Fairness-Aware PageRank

Sotiris Tsioutsiouliklis
University of Ioannina
Greece
stsiouts@cse.uoi.gr

Evaggelia Pitoura
University of Ioannina
Greece
pitoura@cse.uoi.gr

Panayiotis Tsaparas
University of Ioannina
Greece
tsap@cse.uoi.gr

Ilias Kleftakis
University of Ioannina
Greece
ikleftakis@cse.uoi.gr

Nikolaos Mamoulis
University of Ioannina
Greece
nikos@cse.uoi.gr

ABSTRACT

Algorithmic fairness has attracted significant attention in the past years. In this paper, we consider fairness for link analysis and in particular for the celebrated Pagerank algorithm. Given that the nodes in a network belong to groups (for example, based on demographic or other characteristics), we provide a parity-based definition of fairness that imposes constraints on the proportion of Pagerank allocated to the members of each group. We propose two families of fair Pagerank algorithms: the first (Fairness-Sensitive Pagerank) modifies the jump vector of the Pagerank algorithm to enforce fairness; the second (Locally Fair Pagerank) imposes a fair behavior per node. We then define a stronger fairness requirement, termed universal personalized fairness, that asks that the derived personalized pageranks of all nodes are fair. We prove that the locally fair algorithms achieve also universal personalized fairness, and furthermore, we prove that this is the only family of algorithms with this property, establishing an equivalence between universal personalized fairness and local fairness. We also consider the problem of achieving fairness while minimizing the utility loss with respect to the original Pagerank algorithm. We present experiments with real and synthetic networks that examine the fairness of the original Pagerank and demonstrate qualitatively and quantitatively the properties of our algorithms.

CCS CONCEPTS

• **Information systems** → **Data mining**; **Social networks**.

KEYWORDS

networks, pagerank, link analysis, fairness, homophily

ACM Reference Format:
Sotiris Tsioutsiouliklis, Evaggelia Pitoura, Panayiotis Tsaparas, Ilias Kleftakis, and Nikolaos Mamoulis. 2021. Fairness-Aware PageRank. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3442381.3450065

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '21, April 19–23, 2021, Ljubljana, Slovenia
© 2021 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3450065

1 INTRODUCTION

Today, algorithmic systems driven by large amounts of data are increasingly being used in all aspects of life. Often, such systems are being used to assist, or, even replace human decision-making. This increased dependence on algorithms has given rise to the field of algorithmic fairness, where the goal is to ensure that algorithms do not exhibit biases towards specific individuals, or groups of users (see e.g., [13] for a survey). We also live in a connected world where networks, be it, social, communication, interaction, or cooperation networks, play a central role. However, surprisingly, fairness in networks has received less attention.

Link analysis algorithms, such as Pagerank [7], take a graph as input and use the structure of the graph to determine the relative importance of its nodes. The output of the algorithms is a numerical weight for each node that reflects its importance. The weights are used to produce a ranking of the nodes, but also as input features in a variety of machine learning algorithms including classification [8], and search result ranking [7].

Pagerank performs a random walk on the input graph, and weights the nodes according to the stationary probability distribution of this walk. At each step, the random walk restarts with probability γ, where the restart node is selected according to a "jump" distribution vector $\mathbf{v}$. Since its introduction in the Google search engine, Pagerank has been the cornerstone algorithm in several applications (see, e.g., [14]). Previous research on the fairness of centrality measures has considered only degrees and found biases that arise as a network evolves [2, 26], or has studied general notions of fairness in graphs based on the premise that similar nodes should get similar outputs [16]. In this work, we focus on the fairness of the Pagerank algorithm.

As in previous research, we view fairness as lack of discrimination against a protected group defined by the value of a sensitive attribute, such as, gender, or race [13]. We operationalize this view by saying that a link analysis algorithm is ϕ-*fair*, if the fraction of the total weight allocated to the members of the protected group is ϕ. The value of ϕ is a parameter that can be used to implement different fairness policies. For example, by setting ϕ equal to the ratio of the protected nodes in the graph, we ask that the protected nodes have a share in the weights proportional to their share in the population, a property also known as demographic parity [9]. We also consider *targeted* fairness, where we focus on a specific subset of nodes to which we want to allocate weights in a fair manner.

We revisit Pagerank through the lens of our fairness definitions, and we consider the problem of defining families of Pagerank algorithms that are fair. We also define the *utility loss* of a fair algorithm as the difference between its output and the output of the original Pagerank algorithm, and we pose the problem of achieving fairness while minimizing utility.

We consider two approaches for achieving fairness. The first family of algorithms we consider is the *fairness-sensitive* Pagerank family which exploits the jump vector $\mathbf{v}$. There has been a lot of work on modifying the jump vector to obtain variants of Pagerank biased towards a specific set of nodes. The topic-sensitive Pagerank algorithm [15] is such an example, where the probability is assigned to nodes of a specific topic. In this paper, we take the novel approach of using the jump vector to achieve ϕ-fairness. We determine the conditions under which this is feasible and formulate the problem of finding the jump vector that achieves ϕ-fairness while minimizing utility loss as a convex optimization problem.

Our second family of algorithms takes a microscopic view of fairness by looking at the behavior of each individual node in the graph. Implicitly, a link analysis algorithm assumes that links in the graph correspond to endorsements between the nodes. Therefore, we can view each node, as an agent that *endorses* (or *votes for*) the nodes that it links to. Pagerank defines a process that takes these individual actions of the nodes and transforms them into a global weighting of the nodes. We thus introduce, the *locally fair PageRank algorithms*, where each individual node acts fairly by distributing its own pagerank to the protected and non-protected groups according to the fairness ratio ϕ. Local fairness defines a dynamic process that can be viewed as a *fair random walk*, where *at each step* of the random walk (not only at convergence), the probability of being at a node of the protected group is ϕ.

In our first locally fair PageRank algorithm, termed the *neighborhood locally fair* Pagerank algorithm, each node distributes its pagerank fairly among its immediate neighbors, allocating a fraction ϕ to the neighbors in the protected group, and $1-\phi$ to the neighbors in the non-protected group. The *residual-based locally fair* Pagerank algorithms generalizes this idea. Consider a node i that has less neighbors in the protected group than ϕ. The node distributes an equal portion of its pagerank to each of its neighbors and a residual portion $\delta(i)$ to members in the protected group but not necessarily in its own neighborhood. The residual is allocated based on *a residual redistribution policy*, which allows us to control the fairness policy. In this paper, we exploit a residual redistribution policy that minimizes the utility loss.

We then define a stronger fairness requirement, termed universal personalized fairness, that asks that the derived personalized pageranks of all nodes are fair. We prove that the locally fair algorithms achieve also universal personalized fairness. Surprisingly, the locally fair algorithms are the *only* family of algorithms with this property. Thus, we show that an algorithm is locally fair, if and only if, it is universally personalized fair.

We use real and synthetic datasets to study the conditions under which Pagerank and personalized Pagerank are fair. We also evaluate both quantitatively and qualitatively the output of our fairness-aware algorithms.

In summary, in this paper, we make the following contributions:

- We initiate a study of fairness for the Pagerank algorithm.
- We propose the fairness-sensitive Pagerank family that modifies the jump vector so as to achieve fairness, and the locally fair Pagerank family that guarantees that individually each node behaves in a fair manner.
- We prove that local fairness implies universal personalized fairness and also that this is the only family of algorithms with this property, establishing an equivalence between local fairness and universal personalized fairness.
- We perform experiments on several datasets to study the conditions under which Pagerank unfairness emerges and evaluate the utility loss for enforcing fairness.

The remainder of this paper is structured as follows. In Section 2, we provide definitions of fairness and we formulate our problems. In Sections 3 and 4, we introduce the fairness sensitive and the locally fair families of Pagerank algorithms. In Section 5, we discuss personalized fairness and we show an equivalence between local and universal personalized fairness. The results of our experimental evaluation are presented in Section 6. Finally, we present related research in Section 7 and our conclusions in Section 8.

2 DEFINITIONS

In this section, we first present background material and then we define Pagerank fairness.

2.1 Preliminaries

The Pagerank algorithm [7] pioneered link analysis for weighting and ranking the nodes of a graph. It was popularized by its application in the Google search engine, but it has found a wide range of applications in different settings [14]. The algorithm takes as input a graph $G = (V, E)$, and produces a scoring vector, that assigns a weight to each node $v \in V$ in the graph. The scoring vector is the stationary distribution of a random walk on the graph G.

The Pagerank random walk is a random walk with restarts. It is parameterized by the value γ, which is the probability that the random walk will restart at any step. The node from which the random walk restarts is selected according to the jump vector $\mathbf{v}$, which defines a distribution over the nodes in the graph. Typically, the jump probability is set to $\gamma = 0.15$, and the jump vector is set to the uniform vector.

The "organic" part of the random walk is governed by the transition matrix $\mathbf{P}$, which defines the transition probability $P[i, j]$ between any two nodes i and j. The transition matrix is typically defined as the normalized adjacency matrix of graph G. Special care is required for the sink nodes in the graph, that is, nodes with no outgoing edges. In our work, we adopt the convention that, when at a sink node, the random walk performs a jump to a node chosen uniformly at random [14]. That is, the corresponding zero-rows in the matrix $\mathbf{P}$ are replaced by the uniform vector.

The Pagerank vector $\mathbf{p}$ satisfies the equation:

$$\mathbf{p}^T = (1-\gamma)\mathbf{p}^T\mathbf{P} + \gamma\,\mathbf{v}^T \tag{1}$$

It can be computed either by solving the above equation, or by iteratively applying it to any initial probability vector.

