InFoRM: Individual Fairness on Graph Mining

Jian Kang[†], Jingrui He[†], Ross Maciejewski[‡], and Hanghang Tong[†]
[†]University of Illinois at Urbana-Champaign, {jiank2, jingrui, htong}@illinois.edu;
[‡]Arizona State University, rmacieje@asu.edu

ABSTRACT

Algorithmic bias and fairness in the context of graph mining have largely remained nascent. The sparse literature on fair graph mining has almost exclusively focused on group-based fairness notation. However, the notion of individual fairness, which promises the fairness notion at a much finer granularity, has not been well studied. This paper presents the first principled study of Individual Fairness on gRaph Mining (InFoRM). First, we present a generic definition of individual fairness for graph mining which naturally leads to a quantitative measure of the potential bias in graph mining results. Second, we propose three mutually complementary algorithmic frameworks to mitigate the proposed individual bias measure, namely debiasing the input graph, debiasing the mining model and debiasing the mining results. Each algorithmic framework is formulated from the optimization perspective, using effective and efficient solvers, which are applicable to multiple graph mining tasks. Third, accommodating individual fairness is likely to change the original graph mining results without the fairness consideration. We conduct a thorough analysis to develop an upper bound to characterize the cost (i.e., the difference between the graph mining results with and without the fairness consideration). We perform extensive experimental evaluations on real-world datasets to demonstrate the efficacy and generality of the proposed methods.

CCS CONCEPTS

• **Information systems → Data mining**; • **Applied computing → Law, social and behavioral sciences**.

ACM Reference Format:
Jian Kang[†], Jingrui He[†], Ross Maciejewski[‡], and Hanghang Tong[†]. 2020. InFoRM: Individual Fairness on Graph Mining. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, CA, USA*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3394486.3403080

1 INTRODUCTION

In an increasingly connected world, graph mining is playing a more and more important role in many application domains, such as information retrieval [29], community detection [1], recommender systems [36] and security [37]. Decades of research in graph mining have produced a wealth of powerful computational models and algorithms. Despite the remarkable progress, several fundamental questions in relation to algorithmic fairness have remained largely unanswered, e.g., *are the graph mining results fair? If not, how to best mitigate the bias? How would fairness impact the graph mining performance (e.g. ranking precision, classification accuracy)?*

The notion of algorithmic fairness has attracted much attention. To date, researchers have developed a large collection of fair machine learning measures and algorithms, typically for spatial or text data, including statistical measures (e.g., disparate impact [6], statistical parity, equal odds [9]) and causal reasoning-based measures (e.g., counterfactual fairness [30]). Unlike these settings, a major challenge of fair graph mining lies in the non-IID nature of graph data. Since the data samples (i.e., nodes) in a graph are inter-connected, the fundamental IID assumption behind classical fair machine learning methods might be violated. To address this issue, several methods have emerged in recent years, which aim to generalize the traditional fairness notation and bias mitigation algorithms to graph data. For example, fair spectral clustering [15] has been studied to ensure that each cluster contains approximately the same number of elements from each demographic group [4]. Fairness in graph embedding by fulfilling statistical parity has also been explored [3, 24]. Furthermore, there has been research work on fairness in recommendations to enforce statistical parity [11] or other parity-based fairness that measures the differences between model behaviors for advantaged users and disadvantaged users [31]. It is worth pointing out that the vast majority of the existing work on fair graph mining [3, 4, 11, 15, 24, 31] has almost exclusively been focusing on *group-based fairness*.

However, individual fairness [5] has not been well studied in the context of graph mining. The notation of individual fairness is rooted in the Merriam-Webster's dictionary definition of fairness[1]. In the context of algorithmic fairness, this often translates into a generic design principle that *any two individuals who are similar should receive similar algorithmic outcomes*. Compared with the group-based notation, the individual-based approach offers a fairness measure at a much finer granularity (i.e., at the node level). A thorough study of individual fairness in the context of graph mining will complement and expand the current landscape of fair graph mining, which has mostly focused on group-based fairness. Broadly speaking [26], since bias and discrimination could happen in different forms due to the various settings of tasks with machine automation, there have been debates on which fairness notion should be applied in a given context. The fine granularity of individual fairness might provide a natural remedy for such debates.

This paper presents the first principled study of Individual Fairness on gRaph Mining, which is referred to as the InFoRM problem. We focus on three fundamental questions:

[1]In Merriam-Webster, it is defined as *'lack of favoritism from one side or another'*.

Q1. *(InFoRM Measures)* For the traditional machine learning setting, a major concern of individual fairness lies in its requirement of an appropriate similarity measure, which often requires solving a non-trivial problem that may need expert knowledge to address legal, ethical or social concerns [5]. Given the rich similarity measures for the graph data [10, 16, 27], we seek to answer the following: *Given a graph mining model and an arbitrary similarity measure, how can we tell if the mining results are fair? If the results are not fair, how can we quantitatively measure the overall bias?*

Q2. *(InFoRM Algorithms)* Generally speaking, a graph mining method consists of three components, including the input graph, the mining model and the mining results. Each of these three components could introduce and/or amplify the aforementioned bias. *How can we develop generic, effective and efficient algorithms to mitigate such bias by adjusting the input graph, or the mining model, or the mining results, respectively?*

Q3. *(InFoRM Cost)* By mitigating the bias, it is likely to alter the original graph mining results without the fairness consideration, and thus might degrade the mining performance (e.g., ranking, clustering, embedding, etc.). *How can we quantitatively characterize such cost, i.e., to what extent the debiased graph mining results will deviate from the ones without the fairness constraint?*

For Q1, we present a generic definition of individual fairness on graph mining based on the Lipschitz property. The proposed fairness measure naturally enables us to quantify the overall bias of graph mining results by the trace of a quadratic form of the mining results. For Q2, building upon the individual fairness measure from Q1, we propose three mutually complementary algorithmic frameworks to mitigate the proposed bias measure: debiasing the input graph, debiasing the mining model, and debiasing the mining results. For each algorithmic framework, we formulate the framework as an optimization problem, develop effective and efficient solvers, and demonstrate that the framework is applicable to multiple graph mining tasks. In order to debias the input graph, we formulate it as a bi-level optimization problem, where the extra level of optimization can be effectively solved by its KKT conditions [12]. To debias the mining model, we show that the extra time cost incurred due to the fairness consideration is only linear w.r.t. the number of similarity links. To debias the mining results, we develop a closed-form solution that is applicable to *any* graph mining task whose results are in the form of a matrix. For Q3, we develop an upper bound on the difference between debiased mining results and the original mining results. Our analysis reveals that the cost of ensuring individual fairness is closely-related to the input graph structure (e.g., the rank, the spectral norm, etc.).

To our best knowledge, we are the first to study individual fairness on graph mining. In addition to the problem definition, the main contributions of this paper can be summarized as follows.

- **Measure.** We provide a novel definition of individual fairness on graph mining, which is capable of (1) identifying if the mining results are fair w.r.t. any given graph similarity measure, and (2) measuring the individual bias as the trace of a quadratic form in the mining results.
- **Algorithms.** We propose generic, effective and efficient algorithms to mitigate individual bias through the input graph graph, the mining model and mining results.
- **Analysis.** We provide analysis to (1) understand the quality, complexities and applicability of the proposed debiasing algorithms, and (2) reveal the key factors that impact the cost for accommodating individual fairness on graph mining.
- **Evaluations.** We perform extensive empirical evaluations on real-world datasets, which demonstrate that our proposed methods are effective and efficient in reducing bias while preserving the performance of vanilla graph mining models.

2 PROBLEM DEFINITION

In this section, we present the key symbols used throughout the paper (Table 1). Then, we review the general procedure of several classic graph mining algorithms from the optimization viewpoint. Finally, we formally define three problems of individual fairness for graph mining.

Table 1: Table of symbols.

Symbols	Definitions and Descriptions
$\mathbf{A}$	a matrix
$\mathbf{A}'$	transpose of matrix $\mathbf{A}$
$\mathbf{A}^{-1}$	inverse of matrix $\mathbf{A}$
$\mathbf{A}^{+}$	pseudo-inverse of matrix $\mathbf{A}$
$\mathbf{L}_{\mathbf{A}}$	Laplacian matrix of $\mathbf{A}$
$\mathbf{u}$	a vector
$l(\cdot)$	the loss function for a mining task
$\mathbf{S}$	node-node similarity matrix
$\mathbf{Y}$	graph mining results
$\bar{\mathbf{Y}}$	vanilla graph mining results
$\mathbf{Y}^{*}$	debiased graph mining results
θ	a set of parameters
n, m_1	number of nodes and edges
m_2	number of similarity links
r	dimension of mining results $\mathbf{Y}[i,:]$ for node i
c	damping factor for PageRank
k	number of clusters
d	dimension of node embedding

We use bold upper-case letters for matrices (e.g. $\mathbf{A}$), bold lower-case letters for vectors (e.g. $\mathbf{u}$) and lower-case letters for scalars (e.g. c). Regarding matrix indexing conventions, we use rules similar to Numpy. We use $\mathbf{A}[i,j]$ to represent the entry of matrix $\mathbf{A}$ at the i^{th} row and the j^{th} column, $\mathbf{A}[i,:]$ to represent the i^{th} row of matrix $\mathbf{A}$ and $\mathbf{A}[:,j]$ to represent the j^{th} column of matrix $\mathbf{A}$. We use prime to represent the transpose of a matrix (i.e. $\mathbf{A}'$ is the transpose of matrix $\mathbf{A}$) and the superscript plus sign to represent the pseudo-inverse of matrix (i.e. $\mathbf{A}^{+}$ is the pseudo-inverse of matrix $\mathbf{A}$).

A – Graph Mining: An Optimization Pointview. Given a graph with adjacency matrix $\mathbf{A}$, a graph mining algorithm learns the mining results by optimizing a loss function $l(\mathbf{A}, \mathbf{Y}, \theta)$, where $\mathbf{Y} = \text{argmin}_{\mathbf{Y}}\, l(\mathbf{A}, \mathbf{Y}, \theta)$ is the model output (i.e., the mining results) and θ is the set of all parameters that corresponds to a specific mining task. We use three classic graph mining algorithms, including ranking, clustering and embedding, summarized in Table 2.

The first classic algorithm we apply is PageRank [21]. PageRank is a widely used random walk-based ranking algorithm. It outputs the ranking vector $\mathbf{r}$ by minimizing a smoothing term $(\mathbf{r}'(\mathbf{I} - \mathbf{A})\mathbf{r})$

Table 2: Examples of graph mining algorithms.

Mining Tasks	Loss Function l	Mining Results Y	Parameters θ
Ranking (PageRank [21])	$\min_{\mathbf{r}} \quad c\mathbf{r}'(\mathbf{I}-\mathbf{A})\mathbf{r} + (1-c)\|\mathbf{r}-\mathbf{e}\|_2^2$	PageRank vector $\mathbf{r}$	damping factor c teleportation vector $\mathbf{e}$
Clustering (spectral clustering [20])	$\min_{\mathbf{U}} \quad \mathrm{Tr}(\mathbf{U}'\mathbf{L}\mathbf{U}) \quad \text{s.t.} \quad \mathbf{U}'\mathbf{U}=\mathbf{I}$	matrix $\mathbf{U}$	number of clusters k
Graph Embedding (LINE (1st) [25])	$\max_{\mathbf{X}} \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{A}[i,j](\log g(\mathbf{X}[j,:]\mathbf{X}[i,:]') + b\mathbb{E}_{j'\sim P_n}[\log g(-\mathbf{X}[j',:]\mathbf{X}[i,:]')])$	embedding matrix $\mathbf{X}$	embedding dimension d number of negative samples b

and a query-specific term ($\|\mathbf{r}-\mathbf{e}\|_2^2$), with c being a regularization parameter to balance the two terms [2]. The second algorithm is spectral clustering [20], which finds the soft cluster membership matrix $\mathbf{U}$ of nodes in a graph by analyzing the spectrum of its graph Laplacian. It has been shown that spectral clustering is equivalent to finding the eigenvectors that are associated with the k smallest eigenvalues. The final algorithm we use is LINE [25]. Given a graph with n nodes, LINE learns the $n \times d$ embedding matrix $\mathbf{U}$, where each node is mapped into a d-dimensional vector that embeds its structural property.

