CIKM 2021 1-5 NOVEMBER

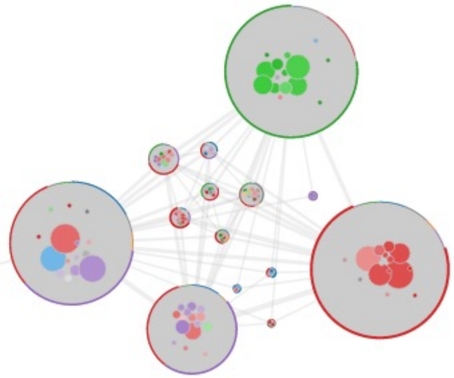
Fair Graph Mining

Jian Kang **Hanghang Tong**

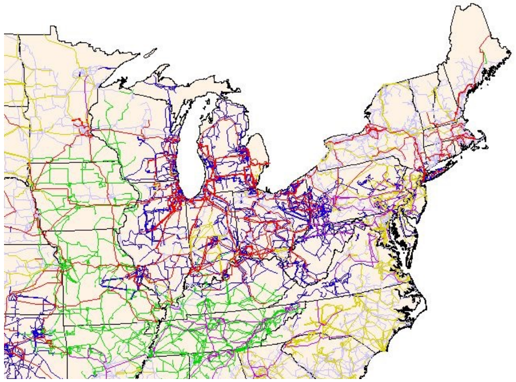
University of Illinois at Urbana-Champaign