The Pagerank algorithm is fully defined by the three parameters we described above: the transition matrix $\mathbf{P}$, the restart probability

γ, and the restart (or jump) vector $\mathbf{v}$. Different settings for these parameters result in different algorithms. Given a graph $G = (V, E)$, let $\mathcal{PR}(G)$ denote the family of all possible Pagerank algorithms on graph G. Each algorithm in $\mathcal{PR}(G)$ corresponds to a triplet $(\mathbf{P}(G), \gamma, \mathbf{v}(G))$ for the parameters of the random walk. This is a very general family that essentially includes all possible random walks defined over the nodes of graph G. We will refer to the algorithm that uses the typical settings as the *original* Pagerank algorithm, OPR, and use $\mathbf{p}_O$ to denote its pagerank vector.

Several variations of the original Pagerank algorithm have been proposed, that modify the above parameters to achieve different goals [14]. We are interested in defining *fair* Pagerank algorithms.

2.2 Fair Pagerank

We focus on graphs where a set of nodes defines a protected group based on the value of some protected attribute. For example, in the case of social, collaboration, or citation networks where each node is a person, protected attributes may correspond to gender, age, race, or religion. In the following for simplicity, we assume binary such attributes, but our algorithms can be extended for the general case. We consider two types of nodes, red and blue nodes, and the corresponding groups denoted R and B respectively. Group R is the protected group. We denote with $r = \frac{|R|}{n}$, and $b = \frac{|B|}{n}$, the fraction of nodes that belong to the red and blue group respectively.

Let $\text{PR} \in \mathcal{PR}(G)$ be a Pagerank algorithm on graph G. We will use $\text{PR}(u)$ to denote the pagerank mass that PR assigns to node u, and, abusing the notation, $\text{PR}(R)$ to denote the total pagerank mass that PR assigns to the nodes in the red group (for the blue group, $\text{PR}(B) = 1 - \text{PR}(R)$). We will say that PR is *fair*, if it assigns weights to each group according to a specified ratio ϕ.

Definition 1 (Pagerank Fairness). *Given a graph $G = (V, E)$ containing the protected group $R \subseteq V$, and a value $\phi \in (0, 1)$, a Pagerank algorithm $\text{PR} \in \mathcal{PR}(G)$ is ϕ-fair on graph G, if $\text{PR}(R) = \phi$.*

The ratio ϕ may be specified so as to implement specific affirmative action policies, or other fairness enhancing interventions. For example, ϕ may be set in accordance to the 80-percent rule advocated by the US Equal Employment Opportunity Commission (EEOC), or some other formulation of disparate impact [12]. Setting $\phi = r$, we ask for a fair Pagerank algorithm that assigns weights proportionally to the sizes of the two groups. In this case, fairness is analogous to demographic parity, i.e., the requirement that the demographics of those receiving a positive outcome are identical to the demographics of the population as a whole [9].

Our goal is to define fair Pagerank algorithms. We say that a *family* of Pagerank algorithms $FPR \subseteq \mathcal{PR}(G)$ is ϕ-fair if all the Pagerank algorithms in the family are ϕ-fair. The first problem we consider is to find such families of algorithms.

Problem 1. *Given a graph $G = (V, E)$ containing a protected group of nodes $R \subseteq V$, and a value $\phi \in (0, 1)$, define a family of algorithms $FPR \subseteq \mathcal{PR}(G)$ that is ϕ-fair.*

We can achieve fairness by modifying the parameters of the Pagerank algorithm. For the following, we assume the jump probability γ to be fixed, and we only consider modifications to the transition matrix $\mathbf{P}$ and the jump vector $\mathbf{v}$. A ϕ-fair family of algorithms is defined by a specific process, parameterized by ϕ, for defining $\mathbf{P}$ and $\mathbf{v}$.

A fair Pagerank algorithm will clearly output a different weight vector than the original Pagerank algorithm. We assume that the weights of the original Pagerank algorithm carry some *utility*, and use these weights to measure the *utility loss* for achieving fairness. Concretely, if $\mathbf{f}$ is the output of a fair Pagerank algorithm FPR and $\mathbf{p}_O$ is the output of the original Pagerank algorithm OPR, we define the utility loss of FPR as: $L(\text{FPR}) = L(\mathbf{f}, \mathbf{p}_O) = \|\mathbf{f} - \mathbf{p}_O\|^2$.

The second problem we consider is finding a fair algorithm that minimizes the utility loss.

Problem 2. *Given a ϕ-fair family of algorithms $FPR \subset \mathcal{PR}(G)$, find an algorithm $\text{PR} \in FPR$ that minimizes the utility loss $L(\text{PR})$.*

Finally, we introduce an extension of the fairness definition, termed *targeted fairness*, that asks for a fair distribution of weights among a specific set of nodes S that is given as input. The subset S contains a protected group of nodes S_R. Targeted fairness asks that a fraction ϕ of the pagerank mass that PR assigns to S goes to the protected group S_R. For example, assume a co-authorship graph G, where S is the set of authors of a particular male-dominated field. We want to allocate to the female authors S_R in this field a fraction ϕ of the total pagerank allocated to S.

Definition 2 (Targeted Fairness). *Given a graph $G = (V, E)$, a subset of nodes $S \subseteq V$ containing a protected group $S_R \subseteq S$, and a value $\phi \in (0, 1)$, a Pagerank algorithm $\text{PR} \in \mathcal{PR}(G)$, is targeted ϕ-fair on the subset S of G, if $\text{PR}(S_R) = \phi \text{PR}(S)$.*

The two problems we defined above can also be defined for targeted fairness.

3 FAIRNESS SENSITIVE PAGERANK

Our first family of algorithms achieves fairness by keeping the transition matrix $\mathbf{P}$ fixed and changing the jump vector $\mathbf{v}$ so as to meet the fairness criterion. We denote this family of algorithms as the *Fairness-Sensitive Pagerank* (*FSPR*) algorithms.

3.1 The *FSPR* Algorithm

First, we note that that pagerank vector $\mathbf{p}$ can be written as a linear function of the jump vector $\mathbf{v}$. Solving Equation (1) for $\mathbf{p}$, we have that $\mathbf{p}^T = \mathbf{v}^T \mathbf{Q}$, where

$$\mathbf{Q} = \gamma \left[\mathbf{I} - (1 - \gamma)\mathbf{P}\right]^{-1}$$

Note that if we set $\mathbf{v} = \mathbf{e}_j$, the vector with $\mathbf{e}_j[j] = 1$ and zero elsewhere, then $\mathbf{p}^T = \mathbf{Q}_j^T$, the j-th row of matrix $\mathbf{Q}$. Therefore, the row vector $\mathbf{Q}_j^T$ corresponds to the personalized pagerank vector of node j. The pagerank vector $\mathbf{p}$ is a linear combination of the personalized pagerank vectors of all nodes, as defined by the jump vector.

Let $\mathbf{p}[R]$ denote the pagerank mass that is allocated to the nodes of the protected category. We have that

$$\mathbf{p}[R] = \sum_{i \in R} \left(\mathbf{v}^T \mathbf{Q}\right)[i] = \mathbf{v}^T \left(\sum_{i \in R} \mathbf{Q}_i\right) = \mathbf{v}^T \mathbf{Q}_R$$

where $\mathbf{Q}_i$ is the i-th column of matrix $\mathbf{Q}$, and $\mathbf{Q}_R$ is the vector that is the sum of the columns of $\mathbf{Q}$ in the set R. $\mathbf{Q}_R[j]$ is the total personalized pagerank that node j allocates to the red group.

For the algorithm to be fair, we need $\mathbf{p}[R] = \phi$. Thus, our goal is to find a jump vector $\mathbf{v}$ such that $\mathbf{v}^T\mathbf{Q}_R = \phi$. Does such a vector always exist? We prove the following:

Lemma 1. *Given the vector $\mathbf{Q}_R$, there exists a vector $\mathbf{v}$ such that $\mathbf{v}^T\mathbf{Q}_R = \phi$, if and only if, there exist nodes i, j such that $\mathbf{Q}_R[i] \leq \phi$ and $\mathbf{Q}_R[j] \geq \phi$*

Proof. We have $\mathbf{p}[R] = \mathbf{v}^T\mathbf{Q}_R = \sum_{j=1}^{N} \mathbf{v}[j]\mathbf{Q}_R[j]$, with $0 \leq \mathbf{v}[j] \leq 1$. If $\mathbf{v}^T\mathbf{Q}_R = \phi$, there must exist i, j with $\mathbf{Q}_R[i] \leq \phi$, and $\mathbf{Q}_R[j] \geq \phi$. Conversely, if there exists two such entries i, j, then we can find values π_i and π_j, such that $\pi_i\mathbf{Q}_R[i] + \pi_j\mathbf{Q}_R[j] = \phi$ and $\pi_i + \pi_j = 1$. □

Complexity. Note that there is a very efficient way to compute the personalized pagerank, $\mathbf{Q}_R[j]$, that node j allocates to the red group. We can add a red and a blue absorbing node in the random walk and estimate the probability for each node j to be absorbed by the corresponding absorbing node. This can be done in the time required for running Pagerank. Thus, it is possible to compute the $\mathbf{Q}_R$ vector without doing matrix inversion to compute $\mathbf{Q}$.

3.2 Minimizing Utility Loss

An implication of Lemma 1 is that, in most cases, there are multiple jump vectors that give a fair pagerank vector. We are interested in the solution that minimizes the utility loss.

To solve this problem we exploit the fact that the utility loss function $L(\mathbf{p_v}, \mathbf{p}_O) = \|\mathbf{p_v} - \mathbf{p}_O\|^2$, where $\mathbf{p_v}$ is the fair pagerank vector and $\mathbf{p}_O$ the original vector, is convex. We also can express the fairness requirement as a linear constraint. Thus, we define the following convex optimization problem.

