A fair label propagation community detection algorithm

GLYKERIA TOULINA, Department of Computer Science & Engineering, University of Ioannina, Greece
PANAYIOTIS TSAPARAS, Department of Computer Science & Engineering, University of Ioannina, and Archimedes/Athena Research Center, Greece

The widespread use of Machine Learning and Data Science algorithms in a variety of applications that affect our everyday lives has raised concerns about possible biases in the output of these algorithms against minorities and under-represented groups. Consequently, in the recent years, there has been a strong research effort towards creating algorithms with fairness guarantees. In this work, we focus on fairness in community detection algorithms. Community detection is an important task in network analysis, where the goal is to partition the nodes of a graph into subsets (communities), such that nodes are densely connected within the communities, while sparsely connected across communities. We use balance as the fairness criterion for a community. We assume that nodes are partitioned into groups, based on a sensitive attribute (e.g., gender for nodes of a social network), and we ask for the different groups to have a balanced representation in the output communities. We propose a novel fair community detection algorithm that builds on the popular Label Propagation approach. Our algorithm is inspired by Physics principles for incorporating fairness in the label propagation process. We present experiments with our approach on different real and synthetic datasets, where we study the properties of our algorithm and compare with baselines.

Additional Key Words and Phrases: fairness, balance, community detection

ACM Reference Format:

Glykeria Toulina and Panayiotis Tsaparas. 2025. A fair label propagation community detection algorithm. 1, 1 (March 2025), 14 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

The AI revolution of the past decade has led to a world where many decisions that affect human lives are assisted by, or deferred entirely to algorithmic systems trained on massive amounts of data. These decisions are often at an individual level, ranging from simple ones, like where to dine, which movie to watch, or what information to consume, to more important ones, such as which school to apply to, what career to follow, or what treatment to receive. Algorithms are also involved in decisions at organizational, institutional and societal level including financial institutions (determining who should get a loan), the judiciary system (affecting sentencing decisions), academics (influencing admissions), or law enforcement (e.g., face recognition systems for suspect identification).

Given the critical role that algorithms play in our lives, there is increased concern as to whether the decisions of these algorithms are ethical and just. These concerns are not unfounded. There is a stream of empirical evidence that suggests that algorithms may exhibit *biases* in their decisions. For example, the COMPASS system, which determines

Authors' Contact Information: Glykeria Toulina, Department of Computer Science & Engineering, University of Ioannina, Greece, gtoulina@cs.uoi.gr; Panayiotis Tsaparas, Department of Computer Science & Engineering, University of Ioannina, and Archimedes/Athena Research Center, Greece, tsap@uoi.gr.

Manuscript submitted to ACM

the risk of recidivism, was shown to be biased towards African-American inmates, while Google Ads was shown to be more likely to show ads for low-paid jobs to women than men. There are several such examples, where automated systems are shown to exhibit bias against specific groups of individuals in very diverse settings [4].

The need for fairness guarantees on the output of algorithms gave rise to the research area of Responsible AI, and Algorithmic Fairness. There has been significant effort in understanding and measuring algorithmic bias and fairness, but also in mitigating these biases and designing fair algorithms [4]. Most of this effort has been directed towards supervised learning tasks. However, recently, there has been interest in designing fair algorithms for unsupervised learning tasks, such as clustering, or network analysis.

In this work, we focus on fairness for the problem of Community Detection in networks, where the goal is to partition the nodes of a graph into subsets (communities), such that nodes are densely connected within the communities, while sparsely connected across communities. Community detection is an important task in network analysis, with several applications, and there is a variety of community detection algorithms, using different approaches [16].

Our objective is to design a community detection algorithm that produces fair communities, where fairness is measured using the *balance* metric [10]. We assume that nodes are partitioned into groups, based on a sensitive attribute (e.g., gender for nodes of a social network), and we ask for the different groups to have a balanced representation in the output communities. Balance is a commonly used metric for fairness in clustering [9], which has also been employed for community detection [23].

A popular community detection algorithm is the Label Propagation (LP) algorithm [29]. The algorithm uses a label propagation method for finding communities. In brief, the algorithm initially assigns to the nodes in the graph distinct labels. Then, it iteratively updates the labels, so that each node receives the most popular label in its neighborhood. The result is that groups of densely connected nodes converge to the same label, which defines the community. The algorithm is successful in identifying dense, well-separated communities. It has the advantage that it does not require the number of communities as input, and it very efficient, since each iteration takes time linear in the number of edges in the graph.

In this paper we propose a fair variant of the Label Propagation algorithm (FLP).We adopt a Physics-inspired approach for our algorithm. In the original algorithm, when considering the labeling of a node, we view each label in the neighborhood of a node as exerting a gravitational force to the node, proportional to the number of nodes in the neighborhood that have this label. The most popular label is the one that pulls the node to its community. To incorporate fairness in the mechanics of the algorithm, we also introduce the notion of an electrostatic charge for each community, which is defined as the imbalance of the different groups in the community. For example, if we have two groups red and blue, and the community is predominantly blue, then the charge of the community is positive, while if it is predominantly red, it is negative. A node also has a charge +1 if it is blue, and -1 if it red. There is an electrostatic force between the node and the neighbors in a community, that it is either attractive, if the node and the community have charges of different polarity, or repellent if they have charges with the same polarity. An attractive force means that the balance of the community improves if the node is added to the community, while a repellent force means that the node addition makes the community more imbalanced.