ILLINOIS

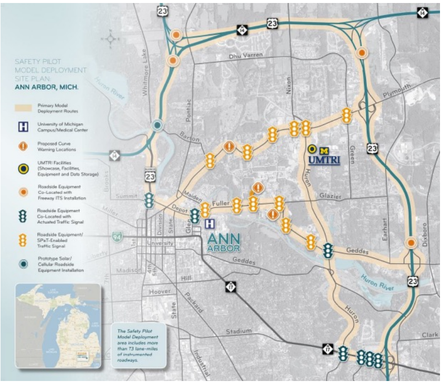
Networks and Graphs are Everywhere



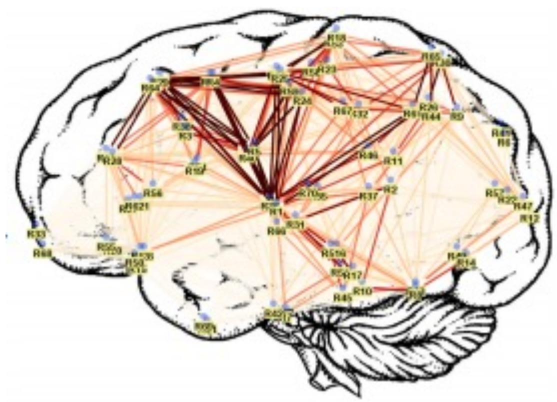
Collaboration Networks



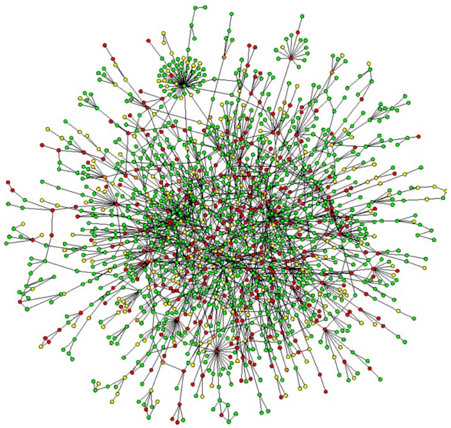
US Power Grid



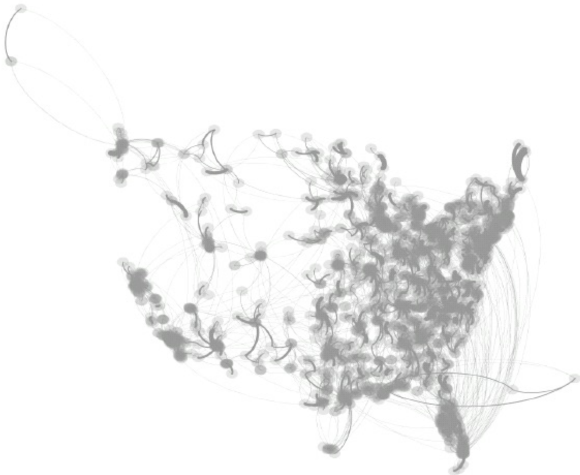
Traffic Network



Brain Networks



Biological Networks

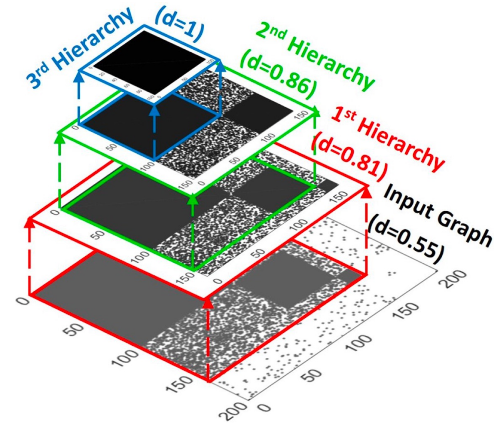


Hospital Networks

This Talk: Graphs = Networks

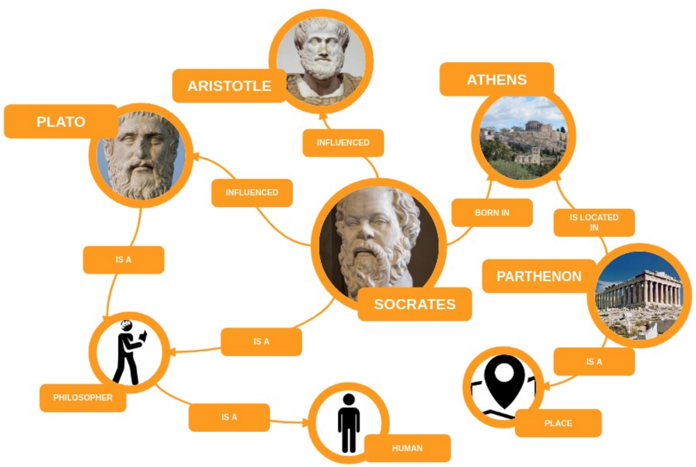
Graph Mining is Widely-Applied

Social science



Finance

Biology

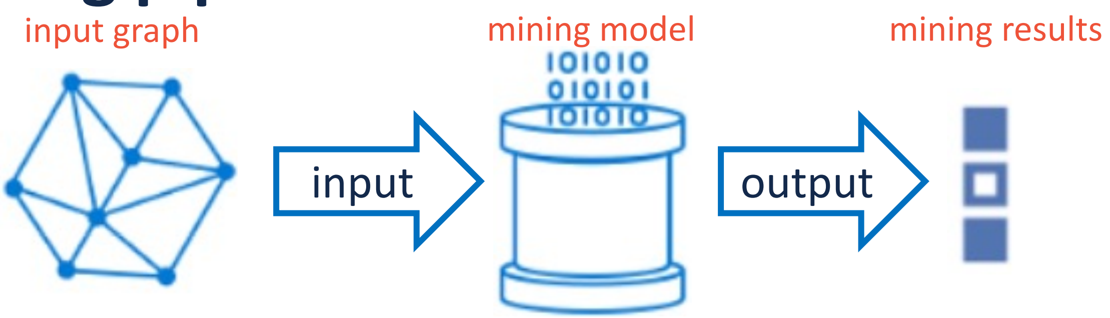


Cognitive science

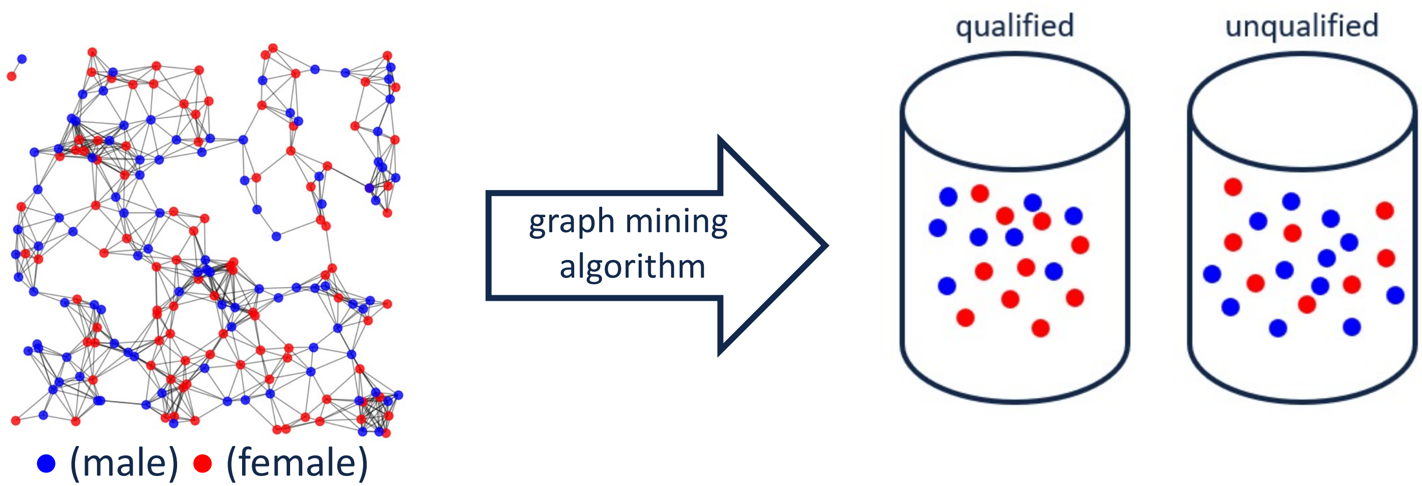
[1] Borgatti, S. P., Mehra, A., Brass, D. J., & Labianca, G.. Network Analysis in the Social Sciences. Science 2009.
[2] Zhang, S., Zhou, D., Yildirim, M. Y., Alcorn, S., He, J., Davulcu, H., & Tong, H.. Hidden: Hierarchical Dense Subgraph Detection with Application to Financial Fraud Detection. SDM 2017.
[3] Wang, S., He, L., Cao, B., Lu, C. T., Yu, P. S., & Ragin, A. B.. Structural Deep Brain Network Mining. KDD 2017.
[4] Ding, M., Zhou, C., Chen, Q., Yang, H., & Tang, J.. Cognitive Graph for Multi-Hop Reading Comprehension at Scale. ACL 2019.

Graph Mining: Pipeline

- **Graph mining pipeline**

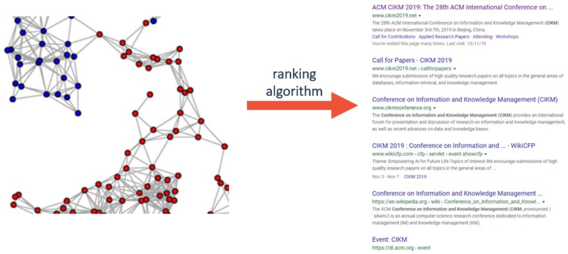


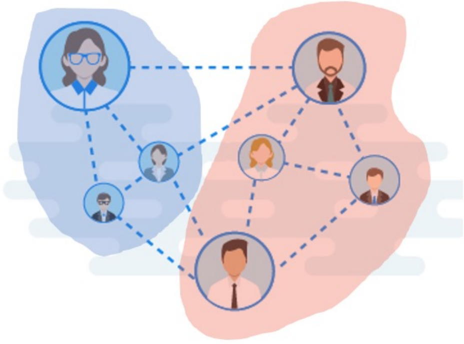
- **Example:** Graduate college admission

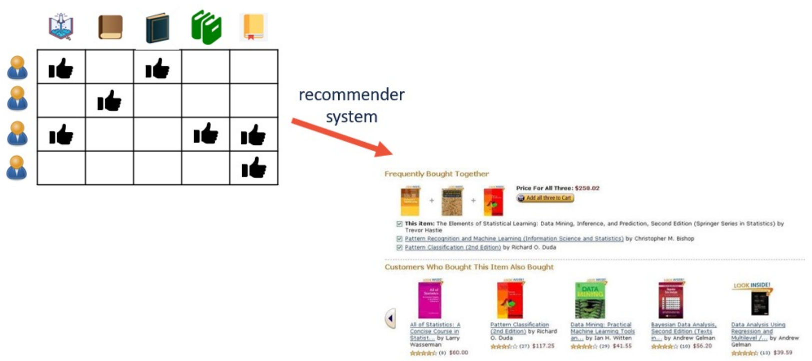


Network Mining: The Who & What Questions

- **Who** are in the same online community?
- **Who** is the key to bridge two academic areas?
- **Who** is the master criminal mind?
- **Who** started a misinformation campaign?
- **Which** items shall we recommend to a user?
- **Which** gene is most relevant to a given disease?
- **Which** webpage is most important?
- **Which** tweet is likely to go viral?
- **Which** transaction looks suspicious?
- …