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \|\mathbf{x}^T\mathbf{Q} - \mathbf{p}_O\|^2 \\ \text{subject to} \quad & \mathbf{x}^T\mathbf{Q}_R = \phi \\ & \sum_{i=1}^{n} \mathbf{x}[i] = 1 \\ & 0 \leq \mathbf{x}[i] \leq 1,\ i = 1, \ldots, n \end{aligned}$$

This problem can be solved using standard convex optimization solvers. In our work, we used the CVXOPT software package[1]. The complexity of the algorithm is dominated by the computation of matrix $\mathbf{Q}$ which requires a matrix inversion. We can speed up this process by exploiting the fact that the rows of $\mathbf{Q}$ are personalized pagerank vectors, which can be computed (in parallel) by performing multiple random walks. We can improve performance further using approximate computation, e.g., [3].

3.3 Targeted Fairness *FSPR* Algorithm

We can formulate a similar convex optimization problem for the targeted fairness problem. Let $\mathbf{Q}_S = \sum_{i \in S} \mathbf{Q}_i$ be the sum of columns of $\mathbf{Q}$ for the nodes in S, and $\mathbf{Q}_{S_R} = \sum_{i \in S_R} \mathbf{Q}_i$be the sum of columns of $\mathbf{Q}$ for the nodes in S_R. We define a convex optimization problem that is exactly the same as in Section 3.2, except for the fact that we replace the constraint $\mathbf{x}^T\mathbf{Q}_R = \phi$ with the constraint $\mathbf{x}^T\mathbf{Q}_{S_R} = \phi\mathbf{x}^T\mathbf{Q}_S$.

[1] https://cvxopt.org/

4 LOCALLY FAIR PAGERANK

Our second family of fair Pagerank algorithms, termed *Locally Fair Pagerank* (*LFPR*), takes a microscopic view of fairness, by asking that *each individual node* acts fairly, i.e., each node distributes its own pagerank to red and blue nodes fairly. In random walk terms, local fairness defines a dynamic process that can be viewed as a random walk that is fair at each step, and not just at convergence.

The *LFPR* contains all Pagerank algorithms, where all rows of the transition matrix $\mathbf{P}$ are ϕ-fair vectors. That is, for every node $i \in V$, $\sum_{j \in R} P[i, j] = \phi$. Also, the jump vector $\mathbf{v}$ is ϕ-fair: $\sum_{j \in R} \mathbf{v}[j] = \phi$.

We now consider specific algorithms from the family of locally fair algorithms.

4.1 The Neighborhood *LFPR* Algorithm

We first consider a node that treats its neighbors fairly by allocating a fraction ϕ of its pagerank to its red neighbors and the remaining $1 - \phi$ fraction to its blue neighbors. In random walk terms, at each node the probability of transitioning to a red neighbor is ϕ and the probability of transitioning to a blue neighbor $1 - \phi$.

Formally, we define the *neighborhood locally fair pagerank* (LFPR_N) as follows. Let $out_R(i)$ and $out_B(i)$ be the number of outgoing edges from node i to red nodes and blue nodes respectively. We define $\mathbf{P}_R$ as the stochastic transition matrix that handles transitions to red nodes, or random jumps to red nodes if such links do not exist:

$$\mathbf{P}_R[i, j] = \begin{cases} \frac{1}{out_R(i)}, & \text{if } (i, j) \in \text{E and } j \in R \\ \frac{1}{|R|}, & \text{if } out_R(i) = 0 \text{ and } j \in R \\ 0, & \text{otherwise} \end{cases}$$

The transition matrix $\mathbf{P}_B$ for the blue nodes is defined similarly. The transition matrix $\mathbf{P}_N$ of the LFPR_N algorithm is defined as:

$$\mathbf{P}_N = \phi\,\mathbf{P}_R + (1 - \phi)\,\mathbf{P}_B$$

We also define a ϕ-fair jump vector $\mathbf{v}_N$ with $\mathbf{v}_N[i] = \frac{\phi}{|R|}$, if $i \in R$, and $\mathbf{v}_N[i] = \frac{1-\phi}{|B|}$, if $i \in B$. The neighborhood locally-fair pagerank vector $\mathbf{p}_N$ is defined as:

$$\mathbf{p}_N^T = (1 - \gamma)\mathbf{p}_N^T\mathbf{P}_N + \gamma\,\mathbf{v}_N^T$$

4.2 The Residual-based *LFPR* Algorithms

We consider an alternative fair behavior for individual nodes. Similarly to the LFPR_N algorithm, each node i acts fairly by respecting the ϕ ratio when distributing its own pagerank to red and blue nodes. However, now node i treats its neighbors the same, independently of their color and assigns to each of them the same portion of its pagerank. When a node is in a "biased" neighborhood, i.e., the ratio of its red neighbors is different than ϕ, to be fair, node i distributes only a fraction of its pagerank to its neigbors, and the remaining portion of its pagerank to nodes in the underrepresented group. We call the remaining portion *residual* and denote it by $\delta(i)$. How $\delta(i)$ is distributed to the underrepresented group is determined by a *residual policy*.

Intuitively, this corresponds to a fair random walker that upon arriving at a node i, with probability 1-$\delta(i)$ follows one of i's outlinks and with probability $\delta(i)$ jumps to one or more node belonging to the group that is locally underrepresented.

We now describe the algorithm formally. We divide the (non sink) nodes in V into two sets, L_R and L_B, based on the fraction of their red and blue neighbors. Set L_R includes all nodes i such that $out_R(i)/out(i) < \phi$, where $out(i)$ the out-degree of node i, that is, the nodes for which the ratio of red nodes in their neighborhoods is smaller than the required ϕ ratio. These nodes have a residual that needs to be distributed to red nodes. Analogously, L_B includes all nodes i such that $out_R(i)/out(i) \geq \phi$, that have a residual to be distributed to blue nodes.

Consider a node i in L_R. To compute $\delta_R(i)$ note that each neighbor of i gets a fraction $\rho_R(i) = \frac{1-\delta_R(i)}{out(i)}$ of i's pagerank. The residual $\delta_R(i)$ of i's pagerank goes to red nodes. In order for node i to be ϕ-fair, we have:

$$\frac{1-\delta_R(i)}{out(i)} out_R(i) + \delta_R(i) = \phi \tag{2}$$

Solving for $\delta_R(i)$ and using the fact that $out(i) = out_R(i) + out_B(i)$ we get $\delta_R(i) = \phi - \frac{(1-\phi)\, out_R(i)}{out_B(i)}$, and $\rho_R(i) = \frac{1-\phi}{out_B(i)}$.

Analogously, for a node i in L_B, we get a residual $\delta_B(i) = (1-\phi) - \frac{\phi\, out_B(i)}{out_R(i)}$ that goes to the blue nodes, and $\rho_B(i) = \frac{\phi}{out_R(i)}$.

For a sink node i, we assume that i belongs to both L_R and L_B with residual $\delta_R(i) = \phi$ and $\delta_B(i) = 1-\phi$.

Example. Consider a node i with 5 out-neighbors, 1 red and 4 blue, and let ϕ be 0.5. This is a "blue-biased"node, that is, $i \in L_R$. With the residual algorithm, each of i's neighbors gets $\rho_R(i) = 0.5/4 = 1/8$ portion of i's pagerank, resulting in red neighbors getting 1/8 and blue neighbors 4/8 of i's pagerank. The residual $\delta_B(i) = 3/8$ goes to nodes in the red group so as to attain the ϕ ratio and make i fair. In terms of the random walker interpretation, a random walker that arrives at i, with probability 5/8 chooses one of i's outlinks uniformly at random and with probability 3/8 jumps to nodes in the red group. □.

Formally, we define $\mathbf{P}_L$ as follows:

$$\mathbf{P}_L[i,j] = \begin{cases} \frac{1-\phi}{out_B(i)}, & \text{if } (i,j) \in E \text{ and } i \in L_R \\ \frac{\phi}{out_R(i)}, & \text{if } (i,j) \in E \text{ and } i \in L_B \\ 0 & \text{otherwise} \end{cases}$$

$\mathbf{P}_L$ handles the transitions of nodes to their neighborhood. To express the residual distribution policy, we introduce matrices $\mathbf{X}$ and $\mathbf{Y}$, that capture the policy for distributing the residual to red and blue nodes respectively. Specifically, $\mathbf{X}[i,j]$ denotes the portion of the $\delta_R(i)$ of node $i \in L_R$ that goes to node $j \in R$, and $\mathbf{Y}[i,j]$ the portion of the $\delta_B(i)$ of node $i \in L_B$ that goes to node $j \in B$.

The locally-fair pagerank vector $\mathbf{p}_L$ is defined as:

$$\mathbf{p}_L^T = (1-\gamma)\mathbf{p}_L^T (\mathbf{P}_L + \mathbf{X} + \mathbf{Y}) + \gamma\, \mathbf{v}_N^T$$

Residual Distribution Policies. The $\mathbf{X}$ and $\mathbf{Y}$ allocation matrices allow us the flexibility to specify appropriate policies for distributing the residual. For example, the LFPR$_N$ algorithm is a special case of the residual-based algorithm, with

$$\mathbf{X}_N[i,j] = \begin{cases} \frac{\delta_R(i)}{out_R(i)}, & \text{if } i \in L_R, (i,j) \in E, \text{ and } j \in R \\ \frac{\delta_R(i)}{|R|}, & \text{if } i \in L_R, out_R(i) = 0, \text{ and } j \in R \\ 0 & \text{otherwise} \end{cases}$$

and $\mathbf{Y}_N[i,j]$ defined analogously.