B – Individual Fairness for Graph Mining. We aim to answer three questions regarding the individual fairness for graph mining (InFoRM). Based on that, we formally define these three problems, and then present our solutions in the subsequent sections.

For Q1 (*InFoRM Measures*), given a graph mining model and an arbitrary similarity measure, we want to (1) determine if the mining results are fair, and if not, (2) quantitatively measure the overall bias. Formally, we define the problem of InFoRM Measures as follows.

Problem 1. *InFoRM Measures.*

Input: (1) a non-negative symmetric node-node similarity matrix $\mathbf{S}$, (2) a graph mining algorithm $\mathbf{Y} = \mathrm{argmin}_{\mathbf{Y}}\, l(\mathbf{A},\mathbf{Y},\theta)$ from Table 2, and (3) a fairness tolerance parameter ϵ;
Output: (1) a binary decision regarding whether or not the mining results are fair, and (2) a bias measure $\texttt{Bias}(\mathbf{Y},\mathbf{S})$ which measures the overall individual bias of the mining results $\mathbf{Y}$ with respect to the similarity matrix $\mathbf{S}$.

For Q2 (*InFoRM Algorithms*), we aim to develop generic, effective and efficient algorithms to mitigate the bias of the mining results $\texttt{Bias}(\mathbf{Y},\mathbf{S})$, by adjusting either the input graph, or the mining model, or the mining results. Formally, we define the problem of InFoRM Algorithms as follows.

Problem 2. *InFoRM Algorithms.*

Input: (1) a non-negative symmetric node-node similarity matrix $\mathbf{S}$, (2) a graph mining algorithm $\mathbf{Y} = \mathrm{argmin}_{\mathbf{Y}}\, l(\mathbf{A},\mathbf{Y},\theta)$ from Table 2, and (3) a bias measure $\texttt{Bias}(\mathbf{Y},\mathbf{S})$ from Problem 1;
Output: a revised model output $\mathbf{Y}^*$ which minimizes (1) the loss function $l(\mathbf{A},\mathbf{Y},\theta)$ and (2) the individual bias $\texttt{Bias}(\mathbf{Y},\mathbf{S})$.

For Q3 (*InFoRM Cost*), we want to quantitatively characterize to what extent the revised graph mining results ($\mathbf{Y}^*$) from Problem 2 will deviate from the graph mining results ($\bar{\mathbf{Y}}$) without the fairness constraint. For clarity, we refer to (1) the original results ($\bar{\mathbf{Y}}$) without the fairness constraint as *vanilla mining results*, and (2) the revised results ($\mathbf{Y}^*$) as *debiased mining results*. Formally, we seek to develop an upper bound of such cost, which is defined as follows

Problem 3. *InFoRM Cost.*

Input: (1) the vanilla mining results $\bar{\mathbf{Y}}$ without consideration of individual fairness, i.e., $\bar{\mathbf{Y}} = \mathrm{argmin}_{\mathbf{Y}}\, l(\mathbf{A},\mathbf{Y},\theta)$ from Table 2, and (2) the debiased mining results $\mathbf{Y}^*$ from Problem 2;
Output: an upper bound of $\|\mathbf{Y}^* - \bar{\mathbf{Y}}\|_F$.

3 PROBLEM 1: INFORM MEASURES

In this section, we address Problem 1, and aim to measure the individual fairness and bias for graph mining. That is, given the graph mining results $\mathbf{Y}$ and a node similarity measure $\mathbf{S}$, we want to determine if the mining results are fair, and if not, we want to quantitatively measure the overall bias.

We follow the generic design principle underlying individual fairness that *any two individuals who are similar should receive similar algorithmic outcome* [5]. In our setting, this implies that if two nodes (i and j) are similar (i.e., $\mathbf{S}[i,j]$ is high), their mining results ($\mathbf{Y}[i,:]$ and $\mathbf{Y}[j,:]$) should be similar as well. We start with the following criteria: the mining results $\mathbf{Y}$ are fair w.r.t. to the node similarity measure $\mathbf{S}$ if the following condition holds.

$$\|\mathbf{Y}[i,:] - \mathbf{Y}[j,:]\|_F^2 \le \frac{\epsilon}{\mathbf{S}[i,j]} \quad \forall i,j = 1,\ldots,n \tag{1}$$

where $\epsilon > 0$ is a constant for tolerance.

According to Eq (1), the difference between the mining results of a pair of nodes i and j is upper bounded by a scalar $\frac{\epsilon}{\mathbf{S}[i,j]}$. The upper bound itself is dependent on the similarity between them $\mathbf{S}[i,j]$. That is, the more similar the node i and the node j, the smaller the upper-bound, and therefore the smaller the difference between $\mathbf{Y}[i,:]$ and $\mathbf{Y}[j,:]$ is likely to be (i.e., the more similar the mining results between them). An illustrative example is shown in Figure 1 in Appendix. Therefore, Eq (1) naturally reflects the aforementioned design principle of individual fairness.

The criteria in Eq (1) requires the inequality constraint to be held for *every* pair of nodes i and j as long as their similarity $\mathbf{S}[i,j]$ is non-zero[3]. Such a constraint might be too restrictive to be fulfilled. Therefore, we further seek for a relaxed criteria to tell if the mining results are fair. Based on Eq. (1), we have

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\|\mathbf{Y}[i,:] - \mathbf{Y}[j,:]\|_F^2\mathbf{S}[i,j] = 2\mathrm{Tr}(\mathbf{Y}'\mathbf{L}_{\mathbf{S}}\mathbf{Y}) \le m\epsilon = \delta \tag{2}$$

where $\mathbf{L}_{\mathbf{S}}$ is the Laplacian matrix of similarity $\mathbf{S}$, m is the number of non-zero elements in $\mathbf{S}$, and $\mathrm{Tr}(\mathbf{Y}'\mathbf{L}_{\mathbf{S}}\mathbf{Y})$ measures the difference of the mining results between all pairs of nodes. Based on Eq. (2), we formally propose the following to (1) determine if the mining results are fair and (2) measure the overall bias.

[2] $0 < c < 1$ is often called the damping factor in PageRank and its variants.

[3] If $\mathbf{S}[i,j] = 0$, the right hand side of Eq. (1) will be infinity, which simply means that it becomes a dummy constraint for nodes i and j.

Definition 1. *(Individual Fairness and Bias). Given a graph mining results* $\mathbf{Y}$ *of size* $n \times r$*, an* $n \times n$ *non-negative, a symmetric node similarity matrix* $\mathbf{S}$ *and a constant* δ *for fairness tolerance,* $\mathbf{Y}$ *is individually fair w.r.t. the similarity measure* $\mathbf{S}$ *if it satisfies*

$$Tr(\mathbf{Y}'\mathbf{L}_\mathbf{S}\mathbf{Y}) \le \delta/2$$

where $\mathbf{L}_\mathbf{S}$ *is the Laplacian matrix of similarity matrix* $\mathbf{S}$. $\mathtt{Bias}(\mathbf{Y}, \mathbf{S}) = Tr(\mathbf{Y}'\mathbf{L}_\mathbf{S}\mathbf{Y})$ *is the overall bias regarding the individual fairness.*

For traditional fair machine learning on spatial data or text data, the notation of individual fairness often has a root in the Lipschitz constant [5]. Here, we show that Eq. (1) can be interpreted from the perspective of the Lipschitz constant.

Definition 2. *(*(D_1, D_2)*-Lipschitz property) [5]. Given a function* f*, denote* $f(x)$ *as the outcome of instance* x*. We say function* f *satisfies* (D_1, D_2)*-Lipschitz property if*

$$D_1(f(x), f(y)) \le LD_2(x, y) \quad \forall\ (x, y),$$

where L *is the Lipschitz constant,* $D_1()$ *and* $D_2()$ *are two functions used to measure the dissimilarty of outcomes and instances, respectively.*

Let $f(i) = \mathbf{Y}[i,:]$, $f(j) = \mathbf{Y}[j,:]$ and define $D_1(f(i), f(j)) = \|f(i) - f(j)\|_2$ and $D_2(i,j) = \frac{1}{\mathbf{S}(i,j)}$. It can be shown that the proposed individual fairness definition in Eq. (1) naturally satisfies (D_1, D_2)-Lipschitz property with ϵ being the Lipschitz constant.

4 PROBLEM 2: INFORM ALGORITHMS

Generally speaking, a graph mining method consists of three major components, including (1) the input graph, (2) the mining model and (3) the mining results. Each of these three components can introduce and/or amplify the proposed bias measure. In this section, we present three complementary solutions to mitigate such bias (i.e., $\mathtt{Bias}(\mathbf{Y}, \mathbf{S})$) from the perspective of each component, namely (1) *debiasing the input graph*, (2) *debiasing the mining model* and (3) *debiasing the mining results.* For each of them, we formulate the bias mitigation problem (i.e., Problem 2) as an optimization problem, propose an effective and efficient algorithm to solve the optimization problem, and instantiate it with the three graph mining tasks in Table 2. Finally, we compare the three proposed solutions.

4.1 Debiasing the Input Graph

Given a graph with adjacency matrix $\mathbf{A}$, if the graph itself is contaminated with bias, it is likely that the bias will be transmitted to, or even amplified in, the mining results $\mathbf{Y}$ if such a graph $\mathbf{A}$ is used to train a graph mining model $l(\mathbf{A}, \mathbf{Y}, \theta)$. The intuition and rationality of debiasing the input graph is as follows. If we have the access to modify the graph and have knowledge about the mining model itself, we aim to learn a new topology of the graph $\tilde{\mathbf{A}}$ so that the bias of mining results based on the modified graph $\tilde{\mathbf{A}}$ is minimized. We also want to make sure that the modified graph $\tilde{\mathbf{A}}$ preserves as much information of $\mathbf{A}$ as possible. Mathematically, we formulate debiasing the input graph method as the following optimization problem.

$$\min_{\tilde{\mathbf{A}}} \quad \|\tilde{\mathbf{A}} - \mathbf{A}\|_F^2 + \alpha \mathrm{Tr}(\mathbf{Y}'\mathbf{L}_\mathbf{S}\mathbf{Y}) \quad \text{s.t.} \quad \mathbf{Y} = \operatorname*{argmin}_{\mathbf{Y}} l(\tilde{\mathbf{A}}, \mathbf{Y}, \theta) \tag{3}$$

where $\alpha > 0$ is the regularization parameter and $\mathbf{L}_\mathbf{S}$ is the Laplacian matrix of the similarity matrix $\mathbf{S}$.

