A Fair Label Propagation Community Detection Algorithm

Glykeria Toulina and Panayiotis Tsaparas

Department of Computer Science & Engineering, University of Ioannina, Ioannina 45110, Greece,
{gtoulina, tsap}@uoi.gr

Abstract. Growing concerns about bias and discrimination in automated systems have spurred a surge of research in algorithmic fairness, which aims to design algorithms with formal fairness guarantees. In this work, we focus on fairness in *community detection*, the task of identifying cohesive subsets of nodes (communities) that are sparsely interconnected. We assume that nodes are partitioned into groups, based on a sensitive attribute (e.g., gender), and aim for *balanced* representation of these groups within the detected communities. We propose a novel fair community detection algorithm that builds on the popular Label Propagation method. Our approach draws inspiration from principles in physics to incorporate fairness into the label propagation process. We present experiments on different real and synthetic datasets, where we study the properties of our algorithm and compare with baselines.

Keywords: community detection, fairness, balance, label propagation

1 Introduction

The widespread use of Machine Learning and Data Science algorithms has raised concerns about possible biases of these algorithms against minorities and under-represented groups. Consequently, in recent years, *algorithmic fairness* has emerged as a key area of research, aiming to design algorithms with formal fairness guarantees. In this work, we focus on fairness for the problem of *community detection* in networks. At a high level, community detection aims to partition the nodes of a graph into subsets (communities), such that nodes are densely connected within the communities, while sparsely connected across communities [8]. Our objective is to design a community detection algorithm that produces *fair* communities, where fairness is measured using the *balance* metric [5]. We assume that nodes are partitioned into groups, based on a sensitive attribute (e.g., the gender of users in a social network), and we ask for a balanced representation of the different groups in the output communities. Balance is a commonly used metric of fairness in clustering [4], which has also been employed in community detection [11].
A popular community detection algorithm is the Label Propagation (LP) algorithm [14], which finds communities by propagating labels along the edges of the graph. In this paper, we propose a fair variant of the LP

algorithm. We adopt a physics-inspired perspective on label propagation, modeling each neighbor as exerting a force that pulls the node to its community. To incorporate fairness, we introduce the notion of an electrostatic charge for a community, defined as the imbalance of the different groups in the community. We define an electrostatic force between the community and the node, which is attractive or repelling, depending on whether the node improves the balance of the community. Our algorithm combines both pulling and electrostatic forces to perform the label propagation, providing a trade-off between community quality and fairness.
In summary, in this paper, we make the following contributions:

- We propose a novel fair community detection algorithm that extends the label propagation algorithm using physics-inspired principles.
- We experimentally assess the fairness–quality trade-off on real and synthetic datasets, showing that our algorithm improves fairness with minimal impact on community quality.
- We compare our algorithm to the fair spectral method (FSP) on real and synthetic datasets, showing improved performance – especially on "difficult" datasets.

2 Related Work

The problem of community detection is a special case of clustering for graph data. There has been considerable amount of work on clustering fairness [4]. The definition of fairness we use is that of *balance*, first introduced in the seminal work about *fairlets* [5]. The work in [5] was followed by several extensions and modifications [2,3,10] that consider variants of the original problem.
There is limited amount of work on community detection fairness. The work most related to ours is that in [11,17], 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 [9] they define a notion of edge fairness for communities, using the modularity metric. 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, using a fairness metric similar to balance. The densest subgraph problem is related to community detection, but has a different objective.

3 Preliminaries

In this section, we introduce the problem of community detection and the Label Propagation algorithm, and the definition of balance.

Community Detection and the Label Propagation algorithm: Given as input a network $G = (V, E)$, the output of a community detection algorithm is a partition of the nodes into k disjoint subsets (communities), $\mathcal{C} = \{C_1, C_2, ..C_k\}$, $C_i \subseteq V$, $C_i \cap C_j = \emptyset$, $\cup_{i=1}^{k} C_i = V$. The

number of communities k may be given as input, or it may be decided by the algorithm. There is a variety of community detection algorithms that use different criteria to produce communities [8].
The *Label Propagation* algorithm (LP), is a popular algorithm for community detection [14].The algorithm starts by assigning each node v a unique label $L(v)$, usually the id of the node. Then it iteratively updates the label of each node, assigning the most frequent label in their neighborhood. If there are ties, the label with the largest id is selected. The algorithm terminates when no node changes label. A community $C_\ell = \{v \in V : L(v) = \ell\}$ is the set of nodes with the same label ℓ.

