [49]. The former conceals the presence of any edge in a graph, while the latter hides both a node and all its adjacent edges. Node LDP is a stronger privacy definition but is more challenging to implement since it requires algorithms to hide more information about the input graph. This paper focuses on edge LDP because it can provide sufficient protection in many scenarios including subgraph counting [28], synthetic graphs [49] and maximizes the availability of perturbed data. Additionally, assortativity represents the likelihood of forming edges between nodes with similar attributes, and the primary focus of its study is the network's edges. In other words, its calculation only requires edge information. From this perspective, edge LDP is sufficient.

*Definition 3.1 (( $\epsilon, \delta$ )-edge LDP [49]).* Let  $n \in \mathbb{N}$ ,  $\epsilon \in \mathbb{R}_{\geq 0}$ , and  $\delta \in [0, 1]$ . For  $i \in \{1, 2, \dots, n\}$ , let  $\mathcal{R}_i$  be a local randomizer of user

$v_i$  that takes  $\mathbf{a}_i$  as input.  $\mathcal{R}_i$  provides  $(\epsilon, \delta)$ -edge LDP if for any two neighbor lists  $\mathbf{a}_i, \mathbf{a}'_i \in \{0, 1\}^n$  that differ in one bit and any  $S \subseteq \text{Range}(\mathcal{R}_i)$ , we have

$$\Pr(\mathcal{R}_i(\mathbf{a}_i) \in S) \leq e^\epsilon \Pr(\mathcal{R}_i(\mathbf{a}'_i) \in S) + \delta. \quad (3)$$

The parameter  $\epsilon$  is referred to as the privacy budget, which reflects the level of privacy protection offered by algorithm  $\mathcal{R}_i$ . When  $\epsilon$  is small (e.g.  $\epsilon \leq 1$  [37]), each bit of  $\mathbf{a}_i$  is strongly protected by edge LDP. The parameter  $\delta$  represents the privacy failure probability and is typically set to a value much smaller than  $\frac{1}{n}$  [15]. If  $\delta = 0$ ,  $\mathcal{R}_i$  provides  $\epsilon$ -edge LDP.

**Randomized Response (RR).** Randomized response is the mainstream obfuscation mechanism in the study of categorical data under LDP. The classic Warner's RR (Randomized Response) [54] is defined as follows. Applying Warner's RR to neighbor lists provides  $\epsilon$ -edge LDP.

*Definition 3.2 (Randomized Response [54]).* Given  $\epsilon \in \mathbb{R}_{\geq 0}$ , the Randomized Response Mechanism  $\mathcal{R}_\epsilon^W : \{0, 1\} \rightarrow \{0, 1\}$  maps  $x \in \{0, 1\}$  to  $y \in \{0, 1\}$  with the probability

$$\Pr(\mathcal{R}_\epsilon^W(x) = y) = \begin{cases} \frac{e^\epsilon}{e^\epsilon + 1} & \text{if } x = y, \\ \frac{1}{e^\epsilon + 1} & \text{otherwise.} \end{cases} \quad (4)$$

Note that two users in a network share information about the existence of an edge between them. That is to say, both users' outputs may reveal sensitive data. Based on this, the following algorithms (which will be proposed in this work) are designed for the lower triangular part of the adjacency matrix  $\mathbf{A}$  to enhance the level of user privacy protection. Specifically, given  $\epsilon \in \mathbb{R}_{\geq 0}$  and a neighbor list  $\mathbf{a}_i \in \{0, 1\}^n$ , the local randomizer  $\mathcal{R}_i$  outputs noisy bits  $(\tilde{a}_{i,1}, \dots, \tilde{a}_{i,i-1}) \in \{0, 1\}^n$  for users with smaller IDs, i.e., for each  $j \in \{1, \dots, i-1\}$ ,  $\tilde{a}_{i,j} = 1 - a_{i,j}$  with probability  $p = \frac{1}{e^\epsilon + 1}$  and  $\tilde{a}_{i,j} = a_{i,j}$  with probability  $1 - p$ .

**Laplacian Mechanism.** The Laplace mechanism [15] is a data obfuscation method introduced by Dwork et al for numerical data to guarantee differential privacy. The mechanism protects privacy by injecting random noise independently sampled from a Laplace distribution into the query statistic, where the level of noise added depends on the global sensitivity of the statistic. The greater the global sensitivity of the statistic, the more noise is added for privacy preservation.

*Definition 3.3 (Global Sensitivity under LDP [15]).* In edge LDP, the global sensitivity of a function  $f : \{0, 1\}^n \rightarrow \mathbb{R}$  is given by

$$\Delta_f = \max_{\mathbf{a}_i, \mathbf{a}'_i \in \{0, 1\}^n, |\mathbf{a}_i - \mathbf{a}'_i|_1 = 1} |f(\mathbf{a}_i) - f(\mathbf{a}'_i)|, \quad (5)$$

where  $|\mathbf{a}_i - \mathbf{a}'_i|_1 = 1$  represents that  $\mathbf{a}_i$  and  $\mathbf{a}'_i$  differ in one bit.

*Definition 3.4 (Local Laplacian Mechanism [15]).* Given  $\epsilon \in \mathbb{R}_{\geq 0}$  and any query function  $f : \{0, 1\}^n \rightarrow \mathbb{R}$  with a global sensitivity  $\Delta_f$ , the Local Laplacian Mechanism is defined as

$$\mathcal{R}_\epsilon^L(f(\mathbf{a}_i)) = f(\mathbf{a}_i) + \text{Lap}\left(\frac{\Delta_f}{\epsilon}\right). \quad (6)$$

**Extended Local Views (ELV).** In the LDP mechanism, each user's local input is his/her neighbor list  $\mathbf{a}_i$ . However, in many real-world scenarios, users typically have a broader local view. For example, in social networks like Twitter and LinkedIn, even with privacy settings enabled, the number of friends of a user's friends is usually accessible. In this paper, in addition to the user's 1-hop local view (i.e., neighbor list) we also focus on the 2-hop Extended Local View, which is defined as follows.

*Definition 3.5 (2-hop Extended Local View [53]).* Given a node  $v_i$ , its 2-hop Extended Local View (ELV) consists of all nodes reachable within 2 hops and all corresponding edges.

Conducting local differential privacy graph statistics in a local setting with 2-hop ELVs can significantly increase privacy risks due to the substantial overlap between different users' ELVs. Specifically, if ELVs from different users overlap, a single edge may contribute to multiple local reports. In other words, inserting/removing an edge in the global graph would result in changes to multiple local reports. Consequently, the presence of an edge is revealed multiple times, leading to an increased privacy risk.

**Decentralized Differential Privacy.** To address this privacy risk in the ELV setting, Sun et al. [53] introduced Decentralized Differential Privacy (DDP), which requires each user to protect not only his/her own privacy but also that of his/her neighbors. DDP is a privacy concept that lies between CDP and LDP. Specifically, DDP follows CDP to define neighboring datasets on the entire graph, but it protects each report locally during data collection. Formally, edge DDP is defined as follows.

*Definition 3.6 (( $\epsilon, \delta$ )-edge DDP [53]).* Let  $n \in \mathbb{N}$ ,  $\epsilon \in \mathbb{R}_{\geq 0}$ , and  $\delta \in [0, 1]$ . A set of local randomizers  $\{\mathcal{R}_i, 1 \leq i \leq n\}$  provides  $(\epsilon, \delta)$ -edge DDP if for any two neighboring graphs  $G, G' \in \mathcal{G}$  that differ in one edge and any  $\{S_i \in \text{Range}(\mathcal{R}_i), 1 \leq i \leq n\}$ ,

$$\Pr((\mathcal{R}_1(G_1) \in S_1), \dots, (\mathcal{R}_n(G_n) \in S_n)) \leq e^\epsilon \Pr((\mathcal{R}_1(G'_1) \in S_1), \dots, (\mathcal{R}_n(G'_n) \in S_n)) + \delta. \quad (7)$$

where  $G_i$  (resp.  $G'_i$ ) is the ELV of user  $v_i$  in graph  $G$  (resp.  $G'$ ).

Since DDP protects an edge in the entire graph  $G$ , the global sensitivity under DDP is defined as

*Definition 3.7 (Global Sensitivity under DDP [53]).* Given a function  $f$ , the global sensitivity of  $f$  is defined as

$$GS_{DDP}(f) = \max_{G, G'} \sum_{i=1}^n |f(G_i) - f(G'_i)|, \quad (8)$$

where  $G$  and  $G'$  are two arbitrary neighboring graphs, and  $G_i$  (resp.  $G'_i$ ) is the ELV of user  $v_i$  in graph  $G$  (resp.  $G'$ ).

The global sensitivity of a function is determined by the function itself, and some functions may have large global sensitivities. For example, in the 2-hop ELV setting, the global sensitivity of the sum of the degrees of a user's friends under DDP is  $4n - 6$ . Since  $n$  can be very large in real-world graphs (e.g., the number of users in social networks), directly perturbing data using the global sensitivity as the noise scale can significantly compromise the accuracy of the estimates. To improve accuracy, local sensitivity is typically used to reduce the noise scale, which is defined as follows.

*Definition 3.8 (Local Sensitivity under DDP [53]).* Given a global graph  $G$  and a function  $f$ , the local sensitivity of  $f$  is defined as

$$LS_{DDP}(f) = \max_{G'} \sum_{i=1}^n |f(G_i) - f(G'_i)|, \quad (9)$$

where  $G'$  is an arbitrary neighboring graph of  $G$  and  $G_i$  (resp.  $G'_i$ ) is the ELV of user  $v_i$  in graph  $G$  (resp.  $G'$ ).

Since local sensitivity is data-dependent, to prevent it from compromising privacy, we derive a probabilistic upper bound for local sensitivity by Lemma 3.9, and use this upper bound as the noise scale for data perturbations.

**LEMMA 3.9 (TAIL BOUND FOR LAPLACE DISTRIBUTION [53]).** *Let  $x$  be any real value, and  $\tilde{x} = x + \text{Lap}(\alpha)$  for some  $\alpha > 0$ . Then, with probability  $1 - \delta$ ,*

$$\tilde{x} + \alpha \cdot \log\left(\frac{1}{2\delta}\right) \geq x \quad (10)$$

### 3.4 Shuffle Model

In recent studies [2, 9, 17], the shuffle model has garnered significant interest due to its privacy amplification effect. The shuffle model, also known as Shuffle DP (SDP for short), introduces an intermediary server called shuffler between the user and the data collector based on LDP, and it works as follows. First, each user  $v_i$  obfuscates his/her personal data using an LDP mechanism  $\mathcal{R}$  common to all users. Then, user  $v_i$  encrypts the obfuscated data and sends it to the shuffler. The shuffler randomly shuffles the encrypted data to ensure anonymization and sends the result to a data collector for which we make no trust assumptions. Finally, the data collector decrypts the shuffled data. Under the assumption that the data collector does not collude with the shuffler, the shuffle model amplifies DP guarantees of the obfuscated data because shuffling removes the link between users and the obfuscated data. Shuffling can amplify privacy without loss of data utility for analytical tasks that are insensitive to data order, such as sum, average and histogram queries. Current research on the shuffle model primarily focuses on determining bounds on the level of privacy obtained after shuffling [20, 23].

In this paper, we use the privacy amplification bound provided by Feldman et al., which is the current state of the art.

**LEMMA 3.10 (PRIVACY AMPLIFICATION BY SHUFFLING [20]).** *Let  $n \in \mathbb{N}$  and  $\epsilon_0 \in \mathbb{R}_{\geq 0}$ . Let  $\mathcal{X}$  be the set of input data for each user. Let  $x_i \in \mathcal{X}$  be input data of the  $i$ -th user and  $\mathbf{x} = (x_1, \dots, x_n) \in \mathcal{X}^n$ . For  $i \in \{1, 2, \dots, n\}$ , let  $\mathcal{R}_i : \mathcal{X} \rightarrow \mathcal{Y}$  be a local randomizer of the  $i$ -th user that provides  $\epsilon_0$ -LDP. Let  $\mathcal{A}_S : \mathcal{X}^n \rightarrow \mathcal{Y}^n$  be the algorithm that given a dataset  $\mathbf{x} \in \mathcal{X}^n$ , samples a uniform random permutation  $\pi$  over  $\{1, 2, \dots, n\}$ , then sequentially computes  $y_{\pi(i)} = \mathcal{R}_i(x_{\pi(i)})$  and outputs  $\mathbf{y} = (y_{\pi(1)}, \dots, y_{\pi(n)})$ . Then for any  $\delta \in [0, 1]$  such that  $\epsilon_0 \leq \log\left(\frac{n}{16 \log(2/\delta)}\right)$ ,  $\mathcal{A}_S$  provides  $(\epsilon, \delta)$ -DP, where*

$$\epsilon \leq \log\left(1 + \frac{e^{\epsilon_0} - 1}{e^{\epsilon_0} + 1} \left(\frac{8\sqrt{e^{\epsilon_0} \log(4/\delta)}}{\sqrt{n}} + \frac{8e^{\epsilon_0}}{n}\right)\right). \quad (11)$$

In the lemma above, the shuffled data  $y_{\pi(1)}, \dots, y_{\pi(n)}$  is protected by  $\epsilon$ -DP, where  $\epsilon \ll \epsilon_0$ . In addition to providing this closed-form upper bound, Feldman et al. [20] developed an efficient algorithm to numerically compute a tighter upper bound, which we refer to as the numerical upper bound. In this paper, we use both the closed-form and numerical upper bounds and also distinguish the impact of each term on the performance of the algorithm.

### 3.5 Utility Metrics

We use MSE and Relative Error as utility metrics in theoretical analysis and experimental evaluation, respectively. Let  $\hat{q}_{ru} : \mathcal{G} \rightarrow \mathbb{R}$  be the assortativity factor estimator. Let  $MSE : \mathbb{R} \rightarrow \mathbb{R}_{\geq 0}$  be the MSE function, which maps the estimate  $\hat{q}_{ru}(G)$  and the true value  $q_{ru}(G)$  to the MSE; i.e.,

$$MSE[\hat{q}_{ru}(G), q_{ru}(G)] = \mathbb{E}[(\hat{q}_{ru}(G) - q_{ru}(G))^2]. \quad (12)$$

Note that the MSE can also be large when the  $q_{ru}(G)$  is large. Therefore, in our experiments, we evaluate the relative error given by

$$RE[\hat{q}_{ru}(G), q_{ru}(G)] = \frac{|\hat{q}_{ru}(G) - q_{ru}(G)|}{\min\{q_{ru}(G), \eta\}}, \quad (13)$$

where  $\eta \in \mathbb{R}_{\geq 0}$  is a small positive value. In general,  $\eta$  is set to  $\frac{n}{10^3}$  [4, 8, 57]. If the relative error is much smaller than 1, the estimation is highly accurate.

As shown in Eq.(1), it is clear that the estimation of the assortativity coefficient  $r$  involves intricate analyses of some terms reflecting inter-node correlations, which poses a challenge to obtaining an accurate estimate of  $r$  under the requirement of privacy protection. Specifically, bias correction is required for the expression  $X/Y$  derived from locally differentially private data to obtain an unbiased privacy estimate of the assortativity coefficient. Here, both  $X$  and  $Y$  denote a random polynomial. Despite recent advances in algorithms for unbiased local privacy estimation of Laplacian variable polynomials [27], it remains unresolved to obtain an unbiased estimate of  $X/Y$  from locally differentially private data.