Eq. (3) is hard to solve due to its bi-level optimization nature. A generic strategy to solve such a bi-level optimization problem is proposed by Mei et al. [19], which reduces the bi-level optimization problem by replacing the lower-level optimization problem with its KKT conditions. By applying this generic strategy to our setting, where the low-level optimization problem is $\mathbf{Y}^* = \operatorname{argmin}_\mathbf{Y} l(\tilde{\mathbf{A}}, \mathbf{Y}, \theta)$, we have

$$\min_{\tilde{\mathbf{A}}} \quad \|\tilde{\mathbf{A}} - \mathbf{A}\|_F^2 + \alpha \mathrm{Tr}(\mathbf{Y}'\mathbf{L}_\mathbf{S}\mathbf{Y}) \quad \text{s.t.} \quad \partial_\mathbf{Y} l(\tilde{\mathbf{A}}, \mathbf{Y}, \theta) = 0 \tag{4}$$

We propose Algorithm 1 to solve Eq. (4). At each iteration of Algorithm 1, we first find the mining results $\tilde{\mathbf{Y}}$ based on the current modified graph $\tilde{\mathbf{A}}$ (Step 3), and then we use the current mining results $\tilde{\mathbf{Y}}$ to further modify graph $\tilde{\mathbf{A}}$ (Steps 4-6). Once we obtain the modified graph $\tilde{\mathbf{A}}$, we use it to generate the debiased mining results $\mathbf{Y}^*$ (Step 7). In order to update $\tilde{\mathbf{A}}$, we apply the gradient descent method to the objective function $J = \|\tilde{\mathbf{A}} - \mathbf{A}\|_F^2 + \alpha \mathrm{Tr}(\tilde{\mathbf{Y}}'\mathbf{L}_\mathbf{S}\tilde{\mathbf{Y}})$. To this end, we compute the partial derivative of J w.r.t. $\tilde{\mathbf{A}}$ as

$$\begin{aligned} \frac{\partial J}{\partial \tilde{\mathbf{A}}} &= 2(\tilde{\mathbf{A}} - \mathbf{A}) + \alpha \frac{\partial \mathrm{Tr}(\tilde{\mathbf{Y}}'\mathbf{L}_\mathbf{S}\tilde{\mathbf{Y}})}{\partial \tilde{\mathbf{A}}} \\ &= 2(\tilde{\mathbf{A}} - \mathbf{A}) + \alpha \left[\mathrm{Tr}\left(\frac{\partial \mathrm{Tr}(\tilde{\mathbf{Y}}'\mathbf{L}_\mathbf{S}\tilde{\mathbf{Y}})}{\partial \tilde{\mathbf{Y}}'} \frac{\partial \tilde{\mathbf{Y}}}{\partial \tilde{\mathbf{A}}[i,j]}\right)\right] \\ &= 2(\tilde{\mathbf{A}} - \mathbf{A}) + \alpha \left[\mathrm{Tr}\left(2\tilde{\mathbf{Y}}'\mathbf{L}_\mathbf{S} \frac{\partial \tilde{\mathbf{Y}}}{\partial \tilde{\mathbf{A}}[i,j]}\right)\right] \end{aligned} \tag{5}$$

where $\left[\mathrm{Tr}\left(2\tilde{\mathbf{Y}}'\mathbf{L}_\mathbf{S} \frac{\partial \tilde{\mathbf{Y}}}{\partial \tilde{\mathbf{A}}[i,j]}\right)\right]$ is a matrix with its element at i^{th} row and j^{th} column as $\mathrm{Tr}\left(2\tilde{\mathbf{Y}}'\mathbf{L}_\mathbf{S} \frac{\partial \tilde{\mathbf{Y}}'}{\partial \tilde{\mathbf{A}}[i,j]}\right)$ for any $1 \le i \le n$, $1 \le j \le j$. The corresponding gradient can be computed as $\frac{dJ}{d\tilde{\mathbf{A}}} = \frac{\partial J}{\partial \tilde{\mathbf{A}}} + \left(\frac{\partial J}{\partial \tilde{\mathbf{A}}}\right)' - diag\left(\frac{\partial J}{\partial \tilde{\mathbf{A}}}\right)$ if $\tilde{\mathbf{A}}$ is an undirected graph; otherwise, we have its gradient as $\frac{dJ}{d\tilde{\mathbf{A}}} = \frac{\partial J}{\partial \tilde{\mathbf{A}}}$.

Algorithm 1: Debiasing the Input Graph

```
Input  : Adjacency matrix A, similarity matrix S, a mining
         algorithm $l(\mathbf{A}, \mathbf{Y}, \theta)$, regularization parameter $\alpha$,
         learning rate $\eta$;
Output : modified topology $\tilde{\mathbf{A}}$ and debiasd mining results $\mathbf{Y}^*$.
1 initialize $\tilde{\mathbf{A}} = \mathbf{A}$;
2 while not converge do
3     find $\tilde{\mathbf{Y}} = \operatorname{argmin}_\mathbf{Y} l(\tilde{\mathbf{A}}, \mathbf{Y}, \theta)$;
4     calculate partial derivative $\frac{\partial J}{\partial \tilde{\mathbf{A}}}$ by Eq. (5);
5     calculate derivative $\frac{dJ}{d\tilde{\mathbf{A}}}$ based on partial derivative $\frac{\partial J}{\partial \tilde{\mathbf{A}}}$;
6     update $\tilde{\mathbf{A}} = \tilde{\mathbf{A}} - \eta \frac{dJ}{d\tilde{\mathbf{A}}}$;
7 return $\tilde{\mathbf{A}}$ and $\mathbf{Y}^* = \operatorname{argmin}_\mathbf{Y} l(\tilde{\mathbf{A}}, \mathbf{Y}, \theta)$;
```

A key step in Eq. (5) is to calculate $\mathbf{H} = \left[\mathrm{Tr}\left(2\tilde{\mathbf{Y}}'\mathbf{L}_\mathbf{S} \frac{\partial \tilde{\mathbf{Y}}}{\partial \tilde{\mathbf{A}}[i,j]}\right)\right]$. Therefore, Algorithm 1 can be applied to a variety of graph mining tasks as long as $\frac{\partial \tilde{\mathbf{Y}}}{\partial \tilde{\mathbf{A}}[i,j]}$ exists. We summarize how to calculate $\mathbf{H}$ in Table 3 for three graph mining tasks. Due to the space limitation, we only presentation details for LINE. Detailed derivations of $\mathbf{H}$ for PageRank and spectral clustering can be found in Appendix.

Algorithm 1 Instantiation with LINE. Denote the $n \times d$ embedding matrix learned by LINE (1st) as $\mathbf{X}$. We apply the chain rule

Table 3: Algorithm 1 instantiations.

Mining Tasks	Mining Results Y	Partial Derivatives H	Remarks
PageRank	$\mathbf{Y} = \mathbf{r} = (1-c)\mathbf{Q}\mathbf{e}$	$\mathbf{H} = 2c\mathbf{Q}'\mathbf{L}_\mathbf{S}\mathbf{r}\mathbf{r}'$	$\mathbf{Q} = (\mathbf{I} - c\tilde{\mathbf{A}})^{-1}$
Spectral clustering	$\mathbf{Y} = \mathbf{U}$ = eigenvectors with smallest eigenvalues	$\mathbf{H} = 2\sum_{i=1}^{k}\left(diag(\mathbf{M}_i'\mathbf{L}_\mathbf{S}\mathbf{u}_i\mathbf{u}_i')\mathbf{1}_{n\times n} - \mathbf{M}_i'\mathbf{L}_\mathbf{S}\mathbf{u}_i\mathbf{u}_i'\right)$	$\lambda_i = i^{\text{th}}$ eigenvalue of $\mathbf{L}_{\tilde{\mathbf{A}}}$ $\mathbf{u}_i$ = eigenvector of $\mathbf{L}_{\tilde{\mathbf{A}}}$ corresponds to λ_i $\mathbf{M}_i = (\lambda_i\mathbf{I} - \mathbf{L}_{\tilde{\mathbf{A}}})^+$
LINE (1st)	$\mathbf{Y}\mathbf{Y}' = \mathbf{Z}$ (see Eq. (6))	$\mathbf{H} = 2f(\tilde{\mathbf{A}} + \tilde{\mathbf{A}}') \circ \mathbf{L}_\mathbf{S} - 2diag(\mathbf{B}\mathbf{L}_\mathbf{S})\mathbf{1}_{n\times n}$	$f()$ calculates Hadamard inverse $\circ$ calculates Hadamard product $\mathbf{B}$ refers to Eq. (9)

and rewrite Eq. (5) as

$$\frac{\partial J}{\partial \tilde{\mathbf{A}}} = 2(\tilde{\mathbf{A}} - \mathbf{A}) + 2\alpha\frac{\partial \mathrm{Tr}(\mathbf{X}'\mathbf{L}_\mathbf{S}\mathbf{X})}{\partial \tilde{\mathbf{A}}} = 2(\tilde{\mathbf{A}} - \mathbf{A}) + 2\alpha\frac{\partial \mathrm{Tr}(\mathbf{L}_\mathbf{S}\mathbf{X}\mathbf{X}')}{\partial \tilde{\mathbf{A}}}$$
$$= 2(\tilde{\mathbf{A}} - \mathbf{A}) + 2\alpha\left[\mathrm{Tr}\left(\mathbf{L}_\mathbf{S}'\frac{\partial \mathbf{X}\mathbf{X}'}{\partial \tilde{\mathbf{A}}[i,j]}\right)\right]$$

Let $\mathbf{Z} = \mathbf{X}\mathbf{X}'$. We use the following method[4] to compute $\frac{\partial \mathbf{Z}}{\partial \tilde{\mathbf{A}}[i,j]}$. First, we have

$$\mathbf{Z}[s,t] = \log\left(\frac{T(\tilde{\mathbf{A}}[s,t] + \tilde{\mathbf{A}}[t,s])}{d_s d_t^{3/4} + d_s^{3/4} d_t}\right) - \log b \tag{6}$$

where d_i is the out degree of node i, $T = \sum_{i=1}^{n} d_i^{3/4}$. Then we have the partial derivative

$$\frac{\partial \mathbf{Z}[s,t]}{\partial \tilde{\mathbf{A}}[i,j]} = \frac{3}{4Td_i^{1/4}} + \frac{1}{\tilde{\mathbf{A}}[s,t] + \tilde{\mathbf{A}}[t,s]}(\mathbb{1}[i=s, j=t] + \mathbb{1}[i=t, j=s])$$
$$- \frac{4d_s^{1/4} + 3d_t^{1/4}}{4(d_s d_t^{1/4} + d_s^{5/4})}\mathbb{1}[i=s] - \frac{3d_s^{1/4} + 4d_t^{1/4}}{4(d_s^{1/4}d_t + d_t^{5/4})}\mathbb{1}[i=t] \tag{7}$$

where $\mathbb{1}$ is the indicator function.

With Eq. (7), we get the following matrix form of derivatives

$$\left[\mathrm{Tr}\left(\mathbf{L}_\mathbf{S}'\frac{\partial \mathbf{Z}}{\partial \tilde{\mathbf{A}}[i,j]}\right)\right] = 2f(\tilde{\mathbf{A}} + \tilde{\mathbf{A}}') \circ \mathbf{L}_\mathbf{S} - 2diag(\mathbf{B}\mathbf{L}_\mathbf{S})\mathbf{1}_{n\times n} \tag{8}$$

where $f(\tilde{\mathbf{A}})$ calculates the Hadamard inverse matrix $\tilde{\mathbf{A}}$, $\circ$ is the Hadamard product operator and $\mathbf{B}$ has the following form

$$\mathbf{B} = \frac{3}{4}f\left(\mathbf{d}^{5/4}(\mathbf{d}^{-1/4})' + \mathbf{d}\mathbf{1}_{1\times n}\right) + f\left(\mathbf{d}^{3/4}(\mathbf{d}^{1/4})' + \mathbf{d}\mathbf{1}_{1\times n}\right) \tag{9}$$

with $\mathbf{d}^x$ being a column vector of $\mathbf{d}^x[i] = d_i^x$.