Balance: To define fairness, we assume that the nodes of the graph are associated with some sensitive attribute A, such as gender, religion or race, that takes t values $\{a_1, ..., a_t\}$, which partition the nodes of the graph into t *groups* $G = \{G_1, G_2, .., G_t\}$, $G_i = \{v \in V : A(v) = a_i\}$. In the following, we will often refer to the attribute values, and the corresponding groups, as *colors*.
The *balance* fairness metric was first defined in the work of Chierichetti et al. [5] for fair 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

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

A perfectly balanced community has equal number of red and blue nodes, resulting in a balance value of 1.
Given the definition of the balance of a community, the balance of a collection of communities $\mathcal{C} = \{C_1, \ldots, C_k\}$ is defined as as the average balance of the communities in $\mathcal{C}$, that is, $bal(\mathcal{C}) = \frac{1}{|\mathcal{C}|}\sum_{C_i \in \mathcal{C}} bal(C_i)$.

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 described in Section 3. For the definition of our algorithm, we take a physics-inspired view of the LP algorithm. We assume that adjacent nodes (u, v) in the graph pull each other with force $F_p(u, v) = 1$. In an iteration of the LP algorithm, if $N_\ell(v)$ is the set of neighbors of v with label ℓ, then the community C_ℓ pulls node v with force $F_p(C_\ell, v) = |N_\ell(v)|$. 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.
To incorporate fairness (balance) in the LP algorithm, we introduce an additional *electrostatic* force between connected nodes. We assign to each node v a *charge* q_v, whose magnitude depends on the *imbalance* of the community node v belongs to. We define the imbalance of a community C as $imb(C) = 1 - bal(C)$, and set $|q_v| = imb(C)$, for the nodes in 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 ($|C^r| > |C^b|$), 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 ($|C^b| > |C^r|$, community C is a blue community), then the charge is negative. Note that a single red 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 with charges q_1 and q_2 at distance d, is governed by Coulomb's law and it has magnitude $|F_e(q_1, q_2)| = K_c \frac{|q_1||q_2|}{d^2}$. 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) exert electrostatic force to each other. The force is directionless, so we only care about the sign and the magnitude of the force. We assume that the nodes are at distance 1, and we set $K_c = 1$. Therefore, we have: $F_e(u, v) = -q_u q_v$.
When considering the label assignment of a node v, the FLP algorithm, computes the electrostatic force that the neighbors of v in the community C_ℓ exert to the node. The charge of the node v has magnitude $|q_v| = 1$, while the charge of a neighbor u in the C_ℓ community has magnitude $|q_u| = imb(C_\ell)$. Summing over $N_\ell(v)$, the neighbors of v in C_ℓ, the electrostatic force of community C_ℓ to node v is:

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

where $\text{sign}(v, C_\ell) = +1$ if v and C_ℓ are of the same color, and $\text{sign}(v, C_\ell) = -1$ 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 C_ℓ has a surplus of the opposite color of v, and adding v to the community will improve its balance. A repellent (negative) force means that the community has a surplus of the color of v, and adding v to the community will further increase the imbalance of the community. The strongest the force, the more unbalanced the community (positively or negatively).
The FLP algorithm computes the total force $F(C_\ell, v)$ that the community C_ℓ exerts on node v, by combining the pulling and electrostatic forces, that is:

$$F(C_\ell, v) = (1 - \lambda)F_p(C_\ell, v) + \lambda F_e(C_\ell, v),$$

where λ is a parameter of the algorithm. By combining the two forces, the algorithm aims to combine two objectives: The quality of the output communities, which is achieved by the pulling force, as in the vanilla LP algorithm, and the fairness of the output communities, which is achieved by the electrostatic force. The parameter λ controls the tradeoff between community quality and fairness. Higher values of λ place more emphasis on fairness. The value $\lambda = 0$ corresponds to the vanilla LP algorithm, while the case $\lambda = 1$ puts all the emphasis on fairness.
Putting everything together, the FLP algorithm operates exactly as the vanilla LP algorithm, iteratively updating the labels of the nodes, each time assigning a node v to the community C_ℓ that exerts the strongest force $F(C_\ell, v)$. The outline of the algorithm is shown in Algorithm 1. The algorithm complexity remains $O(Im)$, with m edges, and I iterations.