The algorithm combines the gravitational and the electrostatic forces to perform the label propagation, providing a trade-off between community quality and fairness. We perform experiments on real and synthetic datasets to explore this trade-off. We also compare with popular fair spectral algorithms, both in terms of fairness, and community quality on real and synthetic data.

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

- We propose a novel fair community detection algorithm. Our algorithm builds upon the popular label propagation algorithm, using Physics-inspired principles.
- We study the trade-off between community quality and fairness using real and synthetic datasets, and we show that our algorithm can achieve fairness without sacrificing quality.
- We compare with a spectral algorithm for finding balanced communities on real and synthetic datasets, and we show that our algorithm performs better in terms of fairness on "difficult" datasets.

The rest of the paper is structured as follows. In Section 2 we present the related work on fairness, and fair clustering and community detection. In Section 3 we present the definition of fairness, and the vanilla label propagation algorithm. In Section 4 we present our algorithm. Section 5 contains the experimental evaluation, and Section 6 concludes the paper.

2 Related Work

Algorithmic fairness is a field that has attracted considerable interest in the past years [4]. It originated from the study of fairness of supervised algorithms like classification [14], but has since expanded to other problems, including unsupervised learning tasks, such as clustering [9], or graph mining [13].

There are typically two approaches to defining fairness: group fairness and individual fairness [4, 14]. We will focus on the former. In group fairness, we assume that our input data is partitioned into groups according the the values of a sensitive attribute; for example, in data regarding individuals, the sensitive attribute may be gender, race, or religion. Group algorithmic fairness requires that the groups are treated equally by the algorithm. There are several ways of defining what equal means [4]. In our setting, we consider representation fairness, where we want the groups to be equally, or proportionally represented in the output of the algorithm.

The problem of community detection that we consider in this paper is an extension of the well-known clustering problem to the case of graph data. There has been considerable amount of work on defining fairness for clustering, and designing fair clustering algorithms [9], that consider both group fairness [3, 5, 8, 10, 17], and individual fairness [2, 6, 7, 22]. The definition of fairness that we will use for our work is that of *balance*, first introduced in the seminal work about *fairlets* [10], This is a group fairness definition that requires that the groups are proportionally represented in the output of the clustering. Intuitively, we want clusters that have members of all groups, and we want to avoid clusters that contain exclusively members of a single group. The work in [10] was followed by several extensions and modifications [3, 5, 19, 20, 30], that consider variants of the original problem.

Similar to clustering, community detection on graphs, asks for a partition of the nodes into sets (communities), such that the induced subgraph defined by a community is well connected, while the connections of nodes across communities are sparse. This is an important problem, with several applications that has received significant attention [15, 16]. There are several algorithms for finding communities, that use different objectives [16]. The Label Propagation Algorithm we consider [29] is a popular algorithm, because of its simplicity and its scalability.

There is limited amount of work on fairness of community detection algorithms. The work most related to ours that in [23, 36] where they define a fair spectral algorithm that aims to achieve balanced communities. The objective is similar to ours, but the technical approach is different. We compare against this algorithm in our experiments.

In [18, 25] they define fairness for communities using the modularity metric. Modularity looks at the connections of the nodes in a community, compared to a null model where edges are created at random. Fairness is defined in this case by considering modularity among and between the groups. The notion of diversity fairness that they define is related to the notion of balance, but the two metrics are distinct.

The work in [1] considers the problem of fairness for the densest subgraph problem, where the goal is to find one or more subgraphs that maximize density. The notion of fairness they consider is similar to balance, since they require for different groups to be equally represented in the selected subgraphs. They consider a spectral algorithm and impose fairness constraints. The densest subgraph problem is related to the community detection, since it looks for subsets of nodes that are densely connected, but has a different objective, as it does not aim to partition the nodes of the graph into communities.

Finally, our work falls within the study of fair algorithms for network analysis [13]. There is significant amount of work in this area, for tasks such as importance computation [34, 35], link recommendations [26], or influence maximization [32, 33]. This work is related, but distinct from the work on community detection.

3 Preliminaries

In this section we introduce the problem of community detection, the definition of fairness, and the Label Propagation algorithm that we will adapt to be fair.

Community Detection: For the following, we assume that we have as input a network $G = (V, E)$, with n nodes $V = \{v_1, v_2, .., v_n\}$ and edge-set $E = \{(v_i, v_j)\}$. The output of a community detection algorithm is a partition of the nodes into k disjoint subsets (communities) $C = \{C_1, C_2, ..C_k\}$, $C_i \subseteq V$, $C_i \cap C_j = \emptyset$, $\cup_{i=1}^{k} C_i = V$. The number of communities k may be given as input, or it is decided by the algorithm. There is a variety of community detection algorithms that use different criteria to produce communities [16]. Typically, we require that the nodes in the community are densely connected, while sparsely connected across communities.

Balance: To define fairness, we assume that the nodes of the graph V are associated with some sensitive attribute A that takes m values $\{a_1, ..., a_m\}$. For example, in a social network, where the nodes of the graph are individuals, the sensitive attribute may be gender, religion, or race. The values of the sensitive attribute, partition the nodes of the graph $V = \{v_1, ..., v_n\}$ into m groups $G = \{G_1, G_2, ..G_m\}$, $G_i = \{v \in V : A(v) = a_i\}$ depending on the attribute value $A(v)$ of the nodes. In the following, we will often refer to the different attribute values, and the corresponding groups, as *colors*.