The next lemma (i.e., lemma 3.11) offers an approach for approximating the unbiased privacy estimate of  $r$ . It is clear that we can approximate the unbiased estimate of the assortativity coefficient  $r$  by calculating the ratio of the unbiased estimates of its numerator and denominator. Of course, a more accurate unbiased estimate of  $r$  can be approximated by Eq.(15a). As shall be seen below, the following method is suitable for a more accurate estimation of  $r$  while many tedious computations need to be performed. This is because the expression for  $r$  involves a large number of terms representing connections between nodes. In this paper, we use a simple approximate estimator for  $r$ . Note that the denominator of  $r$  (denoted by  $r_d$ ) is

$$\begin{aligned} r_d &= M^{-1} \sum_{e_{ij} \in E} \frac{1}{2} (d_i^2 + d_j^2) - \left[ M^{-1} \sum_{e_{ij} \in E} \frac{1}{2} (d_i + d_j) \right]^2 \\ &= \frac{1}{2M} \sum_{i=1}^n d_i^3 - \left[ \frac{1}{2M} \sum_{i=1}^n d_i^2 \right]^2 \end{aligned} \quad (14)$$

Clearly, the denominator  $r_d$  is a polynomial function of node degrees, and its locally differentially private unbiased estimator can be obtained using the debiasing algorithm proposed in [27]. Therefore,

this paper focuses on computing a stable, unbiased estimator of the assortative factor  $r_u$  (numerator) under LDP.

LEMMA 3.11 ([6]). *Let  $X$  and  $Y$  be two random variables,*

$$\mathbb{E}\left(\frac{X}{Y}\right) \approx \frac{\mathbb{E}(X)}{\mathbb{E}(Y)} - \frac{\text{Cov}(X, Y)}{[\mathbb{E}(Y)]^2} + \frac{\mathbb{E}(X)}{[\mathbb{E}(Y)]^3} \cdot \mathbb{V}(Y), \quad (15a)$$

$$\mathbb{V}\left(\frac{X}{Y}\right) \approx \frac{\mathbb{V}(X)}{[\mathbb{E}(Y)]^2} - \frac{2\mathbb{E}(X)\text{Cov}(X, Y)}{[\mathbb{E}(Y)]^3} + \frac{[\mathbb{E}(X)]^2}{[\mathbb{E}(Y)]^4} \mathbb{V}(Y). \quad (15b)$$

## 4 SCHEMES

This section elaborates on our main results. Concretely, we propose three algorithms for estimating the assortativity factor  $r_u$  of network on  $n$  nodes in the requirement of DP.

First of all, we consider the common local setting where each individual is assumed to know only his/her friends. Accordingly, two algorithms **Local<sub>ru</sub>** and **Shuffle<sub>ru</sub>** are presented. **Local<sub>ru</sub>** is built based on a simple assumption that there is only one round of interaction between each user and the data collector. In this model, each user (data provider) perturbs his/her data once and then sends it to the data collector. Due to the limited interaction, the data collector can just gather a restricted amount of information. Consequently, the one-round algorithm is generally suitable for simple data analysis tasks where high accuracy is not critical. In order to obtain more accurate consequences, we add a round of communication between users and the data collector. Put this another way, a two-round algorithm **Shuffle<sub>ru</sub>** is designed. Specifically, the data collector sends query requests to each user two times, allowing for more powerful queries. As shown in the previous works, such as more accurate subgraph counting [28, 53], synthetic graph construction [49], a similar thought has been employed to design some two-round algorithms that can handle more complex data analysis tasks and achieve greater precision. However, this increased accuracy comes at the cost of added complexity and higher communication overhead.

Secondly, we also discuss the other scenario in which each individual has a broader view, i.e., also knowing his/her friends' friends. For instance, with the default setting of Facebook (facebook.com), a user allows each of her friends to see all her connections. The third algorithm **Decentral<sub>ru</sub>** is proposed to get assortativity factor  $r_u$  in this situation.

For convenience and understanding, we summarize the main properties of these three algorithms (i.e., **Local<sub>ru</sub>**, **Shuffle<sub>ru</sub>** and **Decentral<sub>ru</sub>**) in the next table ahead of time.

*Remark* In the rest of this work, we assume that parameter  $M$  is known to all. This is mainly because  $M$  can be accurately calculated using well-established degree distribution estimation techniques [12, 55]. Doing so simplifies the following analysis as well. At the same time, it is clear that parameter  $M$  has little influence on the estimation of network assortativity. It is worth noticing that the analysis approach established below is also suitable for the situation where parameter  $M$  is unknown in advance. Nonetheless, this causes more tedious calculations when conducting the theoretical analysis, which becomes clear to understand based on the following analysis and lemma 3.11. The goal of this paper is to build up a

**Table 2: main properties of algorithm**

	<b>Local<sub>ru</sub></b>	<b>Shuffle<sub>ru</sub></b>	<b>Decentral<sub>ru</sub></b>
$x$ -DP	$\epsilon$ -LDP	$(\epsilon, \delta)$ -DP	$(\epsilon, \delta)$ -DDP
Unbiasedness	yes	yes	yes
MSE	$O(K_1)^1$	$O(K_2)^2$	$O\left(\frac{n^3 d_{\max}^6}{M^4}\right)$
Time complexity	$O(n^2)$	$O(n^2)$	$O(n^2)$
Space complexity	$O(n^2)$	$O(n^2)$	$O(n^2)$

$$^1 K_1 = \frac{n^3 d_{\max}^2 + n^2 d_{\max}^4}{M^2} + \frac{n^3 d_{\max}^6}{M^4}.$$

$$^2 K_2 = \frac{n^{1+\alpha} d_{\max}^4}{M^2} + \frac{n^3 d_{\max}^2}{(\log n)^2 M^2} + \frac{n^3 d_{\max}^6}{(\log n)^2 M^4}.$$

methodology that has wide applications. In a nutshell, we only consider the case where parameter  $M$  is known.

Now, let us divert our attention to the development of the first algorithm **Local<sub>ru</sub>**. It should be noted that all the proofs of both unbiasedness and MSE of three algorithms are deferred to show in Supplementary Material for readability. Also, some notations are abused yet the meanings are clear from the context.

### 4.1 One-Round LDP Algorithm for Assortativity Factor

As shown in Eq.(2), the calculation of the assortativity factor  $r_u$  involves two terms: the degrees of nodes and the edges between them. In other words, we can derive  $r_u$  from the degree sequence  $d_1, d_2, \dots, d_n$  and the network's adjacency matrix  $A$ . Based on this, we propose a one-round algorithm for  $r_u$  under LDP. This algorithm adds random noise to the degrees of all nodes using the Laplace mechanism and obfuscates the lower triangular part of the adjacency matrix  $A$  with Randomized Response simultaneously. The data collector then estimates  $r_u$  from these noisy data sent by each user.

Algorithm 1 shows our one-round LDP algorithm for computing assortativity factor  $r_u$ . It takes a network  $G$  (represented as neighbor lists  $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n \in \{0, 1\}^n$ ) and privacy budgets  $\epsilon_1, \epsilon_2 \in \mathbb{R}_{\geq 0}$  as inputs, and outputs a privacy-preserving estimate of the assortativity factor query  $q_{ru}$ . We denote this algorithm by **Local<sub>ru</sub>** for brevity.

First, user  $v_i$  uses the Randomized Response mechanism  $\mathcal{R}_{\epsilon_1}^W$  that provides  $\epsilon_1$ -LDP to obfuscate  $a_{i,1}, \dots, a_{i,i-1}$  (i.e., the bits with smaller user IDs in the neighbor list  $\mathbf{a}_i$ ) (Line 2). In other words, we apply RR to the lower triangular part of the adjacency matrix  $A$ . Meanwhile, user  $v_i$  calculates his/her degree  $d_i$ , i.e., the number of "1"s in  $\mathbf{a}_i$  (Line 3), and adds noise  $\text{Lap}\left(\frac{1}{\epsilon_2}\right)$  to  $d_i$  (Line 4), where  $\text{Lap}\left(\frac{1}{\epsilon_2}\right)$  is a Laplace random variable with mean 0 and scale  $\frac{1}{\epsilon_2}$ . Then, user  $v_i$  sends the noisy data  $\tilde{\mathbf{R}}_i$  and  $\tilde{d}_i$  to the data collector. Finally, the data collector estimates the assortativity factor  $q_{ru}(G)$  using  $\tilde{\mathbf{R}}_1, \dots, \tilde{\mathbf{R}}_n$  and  $\tilde{d}_1, \dots, \tilde{d}_n$ . Specifically, the privacy estimate  $\hat{q}_{ru}(G)$  of  $q_{ru}(G)$  is as follows

$$\hat{q}_{ru}(G) = \frac{X_1}{M} - \frac{Y_1}{M^2}, \quad (16)$$

**Algorithm 1: Local<sub>ru</sub>.** [v<sub>i</sub>] and [d] represent that the process is run by user v<sub>i</sub> and the data collector, respectively.

**Data:** Graph G represented as neighbor lists a<sub>1</sub>, ..., a<sub>n</sub> ∈ {0, 1}<sup>n</sup>, privacy budget ε<sub>1</sub>, ε<sub>2</sub> ∈ ℝ<sub>≥0</sub>

**Result:** Private estimate q<sub>ru</sub>(G) of q<sub>ru</sub>(G)

```

1 for i = 1 to n do
2   [vi] R̃i ← (R̃ε1W(ai,1), ..., R̃ε1W(ai,i-1)); // Apply RR
   to the lower triangular part of adjacency
   matrix A.
3   [vi] di ← ∑j=1n ai,j;
4   [vi] d̃i ← di + Lap(1/ε2);
5   [vi] Send R̃i and d̃i to the data collector;
6 end
7 [d] X1 ← ∑i=2n ∑j=1i-1 (ãi,j - p) d̃i d̃j / (1 - 2p); // ãi,j is the output
   after applying RR to obfuscate ai,j
8 [d] Y1 ← (1/2 ∑i=1n d̃i2 - (n+2)/ε22) - (5n+4)/ε24;
9 [d] q̃ru(G) ← X1/M + Y1/M2;
10 [d] return q̃ru(G)
    
```

where

$$X_1 = \sum_{i=2}^n \sum_{j=1}^{i-1} \frac{(\tilde{a}_{i,j} - p) \tilde{d}_i \tilde{d}_j}{1 - 2p}, \quad (17a)$$

$$Y_1 = \left( \frac{1}{2} \sum_{i=1}^n \tilde{d}_i^2 - \frac{n+2}{\varepsilon_2^2} \right)^2 - \frac{5n+4}{\varepsilon_2^4}. \quad (17b)$$

Next, we show some theoretical properties of Local<sub>ru</sub>. First, we prove that Local<sub>ru</sub> certainly provides differential privacy.

**THEOREM 4.1.** Local<sub>ru</sub> provides ε-edge LDP and 2ε-edge DDP, where ε = ε<sub>1</sub> + ε<sub>2</sub>.

**PROOF.** Local<sub>ru</sub> applies RR to the lower triangular part of the adjacency matrix A, ensuring that R̃<sub>1</sub>, ..., R̃<sub>n</sub> are protected under ε<sub>1</sub>-edge LDP. Meanwhile, since d̃<sub>i</sub> = d<sub>i</sub> + Lap(1/ε<sub>2</sub>), the perturbed degrees d̃<sub>1</sub>, ..., d̃<sub>n</sub> are protected under ε<sub>2</sub>-edge LDP. According to the sequential composition property of differential privacy [15], the combined sequence R̃<sub>1</sub>, ..., R̃<sub>n</sub>, d̃<sub>1</sub>, ..., d̃<sub>n</sub> are protected under (ε<sub>1</sub> + ε<sub>2</sub>)-edge LDP. Since q̃<sub>ru</sub>(G) is derived from post-processing R̃<sub>1</sub>, ..., R̃<sub>n</sub> and d̃<sub>1</sub>, ..., d̃<sub>n</sub>, q̃<sub>ru</sub>(G) is protected under (ε<sub>1</sub> + ε<sub>2</sub>)-edge LDP by the post-processing immunity property of differential privacy [15]. In a word, Local<sub>ru</sub> provides ε-edge LDP, where ε = ε<sub>1</sub> + ε<sub>2</sub>.

Under the assumption that each individual knows only his/her friends, the local view G<sub>i</sub> of user v<sub>i</sub> is equivalent to a<sub>i</sub>, where i = 1, 2, ..., n. Thus, according to the definition of DDP (see Definition 3.6), for any two neighboring graphs G, G' that differ in one edge,

we have

$$\begin{aligned}
 & \frac{\Pr(\mathcal{R}_1(G_1) \in S_1, \dots, \mathcal{R}_n(G_n) \in S_n)}{\Pr(\mathcal{R}_1(G'_1) \in S_1, \dots, \mathcal{R}_n(G'_n) \in S_n)} \\
 &= \frac{\prod_{i=1}^n \Pr(\mathcal{R}_i(G_i) \in S_i)}{\prod_{i=1}^n \Pr(\mathcal{R}_i(G'_i) \in S_i)} \\
 &= \frac{\Pr(\mathcal{R}_x(\mathbf{a}_x) \in S_x) \cdot \Pr(\mathcal{R}_y(\mathbf{a}_y) \in S_y)}{\Pr(\mathcal{R}_x(\mathbf{a}'_x) \in S_x) \cdot \Pr(\mathcal{R}_y(\mathbf{a}'_y) \in S_y)} \\
 &\leq e^{2\varepsilon}.
 \end{aligned} \quad (18)$$

Therefore, Local<sub>ru</sub> provides 2ε-edge DDP. □

The following theorem demonstrates that Local<sub>ru</sub> provides an unbiased estimate of the assortativity factor.

**THEOREM 4.2.** The estimate q̃<sub>ru</sub>(G) produced by Local<sub>ru</sub> is unbiased, i.e., E(q̃<sub>ru</sub>(G)) = q<sub>ru</sub>(G).

As mentioned above, we also determine the steadiness of local<sub>ru</sub>, which is shown as follows.

**THEOREM 4.3.** When ε<sub>1</sub>, ε<sub>2</sub> are constants, the estimate q̃<sub>ru</sub>(G) produced by Local<sub>ru</sub> provides the following utility guarantee:

$$\text{MSE}(q̃_{ru}(G)) = O\left(\frac{n^3 d_{\max}^2 + n^2 d_{\max}^4}{M^2} + \frac{n^3 d_{\max}^6}{M^4}\right). \quad (19)$$

Since almost all real-world networks are sparse, i.e. M = Θ(n) [1], it follows that we have MSE(q̃<sub>ru</sub>(G)) ≤ O(n<sup>2</sup>), if quantity d<sub>max</sub> ≤ O(n<sup>1/2</sup>). It is well known that this condition d<sub>max</sub> ≤ n<sup>1/2</sup> is commonly seen in real-world networks [42].

Finally, we analyze the time and space complexity of Local<sub>ru</sub>.