Lemma 1. *(Time and space complexities of Algorithm 1 for LINE) It takes $O(min\{m_1, m_2\} + m_2)$ time and $O(min\{m_1, m_2\} + n)$ space to calculate the partial derivatives* $\mathbf{H}$ *where n is the number of nodes and m_1 and m_2 are the number of edges in* $\mathbf{A}$ *and* $\mathbf{S}$*, respectively.*

Proof. See Appendix. □

4.2 Debiasing the Mining Model

The intuition and rationality of debiasing the mining model is as follows. If we directly incorporate the proposed bias measure (Bias(Y, S)) as a regularization term in the loss function of the given mining model (i.e., those listed in Table 2), the generated mining results are likely to have a small bias. Mathematically, we formulate it as

$$\mathbf{Y}^* = \underset{\mathbf{Y}}{\mathrm{argmin}} \quad J = l(\mathbf{A}, \mathbf{Y}, \theta) + \alpha\mathrm{Tr}(\mathbf{Y}'\mathbf{L}_\mathbf{S}\mathbf{Y}) \tag{10}$$

where $\alpha > 0$ is the parameter for regularization.

[4]This method was first developed in [23] in order to establish the relationship between LINE (2nd) and matrix factorization.

To solve Eq. (10), we apply (stochastic) gradient descent/ascent-based methods. Since $\mathbf{Y}$ is, in general, not symmetric, its derivative is $\frac{dJ}{d\mathbf{Y}} = \frac{\partial J}{\partial \mathbf{Y}}$. We have

$$\frac{dJ}{d\mathbf{Y}} = \frac{\partial J}{\partial \mathbf{Y}} = \frac{\partial l(\mathbf{A}, \mathbf{Y}, \theta)}{\partial \mathbf{Y}} + \alpha\frac{\partial \mathrm{Tr}(\mathbf{Y}'\mathbf{L}_\mathbf{S}\mathbf{Y})}{\partial \mathbf{Y}}$$
$$= \frac{\partial l(\mathbf{A}, \mathbf{Y}, \theta)}{\partial \mathbf{Y}} + \alpha(\mathbf{L}_\mathbf{S} + \mathbf{L}_\mathbf{S}')\mathbf{Y} = \frac{\partial l(\mathbf{A}, \mathbf{Y}, \theta)}{\partial \mathbf{Y}} + 2\alpha\mathbf{L}_\mathbf{S}\mathbf{Y} \tag{11}$$

The last equality holds because S is a symmetric matrix and so is its Laplacian matrix $\mathbf{L}_\mathbf{S}$. We can see that, compared with the original graph mining model without the fairness consideration, the extra time to calculate $\mathbf{L}_\mathbf{S}\mathbf{Y}$ is just linear w.r.t. the number of similarity links in S. Based on that, we propose a generic algorithmic framework (i.e. Algorithm 2) to debias the mining model. The key of Algorithm 2 is to solve Eq. (10) (Step 2). This can be done either by (stochastic) gradient descent/ascent method based on Eq. (11), or by a specific algorithm designed for the given graph mining model. For the latter, we give three examples for the mining models in Table 2.

```
Algorithm 2: Debiasing the Mining Model
  Input  : Adjacency matrix A, similarity matrix S, a mining
           model $l(\mathbf{A}, \mathbf{Y}, \theta)$, regularization parameter $\alpha$,
           learning rate $\eta$;
  Output : Debiased mining results $\mathbf{Y}^*$.
1 solve Eq. (10);
2 return $\mathbf{Y}^*$;
```

Algorithm 2 Instantiation with PageRank. Instantiating Eq. (10) with PageRank, we have that $\mathbf{r}^* = \underset{\mathbf{r}}{\mathrm{argmin}}\ J = c\mathbf{r}'(\mathbf{I} - \mathbf{A})\mathbf{r} + (1-c)\|\mathbf{r} - \mathbf{e}\|_F^2 + \alpha\mathbf{r}'\mathbf{L}_\mathbf{S}\mathbf{r}$. We can show that J is a quadratic convex function w.r.t $\mathbf{r}$ as long as the regularization parameter α is positive. Therefore, its optima has a zero derivative $\frac{\partial J}{\partial \mathbf{r}} = 0$. Then we have

$$\frac{\partial J}{\partial \mathbf{r}} = 2\mathbf{r} - 2c\mathbf{A}\mathbf{r} + 2\alpha\mathbf{L}_\mathbf{S}\mathbf{r} - 2(1-c)\mathbf{e} = 0$$
$$\Rightarrow \mathbf{r}^* = c(\mathbf{A} - \frac{\alpha}{c}\mathbf{L}_\mathbf{S})\mathbf{r}^* + (1-c)\mathbf{e} \tag{12}$$

which is equivalent to PageRank on a new transition matrix $\mathbf{A} - \frac{\alpha}{c}\mathbf{L}_\mathbf{S}$. Furthermore, if the similarity matrix S is symmetrically normalized (i.e., $\mathbf{L}_\mathbf{S} = \mathbf{I} - \mathbf{S}$), we have $\mathbf{r}^* = (\frac{c}{1+\alpha}\mathbf{A} + \frac{\alpha}{1+\alpha}\mathbf{S})\mathbf{r}^* + \frac{1-c}{1+\alpha}\mathbf{e}$.

Algorithm 2 Instantiation with Spectral Clustering. Instantiating Eq. (10) with spectral clustering, we have that $\mathbf{U}^* = \underset{\mathbf{U}}{\mathrm{argmin}}\ J = \mathrm{Tr}(\mathbf{U}'\mathbf{L}_\mathbf{A}\mathbf{U}) + \alpha\mathrm{Tr}(\mathbf{U}'\mathbf{L}_\mathbf{S}\mathbf{U}) = \mathrm{Tr}(\mathbf{U}'\mathbf{L}_{\mathbf{A}+\alpha\mathbf{S}}\mathbf{U})$, which is very similar to the loss function of the original spectral clustering without the fairness consideration in Table 2, and both loss functions require $\mathbf{U}$ to be orthonormal. The only difference is that the debiased $\mathbf{U}^*$

is equivalent to eigenvectors of $\mathbf{L}_{\mathbf{A}+\alpha\mathbf{S}}$ with the k smallest eigenvalues instead of the original $\mathbf{L}_{\mathbf{A}}$. In other words, debiased spectral clustering $\mathbf{U}^*$ is essentially spectral clustering on a modified graph with an adjacency matrix $\mathbf{A} + \alpha\mathbf{S}$.

Algorithm 2 Instantiation with LINE. Instantiating Eq. (10) with LINE (1st), we have

$$\begin{aligned} \mathbf{X}^* = \underset{\mathbf{X}}{\text{argmax}} & \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{A}[i,j](\log g(\mathbf{X}[j,:]\mathbf{X}[i,:]') \\ & + b\mathbb{E}_{j'\sim P_n}[\log g(-\mathbf{X}[j',:]\mathbf{X}[i,:]')]) - \alpha \text{Tr}(\mathbf{X}'\mathbf{L_S}\mathbf{X}) \end{aligned} \tag{13}$$

where $g(x) = 1/(1+e^{-x})$ is the sigmoid function.

Due to the unique edge sampling strategy of LINE, we factorize the bias term (i.e., $\text{Tr}(\mathbf{X}'\mathbf{L_S}\mathbf{X})$) and consider it edge-wise. Specifically, for an edge (i,j), LINE (1st) aims to maximize the following objective function

$$\log g(\mathbf{x}_j\mathbf{x}_i') + b\mathbb{E}_{j'\sim P_n}[\log g(-\mathbf{x}_{j'}\mathbf{x}_i')] - \alpha\|\mathbf{x}_i - \mathbf{x}_j\|_2^2\mathbf{S}[i,j] \tag{14}$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are the node embeddings for node i and j (i.e. i^{th} row and j^{th} row in $\mathbf{X}$), respectively. It is worth pointing out that adding such a bias constraint does not increase the time complexity. To see this, we can show that the optimization for one edge in the original LINE takes $O(db)$ time, where b is the number of negative samples and d is the embedding dimension. By adding the bias constraint $\|\mathbf{x}_i - \mathbf{x}_j\|_2^2\mathbf{S}[i,j]$, it would introduce an additional $O(d)$ time per edge, which does not change the overall time complexity in big-O notation.

4.3 Debiasing the Mining Results.

If we do not have access to either the input graph or the graph mining model, we could mitigate the individual bias via a post-processing strategy on the mining results. We formulate this mitigation strategy as a regularized optimization problem below.

$$\mathbf{Y}^* = \underset{\mathbf{Y}}{\text{argmin}} \quad J = \|\mathbf{Y} - \bar{\mathbf{Y}}\|_F^2 + \alpha\text{Tr}(\mathbf{Y}'\mathbf{L_S}\mathbf{Y}) \tag{15}$$

where $\bar{\mathbf{Y}}$ is the vanilla mining results, i,e., the original model output without the consideration of individual fairness.

We can prove that J is a convex function since $\|\mathbf{Y} - \bar{\mathbf{Y}}\|_F^2$ and $\text{Tr}(\mathbf{Y}'\mathbf{L_S}\mathbf{Y})$ are both convex and α is a positive regularization hyperparameter. Thus, the optimal solution for Eq. (15) can be obtained by taking the derivative of J w.r.t. $\mathbf{Y}$ and setting it to zero.

$$\begin{aligned} \frac{\partial J}{\partial \mathbf{Y}} &= \frac{\partial\|\mathbf{Y} - \bar{\mathbf{Y}}\|_F^2}{\partial \mathbf{Y}} + \alpha\frac{\partial\text{Tr}(\mathbf{Y}'\mathbf{L_S}\mathbf{Y})}{\partial \mathbf{Y}} = 0 \\ &\Rightarrow 2\mathbf{Y} - 2\bar{\mathbf{Y}} + 2\alpha\mathbf{L_S}\mathbf{Y} = 0 \Rightarrow (\mathbf{I} + \alpha\mathbf{L_S})\mathbf{Y}^* = \bar{\mathbf{Y}} \end{aligned} \tag{16}$$

Eq. (16) indicates that debiasing the mining results is essentially solving a linear system w.r.t. the debiased mining results. Many linear system solvers can be utilized, e.g., Krylov subspace method, conjugate gradient method, etc. The proposed algorithm for debiasing mining results is summarized in Algorithm 3.

4.4 Analysis and Discussions

The three proposed algorithmic frameworks are mutually complementary with each other. For example, to debias the input graph (Algorithm 1), we modify the input graph and mining results in an iterative way. The potential benefit is that it eliminates or mitigates the bias from the 'origin' (i.e., the input graph), and thus the modified/debiased graph might also help mitigate the bias for other

Algorithm 3: Debiasing the Mining Results

```
Input  : Vanilla graph mining results $\bar{\mathbf{Y}}$, similarity matrix $\mathbf{S}$,
         regularization parameter $\alpha$;
Output : Debiased mining results $\mathbf{Y}^*$.
1 calculate $\mathbf{I} + \alpha\mathbf{L_S}$;
2 solve $(\mathbf{I} + \alpha\mathbf{L_S})\mathbf{Y}^* = \bar{\mathbf{Y}}$;
3 return $\mathbf{Y}^*$;
```

related graph mining models. In order to debias the mining model (Algorithm 2), we need the knowledge of the details of the mining model itself whereas the input graph remains unchanged. On the contrary, neither the knowledge of the input graph nor the mining model is required for debiasing the mining results (Algorithm 3).