The fairness metric we will consider is *balance*. Balance was first defined in the seminal work of Chierichetti et al. [10] for defining fairness and designing fair algorithms for the problem of clustering. The definition can be directly applied to community detection, by simply substituting clusters with communities. For the following, we will assume that we have two groups (colors) of nodes. We will refer to them as the blue group G_b and the red group G_r. For a community C, let C^b and C^r denote the subset of blue and red of nodes in C respectively. We define the balance of community C as

$$\mathrm{bal}(C) = \min\left\{\frac{|C^b|}{|C^r|}, \frac{|C^r|}{|C^b|}\right\} \in [0, 1] \tag{1}$$

A perfectly balanced community would have an equal number of red and blue nodes, resulting in a balance value of 1.

Given the definition of the balance of a community, we define the balance of a collection of communities $C = \{C_1, \ldots, C_k\}$ output by a community detection algorithm, as the weighted average of the balance of the communities in

C, that is,

$$\text{bal}(C) = \sum_{C_i \in C} \frac{|C_i|}{n} \text{bal}(C_i) \tag{2}$$

The balance of the collection of the communities is again in $[0, 1]$, with 1 corresponding to the case of a collection of perfectly balanced communities.

The Label Propagation Algorithm: The label propagation algorithm (LP) was first introduced in [29] for community detection in networks. The main idea of the algorithm is to assign labels to the nodes, and then propagate these labels, each time a node adopting the most popular label in its neighborhood. More specifically, initially each node is assigned a unique label, usually the id of the node. Then the algorithm proceeds iteratively. In each iteration, each node updates their label, adopting the most frequent label in their neighborhood. If there are ties, the adopted label is selected at random. The algorithm terminates when no node changes label.

The result of the LP algorithm is that densely connected nodes are likely to adopt the same label. The algorithm is very efficient, as each iteration takes time linear on the number of the edges in the graph. It also has the advantage that it does not require the number of communities to be specified in advance. It is a popular algorithm commonly used for detecting communities.

A weakness of the algorithm is that, when the updates are asynchronous, the resulting communities depend heavily on the random order in which the nodes are considered, while in the case of synchronous updates, it is not guaranteed that the algorithm will converge [12, 24]. To address these issues, the work in [12] proposed a variation of the LP algorithm that performs semi-synchronous updates of the labels. The algorithm first uses a greedy algorithm to obtain a coloring of the graph, such that no two connected nodes have the same color. Nodes with the same color update their labels synchronously, since the label of the one does not affect the label of the other. In the presence of ties, a predetermined order is followed of selecting the label (the largest label id), so that results are consistent in different runs. The stopping criterion is when all nodes in the graph have the most frequent label in their neighborhood. The pseudocode for the algorithm is shown in Algorithm 1. In the pseudocode, we use $N(v)$ to denote the neighborhood of node v, while $\mathbb{1}$ is an indicator function that takes values 1 or 0, if the input condition is true or not. This version of the LP algorithm is effective, efficient, and stable, making it a solid foundation for our fair algorithm.

4 The Fair Label Propagation Algorithm

In this section we present the Fair Label Propagation algorithm (FLP). Our algorithm builds upon the vanilla LP algorithm we described in Section 3.

To incorporate fairness (balance) in the label propagation process, we take a Physics-inspired view of the LP algorithm. We assume that nodes are objects of with mass, that exert a gravitational force to each other. Recall that, from Newton's law, the gravitational force between two objects of mass m_1 and m_2 at distance d is

$$F_g(m_1, m_2) = K_g \frac{m_1 m_2}{d^2}$$

where K_g is the gravitational constant. In our case, we will assume that nodes have unit mass, and that gravitational force is exerted only between connected nodes that are at distance 1. Therefore, a node v is pulled by all of its neighbors $u \in N(v)$ with force $F_g(u, v) = K_g$.

Consider now an iteration of the LP algorithm. Let $N_\ell(v)$ denote the neighbors of v with label ℓ. The community C_ℓ pulls node v with force $F_g(C_\ell, v) = K_g|N_\ell(V)|$. Node v is assigned to the community C_ℓ that pulls node v the strongest, that is, the most popular community in the neighborhood of v.

Algorithm 1 Semi-Synchronous Label Propagation (LP)

Require: Graph $G = (V, E)$, with n nodes with m edges.
Initialization:
for all $v \in V$ **do**
  Assign a unique label $L(v) = v$.
end for
Network Coloring:
Perform a coloring of the nodes in V, such that $\text{color}(u) \neq \text{color}(v)$ for all $(u, v) \in E$.
Colors = The set of all distinct colors
Iterative Label Propagation:
repeat
  for all color $\in$ Colors **do**
    $V_{\text{color}} = \{v \in V : \text{color}(v) = \text{color}\}$
    for all $v \in V_{\text{color}}$ **do**
      $N_\ell(v) = \{u \in N(v) : L(u) == \ell\}$
      $L(v) = \arg\max_\ell |N_\ell(v)|$
    end for
  end for
until Stopping Criterion is met
Stopping Criterion:
All nodes $v \in V$ have the most frequent label among the labels in $N(v)$.

To incorporate fairness (balance) in the LP algorithm, we introduce an additional *electrostatic* force between connected nodes. In addition to the mass, we also assume that each node v has a *charge* q_v. The magnitude of the charge depends on the *imbalance* of the community node v belongs to. We define the imbalance of a community C as

$$\text{imb}(C) = 1 - \text{bal}(C)$$

A node v that belongs to community C has a charge q_v with magnitude $|q_v| = \text{imb}(C)$.