**THEOREM 4.4.** The time complexity of Local<sub>ru</sub> is O(n<sup>2</sup>), and the space complexity is O(n<sup>2</sup>).

Local<sub>ru</sub> requires obfuscating each bit in the lower triangular part of the adjacency matrix of network G, resulting in a time complexity of O(n<sup>2</sup>). Additionally, this algorithm needs O(n<sup>2</sup>) space to store the adjacency matrix A, hence the space complexity is also O(n<sup>2</sup>).

By far, we have finished the development of Local<sub>ru</sub> and also analyzed it in detail. The next subsection aims to enhance the level of privacy-preserving by bringing the Shuffle Model.

## 4.2 Two-Round DP Algorithm for Assortativity Factor with Shuffling

Generally, multi-round LDP algorithms, which involve multiple rounds of interaction between each user and the data collector, can support more powerful queries and produce more accurate results. Hence, we further design a two-round algorithm for calculating the assortativity factor and, in the meantime, introduce the Shuffle Model to amplify privacy.

Algorithm 2 (denoted by Shuffle<sub>ru</sub>) shows the two-round DP algorithm for the assortativity factor with shuffling. Shuffle<sub>ru</sub> takes as input a graph G (represented as neighbor lists a<sub>1</sub>, a<sub>2</sub>, ..., a<sub>n</sub> ∈ {0, 1}<sup>n</sup>), a privacy budget ε, a failure probability δ and a parameter for privacy budget allocation α.

**Algorithm 2: Shuffle<sub>ru</sub>.**  $[v_i]$ ,  $[s]$  and  $[d]$  represent that the process is run by user  $v_i$ , the shuffler and the data collector, respectively.

**Data:** Neighbor lists  $\mathbf{a}_1, \dots, \mathbf{a}_n \in \{0, 1\}^n$ , privacy budget  $\varepsilon \in \mathbb{R}_{\geq 0}$ , failure probability  $\delta \in [0, 1]$ , parameter for privacy budget allocation  $\alpha \in (0, 1)$

**Result:** Private estimate  $\hat{q}_{ru}(G)$  of  $q_{ru}(G)$

```

1  $\varepsilon_0 \leftarrow \text{LocalPrivacyBudget}(n, \varepsilon, \delta)$ ;
  /* First round
2 for  $i = 1$  to  $n$  do
3    $[v_i]$   $d_i \leftarrow \sum_{j=1}^n a_{i,j}$ ;
4    $[v_i]$   $\tilde{d}_i \leftarrow d_i + \text{Lap}\left(\frac{1}{\alpha\varepsilon_0}\right)$ ;
5    $[v_i]$  Send  $\tilde{d}_i$  to the data collector;
6 end
  /* Second round
7  $[d]$  Return the sequence  $\tilde{d}_1, \dots, \tilde{d}_n$  to each user;
8  $[v_i]$   $\tilde{\mathbf{R}}_i \leftarrow (\mathcal{R}_{\varepsilon_1}^W(a_{i,1}), \dots, \mathcal{R}_{\varepsilon_1}^W(a_{i,i-1}))$ ; // Apply RR
   that provides  $\varepsilon_1$ -LDP to the lower triangular
   part of adjacency matrix  $\mathbf{A}$ , where  $\varepsilon_1 = (1 - \alpha)\varepsilon_0$ .
9 for  $i = 2$  to  $n$  do
10   $[v_i]$   $\hat{r}_i \leftarrow d_i \sum_{j=1}^{i-1} \frac{(\tilde{a}_{i,j} - p)\tilde{d}_j}{1 - 2p}$ ;
11   $[v_i]$  Send  $\hat{r}_i$  to the shuffler;
12 end
13  $[s]$  Sample a random permutation  $\pi$  over  $\{1, 2, \dots, n\}$ ;
14  $[s]$  Send  $\hat{r}_{\pi(1)}, \dots, \hat{r}_{\pi(n)}$  to the data collector;
15  $[d]$   $X_2 \leftarrow \sum_{i=2}^n \hat{r}_i$ ;
16  $[d]$   $Y_2 \leftarrow \left(\frac{1}{2} \sum_{i=1}^n \tilde{d}_i^2 - \frac{n+2}{\varepsilon_0^2}\right)^2 - \frac{5n+4}{\varepsilon_0^4}$ ;
17  $[d]$   $\hat{q}_{ru}(G) \leftarrow X_2/M + Y_2/M^2$ ;
18  $[d]$  return  $\hat{q}_{ru}(G)$ 

```

When running **Shuffle<sub>ru</sub>**, the first step is to call the function **LocalPrivacyBudget** to compute a local privacy budget  $\varepsilon_0$  from parameters  $n, \varepsilon, \delta$  (line 1). Specifically, this function produces  $\varepsilon_0$  such that  $\varepsilon$  is a closed-form or numerical upper bound in the shuffle model with  $n$  users. Given  $\varepsilon_0$ , the closed-form bound can be computed by Eq.(11), while the numerical bound can be obtained using the open source code in [20]<sup>1</sup>. Thus, we can easily compute  $\varepsilon_0$  from  $\varepsilon$  by computing a lookup table for the privacy budget pairs  $(\varepsilon_0, \varepsilon)$  in advance.

In the first round, each user  $v_i$  calculates his/her degree  $d_i$  (line 3), then computes the noisy degree  $\tilde{d}_i = d_i + \text{Lap}\left(\frac{1}{\alpha\varepsilon_0}\right)$  (line 4). Finally,  $v_i$  sends  $\tilde{d}_i$  to the data collector.

In the second round, the data collector returns the noisy degree  $\tilde{d}_1, \dots, \tilde{d}_n$  to each user. Each user  $v_i$  then obfuscates the bits with smaller user IDs in his/her neighbor list  $\mathbf{a}_i$  using the random response mechanism  $\mathcal{R}_{\varepsilon_1}^W$ , and calculates  $\hat{r}_i$  based on both  $\tilde{d}_1, \dots, \tilde{d}_n$  and the obfuscated bits in  $\mathbf{a}_i$  (lines 8 to 10). Next,  $\hat{r}_i$  is sent to the

Shuffler which randomly permutes  $\hat{r}_1, \dots, \hat{r}_n$  and sends the permuted data  $\hat{r}_{\pi(1)}, \dots, \hat{r}_{\pi(n)}$  to the data collector.

Finally, the data collector computes the estimate  $\hat{q}_{ru}(G)$  of  $q_{ru}(G)$  from  $\tilde{d}_1, \dots, \tilde{d}_n$  and  $\hat{r}_{\pi(1)}, \dots, \hat{r}_{\pi(n)}$ . Note that the estimate  $\hat{q}_{ru}$  only involves summing  $\hat{r}_{\pi(1)}, \dots, \hat{r}_{\pi(n)}$  while it is not necessary to know the permutation  $\pi$ . Thus, the expression for  $\hat{q}_{ru}(G)$  is as follows

$$\hat{q}_{ru}(G) = \frac{X_2}{M} - \frac{Y_2}{M^2}, \quad (20)$$

where

$$X_2 = \sum_{i=2}^n \hat{r}_i = \sum_{i=2}^n d_i \sum_{j=1}^{i-1} \frac{(\tilde{a}_{i,j} - p)\tilde{d}_j}{1 - 2p}, \quad (21a)$$

$$Y_2 = \left(\frac{1}{2} \sum_{i=1}^n \tilde{d}_i^2 - \frac{n+2}{\varepsilon_0^2}\right)^2 - \frac{5n+4}{\varepsilon_0^4}. \quad (21b)$$

As above, we need to make a detailed analysis of **Shuffle<sub>ru</sub>**. First, let us state that **Shuffle<sub>ru</sub>** provides the following privacy guarantee.

**THEOREM 4.5.** **Shuffle<sub>ru</sub>** provides  $(\varepsilon, \delta)$ -edge DP.

**PROOF.** Since  $\tilde{d}_i = d_i + \text{Lap}\left(\frac{1}{\alpha\varepsilon_0}\right)$ ,  $\tilde{d}_1, \dots, \tilde{d}_n$  are protected under  $\alpha\varepsilon_0$ -edge LDP. The lower triangular part of the adjacency matrix  $\mathbf{A}$  is obfuscated using RR, so  $\tilde{\mathbf{R}}_1, \dots, \tilde{\mathbf{R}}_n$  are protected under  $\varepsilon_1$ -edge LDP, where  $\varepsilon_1 = (1 - \alpha)\varepsilon_0$ . Note that  $d_i$  is independent of  $\sum_{j=1}^{i-1} \frac{(\tilde{a}_{i,j} - p)\tilde{d}_j}{1 - 2p}$ , by the sequential composition and post-processing immunity of DP,  $\hat{r}_1, \dots, \hat{r}_n$  is protected under  $\varepsilon_0$ -edge LDP. After shuffling, the shuffled sequence  $\hat{r}_{\pi(1)}, \hat{r}_{\pi(2)}, \dots, \hat{r}_{\pi(n)}$  achieves privacy amplification under  $(\varepsilon, \delta)$ -edge DP protection, where  $\varepsilon = g(n, \varepsilon_0, \delta)$  (see Eq.(11) for the expression). By the immunity to post-processing,  $\hat{q}_{ru}(G)$  is still protected under  $(\varepsilon, \delta)$ -edge LDP. Therefore, **Shuffle<sub>ru</sub>** provides  $(\varepsilon, \delta)$ -edge LDP.  $\square$

It is easy to see that **Shuffle<sub>ru</sub>** achieves the amplification of privacy in comparison with **Local<sub>ru</sub>** while a failure probability  $\delta$  emerges.

**THEOREM 4.6.** The estimate  $\hat{q}_{ru}(G)$  produced by **Shuffle<sub>ru</sub>** satisfies  $\mathbb{E}(\hat{q}_{ru}(G)) = q_{ru}(G)$ .

This suggests that **Shuffle<sub>ru</sub>** outputs an unbiased estimation of the assortativity factor. In addition, we have the following statement.

**THEOREM 4.7.** When  $\varepsilon, \delta$  are constants,  $\alpha \in (0, 1)$ ,  $\varepsilon_0 = \log(n) + O(1)$ , the estimate  $\hat{q}_{ru}(G)$  produced by **Shuffle<sub>ru</sub>** provides the following utility guarantee:

$$\text{MSE}(\hat{q}_{ru}(G)) = O\left(\frac{n^{1+\alpha}d_{\max}^4}{M^2} + \frac{n^3d_{\max}^2}{(\log n)^2 M^2} + \frac{n^3d_{\max}^6}{(\log n)^2 M^4}\right). \quad (22)$$

As before, a similar analysis implies that if the quantity  $d_{\max} = O(n^{1/2})$ , then  $\text{MSE}(\hat{q}_{ru}(G)) = O\left(n^{1+\alpha} + \frac{n^2}{(\log n)^2}\right)$ . Therefore, the estimation yielded by **Shuffle<sub>ru</sub>** has better steadiness than that of **Local<sub>ru</sub>**.

**THEOREM 4.8.** The time complexity of **Shuffle<sub>ru</sub>** is  $O(n^2)$ , and the space complexity is  $O(n^2)$ .

<sup>1</sup><https://github.com/apple/ml-shuffling-amplification>

This result may be proved using a similar analysis as used in proof of theorem 4.4, we thus omit it here.

In the next subsection, we will focus on the other setting that is somewhat different from that discussed in the two subsections above. As a consequence, the associated algorithm is established and analyzed in detail.

### 4.3 Computing Assortativity Factor from Extended Local Views

In some social networks, individuals have not only knowledge of their own connections but also a broader subgraph within their local neighborhood. Such a subgraph is called an Extended Local View (ELV) [53]. For example, on Facebook, each user can see all the connections of his/her friends by default. Similarly, in real-life social interactions, we often learn about the relationships among our friends during social events. In these scenarios, individuals can provide information about their neighbors' connections (i.e. 2-hop ELVs) to the data collector.

To accurately compute the assortativity factor while preserving privacy in one such scenario, we propose a DDP algorithm under the assumption that each user knows all the connections of his/her friends. In this algorithm, we apply the Laplace mechanism to perturb each user's degree  $d_i$  and the sum  $T_i = \sum_{j=1}^n a_{i,j} d_j$  of his/her friends' degrees. However, adding an edge can increase  $\{T_i, \dots, T_n\}$  by up to  $4n - 6$ . That is to say, the global sensitivity of  $\{T_i, \dots, T_n\}$  is  $4n - 6$  (according to definition 3.7). When  $n$  is large, we need to add a large amount of noise to  $T_i$  to ensure privacy, which significantly reduces the utility of the algorithm's output.

To address this issue, we adopt a probabilistic upper bound on the local sensitivity in place of the global sensitivity based on Lemma 3.9. Notice that the local sensitivity of  $\{T_1, \dots, T_n\}$  under DDP satisfies

$$LS_{DDP}(T) = \max_{v_i, v_j \in G \wedge i \neq j} 2(d_i + d_j). \quad (23)$$

Specifically, when an edge  $(v_i, v_j)$  is added or removed in  $G$ , (i)  $T_i$  changes by  $d_j$ , (ii)  $T_j$  changes by  $d_i$ , and (iii)  $\sum_{l \neq i, j} T_l$  changes by  $d_i + d_j$ .

It is easy to see that the probabilistic upper bound of  $LS_{DDP}(T)$  can be derived by calculating the probabilistic upper bounds of users' degrees. Specifically, we first perturb the degree of each user using the Laplace mechanism. Then, we use Lemma 3.9 to calculate the probabilistic upper bounds of all the noisy degrees. Finally, by substituting the two largest degree upper bounds into Eq.23, we can obtain the probabilistic upper bound of  $LS_{DDP}(T)$ .

By far, we are ready to propose the third algorithm. Algorithm 3 shows our algorithm for computing the assortativity factor from extended local views. We denote this algorithm by **Decentral<sub>ru</sub>**. It takes several inputs: ELVs of all users  $G_1, G_2, \dots, G_n$ , privacy budgets  $\epsilon_1, \epsilon_2$ , and a failure probability  $\delta$ .

The algorithm begins by splitting the privacy parameter  $\delta$ , which is handled by the data collector. The data collector then broadcasts the parameters to all users, i.e., nodes in the social network. Lines 2–7 are executed by each client, where each user computes his/her noisy degree  $\tilde{d}_i$  and a probabilistic upper bound  $d_i^*$  of the true degree  $d_i$ , and reports these values to the data collector. Based on the received data  $d_1^*, d_2^*, \dots, d_n^*$ , the data collector determines the

---

**Algorithm 3: Decentral<sub>ru</sub>.**  $[v_i]$  and  $[d]$  represent that the process is run by  $v_i$  and the data collector, respectively.