Regarding the applicability of the proposed frameworks, we can always debias the input graph as long as the gradient $\frac{d\mathbf{Y}}{d\mathbf{A}[i,j]}$ in Eq. (5) exists. For debiasing the mining model method, it is applicable as long as a (stochastic) gradient descent solution for the vanilla graph mining algorithm (i.e., the one without the consideration of individual fairness) exists. This is because adding the additional bias term in Eq. (10) would only incur a linear term in computing the gradient. Besides, the convexity of the vanilla graph mining algorithm will not be affected since the bias term itself is convex. For debiasing the mining results method, it can be applied to *any* graph mining model whose mining results $\mathbf{Y}$ are in a matrix form, thanks to its model-agnostic closed-form solution.

The three proposed algorithmic frameworks differ in computational efficiency. First, the debiasing the input graph method is the most time and space-consuming since $\mathbf{H}$ is usually non-trivial to compute and could be a full matrix with $O(n^2)$ space cost. However, for some special mining models, including all three models in Table 2, we can handle this issue by exploring the low-rank structure of $\mathbf{H}$. For example, in PageRank and spectral clustering, $\mathbf{Q}'\mathbf{L_S}\mathbf{r}$ and $\mathbf{M}_i'\mathbf{L_S}\mathbf{u}_i$ are both column vectors of length n where $\mathbf{Q}'\mathbf{L_S}\mathbf{r}$ can be computed by power iterations and $\mathbf{M}_i$ can be efficiently calculated by singular value decomposition (SVD). In LINE, $diag(\mathbf{BL_S})\mathbf{1}_{n\times n}$ is equivalent to vectorize the diagonal of $\mathbf{BL_S}$ as a column vector and multiply with $\mathbf{1}_{1\times n}$. As Lemma 1 says, Algorithm 1 for LINE has a linear complexity. Second, for debiasing the mining model method, the additional time incurred in the gradient computation is linear w.r.t. the number of non-zero elements in $\mathbf{S}$ (m_2, the number of links in the similarity matrix $\mathbf{S}$), according to Eq. (11). Finally, for debiasing the mining results method, it always has a linear time complexity w.r.t. m_2 (the number of links in the similarity matrix $\mathbf{S}$), since we only need to solve a linear system in Eq. (16).

5 PROBLEM 3: INFORM COST

In this section, we address Problem 3 (i.e., InFoRM cost), aiming to characterize how the debiased graph mining results $\mathbf{Y}^*$ would deviate from the vanilla ones $\bar{\mathbf{Y}}$ without the fairness consideration. Among the three algorithms proposed in Section 4, debiasing the mining results method (Algorithm 3), being agnostic to both the input graph and the mining model, has the widest applicability. Therefore, we will mainly focus on characterizing the InFoRM cost of this method. The InFoRM cost of the other two proposed methods (i.e., debiasing the input graph and debiasing the mining model) is dependent on the specific graph and/or the specific mining model. In Appendix, we provide a case study that analyzes the InFoRM cost for PageRank with the debiasing mining model method (Algorithm 2).

For debiasing the mining results method, the solution of Eq. (16) is always optimal since Eq. (15) is a convex optimization problem. Based on this solution, we characterize the cost of debiasing the mining results in Lemma 2.

Lemma 2. *Given a graph of n nodes with the adjacency matrix* $\mathbf{A}$ *and a node-node similarity matrix* $\mathbf{S}$. *Let* $\bar{\mathbf{Y}}$ *be the vanilla mining results without considering the fairness and* $\mathbf{Y}^* = (\mathbf{I}+\alpha\mathbf{L_S})^{-1}\bar{\mathbf{Y}}$ *be the solution of Eq.* (15) *(i.e. the debiased mining results). If* $\|\mathbf{S}-\mathbf{A}\|_F = \delta$, *we have that*

$$\|\mathbf{Y}^* - \bar{\mathbf{Y}}\|_F \le 2\alpha\sqrt{n}(\delta + \sqrt{r(\mathbf{A})}\sigma_{max}(\mathbf{A}))\|\bar{\mathbf{Y}}\|_F$$

where $r(\mathbf{A})$ *is the rank of* $\mathbf{A}$ *and* $\sigma_{max}(\mathbf{A})$ *is the largest singular value of* $\mathbf{A}$.

Proof. Since $\mathbf{Y}^* = (\mathbf{I}+\alpha\mathbf{L_S})^{-1}\bar{\mathbf{Y}}$, by re-arranging terms, we have

$$\|\mathbf{Y}^* - \bar{\mathbf{Y}}\|_F = \alpha\|\mathbf{L_S}\mathbf{Y}^*\|_F \le \alpha\|\mathbf{L_S}\|_F\|(\mathbf{I}+\alpha\mathbf{L_S})^{-1}\|_F\|\bar{\mathbf{Y}}\|_F \quad (17)$$

The last inequality above holds due to the triangle inequality. Since $\|\mathbf{Y}\|_F$ is a constant, our goal is to find upper bounds of $\|\mathbf{L_S}\|_F$ and $\|(\mathbf{I}+\alpha\mathbf{L_S})^{-1}\|_F$ respectively.

First, we derive an upper bound of $\|(\mathbf{I}+\alpha\mathbf{L_S})^{-1}\|_F$. For any matrix $\mathbf{W}$, we have $\|\mathbf{W}\|_F = \sqrt{\sum_{i=1}^{r(\mathbf{W})}\sigma_i^2(\mathbf{W})} \le \sqrt{r(\mathbf{W})}\sigma_{\max}(\mathbf{W})$, where $\sigma_{\max}(\mathbf{W})$ is the largest singular value (i.e. the spectral norm) of matrix $\mathbf{W}$ and $r(\mathbf{W})$ is the rank of matrix $\mathbf{W}$ [7]. Applying it to $\|(\mathbf{I}+\alpha\mathbf{L_S})^{-1}\|_F$, we have the following inequality

$$\|(\mathbf{I}+\alpha\mathbf{L_S})^{-1}\|_F \le \sqrt{n}\sigma_{\max}((\mathbf{I}+\alpha\mathbf{L_S})^{-1}) = \frac{\sqrt{n}}{\sigma_{\min}(\mathbf{I}+\alpha\mathbf{L_S})} = \sqrt{n} \quad (18)$$

The above inequality holds due to the facts that (1) $(\mathbf{I}+\alpha\mathbf{L_S})^{-1}$ is full-rank $n \times n$ matrices; and (2) graph laplacian $\mathbf{L_S}$ is a symmetric singular matrix with smallest singular value being 0, which implies that $\sigma_{\min}(\mathbf{I}+\alpha\mathbf{L_S}) = 1$.

Next, we derive an upper bound of $\|\mathbf{L_S}\|_F$. Denote $d_i = \mathbf{L_S}(i,i)$. We have

$$\begin{aligned}\|\mathbf{L_S}\|_F^2 &= \sum_i (d_i^2 + \sum_{j:j\neq i}\mathbf{S}(i,j)^2) = \sum_i [(\sum_{j:j\neq i}\mathbf{S}(i,j))^2 + \sum_{j:j\neq i}\mathbf{S}(i,j)^2] \\ &\le \sum_i (2\sum_{j:j\neq i}\mathbf{S}(i,j)^2 + \sum_{k:k\neq i}\mathbf{S}(i,k)^2 + \sum_{l:l\neq i}\mathbf{S}(i,l)^2) \le 4\|\mathbf{S}\|_F^2\end{aligned}$$

Taking square root on both sides and applying triangle inequality, we have the upper bound of $\|\mathbf{L_S}\|_F$ as follows.

$$\|\mathbf{L_S}\|_F \le 2\|\mathbf{S}\|_F \le 2(\delta + \|\mathbf{A}\|_F) \le 2(\delta + \sqrt{r(\mathbf{A})}\sigma_{\max}(\mathbf{A})) \quad (19)$$

We complete the proof by combining Eqs. (17), (18) and (19). □

From Lemma 2, we can see that the cost of debiasing the mining results depends on a number of factors, including the size of input graph (i.e., the number of nodes n), the difference δ between $\mathbf{S}$ and $\mathbf{A}$, the rank of the adjacency matrix $r(\mathbf{A})$ and the largest singular value of the adjacency matrix $\sigma_{\max}(\mathbf{A})$. $r(\mathbf{A})$ of many real graphs could be small since they often have an (approximate) low-rank structure. Furthermore, if $\mathbf{A}$ is a symmetrically normalized matrix (i.e., $\mathbf{A} \leftarrow \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, where $\mathbf{D}$ is the degree matrix), its largest singular value is upper bounded by 1. These facts help make the overall upper bound in Lemma 2 to be relatively small.

6 EXPERIMENTAL EVALUATION

In this section, we perform experimental evaluations on our proposed methods. The experiments are designed to answer the following questions:

RQ1. How does the individual fairness constraint impact the graph mining performance?
RQ2. How effective are the proposed debiasing methods?
RQ3. How efficient are the proposed debiasing methods?

6.1 Experimental Settings

We utilize a diverse set of real-world datasets to test our algorithms, all of which are publicly available. Table 4 summarizes the statistics of these datasets. We provide details on experimental settings in Appendix, including dataset description, metrics, baselines, machine configuration and parameter settings. The source code will be released upon publication of the paper.

Table 4: Statistics of datasets.

Domain	Dataset	Nodes	Edges
Collaboration	AstroPh	18,772	198,110
	CondMat	23,133	93,497
Social	Facebook	22,470	171,002
	Twitch	7,126	35,324
Biology	PPI	3,890	76,584

6.2 Main Results

The evaluation of PageRank is shown in Table 5. From the table, we can see that all three proposed methods can effectively reduce the bias with small changes (i.e., Diff and KL columns in Table 5) to the vanilla mining results while being able to preserve the performance (i.e., Prec@50 and NDCG@50) of the vanilla algorithm without fairness consideration. Comparing among these three methods, the debiasing the input graph method takes the longest runtime. However, it is not as effective as the other two methods in terms of reducing the bias[5]. Thus, for spectral clustering and LINE, we mainly evaluate the efficacy on debiasing the mining model and debiasing the mining results. We also provide additional experimental results on debiasing the input graph for spectral clustering and LINE in the Appendix. Evaluation results for spectral clustering and LINE are shown in Tables 7 and 6, respectively. From these tables, we can see that our proposed methods can effectively reduce bias and preserve the performance of the vanilla graph mining algorithm (i.e., Orig. ROC vs. Fair ROC, Orig. F1 vs. Fair F1 for LINE, and NMI for spectral clustering). Interestingly, as shown in Table 6, adding the fairness constraint on LINE sometimes actually improves the link prediction performance (e.g., on Facebook, Twitch and PPI datasets).

7 RELATED WORK

A – Fairness in graph mining has begun to attract more and more research attention in recent years. However, this research has almost exclusively focused on group-based fairness notation. For recommendation, Kamishima et al. [11] were among the first to propose regularization-based collaborative filtering approaches to minimize the average ratings between the protected group and the unprotected group. These methods aim to ensure the statistical independence of predicted ratings from a protected attribute. In

[5]Algorithm 1 for PageRank still enjoys a linear complexity in big-O notation by exploring the low-rank structure of **H** matrix. See details in Appendix.

Table 5: Effectiveness results for PageRank. Lower is better in gray columns. Higher is better in the others.