The polarity of the charge depends on the majority color in the community C. Without loss of generality, we assume that if the majority color is red, that is, $|C^r| > |C^b|$, then the charge is positive. In this case we say that community C is a red community. If the majority color in C is blue (community C is a blue community), that is, $|C^b| > |C^r|$, then the charge is negative. Note that a single red blue node has charge $+1$, a single blue node has charge -1, while a balanced community has charge 0.

The electrostatic force between two charged objects is governed by Coulomb's law. For two objects with charges q_1 and q_2 at distance d, the magnitude of the force is:

$$|F_e(q_1, q_2)| = K_c \frac{|q_1||q_2|}{d^2}$$

The force is attractive if the charges have opposite sign, and repellent if the signs are the same. In our case, we assume that two connected nodes (u, v) are at distance 1, and exert electrostatic force to each other. The force is directionless, so we only care about the sign and the magnitude of the force. Therefore, we have: $F_e(u, v) = -K_c q_u q_v$.

We are now ready to define the fair Label Propagation algorithm (FLP) algorithm. The algorithm proceeds similar to in iterations LP, where at each iteration each node is assigned a label from the labels of the nodes in its neighborhood. When assigning a label to node v, the algorithm also computes the electrostatic force that a community C_ℓ in the neighborhood of v exerts to the node. Note that the charge of the node v has magnitude $|q_v| = 1$, while the charge of neighbor u in the C_ℓ community has charge with magnitude $|q_u| = \text{imb}(C_\ell)$. Therefore, the electrostatic force of community C_ℓ to node v is:

$$F_e(C_\ell, v) = \delta(v, C_\ell) K_c |N_\ell(v)| \text{imb}(C_\ell)$$

where $\delta(v, C_\ell) = +1$ if v and C_ℓ have the same color, and $\delta(v, C_e ll) = -1$ if they have different color.

The electrostatic force captures the effect of node v on the balance of the community C_ℓ. An attractive (positive) force means that the community and the node have different color, the community C_ℓ is unbalanced, and thus it needs node v to improve its balance. The strongest the force, the more unbalanced the community. A repellent (negative) force means that the community and the node have the same color, and thus, adding the node to the community will further increase the imbalance of the community.

The FLP algorithm computes the total force $F(C_\ell, v)$ that the community C_ℓ exerts on node v, by combining the gravitational and electrostatic force, that is:

$$F(C_\ell, v) = F_g(C_\ell, v) + F_e(C_\ell, v)$$

It then assigns the node to community C_ℓ that exerts the highest force. The outline of the algorithm is shown in Algorithm 2.

Algorithm 2 Fair Label Propagation (FLP)

Require: Graph $G = (V, E)$, with n nodes with m edges, the group memberships $G = \{G_b, G_r\}$, and parameters K_g, K_c, such that $K_g + K_c = 1$.

Initialization:
for all $v \in V$ **do**
  Assign a unique label $L(v) = v$.
end for
Network Coloring:
Perform a coloring of the nodes in V, such that $\text{color}(u) \neq \text{color}(v)$ for all $(u, v) \in E$.
Colors = The set of all distinct colors
Iterative Label Propagation:
repeat
  for all color $\in$ Colors **do**
    $V_{\text{color}} = \{v \in V : \text{color}(v) = \text{color}\}$
    for all $v \in V_{\text{color}}$ **do**
      $N_\ell(v) = \{u \in N(v) : L(u) == \ell\}$
      $F_g(C_\ell, v) = K_g |N_\ell(v)|$
      $F_e(C_\ell, v) = \delta(v, C_\ell) K_c |N_\ell(v)| \text{imb}(C_\ell)$
      $F(C_\ell, v) = F_g(C_\ell, v) + F_e(C_\ell, v)$
      $L(v) = \arg\max_\ell F(C_\ell, v)$
    end for
  end for
until Stopping Criterion is met
Stopping Criterion:
All nodes $v \in V$ have the most frequent label among the labels in $N(v)$.

By combining the two forces, the algorithm aims to combine two objectives: The quality of the output communities, which is achieved by the gravitational force, as in the vanilla LP algorithm, and the fairness of the output communities, which is achieved by the electrostatic force. To explore the trade-off between the two, we set the parameters K_g and

K_c, such that $K_g + K_c = 1$. Higher values of K_c place more emphasis on fairness. The value $K_c = 0$ corresponds to the vanilla LP algorithm, while the case $K_c = 1$ puts all the emphasis on fairness. Note that the complexity of the algorithm remains unchanged.

5 Experiments

We now present the experimental evaluation of our proposed algorithm. The goal of the experiments is two-fold: Explore the trade-off between fairness and clustering quality, by varying the parameters of the FLP algorithm; Compare with baselines, and understand how the characteristics of the datasets affect the performance of the algorithms, in terms of both fairness and quality.

5.1 Datasets

For our experiments with use both real and synthetic datasets.

Real Datasets We use the following real datasets:

- *Friendship*: The Friendship Net[1] [27] is a network that corresponds to the friendship relations between students in a high school in Marseilles, France. The sensitive attribute is the gender of the students.
- *Facebook*: The Facebook Net[1] [27] consists of a friends list from students in same high school with the Friendship Net. The sensitive attribute is again the gender.
- *Political Blogs*: The Political Blogs[2] is a network defined over a collection of blogs on US politics, with edges representing hyperlinks between the blogs. The sensitive attribute is the political affiliation (conservative or liberal) of the blog.