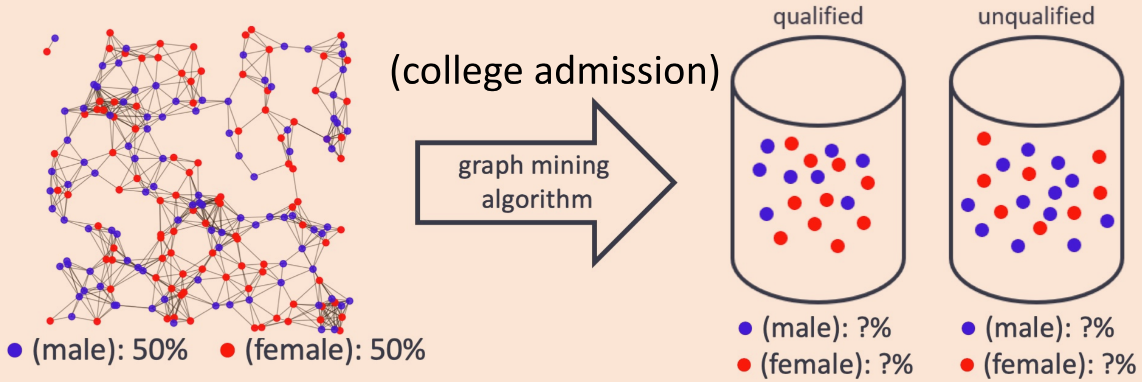






Network Mining: The Why & How Questions

- **How** to ensure the mining is fair?

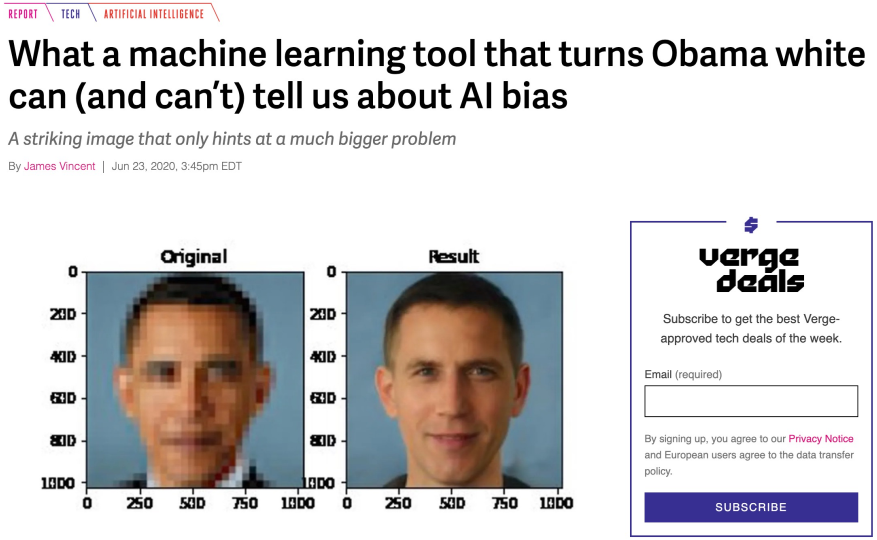


This Tutorial

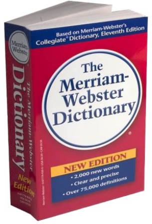
- **Why** are two seemingly different users in the same community?
- **Why** is a particular tweet more likely to go viral than another?
- **Why** does the algorithm 'think' a transaction looks suspicious?
- **How** does an influential researcher bridge two areas?
- **How** do fake reviews skew the recommendation results?
- **How** do the mining results relate to the input graph topology?

Algorithmic Fairness in Machine Learning

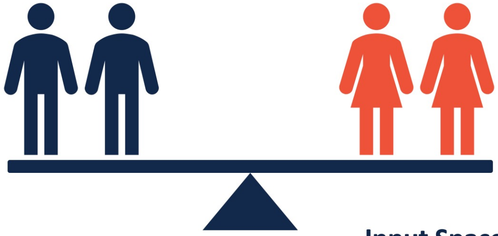
- **Motivation:** Mitigate unintentional bias caused by machine learning (ML) algorithms
- **Examples of discrimination**

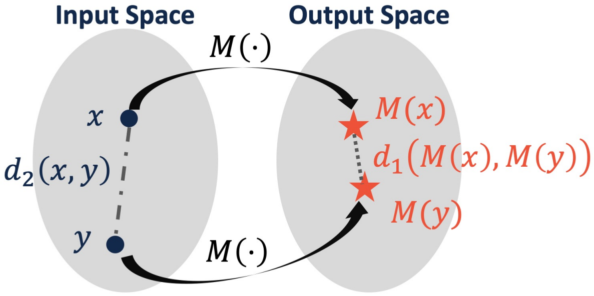


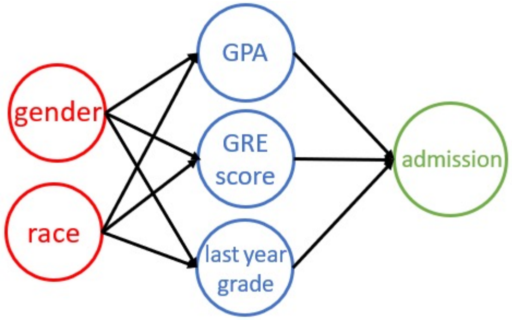
Algorithmic Fairness: Definition



- **Definition of 'fairness':** Lack of favoritism from one side or another
- **Types of fairness**
  - Group fairness
    - Statistical parity
    - Equal opportunity
    - And many more…
  - Individual fairness
  - Counterfactual fairness
  - Rawlsian fairness
  - And many more…





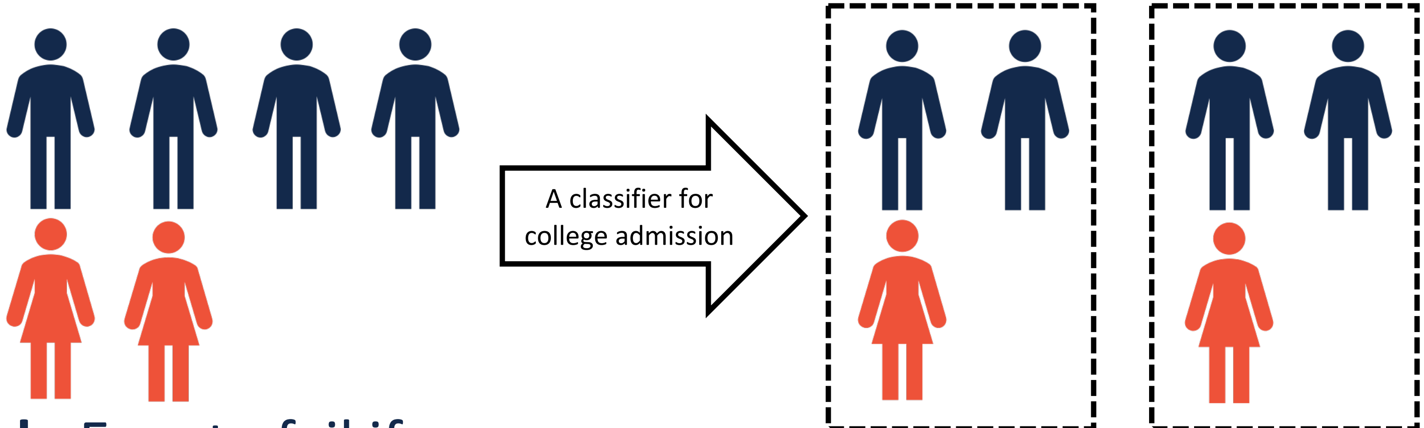


Group Fairness: Statistical Parity

- **Definition:** Equal acceptance rate

$$\Pr_{+}(\hat{y} = c) = \Pr_{-}(\hat{y} = c)$$

  - $\hat{y}$: Model prediction
  - $\Pr_{+}$: Probability of protected group $\Pr_{-}$: Probability of unprotected group
  - Also known as demographic parity
- **Example:** Graduate college admission