We also consider residual policies where all nodes follow the same policy in distributing their residual, that is, each red node gets the same portion of the red residuals and each blue node the same portion and the blue residuals. In this case, the residual policy is expressed through two (column) vectors $\mathbf{x}$ and $\mathbf{y}$, with $\mathbf{x}[j] = 0$ if $j \in B$ and $\mathbf{y}[j] = 0$, $j \in R$. Each node $i \in L_R$ sends a fraction $\delta_R(i)\mathbf{x}[j]$ of its pagerank to red node j, while each node $i \in L_B$ sends a fraction $\delta_B(i)\mathbf{y}[j]$ of its pagerank to blue node j.

Let δ_R be the vector carrying the red residual, and δ_B the vector carrying the blue residual. We have:

$$\mathbf{p}_L^T = (1-\gamma)\mathbf{p}_L^T (\mathbf{P}_L + \delta_R\, \mathbf{x}^T + \delta_B\, \mathbf{y}^T) + \gamma\, \mathbf{v}_N^T.$$

We define two locally fair Pagerank algorithms based on two intuitive policies of distributing the residual:

The *Uniform Locally Fair Pagerank* (LFPR$_U$) algorithm distributes the residual uniformly. Specifically, we define the vector $\mathbf{x}$, as $\mathbf{x}[i] = \frac{1}{|R|}$ for $i \in R$, and the vector $\mathbf{y}$, as $\mathbf{y}[i] = \frac{1}{|B|}$, for $i \in B$.

The *Proportional Locally Fair Pagerank* (LFPR$_P$) algorithm distributes the residual proportionally to the original pagerank weights $\mathbf{p}_O$ Specifically, we define the vector $\mathbf{x}$, as $\mathbf{x}[i] = \frac{\mathbf{p}_O[i]}{\sum_{i \in R} \mathbf{p}_O[i]}$, for $i \in R$, and the vector $\mathbf{y}$, as $\mathbf{y}[i] = \frac{\mathbf{p}_O[i]}{\sum_{i \in B} \mathbf{p}_O[i]}$, for $i \in B$.

4.3 Fairness of the *LFPR* Algorithms

Although each node acts independently of the other nodes in the network, this microscopic view of fairness results in a macroscopic view of fairness. Specifically, we prove the following theorem.

Theorem 1. *The locally fair Pagerank algorithms are ϕ-fair.*

Proof. Let $\mathbf{e}_R$ denote the vector with 1's at the red nodes and zero at the blue nodes. The amount of pagerank that vector $\mathbf{p}_L$ gives to the red nodes can be expressed as $\mathbf{p}_L^T \mathbf{e}_R$. Let $\mathbf{P}_D = \mathbf{P}_L + \mathbf{X} + \mathbf{Y}$, we have:

$$\mathbf{p}_L^T \mathbf{e}_R = (1-\gamma)\mathbf{p}_L^T \mathbf{P}_D \mathbf{e}_R + \gamma \mathbf{v}_N^T \mathbf{e}_R$$

By design we have that $\mathbf{v}_N^T \mathbf{e}_R = \phi$. For transition matrix $\mathbf{P}_D$, due to the local fairness property, we know that each row has probability ϕ of transitioning to a red node. Therefore, $\mathbf{P}_D \mathbf{e}_R = \phi \mathbf{e}$, where $\mathbf{e}$ is the vector of all ones. Note that since $\mathbf{p}_L$ is a distribution we have that $\mathbf{p}_L^T \mathbf{e} = 1$. We thus have: $\mathbf{p}_L^T \mathbf{e}_R = (1-\gamma)\phi + \gamma\phi = \phi$. □

4.4 Minimizing Utility Loss

We now consider how to distribute the residual so as to minimize the utility loss of the locally fair Pagerank. We denote this algorithm as LFPR$_O$. To this end, we compute the $\mathbf{x}$ and $\mathbf{y}$ residual distribution vectors by formulating an optimization problem.

We can write the vector $\mathbf{p}_L$ as a function of the vectors $\mathbf{x}$ and $\mathbf{y}$ as follows:

$$\mathbf{p}_L^T(\mathbf{x},\mathbf{y}) = \gamma\, \mathbf{v}_N^T \left[\mathbf{I} - (1-\gamma)(\mathbf{P}_L + \delta_R\, \mathbf{x}^T + \delta_B\, \mathbf{y}^T)\right]^{-1}$$

We can now define the optimization problem of finding the vectors $\mathbf{x}$ and $\mathbf{y}$ that minimize the loss function $L(\mathbf{p}_L, \mathbf{p}_O) = \|\mathbf{p}_L(\mathbf{x}, \mathbf{y}) - \mathbf{p}_O\|^2$ subject to the constraint that the vectors $\mathbf{x}$ and $\mathbf{y}$ define a distribution over the nodes in R and B respectively.

Since our problem is not convex, we implement a Stochastic Random Search algorithm for solving it, that looks at randomly selected directions for the gradient, and selects the one that causes the largest decrease. We enforce the distribution constraints by adding a penalty term $\lambda \left((\sum_{i=1}^{n} \mathbf{x}_i - 1)^2 + (\sum_{i=1}^{n} \mathbf{y}_i - 1)^2 \right)$. We enforce the positivity constraints through proper bracketing at the line-search step. The complexity of the algorithm is $O(I \cdot K \cdot T_{PR})$, where I is the number of iterations, K the number of random directions that are examined, and T_{PR} the cost of running Pagerank. In our experiments, I and K are in the order of a few hundreds.

4.5 Targeted Fairness *LFPR* Algorithms

We can apply the locally fair algorithms to the targeted fairness problem. Let S_R and S_B be the red and blue nodes in the set S respectively, and let I_S be the set of in-neighbors of S. The idea is that the nodes in I_S should distribute their pagerank to S_R and S_B fairly, such that the portion that goes to nodes in S_R is a ϕ fraction of the total pagerank that goes to the set S. We can implement the same redistribution policies as in the case of the neighborhood local and the residual-based local fair algorithms.

We also need the (global) jump vector $\mathbf{v}$ to obey the ϕ ratio for the nodes in S. We can achieve this by redistributing the probability $|S|/n$ of the jump vector according to the ϕ ratio.

5 PERSONALIZED FAIRNESS

A special case of the Pagerank algorithm is when the restart vector is defined to be the unit vector $\mathbf{e}_i$ that puts all the mass at a single node i. For any Pagerank algorithm $\text{PR} \in \mathcal{PR}$, we can define the corresponding personalized algorithm PR_i by setting $\mathbf{v} = \mathbf{e}_i$.

The output of the algorithm PR_i is a probability vector, where $\text{PR}_i(u)$ is the probability that a random walk that always restarts at node i is at u after infinite number of steps. We say that node i allocates this probability to node u. Personalized random walks have found several applications in network analysis [14]. For example, the probability $\text{PR}_i(u)$ can be seen as a measure of proximity between node i and node u, and it has been used for recommendations.

For a personalized Pagerank algorithm PR_i, we define $\text{PR}_i(R)$ and $\text{PR}_i(B)$ to be the probability that node i allocates to the red and blue groups respectively. Recall that if $\mathbf{v}$ is the jump vector, then $\text{PR}(R) = \sum_{i \in V} \mathbf{v}[i]\text{PR}_i(R)$. We can think of the probability $\text{PR}_i(R)$, as a measure of how a specific node i "views" the red group, while $\text{PR}(R)$ captures the value that the network places on the red nodes on aggregate.

We could define fairness for the PR_i algorithm using the standard fairness definition. However, note that since the random walk jumps with probability γ to node i at every step, this immediately adds probability γ to the category of node i. This probability is due to the random jump and not due to the "organic" random walk, and the structure of the graph. We thus subtract this probability, and we define the vector $\overline{\text{PR}_i}$, where $\overline{\text{PR}_i}(i) = \text{PR}_i(i) - \gamma$, and $\overline{\text{PR}_i}(u) = \text{PR}_i(u)$, for $u \neq i$. Another way to think of this is that an amount γ of probability is reserved for restarting, and the remaining $1 - \gamma$ is allocated through the random walk. This is the probability mass that we want to allocate fairly. We say that the *personalized* Pagerank algorithm PR_i is ϕ-fair if $\overline{\text{PR}_i}(R) = \phi(1 - \gamma)$.

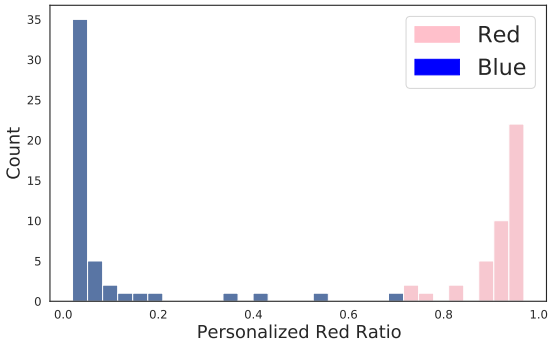


Figure 1: Histogram of personalized red ratio values.

Intuitively, fairness of PR_i implies that node i treats the red and blue groups fairly. For example, if we think of $\overline{\text{PR}_i}(R)$ as a proximity measure, and $\phi = 0.5$, ϕ-fairness implies that node i is equally close to the red and blue groups. Given that this probability is often used for recommendations, this has also implications to the fairness of the recommendations.

Note that a Pagerank algorithm PR may be fair, while the corresponding personalized Pagerank algorithms are not. In Figure 1, we consider the original Pagerank algorithm OPR, and we show the histogram of the $\overline{\text{OPR}_i}(R)$ values for the BOOKS dataset (described in Section 6), for the red and blue nodes, in red and blue respectively. For this dataset, we have that $r = 0.47$ and $\text{OPR}(R)$ is 0.46. That is, the original Pagerank algorithm is almost fair for $\phi = r$. However, we observe that the distribution of the $\overline{\text{OPR}_i}(R)$ values is highly polarized. The values for the red nodes (shown in red) are concentrated in the interval $[0.8, 1]$, while the values for the blue nodes (in blue) are concentrated in the interval $[0, 0.2]$. Thus, although the network as a whole has a fair view of the two groups, the individual nodes are highly biased in favor of their own group.