Debiasing the Input Graph

Datasets	Jaccard Index						Cosine Similarity					
	Diff	KL	Prec@50	NDCG@50	Reduce	Time	Diff	KL	Prec@50	NDCG@50	Reduce	Time
AstroPh	0.059	4.61×10^{-4}	0.840	0.887	16.3%	3632	0.117	1.99×10^{-3}	0.680	0.738	31.9%	3844
CondMat	0.008	1.06×10^{-5}	0.980	0.986	2.16%	1817	0.031	1.57×10^{-4}	0.940	0.957	9.37%	1922
Facebook	0.031	1.83×10^{-4}	0.920	0.943	7.01%	3442	0.072	9.38×10^{-4}	0.760	0.827	16.6%	3623
Twitch	0.109	5.37×10^{-4}	1.000	1.000	24.7%	564.9	0.299	5.41×10^{-3}	0.860	0.899	62.9%	649.3
PPI	0.185	1.90×10^{-3}	0.920	0.944	43.4%	584.4	0.328	8.07×10^{-3}	0.780	0.838	68.7%	636.8

Debiasing the Mining Model

Datasets	Jaccard Index						Cosine Similarity					
	Diff	KL	Prec@50	NDCG@50	Reduce	Time	Diff	KL	Prec@50	NDCG@50	Reduce	Time
AstroPh	0.133	3.28×10^{-3}	0.820	0.871	51.0%	23.08	0.143	4.16×10^{-3}	0.880	0.912	50.4%	26.92
CondMat	0.117	2.43×10^{-3}	0.880	0.915	51.6%	12.02	0.149	4.01×10^{-3}	0.860	0.901	54.6%	12.83
Facebook	0.149	3.33×10^{-3}	0.840	0.884	47.7%	32.41	0.179	4.65×10^{-3}	0.840	0.883	53.3%	33.31
Twitch	0.182	4.97×10^{-3}	0.940	0.958	62.0%	16.18	0.315	1.05×10^{-2}	0.940	0.957	73.9%	12.73
PPI	0.211	4.78×10^{-3}	0.920	0.942	50.8%	10.76	0.280	9.56×10^{-3}	0.900	0.928	67.5%	10.50

Debiasing the Mining Results

Datasets	Jaccard Index						Cosine Similarity					
	Diff	KL	Prec@50	NDCG@50	Reduce	Time	Diff	KL	Prec@50	NDCG@50	Reduce	Time
AstroPh	0.055	1.40×10^{-3}	0.960	0.971	37.4%	0.038	0.094	4.46×10^{-3}	0.960	0.972	49.2%	0.054
CondMat	0.040	8.26×10^{-4}	0.940	0.959	34.4%	0.021	0.082	3.01×10^{-3}	0.780	0.839	48.9%	0.025
Facebook	0.047	1.12×10^{-3}	0.900	0.930	32.6%	0.048	0.086	3.87×10^{-3}	0.960	0.972	44.6%	0.062
Twitch	0.035	9.75×10^{-4}	0.980	0.986	33.9%	0.033	0.101	5.84×10^{-3}	0.940	0.958	44.6%	0.024
PPI	0.045	1.22×10^{-3}	0.940	0.958	27.0%	0.020	0.112	6.97×10^{-3}	0.940	0.958	45.0%	0.019

Table 6: Effectiveness results for LINE. Lower is better in gray columns. Higher is better in the others.

Debiasing the Mining Model

Datasets	Jaccard Index							Cosine Similarity						
	Diff	Orig. ROC	Fair ROC	Orig. F1	Fair F1	Reduce	Time	Diff	Orig. ROC	Fair ROC	Orig. F1	Fair F1	Reduce	Time
AstroPh	0.462	0.973	0.970	0.924	0.914	51.6%	934.7	0.913	0.973	0.966	0.924	0.906	49.5%	923.0
CondMat	0.302	0.963	0.962	0.922	0.920	44.1%	1130	0.439	0.963	0.961	0.922	0.918	41.6%	1133
Facebook	0.323	0.946	0.954	0.888	0.902	49.6%	1099	0.442	0.946	0.957	0.888	0.906	56.0%	1100
Twitch	0.099	0.687	0.690	0.625	0.625	0.64%	333.8	0.152	0.687	0.694	0.625	0.628	0.83%	340.3
PPI	0.238	0.682	0.715	0.618	0.642	5.85%	180.3	0.418	0.682	0.740	0.618	0.669	7.71%	181.6

Debiasing the Mining Results

Datasets	Jaccard Index							Cosine Similarity						
	Diff	Orig. ROC	Fair ROC	Orig. F1	Fair F1	Reduce	Time	Diff	Orig. ROC	Fair ROC	Orig. F1	Fair F1	Reduce	Time
AstroPh	0.365	0.973	0.962	0.924	0.898	83.3%	3.284	0.539	0.973	0.963	0.924	0.902	91.1%	6.461
CondMat	0.215	0.963	0.961	0.922	0.918	71.8%	1.464	0.322	0.963	0.960	0.922	0.915	78.4%	2.213
Facebook	0.304	0.946	0.950	0.888	0.890	88.5%	4.122	0.416	0.946	0.953	0.888	0.891	92.4%	7.394
Twitch	0.457	0.687	0.681	0.625	0.629	95.2%	2.320	0.603	0.687	0.658	0.625	0.616	97.6%	4.343
PPI	0.508	0.682	0.713	0.618	0.642	90.1%	1.031	0.722	0.682	0.634	0.618	0.589	97.0%	2.245

Table 7: Effectiveness results for spectral clustering. Lower is better in gray columns. Higher is better in the others.

Debiasing the Mining Model

Datasets	Jaccard Index				Cosine Similarity			
	Diff	NMI	Reduce	RT	Diff	NMI	Reduce	RT
AstroPh	0.885	0.948	10.2%	333.9	1.085	0.868	23.6%	323.4
CondMat	1.108	0.856	26.4%	383.7	1.186	0.742	35.9%	360.7
Facebook	0.972	0.816	31.9%	549.3	0.897	0.810	37.9%	545.0
Twitch	1.147	0.838	88.3%	26.50	1.145	0.875	87.4%	26.62
PPI	0.994	0.658	67.0%	6.047	0.897	0.667	75.2%	6.244

Debiasing the Mining Results

Datasets	Jaccard Index				Cosine Similarity			
	Diff	NMI	Reduce	Time	Diff	NMI	Reduce	Time
AstroPh	0.071	1.000	24.3%	10.22	0.123	0.984	39.5%	16.46
CondMat	0.071	1.000	34.5%	2.076	0.108	0.985	46.2%	3.196
Facebook	0.056	0.994	24.8%	8.425	0.102	0.994	35.9%	12.81
Twitch	0.150	1.000	90.9%	4.820	0.204	1.000	91.7%	6.513
PPI	0.242	0.811	77.5%	2.896	0.343	0.731	87.4%	4.288

addition, Yao et al. [31] proposed four new metrics to measure the difference in estimation error between predicted ratings and average ground-truth ratings across protected and unprotected groups in order to mitigate population imbalance and observation bias issues. For fair spectral clustering, several recent methods have been proposed. Kleindessner et al. [15] extended the notion of fairness that was originally proposed by Chierichetti et al.[4]. This extension requires that each cluster has a balanced number of elements from different demographic groups, which in turn is rooted in the classic notation of disparate impact [6]. For fairness-aware graph embedding, a generic idea behind the existing work is to ensure the learned node embedding to be independent or uncorrelated with the sensitive attributes. Bose et al. in [3] developed a compositional adversarial framework to ensure statistical parity such that the learned embedding is not biased w.r.t. different combinations of sensitive attributes. Rahman et al. in [24] proposed a fairness definition named equality of representation that builds upon statistical parity and extended node2vec [8] with such notion of fairness.

Different from [3, 24], Palowitch et al. [22] proposed a general GNN-based training paradigm that ensures the orthogonality of metadata (e.g. sensitive information) and the node embedding in order to resolve the metadata leakage issue, where the correlation between metadata and embedding cannot be explicitly modeled and removed by the mining algorithm itself.

B – Individual fairness in machine learning has been studied extensively in non-graph data, such as recidivism data [17], healthcare data [34] and text review data [2]. Compared with group-based fairness, which ensures statistical fairness across the entire population, individual fairness considers the individual merits and mandates fairness at the individual-level. Dwork et al. [5] first proposed the seminal work of individual fairness. Since then, individual fairness has been applied to several different task settings. For example, Zemel et al. [34] learned fair embedding with Euclidean distance as the distance metric. Kim et al. [14] assumed group-based fairness to be an average of individual fairness and learn fair classifier by using oracle to estimate the individual distance. Yona et al. [33] proposed a PAC-based relaxation of individual fairness and show that their proposed fairness can generalize from the training data to underlying population. Lahoti et al. [17] operationalized individual fairness by learning fair representation from the Euclidean space together with pairwise side-information, which assumes that the fair representation is a linear transformation of the original features via trace optimization. It is worth pointing out that trace optimization has been widely used in various data mining tasks, e.g., semi-supervised clustering [28], collaborative filtering with side-information [32], robust PCA [35], etc.

8 CONCLUSION

In this paper, we conduct a principled study of individual fairness on graph mining. We first present quantitative measures for the individual fairness and bias on graph mining. Based on that, we propose three mutually complementary algorithmic frameworks (i.e., debiasing the input graph, debiasing mining model and debiasing mining results) from the optimization perspective, and instantiate each of them with various graph mining tasks (i.e. PageRank, spectral clustering and LINE). Furthermore, we provide theoretical analysis to characterize the cost of individual fairness. Extensive empirical evaluations on a diverse set of real-world datasets demonstrate that our proposed algorithms are effective in reducing individual bias while largely maintaining the performance of various graph mining tasks.

ACKNOWLEDGEMENT

This work is supported by National Science Foundation under grant No. 1947135, 1715385, and 1947203, by the NSF Program on Fairness in AI in collaboration with Amazon under award No. 1939725, Department of Homeland Security under Grant Award Number 17STQAC00001-03-03. The content of the information in this document does not necessarily reflect the position or the policy of the Government or Amazon, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