- **Remark:** Easy to fail if we
  - Select qualified candidates for one group
  - Randomly select candidates for another group

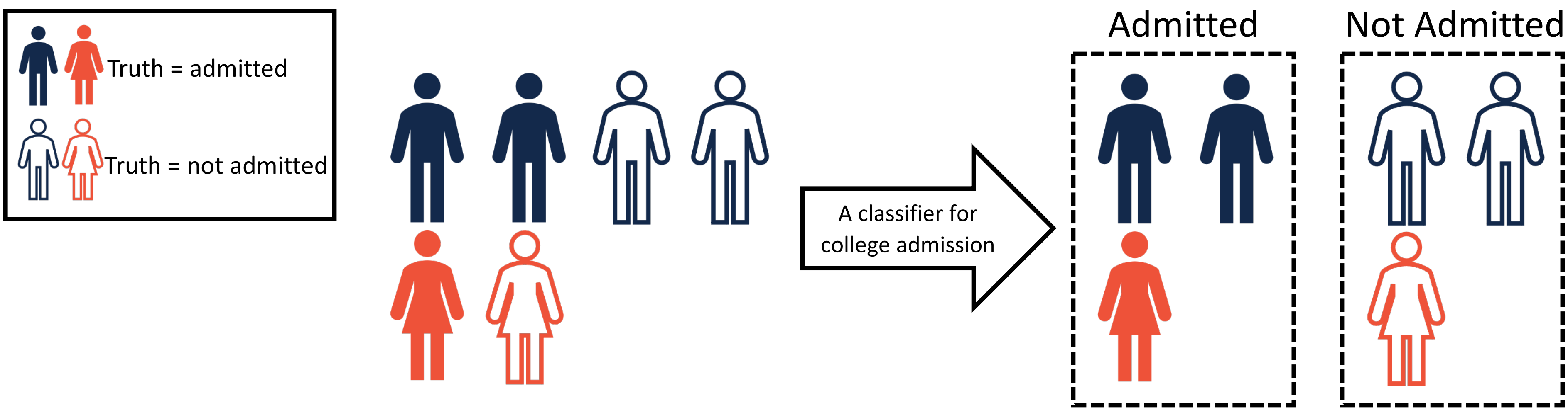
Group Fairness: Equal Opportunity

- **Definition:** Equal true positive rate

$$\Pr_+(\hat{y} = c | y = c) = \Pr_-(\hat{y} = c | y = c)$$

  - $\Pr_+$: Probability of protected group $\Pr_-$: Probability of unprotected group

- **Example:** Graduate college admission



- **Equalized Odds:** Equal opportunity holds for all class labels

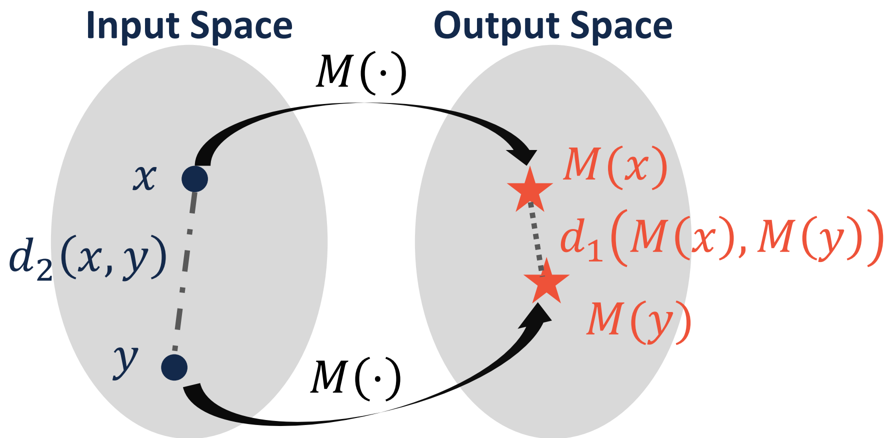
$$\Pr_+(\hat{y} = c | y = c) = \Pr_-(\hat{y} = c | y = c), \quad \forall c$$

Individual Fairness

- **Definition:** Similar individuals should have similar outcomes

$$d_1\big(M(x), M(y)\big) \leq L d_2(x, y)$$

  - M: A mapping from input to output
  - d_1: Distance metric for output
  - d_2: Distance metric for input
  - L: A constant scalar
- **Example**



- **Remarks**
  - Finer granularity than group fairness
  - Hard to find proper distance metrics in practice

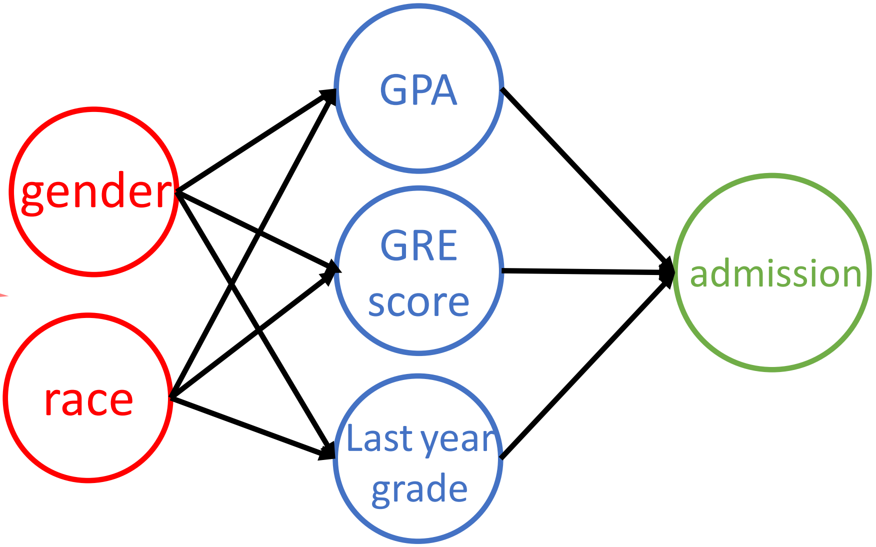
Counterfactual Fairness

- **Definition:** Same outcomes for ‘different versions’ of the same candidate

$$\Pr\left(\hat{y}_{s=s_1} = c | s = s_1, x = \mathbf{x}\right) = \Pr\left(\hat{y}_{s=s_2} = c | s = s_2, x = \mathbf{x}\right)$$