Motivated by this observation, we consider a stronger definition of fairness, where given an algorithm PR we require that *all* derivative Personalized Pagerank versions of this algorithm are fair. That is, it is not sufficient that the algorithm treats the red group fairly on aggregate, but we require that each individual node is also fair.

DEFINITION 3 (UNIVERSAL PERSONALIZED FAIRNESS). *Given a graph $G = (V, E)$ containing the protected group $R \subseteq V$, and a value $\phi \in (0, 1)$, a Pagerank algorithm $PR \in \mathcal{PR}(G)$ is universally personalized ϕ-fair on graph G, if for every node $i \in V$, the personalized Pagerank algorithm PR_i is personalized ϕ-fair.*

Since we want all personalized algorithms to be fair, universal personalized fairness (universal fairness for short) is a property of the transition matrix $\mathbf{P}$ of the Pagerank algorithm. Since the *FSPR* family does not change the matrix $\mathbf{P}$, it is not universally fair. We will show that the locally fair algorithms are universally fair. Furthermore, we can prove that universally fair algorithms

are locally fair. Therefore, universal fairness is equivalent to local fairness.

Theorem 2. *A Pagerank algorithm PR is universally personalized ϕ-fair if and only if it is locally fair.*

Proof. We first prove that if an algorithm PR is locally fair then it is personalized fair. The proof is similar to that of Theorem 1. Let $\mathbf{p}_i$ denote the personalized pagerank vector of algorithm PR_i. We know that $\mathbf{p}_i^T = (1-\gamma)\mathbf{p}_i^T\mathbf{P} + \gamma\mathbf{e}_i^T$ where $\mathbf{e}_i$ is the vector with 1 at the position i, and 0 everywhere else. The amount of probability that PR_i allocates to the red category can be computed as $\text{PR}_i(R) = \mathbf{p}_i^T\mathbf{e}_R$, where $\mathbf{e}_R$ is the vector with 1 at the positions of all red nodes and 0 everywhere else. Multiplying the equation for $\mathbf{p}_i^T$ with $\mathbf{e}_R$ we have:

$$\text{PR}_i(R) = (1-\gamma)\mathbf{p}_i^T\mathbf{P}\mathbf{e}_R + \gamma\mathbf{e}_i^T\mathbf{e}_R$$

Since PR is fair, for every row of the transition matrix $\mathbf{P}$, the probability of transitioning to a red node is ϕ. Therefore we have that $\mathbf{P}\mathbf{e}_R = \phi\mathbf{e}$, where $\mathbf{e}$ is the vector of all 1's. Also, since $\mathbf{p}_i$ defines a distribution $\mathbf{p}_i^T\mathbf{e} = 1$. Therefore $(1-\gamma)\mathbf{p}_i^T\mathbf{P}\mathbf{e}_R = \phi(1-\gamma)$. We have:

$$\text{PR}_i(R) = \phi(1-\gamma) + \gamma\mathbf{e}_i^T\mathbf{e}_R$$

The value of the second term $\gamma\mathbf{e}_i^T\mathbf{e}_R$ depends on whether the node i is red or blue. If i is blue, $\gamma\mathbf{e}_i^T\mathbf{e}_R = 0$, and we have $\text{PR}_i(R) = \phi(1-\gamma)$. If i is red, $\gamma\mathbf{e}_i^T\mathbf{e}_R = \gamma$, and thus $\text{PR}_i(R) = \phi(1-\gamma)+\gamma$, which proves our claim.

For the opposite direction we make use of the fact that the pagerank vector can be written as $\mathbf{p}^T = \mathbf{v}^T\mathbf{Q}$, where $\mathbf{v}$ is the jump vector and $\mathbf{Q} = \gamma\left[\mathbf{I} - (1-\gamma)\mathbf{P}\right]^{-1}$. From Section 3 we know that the i-th row of matrix $\mathbf{Q}$ is equal to the personalized pagerank vector $\mathbf{p}_i$. The product $\mathbf{r} = \mathbf{Q}\mathbf{e}_R$ is a vector where $\mathbf{r}[i] = \text{PR}_i(R)$. We have assumed that the PR algorithm is universally personalized ϕ-fair. Therefore $\mathbf{r}[i] = \phi(1-\gamma)+\gamma$ if i is red, and $\mathbf{r}[i] = \phi(1-\gamma)$ if i is blue. That is, $\mathbf{r} = \phi(1-\gamma)\mathbf{e} + \gamma\mathbf{e}_R$. Using the fact that $\mathbf{r} = \mathbf{Q}\mathbf{e}_R$, and that $\mathbf{Q}\mathbf{e} = \mathbf{e}$, since $\mathbf{Q}$ is stochastic, we have the following derivations:

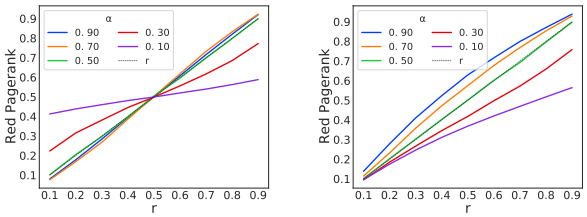
$$\begin{aligned}
\mathbf{Q}\mathbf{e}_R &= \phi(1-\gamma)\mathbf{Q}\mathbf{e} + \gamma\mathbf{e}_R \\
\mathbf{Q}^{-1}\mathbf{Q}\mathbf{e}_R &= \phi(1-\gamma)\mathbf{Q}^{-1}\mathbf{Q}\mathbf{e} + \gamma\mathbf{Q}^{-1}\mathbf{e}_R \\
\mathbf{e}_R &= \phi(1-\gamma)\mathbf{e} + \gamma\frac{1}{\gamma}\left(\mathbf{I} - (1-\gamma)\mathbf{P}\right)\mathbf{e}_R \\
\mathbf{e}_R &= \phi(1-\gamma)\mathbf{e} + \mathbf{e}_R - (1-\gamma)\mathbf{P}\mathbf{e}_R \\
\mathbf{P}\mathbf{e}_R &= \phi\mathbf{e}
\end{aligned}$$

The last equation means that the probability of transitioning from any node in the graph to a red node is ϕ, which proves our claim. □

The theorem holds also when we consider targeted fairness. We can prove that an algorithm is universally personalized targeted fair, if and only if it is locally fair. We omit the details of the proof due to lack of space.

6 EXPERIMENTAL EVALUATION

Our goal is to evaluate Pagerank fairness in different kinds of networks, identify the conditions under which Pagerank unfairness emerges and evaluate the effect of the proposed fair Pagerank algorithms. Previous research has shown that homophily and size imbalance may lead to degree unfairness [2, 11, 26]. Is this the case for Pagerank unfairness?



(a) symmetric (b) asymmetric

Figure 2: Red Pagerank with varying r and α, ϕ-fairness corresponds to the identical line.

Specifically, we address the following three research questions:

RQ1: Under which conditions are the original Pagerank and personalized Pagerank algorithms fair?
RQ2: What is the utility loss incurred by the proposed fair Pagerank algorithms in different networks?
RQ3: What are the qualitative characteristics of the proposed fair Pagerank algorithms?

Datasets. We use both real and synthetic datasets. Our real datasets are the following:

- BOOKS: A network of books about US politics where edges between books represented co-purchasing[2].
- BLOGS: A directed network of hyperlinks between weblogs on US politic [1].
- DBLP: An author collaboration network constructed from DBLP including a subset of data mining and database conferences.
- TWITTER: A political retweet graph from [24].

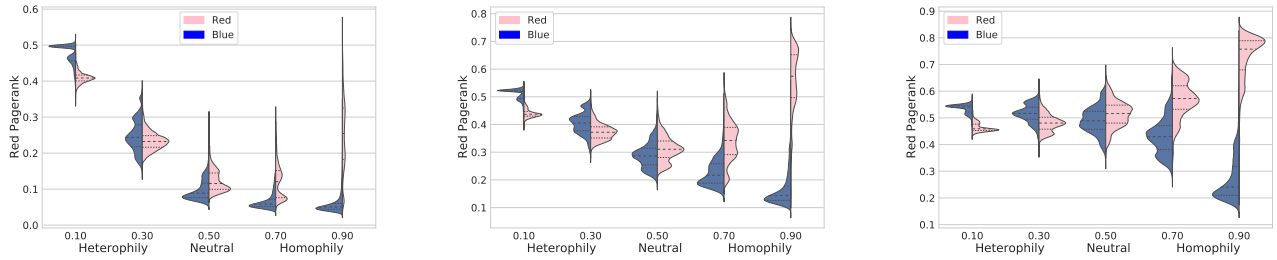
The characteristics of the real datasets are summarized in Table 1. To infer the gender in DBLP, we used the python *gender guesser* package[3]. Regarding *homophily*, we measure for each group, the percentage of the edges that are *cross-edges* that is they point to nodes of the other group divided by the expected number of such edges. We denote these quantities as $cross_R$ and $cross_B$. Values significantly smaller than 1 indicate that the corresponding group exhibits homophily [10].