For all networks, nodes without or unknown attribute value have been removed, and the largest connected component is extracted and used. The characteristics of the datasets, and the balance of the network as a whole are shown in Table 1

Dataset	Nodes	Edges	Features	Network Balance
Friendship Net	134	406	Gender	0.68
Facebook Net	156	1437	Gender	0.81
Political Blogs	1222	16714	Political Affiliation	0.92

Table 1. Real Datasets

Synthetic Datasets

To understand the properties of our algorithm as the characteristics of the input dataset change we also employ synthetic datasets, generated by a variant of the stochastic block model defined in [10]. The model assumes that the nodes are partitioned into k planted communities $T = \{T_1, ..., T_k\}$. We will refer to the planted communities as clusters, to discriminate from the output communities. The nodes are also partitioned into two groups $G = \{G_r, G_b\}$. The model is defined by four parameters: a, b, c, and d, that determine the probability of a connection of two nodes u, v, depending on the group and cluster membership. Specifically: The probability of nodes u and v to be connected, if they belong in the same cluster T_i and the same group G_j is a; The probability of nodes u and v to be connected, if they belong in the same group G_j, but different clusters T_i, T_k is b; The probability of nodes u and v to be connected, if they belong in

[1] http://www.sociopatterns.org/datasets/high-school-contact-and-friendship-networks/
[2] https://websites.umich.edu/~mejn/netdata/

the same cluster T_i but different groups G_r, G_b is c; The probability of nodes u and v to be connected, if they belong in different clusters T_i, T_j and different groups G_r, G_b is d. We have $a > b > c > d$. Therefore, the datasets are constructed such that traditional community detection algorithms will tend to generate monochromatic communities, by placing together nodes from the same group. The goal is to study if the fair community detection algorithms are able to generate fair communities, ideally by recovering the planted clusters.

5.2 Baselines

We compare the FLP algorithm against the Fair Spectral algorithm introduced in [23]. Spectral clustering is a method for clustering data represented as graphs. It relies on Laplacian matrix and aims to minimize the ratio cut objective function [21, 31], which measures the quality of the graph partition. The fair Spectral clustering [23] incorporates fairness constraints in the vanilla Spectral clustering. The goal is that every cluster contains approximately the same number of elements of each group [10]. There are two versions: the unnormalized and the normalized one. The unnormalized fair spectral clustering (UFSP), constructs the Laplacian, projects it onto the null space defined by the fairness constraints, and computes the eigenvectors corresponding to the smallest eigenvalues of the Laplacian. Then it performs k-means clustering, producing fair and high-quality clusters The normalized fair spectral clustering (NFSP), is similar to (UFSP) but adapted to the normalized cut and using the normalized Laplacian. The NFSP algorithm improves the performance when clusters have imbalanced sizes.

5.3 Experimental results

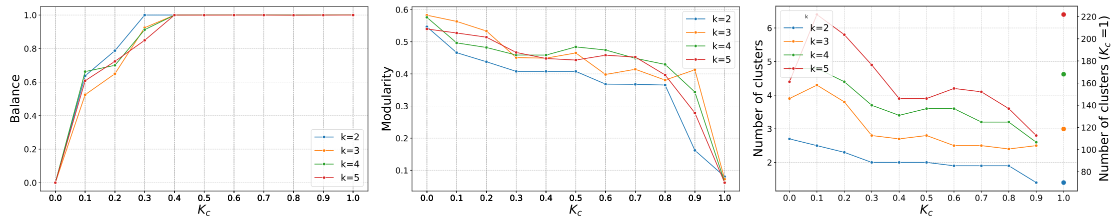
5.3.1 Fairness-Quality trade-off. In our first line of experiments, we aim to understand how the performance of our algorithm changes as we vary the K_c parameter of the algorithm. The parameter K_c determines the effect of the electrostatic force in the selection of a node of the label to adopt, and determines the influence of the fairness criterion in the formation of the communities. The parameter K_c takes values between 0 and 1. The extreme case where $K_c = 0$ corresponds to the vanilla LP algorithm, that does not take fairness considerations into account. The extreme case where $K_c = 1$ makes decisions solely on the account of balance. The values of K_c between 0 and 1, implement a trade-off between finding well-connected communities, and achieving fairness.

In order to study the effect of K_c we use synthetic datasets, for which we have control over the number of planted clusters, the arrangement of the colors on the nodes, and the connectivity between different types of nodes. We set the parameter values as follows: $a = 0.1$,$b = 0.01$, $c = 0.001$, and $d = 0.0001$. The number of groups is 2, and for the number of clusters we use the values $k \in \{2, 3, 4, 5\}$. Each cluster has 400 nodes, 200 of which are red and 200 are blue. The parameter K_c takes values from 0 to 1, in steps of 0.1. We measure fairness, quality, and the number of clusters. Our measurements are averages over 10 different datasets.