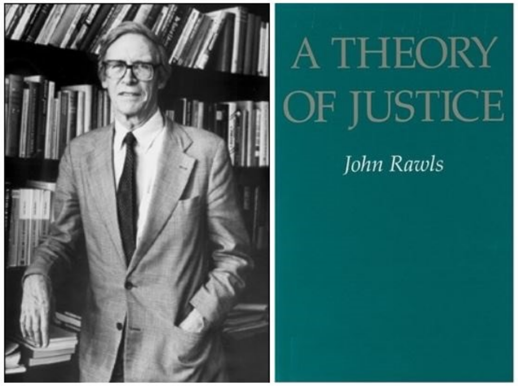
  - $\Pr\left(\hat{y}_{s=s_1} = c | s = s_1, x = \mathbf{x}\right)$: version 1 of $\mathbf{x}$ with sensitive demographic s_1
  - $\Pr\left(\hat{y}_{s=s_2} = c | s = s_2, x = \mathbf{x}\right)$: version 2 of $\mathbf{x}$ with sensitive demographic s_2

- **Example:** Causal graph of graduate college admission



Rawlsian Difference Principle

- **Origin:** Distributive justice



"Inequalities are permissible when they maximize [...] the long-term expectations of the least fortunate group."

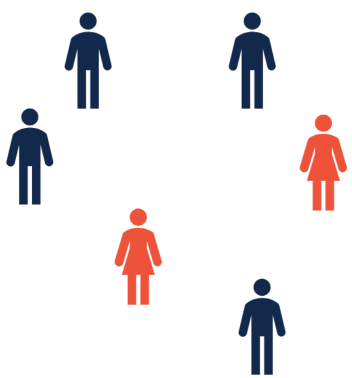
-- John Rawls, 1971

- **Definition:** Impossible to make anyone better off without making at least one other person worse off
- **Formulation in machine learning:** Max-min problem
  - **Min:** The worst-off group with smallest welfare/utility
  - **Max:** Maximization of the corresponding utility

[1] Rawls, J.. A Theory of Justice. Press, Cambridge 1971.

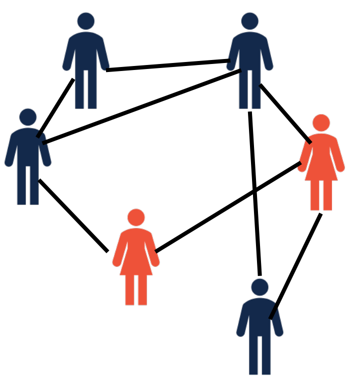
Key Challenge #1: Theoretical Challenge

- **Traditional ML assumption:** Data samples are often IID
- **Non-IID graph data:** Nodes are inter-connected
- **Challenge:** Implication of non-IID nature on
  – Measuring bias
  – Mitigating unfairness
- **Example**



- Individuals are independent
- One cannot affect others

Traditional machine learning

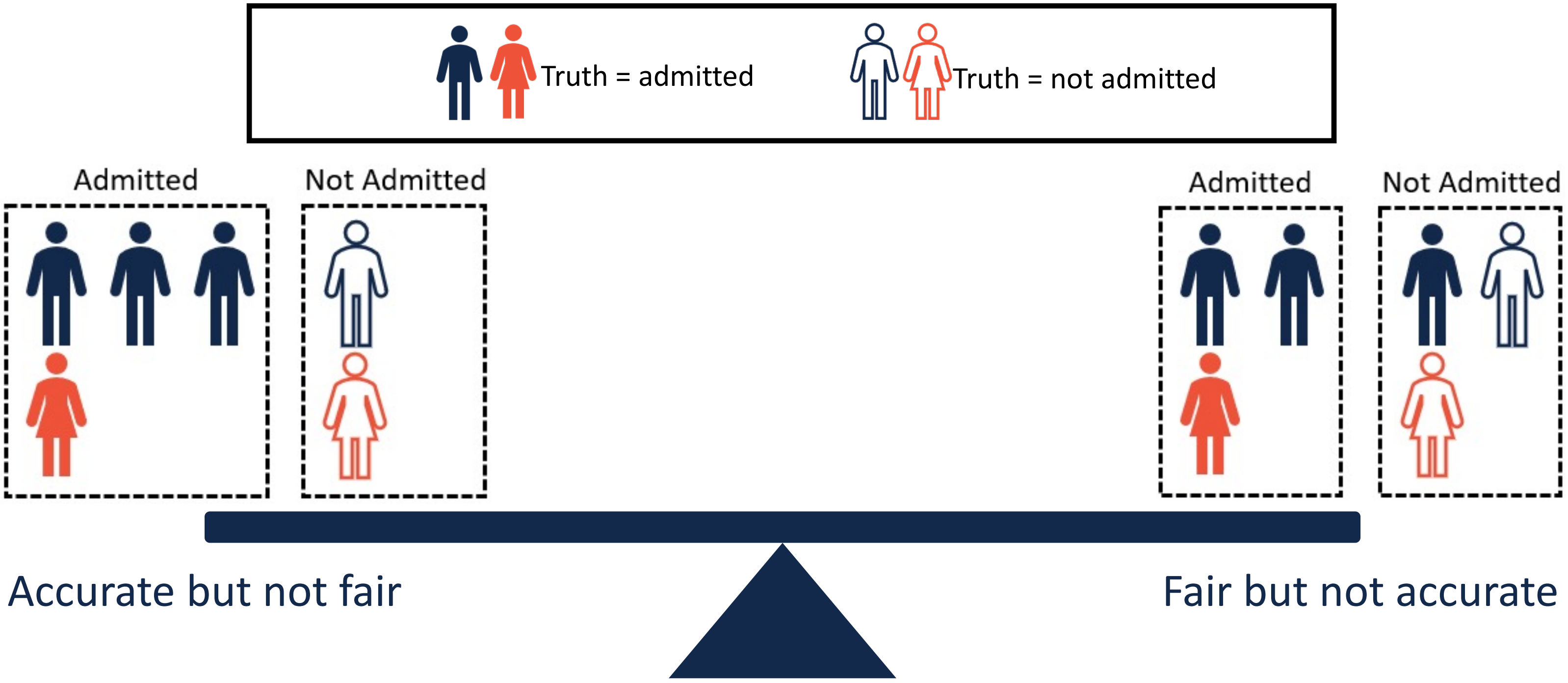


- Individuals are connected
- One can affect others through their connection(s)