---

**Data:** ELVs of all users  $G_1, \dots, G_n$ , privacy budget  $\epsilon_1, \epsilon_2 \in \mathbb{R}_{\geq 0}$ , failure probability  $\delta \in [0, 1]$

**Result:** Private estimate  $\hat{q}_{ru}(G)$  of  $q_{ru}(G)$

---

```

1 [d]  $\delta_1 \leftarrow \delta/2$ ;
2 for  $i = 1$  to  $n$  do
3    $[v_i]$   $d_i \leftarrow \sum_{j=1}^n a_{i,j}$ ;
4    $[v_i]$   $\tilde{d}_i \leftarrow d_i + \text{Lap}\left(\frac{2}{\epsilon_1}\right)$ ;
5    $[v_i]$   $d_i^* \leftarrow \tilde{d}_i + \frac{2}{\epsilon_1} \log\left(\frac{1}{2\delta_1}\right)$ ;
6    $[v_i]$  Send  $\tilde{d}_i$  and  $d_i^*$  to the data collector;
7 end
8 [d] Sort  $\{d_i^*\}$  into  $\{d_{[1]}^*, d_{[2]}^*, \dots, d_{[n]}^*\}$  in descending
   order;
9 [d]  $\Delta \leftarrow 2(d_{[1]}^* + d_{[2]}^*)$ ;
10 for  $i = 1$  to  $n$  do
11    $\tilde{d}_i \leftarrow d_i + \text{Lap}\left(\frac{1}{\epsilon_0}\right)$ ;
12    $[v_i]$   $\tilde{T}_i \leftarrow T_i + \text{Lap}\left(\frac{\Delta}{\epsilon_2}\right)$ ;
13    $[v_i]$  Send  $\tilde{T}_i$  to the data collector;
14 end
15 [d]  $X_3 \leftarrow \frac{1}{2} \sum_{i=1}^n \tilde{d}_i \tilde{T}_i$ ;
16 [d]  $Y_3 \leftarrow \left(\frac{1}{2} \sum_{i=1}^n \tilde{d}_i^2 - \frac{4(n+2)}{\epsilon_1^2}\right)^2 - \frac{16(5n+4)}{\epsilon_1^4}$ ;
17 [d]  $\hat{q}_{ru}(G) \leftarrow X_3/M + Y_3/M^2$ ;
18 [d] return  $\hat{q}_{ru}(G)$ 

```

---

sensitivity required for perturbing  $T_i$ , and sends it back to each user. Then, each user applies the Laplace mechanism to compute the noisy value  $\tilde{T}_i$  and reports it to the data collector. Finally, the data collector estimates  $q_{ru}(G)$  as follows

$$\hat{q}_{ru}(G) = \frac{X_3}{M} - \frac{Y_3}{M^2}, \quad (24)$$

where

$$X_3 = \frac{1}{2} \sum_{i=1}^n \tilde{d}_i \tilde{T}_i, \quad (25a)$$

$$Y_3 = \left(\frac{1}{2} \sum_{i=1}^n \tilde{d}_i^2 - \frac{4(n+2)}{\epsilon_1^2}\right)^2 - \frac{16(5n+4)}{\epsilon_1^4}. \quad (25b)$$

Following the similar analysis as above, we prove that **Decentral<sub>ru</sub>** has the following properties.

**THEOREM 4.9.** **Decentral<sub>ru</sub>** provides  $(\epsilon, \delta)$ -edge DDP, where  $\epsilon = \epsilon_1 + \epsilon_2$  and  $\delta = 2\delta_1$ .

**PROOF.** Obviously, for any  $i \in \{1, \dots, n\}$ ,  $\tilde{d}_i = d_i + \text{Lap}\left(\frac{2}{\epsilon_1}\right)$  is protected with  $\epsilon_1$ -edge DDP. Next, we show that  $\Delta \geq LS_{DDP}(T_i)$  holds with probability at least  $1 - \delta$ . This is equivalent to proving that  $(d_{[1]}^* \geq d_{(1)}) \wedge (d_{[2]}^* \geq d_{(2)})$  holds with probability at least  $1 - \delta$ , where  $d_{(i)}$  denotes the  $i$ -th largest true degree.

Let  $\xi_i \in [0, 1]$  be the total probability that another user's degree other than  $d_{(i)}$  becomes  $d_{[i]}^*$  and meanwhile  $d_{[i]}^* \geq d_{(i)}$ . It is easy to deduce that for any  $i \in \{1, 2, \dots, n\}$ , we have

$$\begin{aligned} \Pr[d_{[i]}^* \geq d_{(i)}] &= \Pr[d_{(i)}^* \geq d_{(i)}] + \xi_i \\ &\geq \Pr[d_{(i)}^* \geq d_{(i)}] \\ &= 1 - \delta_1. \end{aligned} \quad (26)$$

Thus,

$$\begin{aligned} &\Pr\left[\left(d_{[1]}^* \geq d_{(1)}\right) \wedge \left(d_{[2]}^* \geq d_{(2)}\right)\right] \\ &= 1 - \Pr\left[\left(d_{[1]}^* < d_{(1)}\right) \vee \left(d_{[2]}^* < d_{(2)}\right)\right] \\ &\geq 1 - \left[\Pr\left(d_{[1]}^* < d_{(1)}\right) + \Pr\left(d_{[2]}^* < d_{(2)}\right)\right] \\ &= \Pr\left(d_{[1]}^* \geq d_{(1)}\right) + \Pr\left(d_{[2]}^* \geq d_{(2)}\right) - 1 \\ &\geq 1 - 2\delta_1 = 1 - \delta. \end{aligned} \quad (27)$$

Therefore,  $\tilde{T}_i = T_i + \text{Lap}\left(\frac{\Delta}{\varepsilon_2}\right)$  is protected under  $\varepsilon_2$ -edge DDP with probability  $1 - \delta$ . Overall, **Decentral<sub>ru</sub>** provides  $(\varepsilon, \delta)$ -edge DDP, where  $\varepsilon = \varepsilon_1 + \varepsilon_2$  and  $\delta = 2\delta_1$ .  $\square$

**THEOREM 4.10.** *The estimate  $\hat{q}_{ru}(G)$  produced by **Decentral<sub>ru</sub>** satisfies  $\mathbb{E}(\hat{q}_{ru}(G)) = q_{ru}(G)$ .*

**THEOREM 4.11.** *When  $\varepsilon_1, \varepsilon_2$  are constants, the estimate  $\hat{q}_{ru}(G)$  produced by **Decentral<sub>ru</sub>** provides the following utility guarantee:*

$$\text{MSE}(\hat{q}_{ru}(G)) = O\left(\frac{n^3 d_{\max}^6}{M^4}\right). \quad (28)$$

If quantity  $\hat{d}_{\max} = \Theta(n^{1/6})$ , we find that  $\text{MSE}(\hat{q}_{ru}(G)) = O(1)$ . In this setting, the consequence produced by **Decentral<sub>ru</sub>** is overwhelmingly better than that by **Local<sub>ru</sub>**. This implies that a broader view is beneficial to a steadier estimation given  $\hat{d}_{\max} = \Theta(n^{1/6})$ . Furthermore, if  $\hat{d}_{\max} \leq O(n^{1/2})$ , we have also  $\text{MSE}(\hat{q}_{ru}(G)) \leq O(n^2)$ .

**THEOREM 4.12.** *The time complexity of **Decentral<sub>ru</sub>** is  $O(n^2)$ , and the space complexity is  $O(n^2)$ .*

As above, this may also be proved in a similar manner as used in the proof of theorem 4.4.

By far, we finish the development of algorithms **Local<sub>ru</sub>**, **Shuffle<sub>ru</sub>** and **Decentral<sub>ru</sub>**, and give detailed theoretical analysis. In the next section, we are going to conduct experimental evaluations on synthetic datasets and real-world datasets to further clarify the performance of the proposed algorithms.

## 5 EXPERIMENT EVALUATION

In this section, we will conduct experiments on synthetic datasets and real-world datasets to evaluate the performance of our algorithms **Local<sub>ru</sub>**, **Shuffle<sub>ru</sub>** and **Decentral<sub>ru</sub>** proposed in Section 4.

### 5.1 Experimental Set-up

We perform experiments on six datasets whose parameters are summarized in Table 3. The first three are synthetic datasets with power-law degree distribution based on the BA (Barabási-Albert) graph model [48], where  $m$  represents the number of edges of a

newly added node. Here we use the NetworkX library [25] to generate BA synthetic graphs. The other three are real-world datasets from the Stanford Network Analysis Project (SNAP).

**Table 3: datasets**

	$n$	$M$	$d_{\max}$	$d_{\text{avg}}$	$r_u$
BA( $m = 10$ )	10000	99900	464	19.98	-95.22
BA( $m = 50$ )	10000	497500	1237	99.5	49.03
BA( $m = 100$ )	10000	990000	1640	198	736.81
Facebook	4039	88234	1045	43.69	870.36
Deezer	28281	92752	172	6.56	29.19
GitHub	37700	289003	9458	15.33	-162127.53

- **Facebook.** The Facebook online social network dataset (denoted by Facebook) [35] contains friend lists from survey participants using the Facebook application. It provides a social graph  $G = (V, E)$  with 4,039 nodes (Facebook users) and 88,234 undirected edges, where each edge  $(v_i, v_j) \in E$  represents an online friendship between  $v_i$  and  $v_j$ .
- **Deezer.** This dataset (denoted by Deezer) [51] includes a graph  $G$  with 28,281 nodes and 92,752 undirected edges, where nodes represent Deezer users from European countries and edges indicate mutual follower relationships.
- **GitHub.** This dataset (denoted by GitHub) [50] is a social network of GitHub developers with 37,700 nodes and 289,003 edges. Nodes are developers who have starred at least 10 repositories, and edges are the mutual follow relationships between them.

For each algorithm, we evaluate the relative error (as described in subsection 3.5) of the estimate as the value of  $\varepsilon$  varies. In our experiments, we set the range of  $\varepsilon$  to  $(0, 2]$ , which is acceptable in many practical scenarios [37]. To ensure reliable results, we run each algorithm 20 times and use the average relative error of these runs as the final experimental result. Additionally, given that the assortativity factor  $r_u$  reflects the sign of assortativity, it is useful to report the extent to which the privacy estimate of  $r_u$  reflects the true sign of assortativity, in addition to the relative error. Therefore, we also assess the accuracy of the sign of the estimate as  $\varepsilon$  varies. We run each algorithm 100 times and report the proportion of correct signs in the outputs as the sign accuracy of each algorithm. Our source code is available on GitHub<sup>2</sup>.

## 5.2 Experimental Results

**5.2.1 Relation between the Relative Error and  $\varepsilon$ .** We first evaluate the relationship between the relative error and the privacy parameter  $\varepsilon$  in edge DP. In this experiment, for **Local<sub>ru</sub>**, we divide the total privacy budget  $\varepsilon$  into  $\varepsilon_1 = 0.6\varepsilon$  for perturbing the adjacency matrix  $\mathbf{A}$  and  $\varepsilon_2 = 0.4\varepsilon$  for perturbing the users' degrees  $d_1, \dots, d_n$ . For **Shuffle<sub>ru</sub>**, we set the privacy allocation parameter  $\alpha$  to 0.4 and the failure probability  $\delta$  to  $10^{-8}$  ( $\ll n$ ) and use the numerical upper bound in [20] to calculate the local privacy budget  $\varepsilon_0$  in the shuffling model. For **Decentral<sub>ru</sub>**,  $\delta$  is also set to  $10^{-8}$ . Privacy budget  $\varepsilon$

<sup>2</sup><https://github.com/OYJZBYCG/Differential-Privacy-for-Network-Assortativity.git>

is divided into  $\epsilon_1 = 0.4\epsilon$  and  $\epsilon_2 = 0.6\epsilon$  for perturbation of the user's degree  $\{d_1, d_2, \dots, d_n\}$  and number of friends of the user's friends  $\{T_1, T_2, \dots, T_n\}$ , respectively.

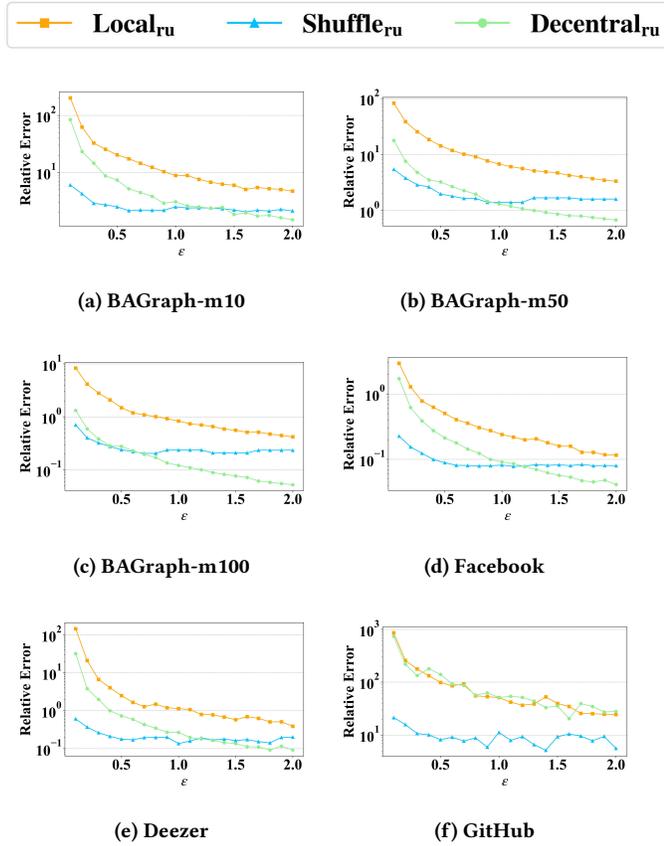


Figure 2: Relation between the relative error and  $\epsilon$  in edge DP.

All the experimental results are shown in Fig.2. It is apparent to see that for  $\epsilon$  in question, **Decentral<sub>ru</sub>** has a better performance than **Local<sub>ru</sub>**. This means that a broader view is conducive to obtaining better estimates. At the same time, given the low level of privacy budget  $\epsilon$ , **Shuffle<sub>ru</sub>** is always optimal in comparison with the other two algorithms as this algorithm has a strictly smaller relative error than them. This is in line with the theoretical analysis in the preceding section. This is mainly because its ingredient, Shuffle Model, owns its merit, i.e., privacy amplification effect. As encountered in the literature [12, 14, 26, 39], all the previous DP-based schemes prefer an assumption of low privacy budget. In a word, **Shuffle<sub>ru</sub>** is the best choice among these proposed algorithms.

**5.2.2 Relation between the Sign Accuracy and  $\epsilon$ .** Next, We evaluate the relationship between the sign accuracy and the privacy parameter  $\epsilon$  in edge DP. Fig.3 shows the results of the experiment. It is clear that the accuracy of **Decentral<sub>ru</sub>** is higher than that of **Local<sub>ru</sub>**, which reinforces the validity of using a broader view to improve our estimation accuracy. In addition, **Shuffle<sub>ru</sub>** is significantly better than the other two, and the accuracy is almost always

100%. This suggests that it is highly effective to take advantage of the shuffle model to improve the estimation accuracy under DP.