Synthetic networks are generated using the biased preferential attachment model introduced in [2]. The graph evolves over time as follows. Let $G_t = (V_t, E_t)$ and $d_t(v)$ denote the graph and the degree of node v at time t, respectively. The process starts with an arbitrary connected graph G_0, with n_0 r red and n_0 $(1-r)$ blue nodes. At time step $t+1$, $t > 0$, a new node v enters the graph. The color of v is red with probability r and blue with probability $1-r$. Node v chooses to connect with an existing node u with probability $\frac{d_t(u)}{\sum_{w\in G_t} d_t(w)}$. If the color of the chosen node u is the same with the color of the new node v, then an edge between them is inserted with probability α; otherwise an edge is inserted with probability

[2] http://www-personal.umich.edu/~mejn/netdata/
[3] https://pypi.org/project/gender-guesser/

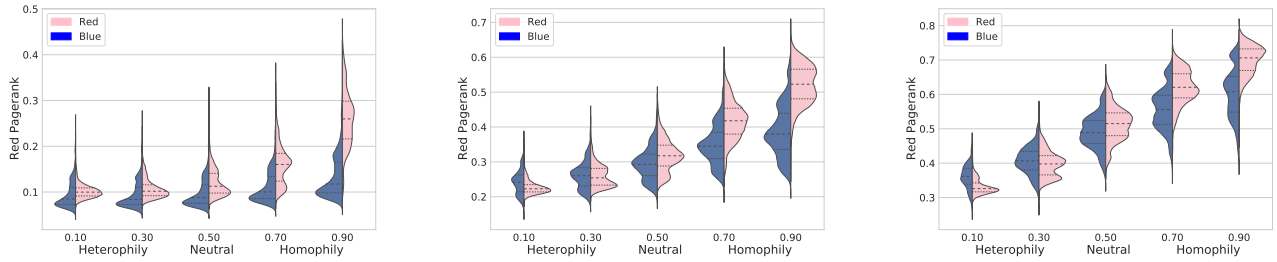
Table 1: Real dataset characteristics. r, b: relative size, $cross_R$, $cross_B$: percentage of cross-edges, p_R, p_B: original pagerank assigned to the protected and unprotected group respectively.

Dataset	#nodes	#edges	Protected attribute	r	b	$cross_R$	$cross_B$	p_R	p_B
BOOKS	92	748	political (left)	0.47	0.53	0.063	0.065	0.46	0.54
BLOGS	1,222	19,089	political (left)	0.48	0.52	0.284	0.036	0.33	0.67
DBLP	13,015	79,972	gender (women)	0.17	0.83	0.96	0.86	0.16	0.84
TWITTER	18,470	61,157	political (left)	0.61	0.39	0.07	0.03	0.57	0.43



(a) $r = 0.1$ (b) $r = 0.3$ (c) $r = 0.5$

Figure 3: Distribution of the red personalized pagerank of the red and blue nodes with varying α in the symmetric case, ϕ-fairness achieved when the red pagerank is r.



(a) $r = 0.1$ (b) $r = 0.3$ (c) $r = 0.5$

Figure 4: Distribution of the red personalized pagerank of the red and blue nodes for varying α in the asymmetric case, ϕ-fairness achieved when the red pagerank is r.

$1 - \alpha$. If no edge is inserted, the process of selecting a neighbor for node v is repeated until an edge is created.

Parameter r controls the group size imbalance. Parameter α controls the level of homophily: $\alpha < 0.5$ corresponds to heterophily, $\alpha = 0.5$ to neutrality and $\alpha > 0.5$ to homophily. We consider: (a) a symmetric case, where α is the same for both groups and (b) an asymmetric case, where we set $\alpha = 0.5$ for the blue group, making it neutral, and vary α for the red group.

The datasets and code are available in GitHub[4].

[4]https://github.com/SotirisTsioutsiouliklis/FairLaR

6.1 When is Pagerank Fair?

We study the conditions under which the original Pagerank and personalized Pagerank algorithms are fair. We assume that the algorithms are fair, if they respect demographic parity, that is, if each group gets pagerank equal to its ratio in the overall population ($\phi = r$). For brevity, we shall call the (personalized) pagerank allocated to red nodes *red (personalized) pagerank* and the (personalized) pagerank allocated to blue nodes *blue (personalized) pagerank* .

First, we study homophily and size imbalance using synthetic datasets. In Figure 2(a), we report the red pagerank for the symmetric and in Figure 2(b) for the asymmetric case. Fairness corresponds to the identity line (red pagerank = r). Values above the line indicate unfairness towards the blue group, while values below the line unfairness towards the red group.

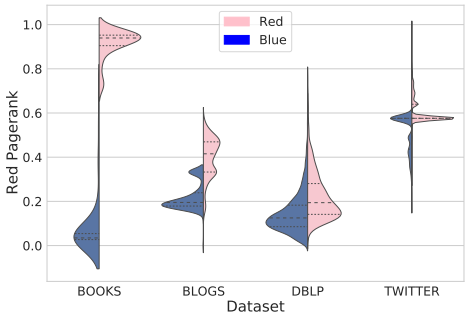


Figure 5: Distribution of the red personalized pagerank of the red and blue nodes for the real datasets, ϕ-fairness when the red pagerank is r (BOOKS $r = 0.47$, BLOGS $r = 0.48$, DBLP $r = 0.17$, AND TWITTER $r = 0.61$).

We also plot the distribution of the red personalized pagerank in Figures 3 and 4 for the symmetric and asymmetric case respectively. To test whether the red personalized pagerank of a node depends on its color, we plot two distributions, one for the red personalized pagerank of the red nodes and one for the red personalized pagerank of the blue nodes. Distributions are plotted in the form of violin plots. Personalized pagerank fairness corresponds to the case in which the two distributions overlap, with their mean on value r. Note that when a group is homophilic, it is expected that a large part of its personalized pagerannk goes to nodes of its own color, e.g., red homophilic nodes have large red personalized pageranks.

Symmetric case (Figures 2(a) and 3): When both groups are homophilic ($\alpha = 0.7, 0.9$), the nodes tend to form two clusters, one with red nodes and one with blue nodes sparsely connected to each other. This leads to an almost fair pagerank (with a very slight unfairness towards the minority group), but highly unfair personalized pageranks. On the contrary, when there is heterophily ($\alpha = 0.1, 0.3$), there are no clusters, nodes tend to connect with nodes of the opposite color, and the larger group favors the smaller one. In this case, the pagerank and personalized pageranks of both the blue and the red nodes are all unfair towards the majority group. This is especially evident when the imbalance in size is large (small r).

Asymmetric case (Figures 2(b) and 4): When the red nodes are homophilic ($\alpha = 0.7, 0.9$), the red group keeps the pagerank to itself. As a result both pagerank and personalized pageranks are unfair towards the blue nodes, especially for larger r. When the red nodes are heterophilic ($\alpha = 0.1, 0.3$), the red group favors the blue group, and as a result, both the pagerank and the personalized pagerank are unfair towards the red nodes, especially for larger r. Thus, independently of the size r, pagerank is unfair to the blue (the neutral) group in case of homophily, and unfair to the red group in the case of heterophily.

Universal fairness: The only case when both pagerank and personalized pageranks are all fair is in a neutral network ($\alpha = 0.5$) with same-size groups ($r = 0.5$) (middle violin plots in Figures 3(c) and 4(c)).

Real datasets: For the real datasets, we report the red pagerank in Table 1 (p_R value) and plot the distributions of the red personalized pagerank of the blue and red nodes in Figure 5. For BOOKS, there is no size imbalance and there is strong symmetric homophily leading to fair pagerank and highly unfair personalized pageranks. For BLOGS, there is no size imbalance, but the blue group is slightly more homophilic, which leads both to unfairness in pagerank for the red group, and unfairness of the personalized pagerank of the blue nodes towards the red ones. For DBLP, we have large size imbalance with the red group being the minority but almost neutral behavior in terms of homophily, leading to an almost fair pagepank, and the majority of personalized pageranks being fair. Finally, for TWITTER, the red group is larger than the blue group but less homophilic which leads to a slight unfairness towards the red group for both pagerank and personalized pageranks.

6.2 What is the Utility Loss for Fairness?

We now look into the utility loss for achieving fairness. We can view utility loss for each network as a measure of the cost we have to pay to achieve ϕ-fairness for this network. First, to assess the utility loss of our algorithms in absolute terms we compute a lower bound for the utility loss.

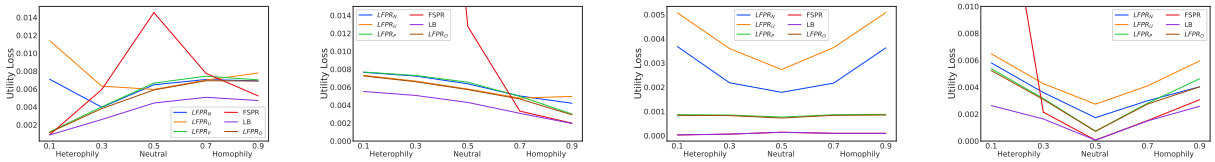
Lower Bound. We compute a lower bound on the utility loss, by constructing the probability vector $\mathbf{w}$ that is ϕ-fair, and it has the minimum utility loss compared to the original pagerank vector $\mathbf{p}_O$. Note that vector $\mathbf{w}$ is not necessarily attainable by any Pagerank algorithm in $\mathcal{PR}$.

To compute $\mathbf{w}$, we start with $\mathbf{p}_O$ and we redistribute the probability between the two groups to make it fair. Let $\mathbf{p}_O(R)$ be the probability assigned to the red group by $\mathbf{p}$. Without loss of generality, assume that $\mathbf{p}_O(R) < \phi$, and let $\Delta = \phi - \mathbf{p}_O(R)$. To make the vector fair, we need to remove Δ probability mass from the nodes in B, and redistribute it to the nodes in R. It is easy to show that to minimize the loss, the optimal redistribution will remove uniformly $\Delta/|B|$ probability from all nodes in B, and add uniformly $\Delta/|R|$ to all nodes in R. This follows from the fact that among all distribution vectors the one with the smallest length is the uniform one. However, this process does not guarantee that the resulting vector will not have negative entries, since some blue nodes may have probability less than $\Delta/|B|$. Let β be the smallest non-zero such probability of any blue node. Our algorithm transfers β probability from all the non-zero blue nodes to the red nodes, and then recursively applies the same procedure for the residual amount of probability that has not been transferred. It is not hard to show that this process will produce a fair vector with the minimum utility loss with respect to $\mathbf{p}_O$.