Graph mining

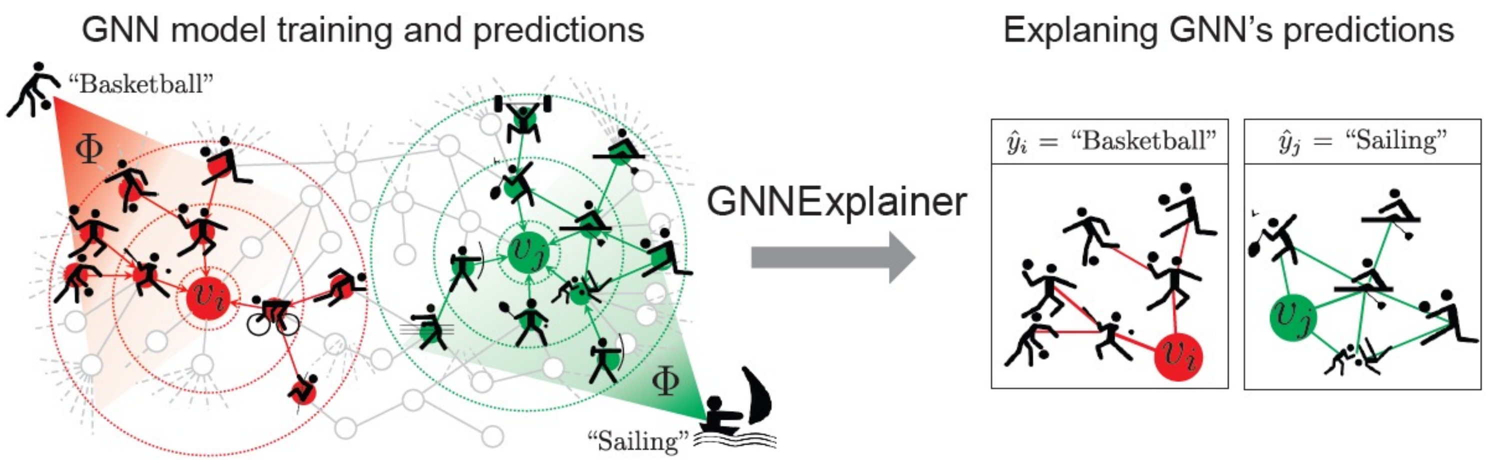
Key Challenge #2: Algorithmic Challenge

- **Dilemma:** Model utility vs. fairness
- **Example:** Graduate college admission with equal opportunity



Related Prob. #1: Explainable Graph Mining

- **Motivation:** **Why** does the mining model make a particular prediction?
- **Goal:** Explain model prediction to non-expert end users
- **Example**

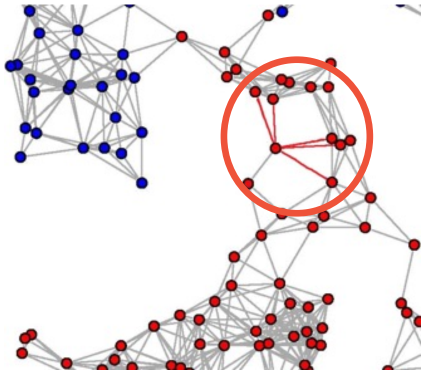


- **Related work:** GNNExplainer, PGM-Explainer, SubgraphX
- **Relationship to fairness:** Explainability helps interpret whether a model uses biased information for prediction to end users

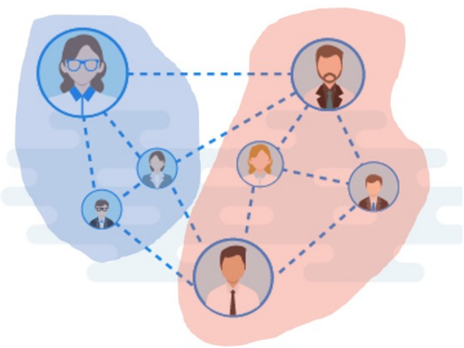
[1] Ying, R., Bourgeois, D., You, J., Zitnik, M., & Leskovec, J.. GNNExplainer: Generating Explanations for Graph Neural Networks. NeurIPS 2019.
[2] Vu, M. N., & Thai, M. T.. PGM-Explainer: Probabilistic Graphical Model Explanations for Graph Neural Networks. NeurIPS 2020.
[3] Yuan, H., Yu, H., Wang, J., Li, K., & Ji, S.. On Explainability of Graph Neural Networks via Subgraph Explorations. ICML 2021.

Related Prob. #2: Graph Mining Auditing

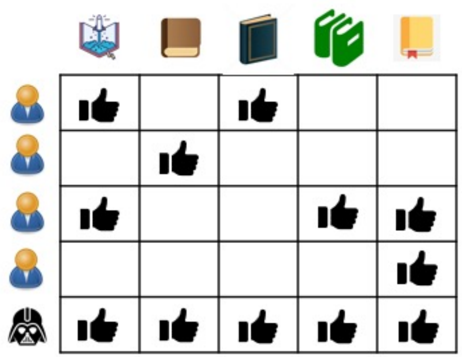
- **Motivation: How** do mining results relate to graph topology?
- **Goal:** Find influential elements w.r.t. the graph mining results
- **Example**



ranking



clustering



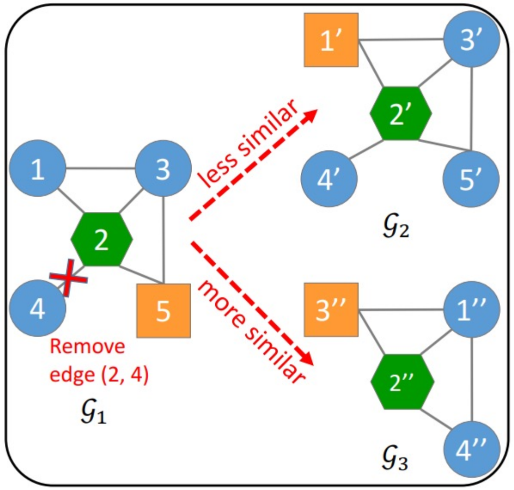
recommendation

- **Related work:** AURORA, N2N, NEAR
- **Relationship to fairness:** Auditing helps determine to what extent a sensitive attribute influences the graph mining results

[1] Kang, J., Wang, M., Cao, N., Xia, Y., Fan, W., & Tong, H.. AURORA: Auditing PageRank on Large Graphs. Big Data 2018.
[2] Kang, J., & Tong, H.. N2N: Network Derivative Mining. CIKM 2019.
[3] Wang, Y., Yao, Y., Tong, H., Xu, F., & Lu, J.. Auditing Network Embedding: An Edge Influence based Approach. TKDE 2021.