Algorithm 1 Fair Label Propagation (FLP)

```
Require: Graph $G = (V, E)$, group partition $\{G_b, G_r\}$, parameter $\lambda$.
 1: Assign a unique label $L(v) = v$ to every node $v \in V$.
 2: repeat
 3:     for all $v \in V$ do
 4:         for all labels $\ell$ do
 5:             $N_\ell(v) = \{u \in N(v) : L(u) = \ell\}$
 6:             $F_p(C_\ell, v) = |N_\ell(v)|$; $F_e(C_\ell, v) = \text{sign}(v, C_\ell)|N_\ell(v)|imb(C_\ell)$
 7:             $F(C_\ell, v) = (1 - \lambda)F_p(C_\ell, v) + \lambda F_e(C_\ell, v)$
 8:         end for
 9:         $L(v) = \arg\max_\ell F(C_\ell, v)$
10:     end for
11: until No label change
```

5 Experiments

The goal of the experiments is two-fold: Explore the fairness–quality trade-off, by varying the parameter λ of the FLP algorithm; Compare against baselines and assess how dataset characteristics affect performance in terms of fairness and clustering quality.

5.1 Experimental Setup

Datasets: In our experiments, we use both real and synthetic datasets. We use the following real datasets:

- *Twitter* [15]: A collection of Twitter users, and their retweet actions.
- *Facebook* [12]: The mutual friendships between a collection of Facebook users, extracted from their ego networks.
- *Deezer* [16]: The mutual follow relationships between a collection of users on Deezer, an online music platform.
- *DrugNet* [18]: The acquaintance relationships between a collection of drug users in Hartford.
- *Friendship* [13]: The mutual friendships between students in a high school in Marseilles, France.

The sensitive attribute is the political affiliation, for the *Twitter* network, and the gender, for all other datasets. The characteristics of the datasets, and the balance, $bal(G)$, of the whole network are shown in Table 1.

We also use synthetic datasets, generated by a variant of the stochastic block model, defined in [11]. The model assumes that the nodes are partitioned into k planted *clusters*, $T = \{T_1, ..., T_k\}$, and two groups $G = \{G_r, G_b\}$. The model is defined by four parameters: a, b, c, and d that determine the probability $\Pr(u, v)$ of an edge between two nodes u, v, depending on the group and cluster membership. Specifically: $\Pr(u, v) = a$, if u, v belong to the same cluster, and the same group; $\Pr(u, v) = b$, if u, v belong to different clusters, but the same group; $\Pr(u, v) = c$, if u, v belong to the same cluster, but in different groups; $\Pr(u, v) = c$, if u, v belong to the different clusters, and different groups.

Table 1: Real Dataset Characteristics

Dataset	**Nodes**	**Edges**	**Groups**	**bal(G)**
Twitter	18,470	48,053	affiliation	0.63
Facebook	4,039	88,234	gender	0.61
Deezer	28,281	92,752	gender	0.80
DrugNet	185	265	gender	0.27
Friendship	127	396	gender	0.67

We have $a > b > c > d$. The datasets are constructed so that traditional algorithms tend to produce monochromatic communities, by separating nodes from different groups. The goal is to study if fair community detection algorithms can generate fair (balanced) communities, ideally by recovering the planted clusters.

Algorithms: In our implementation of the FLP algorithm, we perform semi-synchronous updates of the node labels, adopting the update scheme in [7]. We also perform random permutations of the initial labels of the nodes, to deal with the randomness in the node ordering. We implemented FLP by adapting the open-source implementation provided by the NetworkX library.[1] Our code is publicly available.[2]
We compare the FLP algorithm against the Fair Spectral algorithm (FSP) introduced in [11]. The algorithm incorporates the fairness constraints in the Laplacian matrix. It computes the eigenvectors of the k smallest eigenvalues, and performs k-means clustering on the resulting vectors. In our implementation we use the scalable variant of the algorithm [17], and we employ k-means++ for the clustering step.

Fairness and Quality metrics: We evaluate the output communities in terms of both fairness and quality. Fairness is measured using the balance metric defined in Equation 1. The quality of the communities is measured using *modularity* [6]. Modularity is a popular measure of community quality, that measures the divergence between the actual and the expected number of intra-community edges, if edges were generated at random such that the expected node degrees are preserved. Specifically, the modularity $Q(C)$ of a community C is:

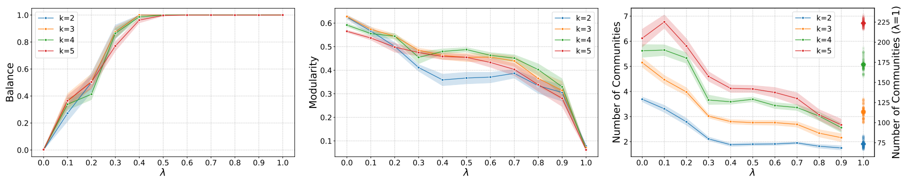
$$Q(C) = \frac{1}{2m}\left(\sum_{u,v\in C} A_{uv} - \frac{|N(u)||N(v)|}{2m}\right),$$

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. The higher the modularity, the more cohesive the community. The modularity of a collection of communities is the sum of the modularities of the communities. Note that the modularity of the whole graph is zero.