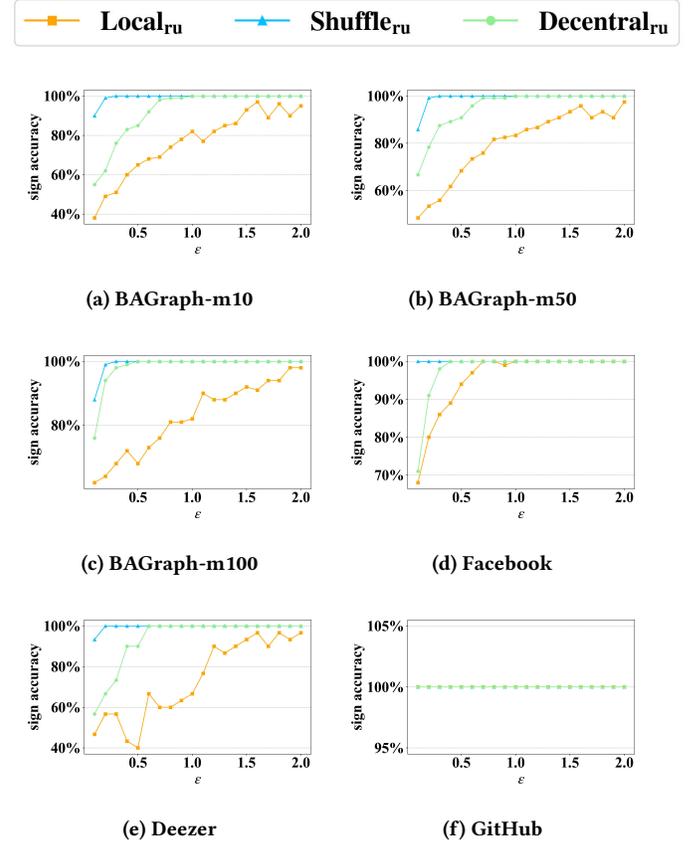


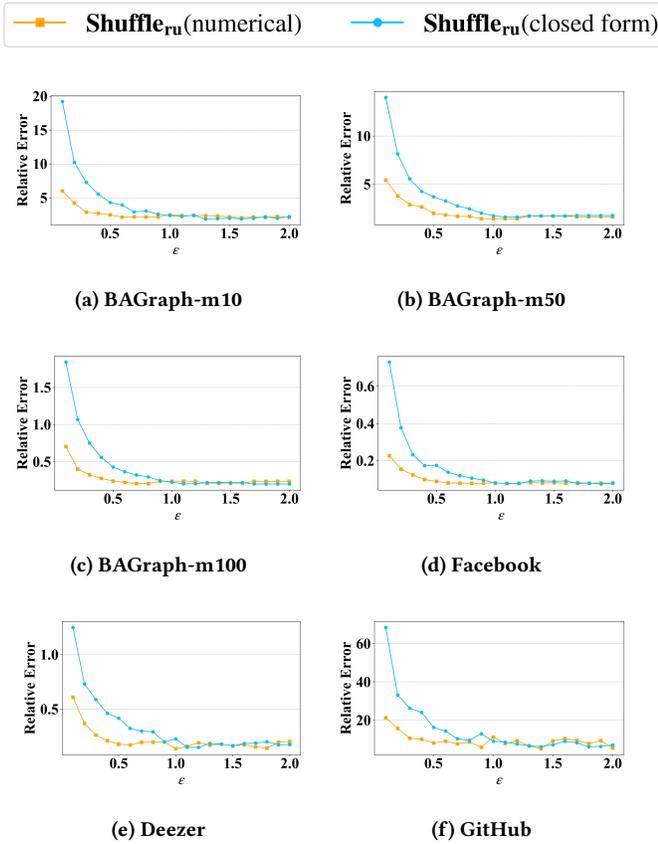
Figure 3: Relation between the sign accuracy and  $\epsilon$  in edge DP.

**5.2.3 Numerical bound vs. closed-form bound in Shuffle<sub>ru</sub>.** As mentioned in subsection 3.4, Feldman et al. provided two types of upper bounds on the privacy level in the shuffle model: a closed-form upper bound (see Eq.(11)) and a numerical upper bound (computed by the algorithm in [20]). Below, we compare the performance of **Shuffle<sub>ru</sub>** using these two bounds. Here, we also set  $\delta = 10^{-8}$ .

Figure 4 shows that when  $\epsilon$  is small, **Shuffle<sub>ru</sub>** equipped with the numerical upper bound achieves a lower relative error, and when  $\epsilon \geq 1$ , the relative errors of both methods are nearly the same. This is because, in both cases, the corresponding local privacy budget  $\epsilon_0$  approaches its maximum value of  $\log\left(\frac{n}{16\log(2/\delta)}\right)$  when  $\epsilon \geq 1$ . In summary, **Shuffle<sub>ru</sub>** with the numerical upper bound generally outperforms the version with the closed-form upper bound.

## 6 DISCUSSION

In this work, we study the estimation of network assortativity under DP for the first time and propose three algorithms **Local<sub>ru</sub>**, **Shuffle<sub>ru</sub>** and **Decentral<sub>ru</sub>**. As mentioned above, the techniques built in the development and analysis of algorithms have a wide



**Figure 4: Numerical bound vs. closed-form bound in  $\text{DeShuffle}_{\text{ru}}$ .**

range of applications. For example, these algorithms can be adapted to analyze the degree-degree correlation distribution of networks. Although we focus only on simple networks, the proposed algorithms can be slightly modified to achieve the same task on other types of networks including weighted networks, directed networks. Hence, we would like to see more applications based on our work in the near future.

On the other hand, our work is just the tip of the iceberg. There is still room for further improvement. For instance, we make use of some relatively rough methods when determining bounds for some quantities including MSE, time and space complexity. Therefore, it is of great interest to capture new bounds using advanced techniques, which is left as our next move. What’s more, it is a notable problem of how to derive the tight upper bound of MSE for DP-algorithm addressing the concern we are discussing in this work. In addition, it is also interesting to propose some new schemes for the purpose of accurately estimating network assortativity under LDP.

It is well known that there is a long history of network structure analysis [42, 45]. At the same time, it has received more attention to analyze structure of network in the requirement of privacy protection, such as subgraph counting [28], community detection [21] and graph synthesis [49]. This work studies another important structural property of network under DP. With the increased awareness

of privacy protection, there is growing emphasis on data privacy disclosure issues. Network, as a class of fundamental yet significant structure depicting data, will continue to gain more attention in the field of privacy-preserving data analysis. In the future, therefore, we explore other structural properties of network from the viewpoint of privacy protection.

## 7 CONCLUSION

In summary, we are the first to consider the problem of how to accurately estimate network assortativity under the demand of privacy protection. Specifically, we propose three DP-based algorithms, i.e.,  $\text{Local}_{\text{ru}}$ ,  $\text{Shuffle}_{\text{ru}}$  and  $\text{Decentral}_{\text{ru}}$ , to address this concern in two distinct scenarios. With rigorous theoretical analysis, we show that the proposed algorithms yield an unbiased estimation for network assortativity. Also, we make use of MSE to measure the steadiness of algorithms. At the same time, we determine time and space complexity of three algorithms. Additionally, we conduct extensive computer simulations, which demonstrate that experimental evaluations are in line with theoretical analysis. At last, we declare some potential applications of the light shed in the development of our algorithms and point out our next move.

## ACKNOWLEDGMENTS

The research was supported by the National Natural Science Foundation of China No. 62403381, the Fundamental Research Funds for the Central Universities No. G2023KY05105 and the Key Research and Development Plan of Shaanxi Province No. 2024GX-YBXM-021.

## REFERENCES

- [1] Réka Albert and Albert-László Barabási. 2002. Statistical mechanics of complex networks. *Reviews of Modern Physics* 74, 1 (2002), 47.
- [2] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. 2019. The privacy blanket of the shuffle model. In *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference*. Springer, 638–667.
- [3] Subhayan Bhattacharya, Sankhamita Sinha, and Sarbani Roy. 2020. Impact of structural properties on network structure for online social networks. *Procedia Computer Science* 167 (2020), 1200–1209.
- [4] Vincent Bindschaedler and Reza Shokri. 2016. Synthesizing plausible privacy-preserving location traces. In *2016 IEEE Symposium on Security and Privacy*. IEEE, 546–563.
- [5] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. 2017. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the 26th symposium on operating systems principles*. 441–459.
- [6] George Casella and Roger Berger. 2024. *Statistical inference*. CRC Press.
- [7] Hui Chang, Bei-Bei Su, Yue-Ping Zhou, and Da-Ren He. 2007. Assortativity and act degree distribution of some collaboration networks. *Physica A: Statistical Mechanics and its Applications* 383, 2 (2007), 687–702.
- [8] Rui Chen, Gergely Acs, and Claude Castelluccia. 2012. Differentially private sequential data publication via variable-length n-grams. In *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM, 638–649.
- [9] Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. 2019. Distributed differential privacy via shuffling. In *Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 375–403.
- [10] Matteo Cinelli, Gianmarco De Francisci Morales, Alessandro Galeazzi, Walter Quattrociocchi, and Michele Starnini. 2021. The echo chamber effect on social media. *Proceedings of the National Academy of Sciences* 118, 9 (2021), e2023301118.
- [11] Gregorio D’Agostino, Antonio Scala, Vinko Zlatić, and Guido Caldarelli. 2012. Robustness and assortativity for diffusion-like processes in scale-free networks. *Europhysics Letters* 97, 6 (2012), 68006.
- [12] Wei Yen Day, Ninghui Li, and Min Lyu. 2016. Publishing graph degree distribution with node differential privacy. In *Proceedings of the 2016 International Conference on Management of Data*. ACM, 123–138.
- [13] John C Duchi, Michael I Jordan, and Martin J Wainwright. 2013. Local privacy and statistical minimax rates. In *2013 IEEE 54th Annual Symposium on Foundations of*

- Computer Science. IEEE, 429–438.
- [14] Cynthia Dwork. 2006. Differential privacy. In *International Colloquium on Automata, Languages, and Programming*. Springer, 1–12.
- [15] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9, 3–4 (2014), 211–407.
- [16] Alexander Edmonds, Aleksandar Nikolov, and Jonathan Ullman. 2020. The power of factorization mechanisms in local and central differential privacy. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 425–438.
- [17] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. 2019. Amplification by shuffling: From local to central differential privacy via anonymity. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2468–2479.
- [18] Úlfar Erlingsson, Vasily Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*. 1054–1067.
- [19] Ernesto Estrada. 2011. Combinatorial study of degree assortativity in networks. *Physical Review E* 84, 4 (2011), 047101.
- [20] Vitaly Feldman, Audra McMillan, and Kunal Talwar. 2022. Hiding among the clones: A simple and nearly optimal analysis of privacy amplification by shuffling. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science*. IEEE, 954–964.
- [21] Nan Fu, Weiwei Ni, Lihe Hou, Dongyue Zhang, and Ruyi Zhang. 2024. Community detection in decentralized social networks with local differential privacy. *Information Sciences* 661 (2024), 120164.
- [22] Alexander A Ganin, Maksim Kitsak, Dayton Marchese, Jeffrey M Keisler, Thomas Seager, and Igor Linkov. 2017. Resilience and efficiency in transportation networks. *Science Advances* 3, 12 (2017), e1701079.
- [23] Antonious M Girgis, Deepesh Data, Suhas Diggavi, Ananda Theertha Suresh, and Peter Kairouz. 2021. On the renyi differential privacy of the shuffle model. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2321–2341.
- [24] Marko Gosak, Rene Markovič, Jurij Dolenšek, Marjan Slak Rupnik, Marko Marhl, Andraž Stožer, and Matjaž Perc. 2018. Network science of biological systems at different scales: A review. *Physics of Life Reviews* 24 (2018), 118–135.
- [25] Aric Hagberg, Pieter J Swart, and Daniel A Schult. 2008. *Exploring network structure, dynamics, and function using NetworkX*. Technical Report. Los Alamos National Laboratory (LANL), Los Alamos, NM (United States).
- [26] Michael Hay, Chao Li, Gerome Miklau, and David Jensen. 2009. Accurate estimation of the degree distribution of private networks. In *2009 Ninth IEEE International Conference on Data Mining*. IEEE, 169–178.
- [27] Quentin Hillebrand, Vorapong Suppakitpaisarn, and Tetsuo Shibuya. 2023. Unbiased locally private estimator for polynomials of laplacian variables. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 741–751.
- [28] Jacob Imola, Takao Murakami, and Kamalika Chaudhuri. 2021. Locally differentially private analysis of graph statistics. In *30th USENIX Security Symposium*. USENIX Association, 983–1000.
- [29] Jacob Imola, Takao Murakami, and Kamalika Chaudhuri. 2022. Communication-Efficient triangle counting under local differential privacy. In *31st USENIX Security Symposium*. USENIX Association, 537–554.
- [30] J. Imola, T. Murakami, and K. Chaudhuri. 2022. Differentially private triangle and 4-cycle counting in the shuffle model. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1505–1519.
- [31] Honglu Jiang, Jian Pei, Dongxiao Yu, Jiguo Yu, Bei Gong, and Xiuzhen Cheng. 2021. Applications of differential privacy in social network analysis: A survey. *IEEE Transactions on Knowledge and Data Engineering* 35, 1 (2021), 108–127.
- [32] Vishesh Karwa, Sofya Raskhodnikova, Adam Smith, and Grigory Yaroslavtsev. 2011. Private analysis of graph structure. *Proceedings of the VLDB Endowment* 4, 11 (2011), 1146–1157.
- [33] Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2013. Analyzing graphs with node differential privacy. In *Theory of Cryptography: 10th Theory of Cryptography Conference*. Springer, 457–476.
- [34] Tomasz J Kozubowski and Saralees Nadarajah. 2010. Multitude of Laplace distributions. *Statistical Papers* 51 (2010), 127–148.
- [35] Jure Leskovec and Julian McAuley. 2012. Learning to discover social circles in ego networks. *Advances in Neural Information Processing Systems* 25 (2012), 539–547.
- [36] CC Leung and HF Chau. 2007. Weighted assortative and disassortative networks model. *Physica A: Statistical Mechanics and its Applications* 378, 2 (2007), 591–602.
- [37] Ninghui Li, Min Lyu, Dong Su, and Weining Yang. 2017. *Differential privacy: From theory to practice*. Springer.
- [38] Fei Ma. 2024. Assortativity in networks. *arXiv* (2024), arXiv:submit/5594716.
- [39] Fei Ma, Renbo Zhu, and Ping Wang. 2024. PTT: Piecewise Transformation Technique for Analyzing Numerical Data under Local Differential Privacy. *IEEE Transactions on Mobile Computing* (2024).
- [40] David Melamed, Ashley Harrell, and Brent Simpson. 2018. Cooperation, clustering, and assortative mixing in dynamic networks. *Proceedings of the National Academy of Sciences* 115, 5 (2018), 951–956.
- [41] Kevin P Murphy. 2012. *Machine learning: a probabilistic perspective*. MIT press.
- [42] Mark Newman. 2018. *Networks*. Oxford University Press.
- [43] Mark EJ Newman. 2002. Assortative mixing in networks. *Physical Review Letters* 89, 20 (2002), 208701.
- [44] Mark EJ Newman. 2003. Mixing patterns in networks. *Physical Review E* 67, 2 (2003), 026126.
- [45] Mark EJ Newman. 2003. The structure and function of complex networks. *SIAM Review* 45, 2 (2003), 167–256.
- [46] Rogier Noldus and Piet Van Mieghem. 2015. Assortativity in complex networks. *Journal of Complex Networks* 3, 4 (2015), 507–542.
- [47] Mahendra Piraveenan, Mikhail Prokopenko, and AY Zomaya. 2008. Local assortativeness in scale-free networks. *Europhysics Letters* 84, 2 (2008), 28002.
- [48] Márton Pósfai and Albert-László Barabási. 2016. *Network science*. Citeseer.
- [49] Zhan Qin, Ting Yu, Yin Yang, Issa Khalil, Xiaokui Xiao, and Kui Ren. 2017. Generating synthetic decentralized social graphs with local differential privacy. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 425–438.
- [50] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. 2021. Multi-scale attributed node embedding. *Journal of Complex Networks* 9, 2 (2021), cnab014.
- [51] Benedek Rozemberczki and Rik Sarkar. 2020. Characteristic functions on graphs: Birds of a feather, from statistical descriptors to parametric models. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. ACM, 1325–1334.
- [52] Alessandra Sala, Xiaohan Zhao, Christo Wilson, Haitao Zheng, and Ben Y Zhao. 2011. Sharing graphs using differentially private graph models. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*. ACM, 81–98.
- [53] Haipei Sun, Xiaokui Xiao, Issa Khalil, Yin Yang, Zhan Qin, Hui Wang, and Ting Yu. 2019. Analyzing subgraph statistics from extended local views with decentralized differential privacy. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 703–717.
- [54] Stanley L Warner. 1965. Randomized response: A survey technique for eliminating evasive answer bias. *J. Amer. Statist. Assoc.* 60, 309 (1965), 63–69.
- [55] Chengkun Wei, Shouling Ji, Changchang Liu, Wenzhi Chen, and Ting Wang. 2020. AsgLDP: Collecting and generating decentralized attributed graphs with local differential privacy. *IEEE Transactions on Information Forensics and Security* 15 (2020), 3239–3254.
- [56] Qian Xiao, Rui Chen, and Kian-Lee Tan. 2014. Differentially private network data release via structural inference. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 911–920.
- [57] Xiaokui Xiao, Gabriel Bender, Michael Hay, and Johannes Gehrke. 2011. iReduct: Differential privacy with reduced relative errors. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. ACM, 229–240.
- [58] Haonan Yan, Xiaoguang Li, Hui Li, Jiamin Li, Wenhai Sun, and Fenghua Li. 2021. Monitoring-based differential privacy mechanism against query flooding-based model extraction attack. *IEEE Transactions on Dependable and Secure Computing* 19, 4 (2021), 2680–2694.
- [59] Qingqing Ye, Haibo Hu, Man Ho Au, Xiaofeng Meng, and Xiaokui Xiao. 2020. Towards locally differentially private generic graph metric estimation. In *2020 IEEE 36th International Conference on Data Engineering*. IEEE, 1922–1925.
- [60] Hailong Zhang, Sufian Latif, Raef Bassily, and Atanas Rountev. 2020. Differentially-private control-flow node coverage for software usage analysis. In *29th USENIX Security Symposium*. USENIX Association, 1021–1038.