Fairness is measured using the balance metric, while the quality of the communities is measured using *modularity* [11, 28]. Modularity is a popular measure of community quality, that measures the divergence between the actual number of intra-community edges, and the expected number of intra-community edges, if edges were generated at random, such that the node degrees are preserved. Specifically, the modularity of a community C, $Q(C)$, is defined as [28]:

$$Q(C) = \frac{1}{2m}\left(\sum_{v,u\in C} A_{uv} - \frac{N_u\, N_v}{2m}\right)$$

where A is the adjacency matrix of G, m the number of edges in G and N_u, N_v the degrees of node u and v respectively. High positive values of modularity indicate well connected communities, while low, or negative values of modularity



(a) Balance

(b) Modularity

(c) Number of Communities

Fig. 1. Fairness-Quality trade-off

sparsely connected communities. The modularity of a collection of communities is the sum of the modularities of the communities. Note that the modularity of the whole dataset is zero.

Figure 1 shows our results. We observe that the vanilla LP algorithm ($K_c = 0$) achieves the highest modularity, but very poor fairness, since the communities created are monochromatic. As the value of K_c increases the fairness of the output improves, reaching the maximum for $K_c \geq 0.4$. At the same time, modularity decreases slowly, up to $K_c = 0.8$, and then drops significantly. For $K_c = 0.9$ the strength of the electrostatic force results in grouping together nodes from different colors even when they belong to different clusters, sometimes in a single community. These communities are sparsely connected, thus bringing modularity down.

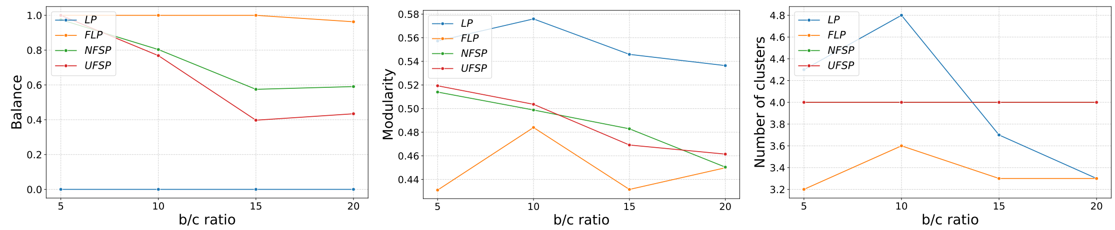
It is interesting to observe what happens in the extreme case when $K_c = 1$. In this case the selection of the label is done based solely on the electrostatic force. This forces bi-chromatic edges to merge form communities. Then communities are built around these bi-chromatic edges, and the number of output communities is usually close to the number of bi-chromatic edges.

Overall, from our results we conclude that a good range for K_c is in $[0.4, 0.7]$, where we strike a good balance between fairness and quality. For the remaining our our experiments, we will use $K_c = 0.5$ as the default value, giving equal weight to the gravitational and electrostatic forces.

5.3.2 Comparison with Fair Spectral on Synthetic Datasets. In the next line of experiments, we evaluate the performance of the FLP algorithm, as we vary the "hardness" of the synthetic datasets we create, and we compare with the fair spectral algorithms. Specifically, we generate synthetic dataset where we vary the ratio b/c of the probability of connecting two nodes u, v, that belong to the different clusters, but are of the same color, over the probability of connecting two nodes that are in the same community but are of different color. We can think the former set of edges as unfair edges that will misguide the algorithm into creating unfair communities, while the latter as fair edges that will guide the algorithm into creating fair communities. The higher the ratio, the harder for an algorithm to bring together nodes of different colors in the same community.

We consider the values $\{5, 10, 15, 20\}$ for the ratio b/c. We control the ratio b/c by varying the parameter c, that is, we use the default parameters $0.1, 0.01, 0.0001$, for the parameters a, b, d, while c takes values $0.002, 0.001, 0.0006, 0.0005$. We set the number of clusters to $k = 4$. We use $K_c = 0.5$ for the FLP algorithm, and we also consider the vanilla LP algorithm. We also ran the spectral algorithms UFSP and NFSP for 4 communities. All the results we report are the averages over 10 different random graphs.

Figure 2 shows our results. We plot the balance, modularity, and number of communities. We observe that our algorithm achieves almost perfect balance for all settings of the ratio b/c. On the other hand, the balance for the spectral



(a) Balance (b) Modularity (c) Number of Communities

Fig. 2. Label Propagation and Spectral algorithms on Synthetic Data: Balance, Modularity, Number of Communities.

algorithms deteriorates as the dataset becomes harder. As bichromatic edges become increasingly rare in comparison to single color edges, the spectral algorithms tend to bring together nodes of the same color and separate nodes of different color, leading to unfairness.

The FLP algorithm has lower modularity than the spectral algorithms. Since the FLP algorithm creates balanced communities, this means that it puts together nodes of different color, which are sparsely connected. On the other hand, the spectral algorithms are more likely to create monochromatic communities, which are better connected, and have higher modularity. As a result modularity for FLP is lower, as expected from the fairness-quality trade-off.

Looking at the number of communities output by the FLP algorithm, we observe that it is on average lower than 4, the number of planted clusters. This indicates that the algorithm sometimes merges different clusters while still preserving balance.

Algorithm	**Balance**	**Modularity**	**#Communities**
Friendship Net			
LP	0.45	0.68	17
FLP	0.60	0.64	14
UFSP	0.66	0.024	14
NFSP	0.53	0.0078	14
Facebook Net			
LP	0.54	0.43	3
FLP	0.72	0.37	5
UFSP	0.80	0.009	5
NFSP	0.81	0.01	5
Political Blogs			
LP	0.05	0.43	11
FLP	0.92	-3.62E-06	2
UFSP	0.92	-3.62E-06	2
NFSP	0.92	0.0006	2

Table 2. Results on multiple datasets

5.3.3 Results on Real Datasets. The last set of experiments is on the real datasets we consider. In Table 2 we show the performance of the different algorithms for the three real datasets we consider. Since we have no ground truth about the "correct" number of communities, we run the spectral algorithms for the same number of communities as those output by the FLP algorithm.