REFERENCES

[1] Punam Bedi and Chhavi Sharma. 2016. Community detection in social networks. *WIREs DMKD* (2016).
[2] Asia J Biega, Krishna P Gummadi, and Gerhard Weikum. 2018. Equity of attention: Amortizing individual fairness in rankings. In *SIGIR*. 405–414.
[3] Avishek Joey Bose and William L. Hamilton. 2019. Compositional Fairness Constraints for Graph Embeddings. In *ICML*.
[4] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. 2017. Fair clustering through fairlets. In *NIPS*. 5029–5037.
[5] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *ITCS*. 214–226.
[6] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and removing disparate impact. In *KDD*.
[7] Gene H. Golub and Charles F. Van Loan. 1996. *Matrix Computations* (third ed.). The Johns Hopkins University Press.
[8] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*. 855–864.
[9] Moritz Hardt, Eric Price, Nati Srebro, et al. 2016. Equality of opportunity in supervised learning. In *NIPS*. 3315–3323.
[10] Glen Jeh and Jennifer Widom. 2002. SimRank: a measure of structural-context similarity. In *KDD*. 538–543.
[11] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. 2012. Enhancement of the Neutrality in Recommendation.
[12] Jian Kang and Hanghang Tong. 2019. N2N: Network Derivative Mining. In *CIKM*. 861–870.
[13] Jian Kang, Meijia Wang, Nan Cao, Yinglong Xia, Wei Fan, and Hanghang Tong. 2018. AURORA: Auditing PageRank on Large Graphs. In *Big Data*. 713–722.
[14] Michael Kim, Omer Reingold, and Guy Rothblum. 2018. Fairness through computationally-bounded awareness. In *NeurIPS*. 4842–4852.
[15] Matthäus Kleindessner, Samira Samadi, Pranjal Awasthi, and Jamie Morgenstern. 2019. Guarantees for Spectral Clustering with Fairness Constraints. In *ICML*.
[16] Danai Koutra, Ankur Parikh, Aaditya Ramdas, and Jing Xiang. 2011. Algorithms for graph similarity and subgraph matching.
[17] Preethi Lahoti, Krishna P Gummadi, and Gerhard Weikum. 2019. Operationalizing Individual Fairness with Pairwise Fair Representations. *PVLDB* (2019).
[18] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.
[19] Shike Mei and Xiaojin Zhu. 2015. Using machine teaching to identify optimal training-set attacks on machine learners. In *AAAI*.
[20] Andrew Y Ng, Michael I Jordan, and Yair Weiss. 2002. On spectral clustering: Analysis and an algorithm. In *NIPS*. 849–856.
[21] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web.* Technical Report. Stanford InfoLab.
[22] John Palowitch and Bryan Perozzi. 2019. MONET: Debiasing Graph Embeddings via the Metadata-Orthogonal Training Unit. *arXiv preprint arXiv:1909.11793* (2019).
[23] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *WSDM*. 459–467.
[24] Tahleen Rahman, Bartlomiej Surma, Michael Backes, and Yang Zhang. 2019. Fairwalk: Towards Fair Graph Embedding. In *IJCAI*.
[25] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *WWW*. 1067–1077.
[26] Sahil Verma and Julia Rubin. 2018. Fairness definitions explained. In *FairWare*.
[27] S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. 2010. Graph kernels. *JMLR* 11, Apr (2010), 1201–1242.
[28] Fei Wang, Tao Li, and Changshui Zhang. 2008. Semi-Supervised Clustering via Matrix Factorization. In *SDM*.
[29] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. 2010. Twitterrank: finding topic-sensitive influential twitterers. In *WSDM*. ACM, 261–270.
[30] Yongkai Wu, Lu Zhang, Xintao Wu, and Hanghang Tong. 2019. PC-Fairness: A Unified Framework for Measuring Causality-Based Fairness. In *NeurIPS*.
[31] Sirui Yao and Bert Huang. 2017. Beyond parity: Fairness objectives for collaborative filtering. In *NIPS*. 2921–2930.
[32] Yuan Yao, Hanghang Tong, Guo Yan, Feng Xu, Xiang Zhang, Boleslaw K Szymanski, and Jian Lu. 2014. Dual-regularized one-class collaborative filtering. In *CIKM*.
[33] Gal Yona and Guy N. Rothblum. 2018. Probably Approximately Metric-Fair Learning. In *ICML*. 5666–5674.
[34] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning fair representations. In *ICML*. 325–333.
[35] Rui Zhang and Hanghang Tong. 2019. Robust Principal Component Analysis with Adaptive Neighbors. In *NeurIPS*. 6959–6967.
[36] Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. 2019. Graph convolutional networks: a comprehensive review. *Computational Social Networks* (2019).
[37] Dawei Zhou, Jingrui He, Hongxia Yang, and Wei Fan. 2018. Sparc: Self-paced network representation for few-shot rare category characterization. In *KDD*.

REPRODUCIBILITY

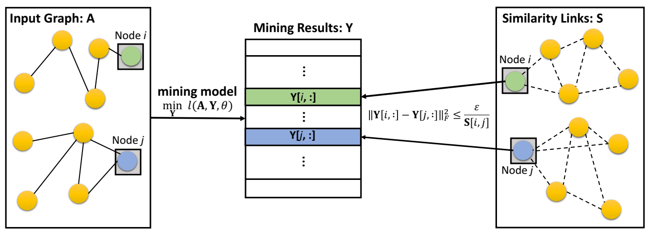
Input Graph: A
Node i
Node j
mining model
$\min_{\mathbf{Y}} \; l(\mathbf{A}, \mathbf{Y}, \theta)$
Mining Results: Y
Y[i, :]
Y[j, :]
$\|\mathbf{Y}[i,:] - \mathbf{Y}[j,:]\|_F^2 \leq \frac{\varepsilon}{\mathbf{S}[i,j]}$
Similarity Links: S
Node i
Node j

Figure 1: An illustrative example of individual fairness for graph mining. S is a node-node similarity matrix. Individual fairness requires that the difference between the mining results be small for a pair of similar nodes i and j.

A – Details of Experimental Setup

Datasets. All datasets are undirected uni-partite graphs. We extract the largest connected components in these datasets for experiments in spectral clustering. The largest one in Table 4 is used to test efficiency of the proposed methods. These datasets are collected from various application-domains, including collaboration networks (Collaboration), social networks (Social), physical infrastructure networks (Infra) and biology network (Biology). We provide the detailed descriptions of these datasets as follows.

- Collaboration Networks. In this type of networks, nodes usually represent researchers. Two researchers are connected if they have collaborated together. We use three collaboration networks in the field of Physics from arXiv preprint archive[6]: Astro Physics (*AstroPh*) and Condense Matter Physics (*CondMat*) [18].
- Social Networks. Here, nodes are users and edges indicate mutual social relationships. Among them, *Facebook* [18] is the page-page network of official Facebook pages, which is collected through Facebook Graph API in November, 2017. *Twitch* [18] is the user-user social network of gamers that streams in English on the popular game streaming website Twitch[7].
- Biology Network. This domain includes the well-known *PPI* [18] network. It is a subgraph of the protein-protein interaction network for Homo Sapiens.

Baseline Methods. We compare the performance of debiased graph mining results with the original graph mining results without consideration of individual fairness.

Similarity Matrix. For each dataset, we construct its node-node similarity S matrix by two different similarity measures: Jaccard index and cosine similarity. For PageRank and LINE, we filter out similarity links smaller than a pre-defined threshold. The threshold is defined as

$$threshold = mean(\mathbf{S}) + 0.75 std(\mathbf{S})$$

where $mean(\mathbf{S})$ and $std(\mathbf{S})$ calculates the mean and standard deviation of all non-zero elements in S.

Metrics. To answer **RQ1**, we use two types of measures. First, we measure the difference between original/vanilla mining results $\bar{\mathbf{Y}}$ and debiased mining results $\mathbf{Y}^*$ as $Diff = \|\mathbf{Y}^* - \bar{\mathbf{Y}}\|_F / \|\bar{\mathbf{Y}}\|_F$. For PageRank, we also measure KL divergence between $\bar{\mathbf{Y}}$ and $\mathbf{Y}^*$ (i.e., $KL(\frac{\mathbf{Y}^*}{\|\mathbf{Y}^*\|_1} || \frac{\bar{\mathbf{Y}}}{\|\bar{\mathbf{Y}}\|_1}) = \sum_i \frac{\mathbf{Y}^*[i]}{\|\mathbf{Y}^*\|_1} \log \frac{\mathbf{Y}^*[i]/\|\mathbf{Y}^*\|_1}{\bar{\mathbf{Y}}[i]/\|\bar{\mathbf{Y}}\|_1}$). We normalize $\bar{\mathbf{Y}}$ and $\mathbf{Y}^*$ since the norm may not equal to 1. Second, we also use a set of mining task specific performance metrics. In detail, for PageRank, we use precision (Prec) and normalized discounted cumulative gain (NDCG). We label the top-K entities in original PageRank as relevant ($rel = 1$) and others as irrelevant ($rel = 0$). Then, we calculate precision at K ($Prec@K = \frac{\#\, of\, relevant\, items}{K}$) and NDCG at K ($NDCG@K = \sum_{i=1}^{K} \frac{rel}{\log(1+i)}$). For spectral clustering, we use normalized mutual information (NMI) to measure the agreement between two cluster assignments before and after debiasing, which is defined as $NMI(C, C_0) = \frac{2MI(C,C_0)}{H(C)+H(C_0)}$, where C_0 and C are the cluster assignments of original and debiased spectral clustering, $\mathrm{MI}(C, C_0)$ is the mutual information between C and C_0, and $H(C)$ is the entropy of assignment C. For LINE, we perform link prediction using debiased mining results and original mining results and compare their F1 score and ROC-AUC score.

To answer **RQ2**, we measure to what extent the individual bias is reduced as $Reduce = 1 - \frac{\mathrm{Tr}((\mathbf{Y}^*)'\mathbf{L}_\mathbf{S}\mathbf{Y}^*)}{\mathrm{Tr}(\bar{\mathbf{Y}}'\mathbf{L}_\mathbf{S}\bar{\mathbf{Y}})}$.

Finally, to answer **RQ3**, we measure the runtime of each proposed method (Time) in seconds.

Parameter Settings. To debias the input graph, for PageRank, we set $\alpha = 1 \times 10^6$ for PPI dataset, $\alpha = 5 \times 10^6$ for other datasets and $\eta = 5 \times 10^{-4}$ for all datasets; for spectral clustering, we set $\alpha = 3 \times 10^5$, $\eta = 0.02$ for Twitch dataset and $\alpha = 1 \times 10^7$, $\eta = 0.05$ for PPI dataset; for LINE, we set $\alpha = 0.25$, $\eta = 0.5$ for Twitch dataset and $\alpha = 10$, $\eta = 0.025$ for PPI dataset. To debias the mining model and the mining results, we set $\alpha = 0.5$ for all mining tasks.

Besides, for PageRank, we set its damping factor $c = 0.85$ and symmetrically normalize the adjacency matrix **A** and similarity matrix **S** to ensure convergence of power iterations; for spectral clustering, we set the number of clusters as 10; for LINE, we randomly select 85% of all edges as training set, 5% as validation set and 10% as test set. During model training, we sample $3200 \times n$ edges for each dataset, where n is the number of nodes, and use the same learning rate as in [25].

Machine Configuration. All experiments are performed on a Windows PC with i7-9800X CPU and 64GB RAM. All datasets are publicly available. All codes are programmed in Python 3.7. The source code will be released upon publication of the paper.

B – Additional Experimental Results

Additional experimental results on debiasing the input graph for spectral clustering and LINE are shown in Tables 9 and 8, respectively. From the table, we can see that our proposed debiasing the input graph method can effectively reduce the bias while preserving the performance of vanilla algorithm, which is consistent with our evaluation results shown in Section 6.2.

C – Debiasing the Input Graph

Algorithm 1 Instantiation with PageRank. Given a symmetric normalized graph **A**, by Table 2, PageRank on graph essentially calculates the fixed-point solution: $\mathbf{r} = (1 - c)(\mathbf{I} - c\mathbf{A})^{-1}\mathbf{e}$, where c is the damping factor and **e** is the teleportation vector. With that in

[6]https://arxiv.org/
[7]https://www.twitch.tv/

Table 8: Effectiveness results for LINE. Lower is better in gray columns. Higher is better in the others.

Datasets	Jaccard Index							Cosine Similarity						
	Diff	Orig. ROC	Fair ROC	Orig. F1	Fair F1	Reduce	Time	Diff	Orig. ROC	Fair ROC	Orig. F1	Fair F1	Reduce	Time
Twitch	1.079	0.687	0.691	0.625	0.622	1.92%	1878	1.267	0.687	0.662	0.625	0.606	12.1%	1999
PPI	0.674	0.682	0.678	0.618	0.620	2.06%	1656	0.699	0.682	0.686	0.618	0.621	1.22%	1779

Table 9: Effectiveness results for spectral clustering. Lower is better in gray columns. Higher is better in the others.