Figures 6 and 7 report the utility loss for selected synthetic networks and Figure 8 for different values of ϕ for the real datasets. *LB* is the lower bound on utility loss.

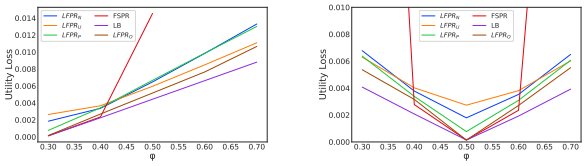
Effect of ϕ: In all cases, loss increases as the requested ϕ deviates from the red pagerank originally assigned to the red group (Figures 7 and 8).

***FSPR*:** In some cases FSPR incurs high utility loss and, occasionally, it is even unable to find an appropriate solution. FSPR achieves fairness by changing the jump vector of the Pagerank algorithm. The overall pagerank vector is a linear combination of the personalized pagerank vectors, with the jump vector providing the coefficients of this linear combination. FSPR increases the jump probability for



(a) symmetric, $r = 0.3$ (b) asymmetric, $r = 0.3$ (c) symmetric, $r = 0.5$ (d) asymmetric, $r = 0.5$

Figure 6: Utility loss for synthetic networks, $\phi = 0.5$.



(a) $r = 0.3$ (b) $r = 0.5$

Figure 7: Utility loss for the synthetic datasets, $\alpha = 0.5$.

the vectors that favor the group it wants to promote and takes away probability from the vectors that favor the other group. However, when there are few, or no appropriate such vectors, FSPR is forced to make extreme choices (assign a lot of probability to a few vectors) thus incurring high utility loss, or it is unable to find any solution.

There are several such examples in our datasets. For the DBLP dataset (Figure 8(c)), for small values of ϕ ($\phi \leq 0.3$), the utility loss of FSPR is close to the optimal, but for $\phi \geq 0.4$, it skyrockets. Looking at Figure 5, we see that there are very few personalized pagerank vectors with red pagerank larger than 0.4. As a result, for $\phi \geq 0.4$, FSPR is forced to allocate all the probability of the jump vector to these nodes, leading to high utility loss. It is also interesting to observe the effect of homophily, or lack of, on the utility loss of FSPR. In Figure 6(a) and (b), utility loss peaks when the network is neutral ($\alpha = 0.5$). In this case, there is no personalized pagerank vector that strongly favors one group over the other.

***LFPR*:** Overall, for the locally fair family of algorithms, the utility loss function is smoother, avoiding high peaks. The locally fair algorithms are directly affected by homophily, since this determines the composition of the neighborhoods of the nodes. As we deviate from neutrality, the loss increases (Figure 6). This holds especially for the LFPR_N algorithm. This can be seen very clearly in BOOKS (Figure 8(a)), where FSPR almost achieves the lower bound, while LFPR_N incurs high utility loss because of BOOKS being very homophilic. The utility loss of LFPR_U and LFPR_P follows in general the same trend as LFPR_N. Finally, LFPR_O redistributes any residual Pagerank so that the utility loss is optimized and in many cases its utility loss is close to the lower bound (*LB*).

Summary: FSPR works well when there are enough unfair nodes (i.e., nodes with unfair personalized pageranks), as it can distribute the jump probability to them to achieve fairness. On the contrary, locally fair algorithms have high utility loss when there are many unfair nodes. LFPR_N is the most sensitive to homophily. The utility loss of LFPR_N can be seen as a measure of local unfairness. Overall, the locally fair algorithms are more stable than FSPR. LFPR_O works very well in terms of utility loss and in many cases it achieves utility close to the lower bound.

6.3 Qualitative Evaluation

In this section, we provide some qualitative experiments, to better understand the properties of the fair algorithms.

Visualization. In Figures 9 and 10, we visualize the results of the algorithms for the BOOKS and the DBLP dataset respectively, for $\phi = 0.5$. Red nodes are colored red, and blue nodes are colored blue. Their size depends on the value of the quantity we visualize. We visualize the pagerank values for the original Pagerank, FSPR and LFPR_N algorithms, and the jump vector probabilities for FSPR.

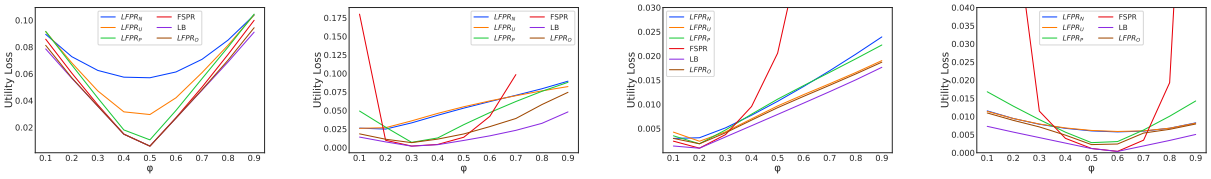
For BOOKS, where the original red pagerank is close to ϕ, FSPR is very similar to the original Pagerank algorithm. BOOKS is homophilic and the jump vector assigns rather uniform weights to almost all nodes. On the other hand, LFPR_N promotes heavily nodes connecting the two opposite groups, i.e., nodes that are minorities in their neighborhoods. We observe a different result in DBLP, where ϕ is much larger than the original red pagerank. LFPR_N distributes weights broadly in the red community, while FSPR is much more aggressive. This is especially evident in the jump vector which promotes a few nodes in the periphery.

Table 2: Top-10 authors with $\phi = 0.3$; the number in parenthesis is the position of the author in the original Pagerank (OPR) (female authors in bold).

OPR	FSPR	LFPR_N
C. Faloutsos	C. Faloutsos (1)	C. Faloutsos (1)
G. Weikum	G. Weikum (2)	G. Weikum (2)
P. S. Yu	P. S. Yu (3)	**J. Widom** (38)
M. Stonebraker	M. Stonebraker (4)	M. Stonebraker (4)
M. J. Franklin	M. J. Franklin (5)	P. S. Yu (3)
H. Garcia-Molina	H. Garcia-Molina (6)	**S. T. Dumais** (28)
D. Kossmann	D. Kossmann (7)	**M. Lalmas** (27)
W. Lehner	**E. A. Rundensteiner** (22)	P. Serdyukov (17)
M. J. Carey	R. Agrawal (11)	**E. Bertino** (25)
M. de Rijke	W. Lehner (8)	**E. A. Rundensteiner** (22)

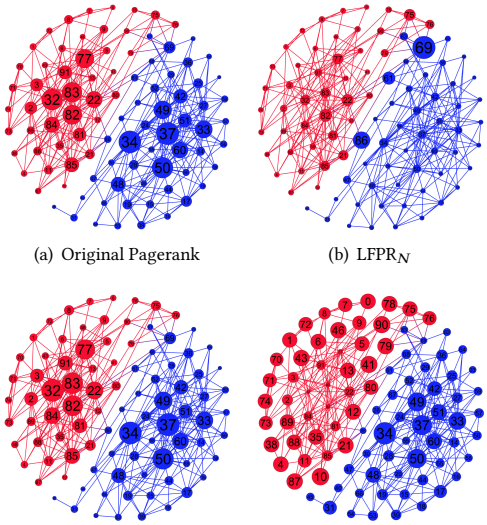
Anecdotal Examples. We will use the DBLP dataset for a qualitative evaluation of the results of the algorithms. Recall that for this dataset, women are the minority with $r = 0.17$, and the original red pagerank is equal to 0.16.

To achieve a fairer representation of women, we apply the LFPR_N and FSPR algorithms with $\phi = 0.3$. In Table 2, we present the first 10 authors for each algorithm. In the original Pagerank algorithm



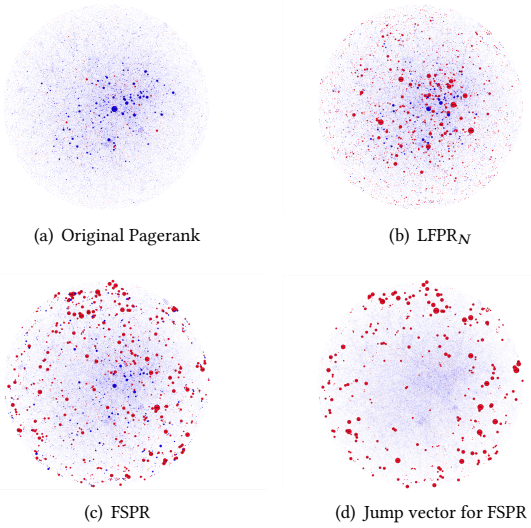
(a) BOOKS (b) BLOGS (c) DBLP (d) TWITTER

Figure 8: Utility loss for the real networks, (original red pagerank, BOOKS: 0.48, BLOGS: 0.33, DBLP: 0.16, AND TWITTER: 0.57).



(a) Original Pagerank (b) LFPR$_N$ (c) FSPR (d) Jump vector for FSPR

Figure 9: Visualization of the BOOK **dataset.**



(a) Original Pagerank (b) LFPR$_N$ (c) FSPR (d) Jump vector for FSPR

Figure 10: Visualization of the DBLP **dataset.**

OPR, there is no female author in the top-10 (the first one appears in position 22). FSPR achieves fairness but also minimizes utility loss, so the result is fair but also close to that of OPR. LFPR$_N$ asks that all authors have ϕ-fair inter-gender collaborations resulting in a large number of female authors appearing in the top-10 positions.