Related Prob. #3: Adversarial Attacks on Graph Mining

- **Motivation:** **Why** do mining results sensitive to malicious manipulations?
- **Goal:** Fool the mining model by a few manipulations on the input graph
- **Example**

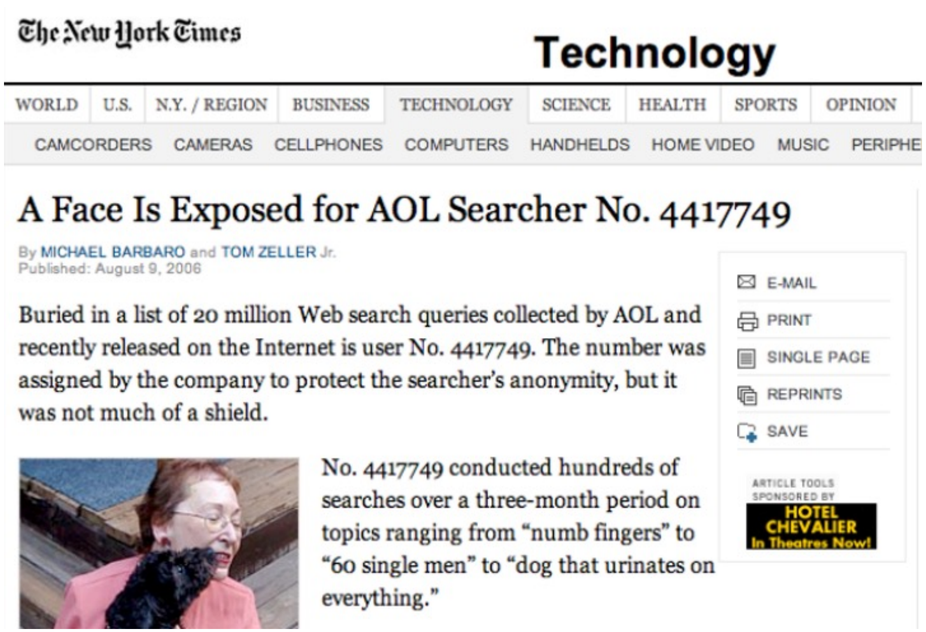


- **Related work:** Nettack, Mettack, Admiring
- **Relationship to fairness:** Malicious users can
  – Manipulate the private sensitive information of other users
  – Attack the model to make a fair mining model biased

[1] Zügner, D., Akbarnejad, A., & Günnemann, S.. Adversarial Attacks on Neural Networks for Graph Data. KDD 2018.
[2] Zügner, D., & Günnemann, S.. Adversarial Attacks on Graph Neural Networks via Meta Learning. ICLR 2019.
[3] Zhou, Q., Li, L., Cao, N., Ying, L., & Tong, H.. ADMIRING: Adversarial Multi-Network Mining. ICDM 2019.

Related Prob. #4: Privacy-Preserving Graph Mining

- **Motivation: Why** can we infer private information by data analysis?
- **Goal:** Prevent the data or mining model from leaking private information
- **Example**



The New York Times

Technology

WORLD | U.S. | N.Y. / REGION | BUSINESS | TECHNOLOGY | SCIENCE | HEALTH | SPORTS | OPINION

CAMCORDERS | CAMERAS | CELLPHONES | COMPUTERS | HANDHELDS | HOME VIDEO | MUSIC | PERIPHE

A Face Is Exposed for AOL Searcher No. 4417749

By MICHAEL BARBARO and TOM ZELLER Jr.
Published: August 9, 2006

Buried in a list of 20 million Web search queries collected by AOL and recently released on the Internet is user No. 4417749. The number was assigned by the company to protect the searcher's anonymity, but it was not much of a shield.

No. 4417749 conducted hundreds of searches over a three-month period on topics ranging from "numb fingers" to "60 single men" to "dog that urinates on everything."

E-MAIL
PRINT
SINGLE PAGE
REPRINTS
SAVE

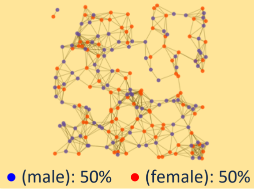
ARTICLE TOOLS SPONSORED BY
HOTEL CHEVALIER In Theatres Now!

- AOL releases anonymized search logs of 650k users
- People find out the identity of one searcher using her search logs in a few days

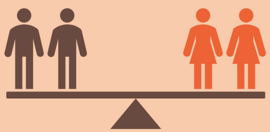
- **Related work:** T_λ, dK-graph, VFGNN
- **Relationship to fairness:** Preserving privacy on sensitive information may help ensure fairness

[1] Ding, X., Zhang, X., Bao, Z., & Jin, H.. Privacy-Preserving Triangle Counting in Large Graphs. CIKM 2018.
[2] Wang, Y., & Wu, X.. Preserving Differential Privacy in Degree-Correlation based Graph Generation. TDP 2013.
[3] Zhou, J., Chen, C., Zheng, L., Wu, H., Wu, J., Zheng, X., ... & Wang, L.. Vertically Federated Graph Neural Network for Privacy-Preserving Node Classification. *arXiv 2020*.

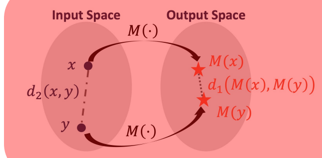
Roadmap



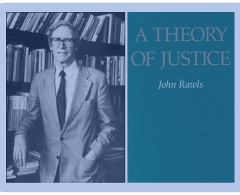
Introduction



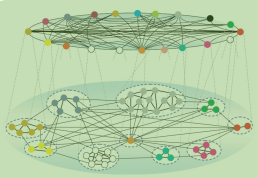
Part I: Group Fairness on Graphs



Part II: Individual Fairness on Graphs



Part III: Other Fairness Definitions on Graphs



Part IV: Future Research Directions

Overview of Part I

Group Fairness on Graphs

Fairness on Graph Ranking

- Fairness-aware PageRank

Fairness on Graph Clustering

- Fairness-aware spectral clustering

Fairness on Graph Embedding

- Adversarial learning-based method
- Random walk-based method
- Bayesian method

Preliminary: PageRank

- **Assumption**
  - Important webpage → linked by many others
- **Formulation**
  - Iteratively solve the following linear system

$$\mathbf{r} = c\mathbf{A}^T\mathbf{r} + (1-c)\mathbf{e}$$

  - $\mathbf{A}$: transition matrix $\mathbf{r}$: PageRank vector
    c: damping factor $\mathbf{e}$: teleportation vector
- **Closed-form solution**

$$\mathbf{r} = (1-c)(\mathbf{I} - c\mathbf{A}^T)^{-\mathbf{1}}\mathbf{e}$$

- Many variants exist, including
  - Personalized PageRank (PPR)
  - Random Walk with Restart (RWR)
  - And many more…

[1] Page, L., Brin, S., Motwani, R., & Winograd, T.. The PageRank Citation Ranking: Bringing Order to the Web. Stanford InfoLab 1999.
[2] Haveliwala, T. H.. Topic-sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search. TKDE 2003.
[3] Tong, H., Faloutsos, C., & Pan, J. Y.. Fast Random Walk with Restart and Its Applications. ICDM 2006.