## SUPPLEMENTARY MATERIAL

Below are more details that are omitted in the main text due to the space limitation.

### Proof of Theorem 4.2

**Theorem 4.2** *The estimate  $\hat{q}_{ru}(G)$  produced by  $\text{Local}_{ru}$  is unbiased, i.e.,  $\mathbb{E}(\hat{q}_{ru}(G)) = q_{ru}(G)$ .*

Before beginning with the detailed proof, we need to introduce a lemma as below.

LEMMA 7.1 ([34]). *Given a random variable  $X \sim \text{Lap}(x, b)$ , then*

$$\mathbb{E}(X^r) = \sum_{k=0}^r \left\{ \frac{1}{2} \left[ 1 + (-1)^k \right] \frac{r!}{(r-k)!} b^k x^{r-k} \right\}. \quad (\text{A.1})$$

Now, let us give the proof of Theorem 4.2.

PROOF. It is clear to see that  $\tilde{a}_{i,j}$  is in fact a Bernoulli random variable and  $\tilde{d}_i \sim \text{Lap}\left(d_i, \frac{1}{\varepsilon_2}\right)$ . Then, we have

$$\mathbb{E}(\tilde{a}_{i,j}) = a_{i,j}(1-p) + (1-a_{i,j})p, \quad (\text{A.2a})$$

$$\mathbb{E}(\tilde{d}_i^2) = d_i^2 + \frac{2}{\varepsilon_2^2}. \quad (\text{by Lemma 7.1}) \quad (\text{A.2b})$$

$$\mathbb{E}(\tilde{d}_i^4) = d_i^4 + \frac{12}{\varepsilon_2^2}d_i^2 + \frac{24}{\varepsilon_2^4} \quad (\text{by Lemma 7.1}) \quad (\text{A.2c})$$

Next, we move to the proof of unbiasedness of  $\hat{q}_{ru}(G)$ . First, we obtain

$$\begin{aligned} \mathbb{E}(X_1) &= \sum_{i=1}^n \sum_{j=1}^{i-1} \frac{E(\tilde{a}_{i,j}) - p}{1-2p} E(\tilde{d}_i) E(\tilde{d}_j) \\ &= \sum_{i=1}^n \sum_{j=1}^{i-1} \frac{[a_{i,j}(1-p) + (1-a_{i,j})p - p] d_i d_j}{1-2p} \\ &= \sum_{i=1}^n \sum_{j=1}^{i-1} a_{i,j} d_i d_j \\ &= \sum_{(v_i, v_j) \in E} d_i d_j. \end{aligned} \quad (\text{A.3})$$

Similarly, it is easy to check

$$\begin{aligned} \mathbb{E}(Y_1) &= \mathbb{E}\left[\left(\frac{1}{2} \sum_{i=1}^n \tilde{d}_i^2 - \frac{n+2}{\varepsilon_2^2}\right)^2 - \frac{5n+4}{\varepsilon_2^4}\right] \\ &= \mathbb{V}\left(\frac{1}{2} \sum_{i=1}^n \tilde{d}_i^2 - \frac{n+2}{\varepsilon_2^2}\right) - \frac{5n+4}{\varepsilon_2^4} \\ &\quad + \left[\mathbb{E}\left(\frac{1}{2} \sum_{i=1}^n \tilde{d}_i^2 - \frac{n+2}{\varepsilon_2^2}\right)\right]^2 \\ &= \frac{1}{4} \sum_{i=1}^n \left[\mathbb{E}(\tilde{d}_i^4) - \left[\mathbb{E}(\tilde{d}_i^2)\right]^2\right] - \frac{5n+4}{\varepsilon_2^4} \\ &\quad + \left[\frac{1}{2} \sum_{i=1}^n \mathbb{E}(\tilde{d}_i^2) - \frac{n+2}{\varepsilon_2^2}\right]^2 \\ &= \frac{1}{4} \sum_{i=1}^n \left(\frac{8}{\varepsilon_2^2} d_i^2 + \frac{20}{\varepsilon_2^4}\right) - \frac{5n+4}{\varepsilon_2^4} \\ &\quad + \left[\frac{1}{2} \sum_{i=1}^n \left(\frac{2}{\varepsilon_2^2} + d_i^2\right) - \frac{n+2}{\varepsilon_2^2}\right]^2 \\ &= \frac{2}{\varepsilon_2^2} \sum_{i=1}^n d_i^2 + \frac{5n}{\varepsilon_2^4} + \left[\frac{1}{2} \sum_{i=1}^n d_i^2 - \frac{2}{\varepsilon_2^2}\right]^2 - \frac{5n+4}{\varepsilon_2^4} \\ &= \left[\sum_{(v_i, v_j) \in E} \frac{1}{2} (d_i + d_j)\right]^2. \end{aligned} \quad (\text{A.4})$$

To sum up, we verify that

$$\mathbb{E}[\hat{q}_{ru}(G)] = \frac{\mathbb{E}(X_1)}{M} - \frac{\mathbb{E}(Y_1)}{M^2} = q_{ru}(G). \quad (\text{A.5})$$

□

### Proof of Theorem 4.3

**Theorem 4.3** When  $\varepsilon_1, \varepsilon_2$  are constants, the estimate  $\hat{q}_{ru}(G)$  produced by **Local<sub>ru</sub>** provides the following utility guarantee:

$$\text{MSE}\left(\frac{n^3 d_{\max}^2 + n^2 d_{\max}^4 + n^3 d_{\max}^6}{M^2} + \frac{n^3 d_{\max}^6}{M^4}\right). \quad (\text{B.1})$$

Let us introduce a lemma to succeed in verifying Theorem 4.3.

LEMMA 7.2 ([30]). Let  $x_1, x_2$  be two random variables, then  $\mathbb{V}(x_1 + x_2) \leq 4 \max\{\mathbb{V}(x_1), \mathbb{V}(x_2)\}$ .

From now on, we show the detailed proof of Theorem 4.3 as follows.

PROOF. Due to  $\mathbb{V}(a_{i,j}) = \mathbb{E}(a_{i,j}^2) - [\mathbb{E}(a_{i,j})]^2 = p(1-p)$ , we can obtain

$$\begin{aligned} \mathbb{E}(\hat{a}_{i,j}^2) &= \mathbb{V}(\hat{a}_{i,j}) + [\mathbb{E}(\hat{a}_{i,j})]^2 \\ &= \frac{p(1-p)}{(1-2p)^2} + a_{i,j}^2 \\ &= \frac{p(1-p)}{(1-2p)^2} + a_{i,j}. \end{aligned} \quad (\text{B.2})$$

From Theorem 4.2, it is clear to the eye that the estimate  $\hat{q}_{ru}(G)$  produced by **Local<sub>ru</sub>** is unbiased. By the bias-variance decomposition [41], the mean squared error (MSE) of  $\hat{q}_{ru}(G)$  is equal to its variance. Let  $P = \frac{X_1}{M}$  and  $Q = -\frac{Y_1}{M^2}$ , then

$$\begin{aligned} \text{MSE}(\hat{q}_{ru}(G)) &= \mathbb{V}(\hat{q}_{ru}(G)) = \mathbb{V}(P + Q) \\ &\leq 4 \max\{\mathbb{V}(P), \mathbb{V}(Q)\}. \end{aligned} \quad (\text{B.3})$$

(by Lemma 7.2)

Now, our task is to calculate  $\mathbb{V}(P)$  and  $\mathbb{V}(Q)$  separately.

For  $\mathbb{V}(P)$ , since  $\mathbb{V}(P) = M^{-2} \mathbb{V}(X_1)$ , we need to focus on calculation of  $\mathbb{V}(X_1)$ . For ease of presentation, we define  $\tilde{B}_{i,j} = \frac{(\tilde{a}_{i,j} - p)\tilde{d}_i \tilde{d}_j}{1-2p}$ , then

$$\begin{aligned} \mathbb{V}(X_1) &= \mathbb{V}\left(\sum_{i=2}^n \sum_{j=1}^{i-1} \tilde{B}_{i,j}\right) \\ &= \sum_{i=2}^n \mathbb{V}\left(\sum_{j=1}^{i-1} \tilde{B}_{i,j}\right) + \sum_{2 \leq k, l \leq n, k \neq l} \text{Cov}\left(\sum_{j=1}^{k-1} \tilde{B}_{k,j}, \sum_{j=1}^{l-1} \tilde{B}_{l,j}\right) \\ &= \sum_{i=2}^n \sum_{j=1}^{i-1} \mathbb{V}(\tilde{B}_{i,j}) + \sum_{i=2}^n \sum_{1 \leq k, l \leq i-1, k \neq l} \text{Cov}(\tilde{B}_{i,k}, \tilde{B}_{i,l}) \\ &\quad + \sum_{2 \leq k, l \leq n, k \neq l} \sum_{j=1}^{k-1} \sum_{t=1}^{l-1} \text{Cov}(\tilde{B}_{k,j}, \tilde{B}_{l,t}) \\ &= P_1 + P_2 + P_3, \end{aligned} \quad (\text{B.4})$$

where  $P_1 = \sum_{i=2}^n \sum_{j=1}^{i-1} \mathbb{V}(\tilde{B}_{i,j})$ ,  $P_2 = \sum_{i=2}^n \sum_{1 \leq k, l \leq i-1, k \neq l} \text{Cov}(\tilde{B}_{i,k}, \tilde{B}_{i,l})$  and  $P_3 = \sum_{2 \leq k, l \leq n, k \neq l} \sum_{j=1}^{k-1} \sum_{t=1}^{l-1} \text{Cov}(\tilde{B}_{k,j}, \tilde{B}_{l,t})$ .

Next, we calculate  $P_1$ ,  $P_2$  and  $P_3$  respectively, and obtain

$$\begin{aligned}
 P_1 &= \sum_{i=2}^n \sum_{j=1}^{i-1} \mathbb{V}(\tilde{B}_{i,j}) \\
 &= \sum_{i=2}^n \sum_{j=1}^{i-1} \left[ \mathbb{E} \left[ \left( \tilde{B}_{i,j} \right)^2 \right] - \left[ \mathbb{E}(\tilde{B}_{i,j}) \right]^2 \right] \\
 &= \sum_{i=2}^n \sum_{j=1}^{i-1} \left[ \left( \frac{p(1-p)}{(1-2p)^2} + a_{i,j} \right) \left( \frac{2}{\epsilon_2^2} + d_i^2 \right) \left( \frac{2}{\epsilon_2^2} + d_j^2 \right) \right] \\
 &\quad - \sum_{i=2}^n \sum_{j=1}^{i-1} a_{i,j} d_i^2 d_j^2 \\
 &= \sum_{i=2}^n \sum_{j=1}^{i-1} \left[ \frac{p(1-p)}{(1-2p)^2} \left( \frac{2}{\epsilon_2^2} + d_i^2 \right) \left( \frac{2}{\epsilon_2^2} + d_j^2 \right) \right] \\
 &\quad + \sum_{i=2}^n \sum_{j=1}^{i-1} \frac{2}{\epsilon_2^2} a_{i,j} \left( \frac{2}{\epsilon_2^2} + d_i^2 + d_j^2 \right) \\
 &\leq \sum_{i=2}^n \sum_{j=1}^{i-1} \left[ \frac{p(1-p)}{(1-2p)^2} \left( \frac{2}{\epsilon_2^2} + d_{\max}^2 \right) \left( \frac{2}{\epsilon_2^2} + d_{\max}^2 \right) \right] \\
 &\quad + \sum_{i=2}^n \sum_{j=1}^{i-1} \frac{2}{\epsilon_2^2} \left( \frac{2}{\epsilon_2^2} + 2d_{\max}^2 \right) \\
 &= O\left(n^2 d_{\max}^4\right),
 \end{aligned} \tag{B.5a}$$