Table 3: Top-3 female authors by conference, ϕ = 0.3.

	SIGIR	SIGMOD
FSPR	Mounia Lalmas Susan T. Dumais Juliana Freire	Elke A. Rundensteiner Elisa Bertino Tova Milo
LFPR$_N$	Susan T. Dumais Mounia Lalmas Emine Yilmaz	Jennifer Widom Elke A. Rundensteiner Fatma Ozcan

We also use DBLP to study the targeted fair Pagerank algorithms. In this case, we want to enforce fair behavior towards authors in specific conferences. We consider two such conferences, SIGIR and SIGMOD, and select S to include authors of each one of them. In Table 3, we show the top-3 women authors for each conference according to our algorithms. We observe that the algorithms produce different results depending on the conference, each time promoting women that are authorities in their respective fields, such as, Suzan Dumais when S is SIGIR, and Jennifer Widom when S is SIGMOD.

7 RELATED WORK

Algorithmic fairness. Recently, there has been increasing interest in algorithmic fairness, especially in the context of machine learning. Fairness is regarded as the lack of discrimination on the basis of some protective attribute. Various definition of fairness having proposed especially for classification [9, 13, 19, 21]. We use a group-fairness definition, based on parity. Approaches to handing fairness can be classified as *pre-processing*, that modify the input data, *in-processing*, that modify the algorithm and *post-processing* ones, that modify the output. We are mostly interested in in-processing techniques.

There is also prior work on fairness in ranking [4, 5, 27, 28]. All of these works consider ranking as an ordered list of items, and use different rules for defining and enforcing fairness that consider different prefixes of the ranking [27, 28], pair-wise orderings [4], or exposure and presentation bias [5, 25].

Our goal in this paper is not to propose a new definition of ranking fairness, but rather to initiate a study of fairness in link

analysis. A distinguishing aspect of our approach is that we take into account the actual Pagerank weights of the nodes, not just their ranking. Furthermore, our focus in this paper, is to design in-processing algorithms that incorporate fairness in the inner working of the Pagerank algorithm. We present a post-processing approach as a means to estimate a lower bound on the utility loss. None of the previous approaches considers ranking in networks, so the proposed approaches are novel.

Fairness in networks. There has been some recent work on network fairness in the context of graph embeddings [6, 18, 23]. The work in [6] follows an in-processing approach that extends the learning function with regulatory fairness enforcing terms, while the work in [18] follows a post-processing approach so as to promote link recommendations to nodes belonging to specific groups. Both works are not related to our approach. The work in [23] extends the node2vec graph embedding method by modifying the random walks used in node2vec with fair walks, where nodes are partitioned into groups and each group is given the same probability of being selected when a node makes a transition. The random walk introduced in [23] has some similarity with the random walk interpretation of LFPR_N. It would be interesting to see, whether our extended residual-based algorithms could be utilized also in the context of graph embeddings, besides its use in link analysis.

There are also previous studies on the effect of homophily, preferential attachment and imbalances in group sizes. It was shown that the combination of these three factors leads to uneven degree distributions between groups [2]. Recent work shows that this phenomenon is exaggerated by many link recommendation algorithms [26]. Evidence of inequality between degree distribution of minorities and majorities was also found in many real networks [17]. Our work extends this line of research by looking at Pagerank values instead of degrees. Along this lines, recent work studies in depth how homophily and size imbalance can affect the visibility that different groups get in link recommendations, i.e, how often nodes in each group get recommended [11]. Very recent work also looks at graph mining algorithms in general from the perspective of individual fairness, where the goal is to produce a similar output for similar nodes [16].

Finally, there is previous work on diversity in network ranking. The goal is to find important nodes that also maximally cover the nodes in the network [20, 29]. Our problem is fundamentally different, since we look for scoring functions that follow a parity constraint.

8 CONCLUSIONS

In this paper, we initiate a study of fairness for Pagerank. We provide definitions of fairness, and we propose two approaches for achieving fairness: one that modifies the jump vector, and one that imposes a fair behavior per node. We prove that the latter is equivalent to a stronger notion of fairness that also guarantees personalized Pagerank fairness. We also consider the problem of attaining fairness while minimizing the utility loss of Pagerank. Our experiments demonstrate the behavior of our different algorithms.

There are many direction for future work. First, we would like to study the role of γ, i.e., the jump probability. Then, it would be interesting to explore other notions of Pagerank fairness, besides ϕ-fairness, for instance ones based on rank-aware fairness [22]. Furthermore, we plan to explore further applications of the theory behind our fair Pagerank algorithms to derive novel notions of fair random walks.

ACKNOWLEDGMENTS

Research work supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the "1st Call for H.F.R.I. Research Projects to Support Faculty Members & Researchers and Procure High-Value Research Equipment" Project Number: HFRI-FM17-1873, GraphTempo.

REFERENCES

[1] L. A. Adamic and N. S. Glance. 2005. The political blogosphere and the 2004 U.S. election: divided they blog. In *LinkKDD*.
[2] C. Avin, B. Keller, Z. Lotker, C. Mathieu, D. Peleg, and Y. A. Pignolet. 2015. Homophily and the Glass Ceiling Effect in Social Networks. In *ITCS*. 41–50.
[3] B. Bahmani, A. Chowdhury, and A. Goel. 2010. Fast Incremental and Personalized PageRank. *Proc. VLDB Endow.* 4, 3 (2010), 173–184.
[4] A. Beutel, J. Chen, T. Doshi, H. Qian, L. Wei, Y. Wu, L. Heldt, Z. Zhao, L. Hong, E. H. Chi, and C. Goodrow. 2019. Fairness in Recommendation Ranking through Pairwise Comparisons. In *KDD*. 2212–2220.
[5] A. J. Biega, K. P. Gummadi, and G. Weikum. 2018. Equity of Attention: Amortizing Individual Fairness in Rankings. In *SIGIR*. 405–414.
[6] A. Joey Bose and W. Hamilton. 2019. Compositional Fairness Constraints for Graph Embeddings. In *ICML*. 715–724.
[7] S. Brin and L. Page. 1998. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems* 30, 1 (1998), 107 – 117.
[8] C. Castillo, De. Donato, A. Gionis, V. Murdock, and F. Silvestri. 2007. Know Your Neighbors: Web Spam Detection Using the Web Topology. In *SIGIR*.
[9] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. S. Zemel. 2012. Fairness through awareness. In *Innovations in Theoretical Computer Science*. 214–226.
[10] D. A. Easley and J. M. Kleinberg. 2010. *Networks, Crowds, and Markets*. Cambridge University Press.
[11] F. Fabbri, F. Bonchi, L. Boratto, and C. Castillo. [n.d.]. The Effect of Homophily on Disparate Visibility of Minorities in People Recommender Systems. In *ICWSM*.
[12] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian. 2015. Certifying and Removing Disparate Impact. In *KDD*. 259–268.
[13] S.e A. Friedler, C. Scheidegger, S. Venkatasubramanian, S. Choudhary, E. P. Hamilton, and D. Roth. 2019. A comparative study of fairness-enhancing interventions in machine learning. In *FAT**. 329–338.
[14] D. F. Gleich. 2015. PageRank Beyond the Web. *SIAM Rev.* 57, 3 (2015), 321–363.
[15] T. H. Haveliwala. 2002. Topic-Sensitive PageRank. In *WWW*.
[16] J. Kang, J. He, R. Maciejewski, and H. Tong. 2020. InFoRM: Individual Fairness on Graph Mining. In *KDD*.
[17] F. Karimi, M. Génois, C. Wagner, P. Singer, and M. Strohmaier. 2018. Homophily influences ranking of minorities in social networks. *Nature Scientific Reports* 8 (2018).
[18] F. Masrour, T. Wilson, H. Yan, P-N Tan, and A-H Esfahanian. 2020. Bursting the Filter Bubble: Fairness-Aware Network Link Prediction. In *AAAI*.
[19] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan. 2019. A Survey on Bias and Fairness in Machine Learning. *CoRR* abs/1908.09635 (2019). arXiv:1908.09635 http://arxiv.org/abs/1908.09635
[20] Q. Mei, J. Guo, and D. R. Radev. 2010. DivRank: the interplay of prestige and diversity in information networks. In *KDD*. 1009–1018.
[21] A. Narayanan. 2018. 21 Fairness Definitions and Their Politics. (tutorial). In *FAT*.
[22] E. Pitoura, G. Koutrika, and K. Stefanidis. 2020. Fairness in Rankings and Recommenders. In *EDBT*. 651–654.
[23] T. A. Rahman, B. Surma, M. Backes, and Y. Zhang. 2019. Fairwalk: Towards Fair Graph Embedding. In *IJCAI*. 3289–3295.
[24] Ryan A. Rossi and Nesreen K. Ahmed. 2015. The Network Data Repository with Interactive Graph Analytics and Visualization. In *AAAI*.
[25] A. Singh and T. Joachims. 2018. Fairness of Exposure in Rankings. In *KDD*.
[26] A. Stoica, C. J. Riederer, and A. Chaintreau. 2018. Algorithmic Glass Ceiling in Social Networks: The effects of social recommendations on network diversity. In *WebConf*. 923–932.
[27] K. Yang and J. Stoyanovich. 2017. Measuring Fairness in Ranked Outputs. In *SSDBM*.
[28] M. Zehlike, F. Bonchi, C. Castillo, S. Hajian, M. Megahed, and R. Baeza-Yates. 2017. FA*IR: A Fair Top-k Ranking Algorithm. In *CIKM*.
[29] X. Zhu, A. B. Goldberg, J. Van Gael, and D. Andrzejewski. 2007. Improving Diversity in Ranking using Absorbing Random Walks. In *HLT-NAACL*. 97–104.