[1] `https://networkx.org/documentation/stable/_modules/networkx/algorithms/community/label_propagation.html`
[2] `https://github.com/elidek-themis/flp`

5.2 Experimental results

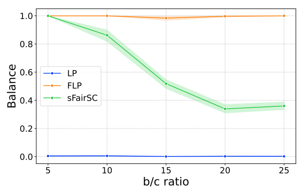
Fairness-Quality trade-off: We first study the effect of the parameter λ on the performance of our algorithm. The parameter λ controls the contribution of the electrostatic force in the label assignment, and the influence of the fairness criterion on the community formation process. The values of λ between 0 and 1 implement a trade-off between finding well-connected communities, and achieving fairness.
To study the effect of λ, we use synthetic datasets, for which we have control over the dataset characteristics. 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 (red/blue), 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 λ 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, and 10 different label permutations per dataset.



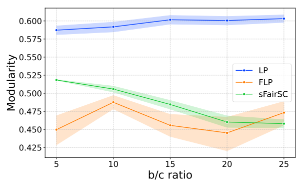
(a) Balance (b) Modularity (c) Number of Communities

Fig. 1: Fairness-Quality trade-off

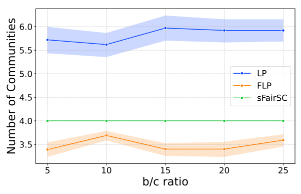
Figure 1 shows our results. We observe that the vanilla LP algorithm ($\lambda = 0$) achieves the highest modularity, but very poor fairness, since the communities created are monochromatic. As the value of λ increases the fairness of the output improves, reaching the maximum for $\lambda \geq 0.4$. At the same time, modularity decreases slowly, up to $\lambda = 0.8$, and then drops sharply. For $\lambda = 0.9$, the strength of the electrostatic force results in grouping together nodes from different colors even when they belong to different clusters. These communities are sparsely connected, thus bringing modularity down. Sometimes, a single community is output, with modularity zero.
In the extreme case where $\lambda = 1$, the label selection is based solely on the electrostatic force. This forces bi-chromatic edges to merge, and same-color edges to split. Communities are built around bi-chromatic edges, resulting in a large number of communities, as shown in the right axis of Figure 1c, usually close to the number of bi-chromatic edges.
Overall, we observe that for λ in $[0.4, 0.7]$ we strike a good balance between fairness and quality. For the following, we use $\lambda = 0.5$ as the default value.



(a) Balance



(b) Modularity



(c) Number of Communities

Fig. 2: Comparison on synthetic datasets of different hardness.

Comparison on Synthetic Datasets: We now evaluate the performance of the FLP algorithm, as we vary the "hardness" of the synthetic datasets, and we compare with the fair spectral algorithm (FSP). Specifically, we generate synthetic datasets where we vary the ratio b/c of the probability of connecting two nodes of the same color, that belong to different clusters, over the probability of connecting two nodes of different color that are in the same cluster. We think of the former edges as impeding the creation of fair communities, while the latter as facilitating the creation of fair communities. The higher the ratio, the harder for an algorithm to bring together nodes of different color in the same community.
We consider the values $\{5, 10, 15, 20, 25\}$ for the ratio b/c. We vary the ratio b/c by varying the parameter c. We use the default values for the parameters a, b, d $(0.1, 0.01, 0.0001)$, and we vary the parameter c to take values 0.002, 0.001, 0.0006, 0.0005, 0.0004. We set the number of clusters to $k = 4$. We use $\lambda = 0.5$ for the FLP algorithm, and we also consider the vanilla LP algorithm. Again, we report averages over 10 different random graphs, and 10 label permutations per graph.
Figure 2 shows our results. 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 algorithm deteriorates as the dataset becomes harder. As bi-chromatic edges become increasingly rare in comparison to single color edges, the FSP algorithm tends 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 algorithm. This is expected from the fairness-quality trade-off, since the FLP algorithm brings together nodes of different color, which are sparsely connected, while the FSP algorithm is more likely to create monochromatic communities, which have higher modularity.
We observe that the FLP algorithm on average outputs less than 4 communities, the number of planted clusters. This indicates that the FLP algorithm is more aggressive in merging clusters to preserve balance.