$$\begin{aligned}
 P_2 &= \sum_{i=2}^n \sum_{1 \leq k, l \leq i-1, k \neq l} \text{Cov}(\tilde{B}_{i,k}, \tilde{B}_{i,l}) \\
 &= \sum_{i=2}^n \sum_{1 \leq k, l \leq i-1, k \neq l} \left[ \mathbb{E}(\tilde{B}_{i,k} \tilde{B}_{i,l}) - \mathbb{E}(\tilde{B}_{i,k}) \mathbb{E}(\tilde{B}_{i,l}) \right] \\
 &= \sum_{i=2}^n \sum_{1 \leq k, l \leq i-1, k \neq l} \left[ a_{i,k} a_{i,l} \left( \frac{2}{\epsilon_2^2} + d_i^2 \right) d_k d_l \right] \\
 &\quad - \sum_{i=2}^n \sum_{1 \leq k, l \leq i-1, k \neq l} a_{i,k} a_{i,l} d_i^2 d_k d_l \\
 &= \frac{2}{\epsilon_2^2} \sum_{i=2}^n \sum_{1 \leq k, l \leq i-1, k \neq l} a_{i,k} a_{i,l} d_k d_l \\
 &\leq \frac{2}{\epsilon_2^2} \sum_{i=2}^n \sum_{1 \leq k, l \leq i-1, k \neq l} d_{\max}^2 \\
 &= O\left(n^3 d_{\max}^2\right),
 \end{aligned} \tag{B.5b}$$

$$\begin{aligned}
 P_3 &= \sum_{2 \leq k, l \leq n, k \neq l} \sum_{j=1}^{k-1} \sum_{t=1}^{l-1} \text{Cov}(\tilde{B}_{k,j}, \tilde{B}_{l,t}) \\
 &= \sum_{2 \leq k, l \leq n, k \neq l} \sum_{j=1}^{k-1} \sum_{t=1}^{l-1} \mathbb{E}(\tilde{B}_{k,j} \tilde{B}_{l,t}) \\
 &\quad - \sum_{2 \leq k, l \leq n, k \neq l} \sum_{j=1}^{k-1} \sum_{t=1}^{l-1} \mathbb{E}(\tilde{B}_{k,j}) \mathbb{E}(\tilde{B}_{l,t}) \\
 &= \sum_{2 \leq k, l \leq n, k \neq l} \sum_{j=1}^{k-1} \sum_{t=1}^{l-1} a_{k,j} a_{l,t} \mathbb{E}(\tilde{d}_k \tilde{d}_j \tilde{d}_l \tilde{d}_t) \\
 &\quad - \sum_{2 \leq k, l \leq n, k \neq l} \sum_{j=1}^{k-1} \sum_{t=1}^{l-1} a_{k,j} d_k d_j a_{l,t} d_l d_t \\
 &= 2 \sum_{2 \leq k < l \leq n, k \neq l} \sum_{j=1}^{k-1} \left[ \frac{2}{\epsilon_2^2} a_{k,j} a_{l,t} d_k d_l + \frac{2}{\epsilon_2^2} a_{k,j} a_{l,t} d_j d_l \right] \\
 &= \frac{4}{\epsilon_2^2} \sum_{2 \leq k < l \leq n, k \neq l} \sum_{j=1}^{k-1} [a_{k,j} a_{l,t} d_k d_l + a_{k,j} a_{l,t} d_j d_l] \\
 &\leq \frac{4}{\epsilon_2^2} \sum_{2 \leq k < l \leq n, k \neq l} \sum_{j=1}^{k-1} 2d_{\max}^2 \\
 &= O\left(n^3 d_{\max}^2\right).
 \end{aligned} \tag{B.5c}$$

Thus,

$$\mathbb{V}(X_1) = O\left(n^3 d_{\max}^3 + n^2 d_{\max}^4\right). \tag{B.6}$$

This leads to the following expression

$$\mathbb{V}(P) = O\left(\frac{n^3 d_{\max}^3 + n^2 d_{\max}^4}{M^2}\right). \tag{B.7}$$

Below we move on to the calculation of  $\mathbb{V}(Q)$ . By Lemma 7.1, we first derive

$$\mathbb{E}(\tilde{d}_i^6) = d_i^6 + \frac{30}{\epsilon_2^2} d_i^4 + \frac{360}{\epsilon_2^4} d_i^2 + \frac{720}{\epsilon_2^6}, \tag{B.8a}$$

$$\mathbb{E}(\tilde{d}_i^8) = d_i^8 + \frac{56}{\epsilon_2^2} d_i^6 + \frac{1680}{\epsilon_2^4} d_i^4 + \frac{20160}{\epsilon_2^6} d_i^2 + \frac{40320}{\epsilon_2^8}. \tag{B.8b}$$

To make further progress, we write

$$\begin{aligned} \mathbb{V}(\tilde{d}_i^2) &= \mathbb{E}(\tilde{d}_i^4) - \left[\mathbb{E}(\tilde{d}_i^2)\right]^2 \\ &= \left(d_i^4 + \frac{12}{\varepsilon_2^2}d_i^2 + \frac{24}{\varepsilon_2^4}\right) - \left(d_i^2 + \frac{2}{\varepsilon_2^2}\right)^2 \\ &= \frac{8}{\varepsilon_2^2}d_i^2 + \frac{20}{\varepsilon_2^4}, \end{aligned} \quad (\text{B.9})$$

and

$$\begin{aligned} \mathbb{V}(\tilde{d}_i^4) &= \mathbb{E}(\tilde{d}_i^8) - \left[\mathbb{E}(\tilde{d}_i^4)\right]^2 \\ &= \frac{32}{\varepsilon_2^2}d_i^6 + \frac{1488}{\varepsilon_2^4}d_i^4 + \frac{19584}{\varepsilon_2^6}d_i^2 + \frac{39744}{\varepsilon_2^8}. \end{aligned} \quad (\text{B.10})$$

Armed with the consequences above, let us derive quantity  $\mathbb{V}(Q)$  as below

$$\begin{aligned} \mathbb{V}(Q) &= \mathbb{V}\left(-\frac{Y_1}{M^2}\right) = \frac{1}{M^4}\mathbb{V}(Y_1) \\ &= \frac{1}{M^4}\mathbb{V}\left[\left(\frac{1}{2}\sum_{i=1}^n \tilde{d}_i^2 - \frac{n+2}{\varepsilon_2^2}\right)^2 - \frac{5n+4}{\varepsilon_2^4}\right] \\ &= \frac{1}{M^4}\mathbb{V}\left[\frac{1}{4}\left(\sum_{i=1}^n \tilde{d}_i^2\right)^2 - \frac{n+2}{\varepsilon_2^2}\sum_{i=1}^n \tilde{d}_i^2\right] \\ &= \frac{1}{M^4}\mathbb{V}\left[\frac{1}{4}\left(\sum_{i=1}^n \tilde{d}_i^4 + \sum_{i=1}^n \sum_{j=1, j \neq i}^n \tilde{d}_i^2 \tilde{d}_j^2\right) - \frac{n+2}{\varepsilon_2^2}\sum_{i=1}^n \tilde{d}_i^2\right] \\ &= \frac{1}{M^4}\mathbb{E}\left[\left(\frac{1}{4}\sum_{i=1}^n \tilde{d}_i^4 + \frac{1}{4}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \tilde{d}_i^2 \tilde{d}_j^2 - \frac{n+2}{\varepsilon_2^2}\sum_{i=1}^n \tilde{d}_i^2\right)^2\right] \\ &\quad - \frac{1}{M^4}\left[\mathbb{E}\left(\frac{1}{4}\sum_{i=1}^n \tilde{d}_i^4 + \frac{1}{4}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \tilde{d}_i^2 \tilde{d}_j^2 - \frac{n+2}{\varepsilon_2^2}\sum_{i=1}^n \tilde{d}_i^2\right)\right]^2 \\ &= \frac{1}{M^4}Q_1 - \frac{1}{M^4}Q_2, \end{aligned} \quad (\text{B.11})$$

where

$$\begin{aligned} Q_1 &= \mathbb{E}\left[\left(\frac{1}{4}\sum_{i=1}^n \tilde{d}_i^4 + \frac{1}{4}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \tilde{d}_i^2 \tilde{d}_j^2 - \frac{n+2}{\varepsilon_2^2}\sum_{i=1}^n \tilde{d}_i^2\right)^2\right], \\ Q_2 &= \left[\mathbb{E}\left(\frac{1}{4}\sum_{i=1}^n \tilde{d}_i^4 + \frac{1}{4}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \tilde{d}_i^2 \tilde{d}_j^2 - \frac{n+2}{\varepsilon_2^2}\sum_{i=1}^n \tilde{d}_i^2\right)\right]^2. \end{aligned}$$

Analogously, we need to calculate  $Q_1$  and  $Q_2$  separately. In essence, it is easy to derive

$$\begin{aligned} Q_1 &= \mathbb{E}\left[\left(\frac{1}{4}\sum_{i=1}^n \tilde{d}_i^4 + \frac{1}{4}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \tilde{d}_i^2 \tilde{d}_j^2 - \frac{n+2}{\varepsilon_2^2}\sum_{i=1}^n \tilde{d}_i^2\right)^2\right] \\ &= \frac{1}{16}\mathbb{E}\left[\left(\sum_{i=1}^n \tilde{d}_i^4\right)^2\right] + \frac{1}{16}\mathbb{E}\left[\left(\sum_{i=1}^n \sum_{j=1, j \neq i}^n \tilde{d}_i^2 \tilde{d}_j^2\right)^2\right] \\ &\quad + \frac{(n+2)^2}{\varepsilon_2^4}\mathbb{E}\left[\left(\sum_{i=1}^n \tilde{d}_i^2\right)^2\right] \\ &\quad + \frac{1}{8}\mathbb{E}\left[\left(\sum_{i=1}^n \tilde{d}_i^4\right)\left(\sum_{i=1}^n \sum_{j=1, j \neq i}^n \tilde{d}_i^2 \tilde{d}_j^2\right)\right] \\ &\quad - \frac{n+2}{2\varepsilon_2^2}\mathbb{E}\left[\left(\sum_{i=1}^n \tilde{d}_i^4\right)\left(\sum_{i=1}^n \tilde{d}_i^2\right)\right] \\ &\quad - \frac{n+2}{2\varepsilon_2^2}\mathbb{E}\left[\left(\sum_{i=1}^n \sum_{j=1, j \neq i}^n \tilde{d}_i^2 \tilde{d}_j^2\right)\left(\sum_{i=1}^n \tilde{d}_i^2\right)\right] \\ &= \frac{1}{16}\sum_{i=1}^n \mathbb{E}(\tilde{d}_i^8) + \frac{3}{16}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^4)\mathbb{E}(\tilde{d}_j^4) \\ &\quad + \frac{1}{4}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^6)\mathbb{E}(\tilde{d}_j^2) \\ &\quad + \frac{3}{8}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \sum_{k=1, k \neq i, j}^n \mathbb{E}(\tilde{d}_i^4)\mathbb{E}(\tilde{d}_j^2)\mathbb{E}(\tilde{d}_k^2) \\ &\quad + \frac{1}{16}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \sum_{k=1, k \neq i, j}^k \sum_{s=1, s \neq i, j, k}^n \mathbb{E}(\tilde{d}_i^2)\mathbb{E}(\tilde{d}_j^2)\mathbb{E}(\tilde{d}_k^2)\mathbb{E}(\tilde{d}_s^2) \\ &\quad + \frac{(n+2)^2}{\varepsilon_2^4}\sum_{i=1}^n \mathbb{E}(\tilde{d}_i^4) + \frac{(n+2)^2}{\varepsilon_2^4}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^2)\mathbb{E}(\tilde{d}_j^2) \\ &\quad - \frac{n+2}{2\varepsilon_2^2}\sum_{i=1}^n \mathbb{E}(\tilde{d}_i^6) - \frac{3(n+2)}{2\varepsilon_2^2}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^4)\mathbb{E}(\tilde{d}_j^2) \\ &\quad - \frac{n+2}{2\varepsilon_2^2}\sum_{i=1}^n \sum_{j=1, j \neq i}^n \sum_{k=1, k \neq i, j}^n \mathbb{E}(\tilde{d}_i^2)\mathbb{E}(\tilde{d}_j^2)\mathbb{E}(\tilde{d}_k^2), \end{aligned} \quad (\text{B.12a})$$

$$\begin{aligned}
Q_2 &= \left[ \mathbb{E} \left( \frac{1}{4} \sum_{i=1}^n \tilde{d}_i^4 + \frac{1}{4} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \tilde{d}_i^2 \tilde{d}_j^2 - \frac{n+2}{\varepsilon_2^2} \sum_{i=1}^n \tilde{d}_i^2 \right) \right]^2 \\
&= \left[ \frac{1}{4} \sum_{i=1}^n \mathbb{E}(\tilde{d}_i^4) + \frac{1}{4} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^2) \mathbb{E}(\tilde{d}_j^2) - \frac{n+2}{\varepsilon_2^2} \sum_{i=1}^n \mathbb{E}(\tilde{d}_i^2) \right]^2 \\
&= \frac{1}{16} \left[ \sum_{i=1}^n \mathbb{E}(\tilde{d}_i^4) \right]^2 + \frac{1}{16} \left[ \sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^2) \mathbb{E}(\tilde{d}_j^2) \right]^2 \\
&\quad + \frac{(n+2)^2}{\varepsilon_2^4} \left[ \sum_{i=1}^n \mathbb{E}(\tilde{d}_i^2) \right]^2 \\
&\quad + \frac{1}{8} \left[ \sum_{i=1}^n \mathbb{E}(\tilde{d}_i^4) \right] \left[ \sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^2) \mathbb{E}(\tilde{d}_j^2) \right] \\
&\quad - \frac{n+2}{2\varepsilon_2^2} \left[ \sum_{i=1}^n \mathbb{E}(\tilde{d}_i^4) \right] \left[ \sum_{i=1}^n \mathbb{E}(\tilde{d}_i^2) \right] \\
&\quad - \frac{n+2}{2\varepsilon_2^2} \left[ \sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^2) \mathbb{E}(\tilde{d}_j^2) \right] \left[ \sum_{i=1}^n \mathbb{E}(\tilde{d}_i^2) \right] \\
&= \frac{1}{16} \sum_{i=1}^n \left[ \mathbb{E}(\tilde{d}_i^4) \right]^2 + \frac{1}{16} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^4) \mathbb{E}(\tilde{d}_j^4) \\
&\quad + \frac{1}{4} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^4) \mathbb{E}(\tilde{d}_j^2) \mathbb{E}(\tilde{d}_k^2) \\
&\quad + \frac{1}{8} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \sum_{k=1, k \neq i, j}^n \mathbb{E}(\tilde{d}_i^4) \mathbb{E}(\tilde{d}_j^2) \mathbb{E}(\tilde{d}_k^2) \\
&\quad + \frac{1}{8} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \left[ \mathbb{E}(\tilde{d}_i^2) \right]^2 \left[ \mathbb{E}(\tilde{d}_j^2) \right]^2 \\
&\quad + \frac{1}{4} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \sum_{k=1, k \neq i, j}^n \left[ \mathbb{E}(\tilde{d}_i^2) \right]^2 \mathbb{E}(\tilde{d}_j^2) \mathbb{E}(\tilde{d}_k^2) \\
&\quad + \frac{1}{16} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \sum_{k=1, k \neq i, j}^k \sum_{s=1, s \neq i, j, k}^n \mathbb{E}(\tilde{d}_i^2) \mathbb{E}(\tilde{d}_j^2) \mathbb{E}(\tilde{d}_k^2) \mathbb{E}(\tilde{d}_s^2) \\
&\quad + \frac{(n+2)^2}{\varepsilon_2^4} \sum_{i=1}^n \left[ \mathbb{E}(\tilde{d}_i^2) \right]^2 + \frac{(n+2)^2}{\varepsilon_2^4} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^2) \mathbb{E}(\tilde{d}_j^2) \\
&\quad - \frac{n+2}{2\varepsilon_2^2} \sum_{i=1}^n \mathbb{E}(\tilde{d}_i^4) \mathbb{E}(\tilde{d}_i^2) - \frac{n+2}{2\varepsilon_2^2} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^4) \mathbb{E}(\tilde{d}_j^2) \\
&\quad - \frac{n+2}{\varepsilon_2^2} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \mathbb{E}(\tilde{d}_i^2) \mathbb{E}(\tilde{d}_j^2) \mathbb{E}(\tilde{d}_k^2) \\
&\quad - \frac{n+2}{2\varepsilon_2^2} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \sum_{k=1, k \neq i, j}^n \mathbb{E}(\tilde{d}_i^2) \mathbb{E}(\tilde{d}_j^2) \mathbb{E}(\tilde{d}_k^2).
\end{aligned} \tag{B.12b}$$