Interestingly, each dataset exhibits a different behavior. For the *Political Blogs* dataset we observe that all fair algorithms achieve balance equal to the network balance. This is due to the fact that all algorithms form a giant community that contains almost all nodes, and a second very small one, This is also indicated by the fact that modularity is close to zero (which is the case when we have a single community). In the *Political Blogs* graph the red and blue groups are highly segregated, so it is hard to form meaningfully connected communities that are balanced. The result is that the algorithms merge almost all the nodes in a single community.

For the *Facebook* dataset, we observe that the spectral algorithms perform better than FLP in terms of fairness, achieving a better balance. However, the modularity of the resulting communities is close to zero, indicating very poor connectivity between the nodes in the community. The FLP algorithm achieves modularity competitive to that of the LP algorithm, while improving significantly the balance of the communities compared to LP. Therefore, it strikes a better balance between fairness and quality in the output.

Finally, we observe a similar behavior for the *Friendship* dataset. In this case, the FLP algorithm achieves the second-best balance, between the UFSP algorithm and the NFSP algorithm. It is again considerably better in terms of modularity, indicating better connected communities.

6 Discussion and Conclusion

In this paper we consider the problem of fairness for the problem of community detection on graphs. We consider balance as the fairness criterion for communities, and we propose an algorithm for creating well-connected and fair communities. Our algorithm is inspired by Physics principles, and it modifies the popular label propagation algorithm, introducing fairness in the propagation. We evaluate our algorithm experimentally, and we compare against fair spectral algorithms. We show that our algorithm is able to achieve a good trade-off between balance and quality.

For future work, it would be interesting to extend our algorithm to handle sensitive attributes with more than two values. Furthermore, we want to better exploit the distance in the definition of the gravitational and electrostatic forces, which is currently simply set to 1 for connected nodes. Finally, it would be interesting to explore if a variation of our algorithm could be applicable to numerical data, where there is no graph structure.

Acknowledgments: This work has been implemented within the framework of the H.F.R.I call “Basic Research Financing" (H.F.R.I. Project Number: 016636), under the National Recovery and Resilience Plan "Greece 2.0," funded by the European Union - NextGenerationEU.

References

[1] Aris Anagnostopoulos, Luca Becchetti, Adriano Fazzone, Cristina Menghini, and Chris Schwiegelshohn. 2020. Spectral Relaxations and Fair Densest Subgraphs. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (Eds.). ACM, 35–44. https://doi.org/10.1145/3340531.3412036

[2] Nihesh Anderson, Suman K. Bera, Syamantak Das, and Yang Liu. 2020. Distributional Individual Fairness in Clustering. arXiv:2006.12589 [cs.LG] https://arxiv.org/abs/2006.12589

[3] Arturs Backurs, Piotr Indyk, Krzysztof Onak, Baruch Schieber, Ali Vakilian, and Tal Wagner. 2019. Scalable Fair Clustering. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 405–413. https://proceedings.mlr.press/v97/backurs19a.html

[4] Solon Barocas, Moritz Hardt, and Arvind Narayanan. 2023. *Fairness and Machine Learning: Limitations and Opportunities.* MIT Press.

[5] Suman Bera, Deeparnab Chakrabarty, Nicolas Flores, and Maryam Negahbani. 2019. Fair Algorithms for Clustering. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2019/file/fc192b0c0d270dbf41870a63a8c76c2f-Paper.pdf

[6] Brian Brubach, Darshan Chakrabarti, John Dickerson, Samir Khuller, Aravind Srinivasan, and Leonidas Tsepenekas. 2020. A pairwise fair and community-preserving approach to k-center clustering. In *International conference on machine learning*. PMLR, 1178–1189.

[7] Darshan Chakrabarti, John P. Dickerson, Seyed A. Esmaeili, Aravind Srinivasan, and Leonidas Tsepenekas. 2022. A New Notion of Individually Fair Clustering: α-Equitable k-Center. arXiv:2106.05423 [cs.LG] https://arxiv.org/abs/2106.05423

[8] Xingyu Chen, Brandon Fain, Liang Lyu, and Kamesh Munagala. 2019. Proportionally Fair Clustering. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 1032–1041. https://proceedings.mlr.press/v97/chen19d.html

[9] Anshuman Chhabra, Karina Masalkovaitė, and Prasant Mohapatra. 2021. An overview of fairness in clustering. *IEEE Access* 9 (2021), 130698–130720.

[10] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. 2017. Fair clustering through fairlets. *Advances in neural information processing systems* 30 (2017).

[11] A. Clauset, M. E. J. Newman, and C. Moore. 2004. Finding community structure in very large networks. *Phys. Rev. E* 70 (2004). Issue 6.