Results on Real Datasets: Finally, we compare our algorithms on the real datasets. Table 2 shows the results. For the Label Propagation algorithms the results are the averages of 10 different runs with different label permutations. Since we have no ground truth for the "correct" number of communities, we also perform 10 runs of the FSP algorithm

Table 2: Algorithm results on real datasets

Dataset	Algorithm	Balance	Modularity	Communities
Twitter	LP	0.010	**0.549**	1189.3
	FLP	**0.579**	0.040	69.0
	FSP	0.018	0.083	69.0
Facebook	LP	0.460	0.734	38.2
	FLP	**0.654**	**0.798**	66.5
	FSP	0.563	0.596	66.5
Deezer	LP	0.479	**0.469**	1697.8
	FLP	**0.767**	0.462	1728.5
	FSP	0.493	0.452	1728.5
DrugNet	LP	0.218	0.602	43.0
	FLP	**0.367**	0.656	17.2
	FSP	0.345	**0.683**	17.2
Friendship	LP	0.390	**0.662**	16.1
	FLP	**0.632**	0.659	13.0
	FSP	0.398	0.659	13.0

with the number of communities output by the FLP algorithm in the corresponding run, and report average values.
The first observation is that the FLP algorithm performs best in terms of balance on all datasets, sometimes being the clear winner (e.g., on the *Friendship* or *Deezer* networks). It is also competitive in terms of modularity: It has the best modularity for *Facebook* and close to the best in all datasets except for *Twitter*. Perhaps surprisingly, the LP algorithm does not always yield the best modularity, although its modularity is close to the best. This indicates that the fair algorithms are able to maintain community quality, while achieving fairness.
For the *Twitter* dataset, the FLP algorithm typically produces a very large community, and several smaller ones, yielding high balance, but low modularity. A similar behavior is also observed for the FSP algorithm, however, in this case, the small communities are monochromatic, bringing down the average balance.

6 Conclusion

In this paper, we addressed fairness in community detection by proposing a physics-inspired fair variant of the Label Propagation algorithm. Experiments on real and synthetic datasets show that our method effectively identifies fair, high-quality communities, and outperforms existing baselines. As future work, we plan to explore alternative formulations of the pulling and electrostatic forces, as well as extensions of our algorithm for vector datasets.

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. Anagnostopoulos, A., Becchetti, L., Fazzone, A., Menghini, C., Schwiegelshohn, C.: Spectral relaxations and fair densest subgraphs. In: ACM CIKM (2020)
2. Backurs, A., Indyk, P., Onak, K., Schieber, B., Vakilian, A., Wagner, T.: Scalable fair clustering. In: ICML (2019)
3. Bera, S.K., Chakrabarty, D., Flores, N.J., Negahbani, M.: Fair algorithms for clustering. In: NeurIPS (2019)
4. Chhabra, A., Masalkovaitė, K., Mohapatra, P.: An overview of fairness in clustering. IEEE Access **9** (2021)
5. Chierichetti, F., Kumar, R., Lattanzi, S., Vassilvitskii, S.: Fair clustering through fairlets. NeurIPS (2017)
6. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Phys. Rev. E **70** (2004)
7. Cordasco, G., Gargano, L.: Community detection via semi-synchronous label propagation algorithms. In: BASNA (2010)
8. Fortunato, S.: Community detection in graphs. Physics Reports **486**(3–5) (2010)
9. Gkartzios, C., Pitoura, E., Tsaparas, P.: Fair network communities through group modularity. In: WWW (2025)
10. Gupta, S., Dukkipati, A.: Consistency of constrained spectral clustering under graph induced fair planted partitions. In: NeurIPS (2022)
11. Kleindessner, M., Samadi, S., Awasthi, P., Morgenstern, J.: Guarantees for spectral clustering with fairness constraints. In: ICML (2019)
12. Leskovec, J., Mcauley, J.: Learning to discover social circles in ego networks. In: NeurIPS (2012)
13. Mastrandrea, R., Fournet, J., Barrat, A.: Contact patterns in a high school: A comparison between data collected using wearable sensors, contact diaries and friendship surveys. PLOS ONE **10**(9) (2015)
14. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E **76** (2007)
15. Rossi, R.A., Ahmed, N.K.: The network data repository with interactive graph analytics and visualization. In: AAAI (2015)
16. Rozemberczki, B., Sarkar, R.: Characteristic Functions on Graphs: Birds of a Feather, from Statistical Descriptors to Parametric Models. In: ACM CIKM (2020)
17. Wang, J., Lu, D., Davidson, I., Bai, Z.: Scalable spectral clustering with group fairness constraints. In: AISTATS (2023)
18. Weeks, M.R., Clair, S., Borgatti, S.P., Radda, K., Schensul, J.J.: Social networks of drug users in high-risk sites: Finding the connections. AIDS and Behavior **6**(2) (2002)