Datasets	Jaccard Index				Cosine Similarity			
	Diff	NMI	Reduce	Time	Diff	NMI	Reduce	Time
Twitch	0.031	1.000	5.44%	1698	0.107	1.000	24.5%	1714
PPI	1.035	0.914	19.5%	829.3	0.933	0.849	24.1%	985.1

mind, we can rewrite Eq. (5) as

$$\frac{\partial J}{\partial \tilde{\mathbf{A}}} = 2(\tilde{\mathbf{A}} - \mathbf{A}) + 2\alpha \left[\mathbf{r}'\mathbf{L}_\mathbf{S} \frac{\partial \mathbf{r}}{\partial \tilde{\mathbf{A}}[i,j]} \right] \tag{20}$$

where $\mathbf{r} = (1-c)(\mathbf{I} - c\tilde{\mathbf{A}})^{-1}\mathbf{e}$. Based on [13], we have $\frac{\partial \mathbf{r}}{\partial \tilde{\mathbf{A}}[i,j]} = c\mathbf{r}[j](\mathbf{I} - c\tilde{\mathbf{A}})^{-1}[:,i]$. Then, define $\mathbf{Q} = (\mathbf{I} - c\tilde{\mathbf{A}})^{-1}$, we can further simplify Eq. (20) and get

$$\frac{\partial J}{\partial \tilde{\mathbf{A}}} = 2(\tilde{\mathbf{A}} - \mathbf{A}) + 2c\alpha \mathbf{Q}'\mathbf{L}_\mathbf{S}\mathbf{r}\mathbf{r}' \tag{21}$$

Then, we can easily learn its debiased topology by applying Algorithm 1 with Eq. (21).

Algorithm 1 Instantiation with Spectral Clustering. For spectral clustering, as shown in Table 2, given an undirected graph with adjacency matrix $\mathbf{A}$, it finds the soft cluster membership matrix $\mathbf{U}$ as the eigenvectors of $\mathbf{L}_\mathbf{A}$ associated with the smallest k eigenvalues. With that in mind, we first rewrite Eq. (5) as

$$\frac{\partial J}{\partial \tilde{\mathbf{A}}} = 2(\tilde{\mathbf{A}} - \mathbf{A}) + 2\alpha \frac{\partial \mathrm{Tr}(\mathbf{U}'\mathbf{L}_\mathbf{S}\mathbf{U})}{\partial \tilde{\mathbf{A}}} \tag{22}$$

However, directly calculating $\frac{\partial \mathrm{Tr}(\mathbf{U}'\mathbf{L}_\mathbf{S}\mathbf{U})}{\partial \tilde{\mathbf{A}}}$ is hard, we resort to chain rule. First, to calculate $\frac{\partial \mathrm{Tr}(\mathbf{U}'\mathbf{L}_\mathbf{S}\mathbf{U})}{\partial \mathbf{L}_{\tilde{\mathbf{A}}}}$, we denote $\mathbf{u}_i$ as the i^{th} column of $\mathbf{U}$ and write

$$\frac{\partial \mathrm{Tr}(\mathbf{U}'\mathbf{L}_\mathbf{S}\mathbf{U})}{\partial \mathbf{L}_{\tilde{\mathbf{A}}}} = 2\left[\mathrm{Tr}((\mathbf{L}_\mathbf{S}\mathbf{U})' \frac{\partial \mathbf{U}}{\partial \mathbf{L}_{\tilde{\mathbf{A}}}[i,j]}) \right] = 2\sum_{i=1}^{k} \left[\mathbf{u}_i'\mathbf{L}_\mathbf{S} \frac{\partial \mathbf{u}_i}{\partial \mathbf{L}_{\tilde{\mathbf{A}}}[i,j]} \right] \tag{23}$$

Denote $\mathbf{M}_i = (\lambda_i \mathbf{I} - \mathbf{L}_{\tilde{\mathbf{A}}})^+$ where λ_i is the i^{th} eigenvalue. Written in a matrix form, by the derivative of eigenvectors, we have

$$\left[\mathbf{u}_i'\mathbf{L}_\mathbf{S} \frac{\partial \mathbf{u}_i}{\partial \mathbf{L}_{\tilde{\mathbf{A}}}[i,j]} \right] = \left[\mathbf{u}_i'\mathbf{L}_\mathbf{S}\mathbf{M}[:,i]\mathbf{u}_i[j] \right] = \mathbf{M}_i'\mathbf{L}_\mathbf{S}\mathbf{u}_i\mathbf{u}_i' \tag{24}$$

Then, based on [12], we get

$$\begin{aligned} \frac{\partial \mathrm{Tr}(\mathbf{U}'\mathbf{L}_\mathbf{S}\mathbf{U})}{\partial \tilde{\mathbf{A}}} &= diag\left(\frac{\partial \mathrm{Tr}(\mathbf{U}'\mathbf{L}_\mathbf{S}\mathbf{U})}{\partial \mathbf{L}_{\tilde{\mathbf{A}}}} \right) \mathbf{1}_{n\times n} - \frac{\partial \mathrm{Tr}(\mathbf{U}'\mathbf{L}_\mathbf{S}\mathbf{U})}{\partial \mathbf{L}_{\tilde{\mathbf{A}}}} \\ &= 2\sum_{i=1}^{k} \left(diag\left(\mathbf{M}_i'\mathbf{L}_\mathbf{S}\mathbf{u}_i\mathbf{u}_i' \right) \mathbf{1}_{n\times n} - \mathbf{M}_i'\mathbf{L}_\mathbf{S}\mathbf{u}_i\mathbf{u}_i' \right) \end{aligned} \tag{25}$$

where $\mathbf{1}_{n\times n}$ is an $n \times n$ matrix filled with 1. To learn the debiased topology, we can apply Algorithm 1 by combining Eq. (22) and (25).

D – Proof of Lemma 1

Proof. It takes $O(min\{m_1, m_2\})$ time to calculate $f(\mathbf{A} + \mathbf{A}') \circ \mathbf{L}_\mathbf{S}$ and $O(m_2)$ time to calculate $diag(\mathbf{B}\mathbf{L}_\mathbf{S})$. Thus the overall time complexity is $O(min\{m_1, m_2\} + m_2)$. For space complexity, it takes $O(min\{m_1, m_2\})$ space to save $f(\mathbf{A} + \mathbf{A}') \circ \mathbf{L}_\mathbf{S}$ in sparse format and $O(n)$ space to save $diag(\mathbf{B}\mathbf{L}_\mathbf{S})$. Therefore, the overall space complexity is $O(min\{m_1, m_2\} + n)$. □

E – Cost of Debiasing the Mining Model: A Case Study on PageRank

Given a graph with adjacency matrix $\mathbf{A}$, similarity matrix $\mathbf{S}$ and regularization parameter α, the cost of debiasing the mining model method on PageRank is summarized in Lemma 3.

Lemma 3. *Given a graph with the symmetric normalized adjacency matrix* $\mathbf{A}$ *and node-node similarity matrix* $\mathbf{S}$*, let* $\bar{\mathbf{r}}$ *be the PageRank vector without considering the fairness and* $\mathbf{r}^*$ *be the debiased PageRank vector as in Eq. (12). If teleportation vector* $\|\mathbf{e}\|_1 = 1$ *and similarity matrix* $\|\mathbf{S} - \mathbf{A}\|_F = \delta$*, it satisfies*

$$\|\mathbf{r}^* - \bar{\mathbf{r}}\|_F \le \frac{2\alpha n}{1-c}(\delta + \sqrt{r(\mathbf{A})}\sigma_{max}(\mathbf{A}))$$

where c is the damping factor and α *is the regularization parameter for individual fairness.*

Proof. Recall that debiasing the mining model on PageRank is equivalent to solving the linear system $\mathbf{r} = c(\mathbf{A} - \frac{\alpha}{c}\mathbf{L}_\mathbf{S})\mathbf{r} + (1-c)\mathbf{e}$. After rearranging terms, we can get its closed-form solution as $\mathbf{r}^* = (1-c)(\mathbf{I} - c\mathbf{A} + \alpha\mathbf{L}_\mathbf{S})^{-1}\mathbf{e}$. If we do not consider individual fairness constraint, we can easily set $\mathbf{L}_\mathbf{S} = \mathbf{0}$ and get $\bar{\mathbf{r}} = (1-c)(\mathbf{I} - c\mathbf{A})^{-1}\mathbf{e}$. Then we have the cost of individual fairness in PageRank as

$$\begin{aligned} \|\mathbf{r}^* - \bar{\mathbf{r}}\|_F &= (1-c)\|((\mathbf{I} - c\mathbf{A} + \alpha\mathbf{L}_\mathbf{S})^{-1} - (\mathbf{I} - c\mathbf{A})^{-1})\mathbf{e}\|_F \\ &\le (1-c)\|((\mathbf{I} - c\mathbf{A} + \alpha\mathbf{L}_\mathbf{S})^{-1} - (\mathbf{I} - c\mathbf{A})^{-1})\|_F\|\mathbf{e}\|_F \\ &\le (1-c)\|((\mathbf{I} - c\mathbf{A} + \alpha\mathbf{L}_\mathbf{S})^{-1} - (\mathbf{I} - c\mathbf{A})^{-1})\|_F \\ &= (1-c)\|(\mathbf{I} - c\mathbf{A} + \alpha\mathbf{L}_\mathbf{S})^{-1} \cdot \alpha\mathbf{L}_\mathbf{S} \cdot (\mathbf{I} - c\mathbf{A})^{-1}\|_F \\ &\le \alpha(1-c)\|(\mathbf{I} - c\mathbf{A} + \alpha\mathbf{L}_\mathbf{S})^{-1}\|_F \cdot \|\mathbf{L}_\mathbf{S}\|_F \cdot \|(\mathbf{I} - c\mathbf{A})^{-1}\|_F \end{aligned} \tag{26}$$

Since $\mathbf{A}$ is symmetric normalized matrix, its Laplacian matrix is $\mathbf{I} - \mathbf{A}$, which reveals that $\mathbf{I} - c\mathbf{A} = (1-c)\mathbf{I} + \mathbf{L}_{c\mathbf{A}}$ and $\mathbf{I} - c\mathbf{A} + \alpha\mathbf{L}_\mathbf{S} = (1-c)\mathbf{I} + \mathbf{L}_{c\mathbf{A}+\alpha\mathbf{S}}$. Define $\mathbf{C} = (1-c)\mathbf{I} + \mathbf{L}_{c\mathbf{A}}$ and $\mathbf{D} = (1-c)\mathbf{I} + \mathbf{L}_{c\mathbf{A}+\alpha\mathbf{S}}$. Based on Eq. (18), we have the following two inequalities holds

$$\begin{aligned} \|(\mathbf{D})^{-1}\|_F &\le \sqrt{n}\sigma_{\max}(((1-c)\mathbf{I} + \mathbf{L}_{c\mathbf{A}+\alpha\mathbf{S}})^{-1}) \\ &= \frac{\sqrt{n}}{\sigma_{\min}((1-c)\mathbf{I} + \mathbf{L}_{c\mathbf{A}+\alpha\mathbf{S}})} = \frac{\sqrt{n}}{1-c} \end{aligned} \tag{27}$$

$$\|(\mathbf{C})^{-1}\|_F \le \sqrt{n}\sigma_{\max}(((1-c)\mathbf{I} + \mathbf{L}_{c\mathbf{A}})^{-1}) = \frac{\sqrt{n}}{1-c} \tag{28}$$

Combine Eq. (27), (28) with Eq. (26), we have $\|\mathbf{r}^* - \bar{\mathbf{r}}\|_F \le \frac{\alpha n}{1-c}\|\mathbf{L}_\mathbf{S}\|_F$. As shown in Eq. (19), we have $\|\mathbf{L}_\mathbf{S}\|_F \le 2(\delta + \sqrt{r(\mathbf{A})}\sigma_{\max}(\mathbf{A}))$. Thus, we have $\|\mathbf{r}^* - \bar{\mathbf{r}}\|_F \le \frac{2\alpha n}{1-c}(\delta + \sqrt{r(\mathbf{A})}\sigma_{\max}(\mathbf{A}))$. □

Similar in Section 5, the cost of debiasing the mining model with PageRank depends on the number of nodes n, rank of adjacency matrix $r(\mathbf{A})$ and the largest singular value $\sigma_{\max}(\mathbf{A})$.