[12] Gennaro Cordasco and Luisa Gargano. 2010. Community detection via semi-synchronous label propagation algorithms. In *2010 IEEE International Workshop on: Business Applications of Social Network Analysis (BASNA)*. 1–8. https://doi.org/10.1109/BASNA.2010.5730298

[13] Yushun Dong, Jing Ma, Song Wang, Chen Chen, and Jundong Li. 2023. Fairness in Graph Mining: A Survey. *IEEE Transactions on Knowledge and Data Engineering* 35, 10 (2023), 10583–10602. https://doi.org/10.1109/TKDE.2023.3265598

[14] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference* (Cambridge, Massachusetts) *(ITCS '12)*. Association for Computing Machinery, New York, NY, USA, 214–226. https://doi.org/10.1145/2090236.2090255

[15] D. Easley and J. Kleinberg. 2010. *Networks, Crowds, and Markets: Reasoning about a Highly Connected World*. Cambridge University Press. https://books.google.gr/books?id=atfCl2agdi8C

[16] Santo Fortunato. 2010. Community detection in graphs. *Physics Reports* 486, 3–5 (Feb. 2010), 75–174. https://doi.org/10.1016/j.physrep.2009.11.002

[17] Mehrdad Ghadiri, Samira Samadi, and Santosh Vempala. 2021. Socially Fair k-Means Clustering. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (Virtual Event, Canada) *(FAccT '21)*. Association for Computing Machinery, New York, NY, USA, 438–448. https://doi.org/10.1145/3442188.3445906

[18] Christos Gkartzios, Evaggelia Pitoura, and Panayiotis Tsaparas. 2025. Fair Network Communities through Group Modularity. In *Proceedings of the ACM Web Conference 2025 (WWW '25)* (Sydney, NSW, Australia). ACM. https://doi.org/10.1145/3696410.3714625

[19] Shubham Gupta and Ambedkar Dukkipati. 2021. Protecting individual interests across clusters: Spectral clustering with guarantees. *CoRR* (2021).

[20] Shubham Gupta and Ambedkar Dukkipati. 2022. Consistency of constrained spectral clustering under graph induced fair planted partitions. *Advances in Neural Information Processing Systems* 35 (2022), 13527–13540.

[21] L. Hagen and A.B. Kahng. 1992. New spectral methods for ratio cut partitioning and clustering. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 11, 9 (1992), 1074–1085. https://doi.org/10.1109/43.159993

[22] Matthäus Kleindessner, Pranjal Awasthi, and Jamie Morgenstern. 2020. A Notion of Individual Fairness for Clustering. arXiv:2006.04960 [stat.ML] https://arxiv.org/abs/2006.04960

[23] Matthäus Kleindessner, Samira Samadi, Pranjal Awasthi, and Jamie Morgenstern. 2019. Guarantees for spectral clustering with fairness constraints. In *International conference on machine learning*. PMLR, 3458–3467.

[24] Xin Liu and Tsuyoshi Murata. 2009. How Does Label Propagation Algorithm Work in Bipartite Networks?. In *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, Vol. 3. 5–8. https://doi.org/10.1109/WI-IAT.2009.217

[25] Konstantinos Manolis and Evaggelia Pitoura. 2023. Modularity-Based Fairness in Community Detection. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining*. 126–130.

[26] Farzan Masrour, Tyler Wilson, Heng Yan, Pang-Ning Tan, and Abdol Esfahanian. 2020. Bursting the filter bubble: Fairness-aware network link prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 841–848.

[27] Rossana Mastrandrea, Julie Fournet, and Alain Barrat. 2015. Contact Patterns in a High School: A Comparison between Data Collected Using Wearable Sensors, Contact Diaries and Friendship Surveys. *PLOS ONE* 10, 9 (09 2015), 1–26. https://doi.org/10.1371/journal.pone.0136497

[28] M. E. J. Newman. 2004. Fast algorithm for detecting community structure in networks. *Phys. Rev. E* 69 (2004).

[29] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. 2007. Near linear time algorithm to detect community structures in large-scale networks. *Phys. Rev. E* 76 (Sep 2007), 036106. Issue 3. https://doi.org/10.1103/PhysRevE.76.036106

[30] Clemens Rösner and Melanie Schmidt. 2018. Privacy preserving clustering with constraints. arXiv:1802.02497 [cs.CC] https://arxiv.org/abs/1802.02497

[31] Jianbo Shi and J. Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, 8 (2000), 888–905. https://doi.org/10.1109/34.868688

[32] Ana-Andreea Stoica and Augustin Chaintreau. 2019. Fairness in Social Influence Maximization. In *Companion Proceedings of The 2019 World Wide Web Conference* (San Francisco, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 569–574. https://doi.org/10.1145/3308560.3317588

[33] Alan Tsang, Bryan Wilder, Eric Rice, Milind Tambe, and Yair Zick. 2019. Group-Fairness in Influence Maximization. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 5997–6005. https://doi.org/10.24963/ijcai.2019/831

[34] Sotiris Tsioutsiouliklis, Evaggelia Pitoura, Konstantinos Semertzidis, and Panayiotis Tsaparas. 2022. Link Recommendations for PageRank Fairness. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 3541–3551. https://doi.org/10.1145/3485447.3512249

[35] Sotiris Tsioutsiouliklis, Evaggelia Pitoura, Panayiotis Tsaparas, Ilias Kleftakis, and Nikos Mamoulis. 2021. Fairness-Aware PageRank. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 3815–3826. https://doi.org/10.1145/3442381.3450065

[36] Ji Wang, Ding Lu, Ian Davidson, and Zhaojun Bai. 2023. Scalable spectral clustering with group fairness constraints. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 6613–6629.