From Eqs.(B.12a) and (B.12b), we obtain

$$\begin{aligned}
\mathbb{V}(Q) &= \frac{1}{16M^4} \sum_{i=1}^n \left[ \mathbb{E}(\tilde{d}_i^8) - \left[ \mathbb{E}(\tilde{d}_i^4) \right]^2 \right] \\
&\quad + \frac{1}{4M^4} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \left[ \mathbb{E}(\tilde{d}_i^6) - \mathbb{E}(\tilde{d}_i^4) \mathbb{E}(\tilde{d}_i^2) \right] \mathbb{E}(\tilde{d}_j^2) \\
&\quad + \frac{1}{8M^4} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \left[ \mathbb{E}(\tilde{d}_i^4) \mathbb{E}(\tilde{d}_j^4) - \left[ \mathbb{E}(\tilde{d}_i^2) \right]^2 \left[ \mathbb{E}(\tilde{d}_j^2) \right]^2 \right] \\
&\quad + \frac{1}{4M^4} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \sum_{k=1, k \neq i, j}^n \left[ \mathbb{E}(\tilde{d}_i^4) - \left[ \mathbb{E}(\tilde{d}_i^2) \right]^2 \right] \mathbb{E}(\tilde{d}_j^2) \mathbb{E}(\tilde{d}_k^2) \\
&\quad + \frac{(n+2)^2}{\varepsilon_2^4 M^4} \sum_{i=1}^n \left[ \mathbb{E}(\tilde{d}_i^4) - \left[ \mathbb{E}(\tilde{d}_i^2) \right]^2 \right] \\
&\quad - \frac{n+2}{2\varepsilon_2^2 M^4} \sum_{i=1}^n \left[ \mathbb{E}(\tilde{d}_i^6) - \mathbb{E}(\tilde{d}_i^4) \mathbb{E}(\tilde{d}_i^2) \right] \\
&\quad - \frac{n+2}{\varepsilon_2^2 M^4} \sum_{i=1}^n \sum_{j=1, j \neq i}^n \left[ \mathbb{E}(\tilde{d}_i^4) - \left[ \mathbb{E}(\tilde{d}_i^2) \right]^2 \right] \mathbb{E}(\tilde{d}_j^2) \\
&= O\left(\frac{n^3 d_{\max}^6}{M^4}\right).
\end{aligned} \tag{B.13}$$

Finally, from Eqs.(B.7) and (B.13) it follows that

$$\text{MSE} = O\left(\frac{n^3 d_{\max}^2 + n^2 d_{\max}^4 + n^3 d_{\max}^6}{M^2} + \frac{n^3 d_{\max}^6}{M^4}\right). \tag{B.14}$$

□

## Proof of Theorem 4.6

**Theorem 4.6** *The estimate  $\hat{q}_{ru}(G)$  produced by  $\text{Shuffle}_{ru}$  satisfies  $\mathbb{E}(\hat{q}_{ru}(G)) = q_{ru}(G)$ .*

**PROOF.** First, we prove that  $X_2$  is unbiased, i.e.,

$$\begin{aligned}
\mathbb{E}(X_2) &= \mathbb{E}\left(\sum_{i=2}^n \hat{r}_i\right) \\
&= \mathbb{E}\left[\sum_{i=2}^n d_i \sum_{j=1}^{i-1} \frac{(\tilde{a}_{i,j} - p) \tilde{d}_j}{1 - 2p}\right] \\
&= \sum_{i=2}^n d_i \sum_{j=1}^{i-1} \mathbb{E}\left(\frac{\tilde{a}_{i,j} - p}{1 - 2p}\right) \mathbb{E}(\tilde{d}_j) \\
&= \sum_{i=2}^n d_i \sum_{j=1}^{i-1} a_{ij} d_j \\
&= \sum_{(v_i, v_j) \in E} d_i d_j.
\end{aligned} \tag{C.1}$$

We now consider  $Y_2$ . Since  $Y_2$  has the similar form as  $Y_1$ , and the unbiasedness of  $Y_1$  has been already proven in the proof of Theorem 4.2, it follows that  $Y_2$  is also unbiased. Thus, we have

$$\mathbb{E}(Y_2) = \left[ \sum_{(v_i, v_j) \in E} \frac{1}{2} (d_i + d_j) \right]^2. \tag{C.2}$$

Armed with the results above, we come to □

$$\mathbb{E}(\hat{q}_{ru}(G)) = q_{ru}(G). \quad (\text{C.3})$$

□

### Proof of Theorem 4.7

**Theorem 4.7** When  $\varepsilon, \delta$  are constants,  $\alpha \in (0, 1)$ ,  $\varepsilon_0 = \log(n) + O(1)$ , the estimate  $\hat{q}_{ru}(G)$  produced by **Shuffle<sub>ru</sub>** provides the following utility guarantee:

$$\text{MSE}(\hat{q}_{ru}(G)) = O\left(\frac{n^{1+\alpha}d_{\max}^4}{M^2} + \frac{n^3d_{\max}^2}{(\log n)^2 M^2} + \frac{n^3d_{\max}^6}{(\log n)^2 M^4}\right). \quad (\text{D.1})$$

**PROOF.** Let  $U = \frac{X_2}{M}$  and  $W = -\frac{Y_2}{M^2}$ , then the MSE of  $\hat{q}_{ru}(G)$  by **Shuffle<sub>ru</sub>** can be written as follows

$$\begin{aligned} \text{MSE}(\hat{q}_{ru}(G)) &= \mathbb{V}(\hat{q}_{ru}(G)) = \mathbb{V}(U + W) \\ &\leq 4 \max\{\mathbb{V}(U), \mathbb{V}(W)\}. \quad (\text{by Lemma 7.2}) \end{aligned} \quad (\text{D.2})$$

We now need to calculate  $\mathbb{V}(U)$  and  $\mathbb{V}(W)$  separately. Since the expression for  $W$  is similar to that of  $Q$  in the proof of Theorem 4.3, this leads to  $\mathbb{V}(W) \leq O\left(\frac{n^3d_{\max}^6}{(\log n)^2 M^4}\right)$ . Next, we only need to compute  $\mathbb{V}(U)$  to establish the upper bound of  $\text{MSE}(\hat{q}_{ru}(G))$ . For ease of presentation, we define  $\tilde{C}_{i,j} = \frac{(\tilde{a}_{i,j}-p)\tilde{d}_j}{1-2p}$ , then

$$\begin{aligned} \mathbb{V}(U) &= M^{-2} \mathbb{V}\left(\sum_{i=2}^n d_i \sum_{j=1}^{i-1} \tilde{C}_{i,j}\right) \\ &= M^{-2} \sum_{i=2}^n d_i^2 \mathbb{V}\left(\sum_{j=1}^{i-1} \tilde{C}_{i,j}\right) \\ &\quad + 2M^{-2} \sum_{2 \leq k < l \leq n} \text{Cov}\left(\sum_{j=1}^{k-1} d_k \tilde{C}_{k,j}, \sum_{h=1}^{l-1} d_l \tilde{C}_{l,h}\right) \\ &= M^{-2} \sum_{i=2}^n \frac{d_i^2}{(1-2p)^2} \sum_{j=1}^{i-1} \left[ p(1-p)d_j^2 + \frac{2(1-2p)^2 a_{i,j}}{\alpha^2 \varepsilon_0^2} + \frac{2p(1-p)}{\alpha^2 \varepsilon_0^2} \right] \\ &\quad + \frac{4M^{-2}}{\alpha^2 \varepsilon_0^2} \sum_{2 \leq k < l \leq n} \sum_{j=1}^{k-1} a_{k,j} a_{l,j} d_k d_l \\ &\leq \frac{p(1-p)}{(1-2p)^2 M^2} \sum_{i=2}^n \sum_{j=1}^{i-1} d_{\max}^4 + \frac{4}{\alpha^2 \varepsilon_0^2 M^2} \sum_{2 \leq k < l \leq n} (k-1) d_{\max}^2 \\ &= \frac{p(1-p)n(n-1)}{2(1-2p)^2 M^2} d_{\max}^4 + \frac{2n(n-1)(n-2)}{3\alpha^2 \varepsilon_0^2 M^2} d_{\max}^2 \\ &= \frac{n^\alpha(n-1)}{M^2} d_{\max}^4 + \frac{n(n-1)(n-2)}{(\log n)^2 M^2} d_{\max}^2 \\ &= O\left(\frac{n^{1+\alpha}d_{\max}^4}{M^2} + \frac{n^3d_{\max}^2}{(\log n)^2 M^2}\right). \end{aligned} \quad (\text{D.3})$$

Therefore, we gain

$$\text{MSE}(\hat{q}_{ru}(G)) = O\left(\frac{n^{1+\alpha}d_{\max}^4}{M^2} + \frac{n^3d_{\max}^2}{(\log n)^2 M^2} + \frac{n^3d_{\max}^6}{(\log n)^2 M^4}\right). \quad (\text{D.4})$$

### Proof of Theorem 4.10

**Theorem 4.10** The estimate  $\hat{q}_{ru}(G)$  produced by **Decentral<sub>ru</sub>** satisfies  $\mathbb{E}(\hat{q}_{ru}(G)) = q_{ru}(G)$ .

**PROOF.** Since  $\tilde{T}_i \sim \text{Lap}\left(\frac{d_{\max}}{\varepsilon_2}\right)$ , by Lemma 7.1, we have

$$\mathbb{E}(\tilde{T}_i) = T_i + \frac{2d_{\max}^2}{\varepsilon_2^2}. \quad (\text{E.1})$$

Then, we can easily obtain

$$\begin{aligned} \mathbb{E}(X_2) &= \mathbb{E}\left(\frac{1}{2} \sum_{i=1}^n \tilde{d}_i \tilde{T}_i\right) \\ &= \frac{1}{2} \sum_{i=1}^n d_i T_i \\ &= \sum_{(v_i, v_j) \in E} d_i d_j, \end{aligned} \quad (\text{E.2})$$

Following a similar derivation as in Eq.(A.4), we have

$$\mathbb{E}(Y_2) = \left[ \sum_{(v_i, v_j) \in E} \frac{1}{2} (d_i + d_j) \right]^2. \quad (\text{E.3})$$

Finally, we show that

$$\mathbb{E}(\hat{q}_{ru}(G)) = q_{ru}(G). \quad (\text{E.4})$$

□

### Proof of Theorem 4.11

**Theorem 4.11** When  $\varepsilon_1, \varepsilon_2$  are constants, the estimate  $\hat{q}_{ru}(G)$  produced by **Decentral<sub>ru</sub>** provides the following utility guarantee:

$$\text{MSE}(\hat{q}_{ru}(G)) = O\left(\frac{n^3d_{\max}^6}{M^4}\right). \quad (\text{F.1})$$

**PROOF.** Since  $\tilde{T}_i \sim \text{Lap}\left(T_i, \frac{d_{\max}}{\varepsilon_2}\right)$ , then by Lemma 7.1, we can get

$$\mathbb{E}(\tilde{T}_i^2) = T_i^2 + \frac{2d_{\max}^2}{\varepsilon_2^2}, \quad (\text{F.2})$$

Below we calculate the MSE of  $q_{ru}(G)$  produced by **Decentral<sub>ru</sub>**. Let  $H = \frac{X_3}{M}$  and  $S = -\frac{Y_3}{M^2}$ , then

$$\begin{aligned} \text{MSE}(\hat{q}_{ru}(G)) &= \mathbb{V}(\hat{q}_{ru}(G)) = \mathbb{V}(H + S) \\ &\leq 4 \max\{\mathbb{V}(H), \mathbb{V}(S)\}. \quad (\text{by Lemma 7.2}) \end{aligned} \quad (\text{F.3})$$

Next, we calculate  $\mathbb{V}(H)$  and  $\mathbb{V}(S)$  respectively.

$$\begin{aligned}
 \mathbb{V}(H) &= \mathbb{V}\left(\frac{X_3}{M}\right) = M^{-2}\mathbb{V}(X_3) \\
 &= \frac{1}{4M^2} \sum_{i=1}^n \mathbb{V}(\hat{d}_i \hat{T}_i) \\
 &= \frac{1}{4M^2} \sum_{i=1}^n \left[ \mathbb{E}(\hat{d}_i^2 \hat{T}_i^2) - \mathbb{E}(\hat{d}_i \hat{T}_i)^2 \right] \\
 &= \frac{1}{4M^2} \sum_{i=1}^n \left[ \mathbb{E}(\hat{d}_i^2) \mathbb{E}(\hat{T}_i^2) - \mathbb{E}(\hat{d}_i)^2 \mathbb{E}(\hat{T}_i)^2 \right] \\
 &= \frac{1}{4M^2} \sum_{i=1}^n \left[ \left( d_i^2 + \frac{2}{\varepsilon_1^2} \right) \left( T_i^2 + \frac{2(d_{[1]}^* + d_{[2]}^*)^2}{\varepsilon_2^2} \right) - d_i^2 T_i^2 \right] \\
 &= \frac{1}{4M^2} \sum_{i=1}^n \frac{2\varepsilon_2^2 T_i^2 + 2\varepsilon_1^2 d_i^2 (d_{[1]}^* + d_{[2]}^*)^2 + 4(d_{[1]}^* + d_{[2]}^*)^2}{\varepsilon_1^2 \varepsilon_2^2} \\
 &\leq \frac{1}{4M^2} \sum_{i=1}^n \frac{2\varepsilon_2^2 d_{\max}^4 + 2\varepsilon_1^2 d_{\max}^2 (d_{[1]}^* + d_{[2]}^*)^2 + 4(d_{[1]}^* + d_{[2]}^*)^2}{\varepsilon_1^2 \varepsilon_2^2} \\
 &= O\left(\frac{nd_{\max}^2 (d_{[1]}^* + d_{[2]}^*)^2}{M^2}\right).
 \end{aligned}$$

(F.4)

Since the expression for  $S$  is similar to the expression for  $Q$  in the proof of Theorem 4.3, we obtain  $\mathbb{V}(S) \leq O\left(\frac{n^3 d_{\max}^6}{M^4}\right)$ .

To sum up, we have

$$\text{MSE}(\hat{q}_{ru}(G)) = O\left(\frac{n^3 d_{\max}^6}{M^4}\right). \quad (\text{F.5})$$

□