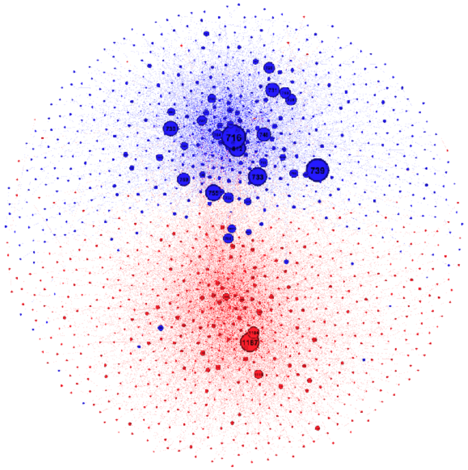
Fairness Measures for PageRank

- **ϕ-fair PageRank**
  - **Given:** A graph G
  - **Definition:** A PageRank vector is ϕ-fair if ϕ fraction of total PageRank mass is allocated to the protected group
  - **Variants and generalizations**
    - Statistical Parity: $\phi =$ fraction of protected group
    - Affirmative Action: $\phi =$ a desired ratio (e.g., 20%)
- **Targeted ϕ-fair PageRank**
  - **Given:** (1) A graph G and (2) a subset of nodes $\mathcal{S}$
  - **Definition:** A PageRank vector is targeted ϕ-fair if ϕ fraction of PageRank mass w.r.t. the subset $\mathcal{S}$ is allocated to the protected group in subset $\mathcal{S}$

[1] Tsioutsiouliklis, S., Pitoura, E., Tsaparas, P., Kleftakis, I., & Mamoulis, N.. Fairness-Aware PageRank. WWW 2021.

Prob. Defn.: Fairness-Aware PageRank

- **Given**
  - A graph with transition matrix $\mathbf{A}$
  - Partitions of nodes
    - Red nodes ($\mathcal{R}$): protected group
    - Blue nodes ($\mathcal{B}$): unprotected group
- **Find:** A fair PageRank vector $\tilde{\mathbf{r}}$ that is
  - ϕ-fair
  - Close to the original PageRank vector $\mathbf{r}$



Fairness-aware PageRank

- **Recap:** PageRank
  - Closed-form Solution

$$\mathbf{r} = (1 - c)(\mathbf{I} - c\mathbf{A}^T)^{-\mathbf{1}}\mathbf{e}$$

- **Parameters in PageRank**
  - **Damping factor c:** Avoid sinks in the random walk (i.e., nodes without outgoing links)
  - **Teleportation vector e:** Control the starting node where a random walker restarts
    - **Question:** Can we let the walker restart at a protected node or a node near many protected nodes? ← Solution #1: Fairness-sensitive PageRank
  - **Transition matrix $\mathbf{A}$**: Control the next step where the walker goes to
    - **Question:** Can we let the walker go to the protected nodes more frequently?

Solution #1: Fairness-sensitive PageRank

- **Intuition**
  - Find a teleportation vector $\mathbf{e}$ to make PageRank vector ϕ-fair
  - Keep transition matrix $\mathbf{A}$ and $\mathbf{Q}^T = (1-c)(\mathbf{I} - c\mathbf{A}^T)^{-1}$ fixed
- **Observation:** Mass of PageRank $\mathbf{r}$ w.r.t. red nodes $\mathcal{R}$

$$\mathbf{r}(\mathcal{R}) = \mathbf{Q}^T[\mathcal{R},:]\mathbf{e}$$

  - $\mathbf{Q}^T[\mathcal{R},:]$: Rows of $\mathbf{Q}^T$ w.r.t. nodes in set $\mathcal{R}$
- **(Convex) optimization problem**

$$\begin{aligned} \min_{v} \quad & \boxed{\|\mathbf{Q}^T\mathbf{e} - \mathbf{r}\|^2} \\ \text{s.t.} \quad & \boxed{\begin{aligned} &\mathbf{e}[i] \in [0,1], \forall i \\ &\|\mathbf{e}\|_1 = 1 \end{aligned}} \\ & \boxed{\|\mathbf{Q}^T[:,\mathcal{R}]\mathbf{e}\|_1 = \phi\ \|\mathbf{r}\|_1} \end{aligned}$$

The fair PageRank $\mathbf{Q}^T\mathbf{e}$ is as close as possible to the original PageRank $\mathbf{r}$

The teleportation vector $\mathbf{e}$ is a probability distribution

The fair PageRank $\mathbf{Q}^T\mathbf{e}$ needs to be ϕ-fair

  - Can be solved by any convex optimization solvers

Example: Fairness-sensitive PageRank

- Define $\phi = 1/3$ and the protected node is the red node
- **Original PageRank**

rows w.r.t. blue nodes

row w.r.t. red nodes

$$\mathbf{Q}^T = \begin{bmatrix} 0.8 & 0.7 & 0.3 \\ 0.7 & 0.9 & 0.5 \\ 0.3 & 0.5 & 0.7 \end{bmatrix}, \quad \mathbf{e} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

$$\mathbf{r} = \mathbf{Q}^T\mathbf{e} = \begin{bmatrix} 0.6 \\ 0.7 \\ 0.5 \end{bmatrix}$$

Not ϕ-fair!

$$\frac{0.5}{0.6+0.7+0.5} < \frac{1}{3}$$

- **Fairness-sensitive PageRank**

rows w.r.t. blue nodes

row w.r.t. red nodes

$$\mathbf{Q}^T = \begin{bmatrix} 0.8 & 0.7 & 0.3 \\ 0.7 & 0.9 & 0.5 \\ 0.3 & 0.5 & 0.7 \end{bmatrix}, \quad \tilde{\mathbf{e}} = \begin{bmatrix} 1/6 \\ 1/6 \\ 2/3 \end{bmatrix}$$

$$\tilde{\mathbf{r}} = \mathbf{Q}^T\tilde{\mathbf{e}} = \begin{bmatrix} 0.45 \\ 0.6 \\ 0.6 \end{bmatrix}$$

ϕ-fair

$$\frac{0.6}{0.45+0.6+0.6} > \frac{1}{3}$$

Fairness-aware PageRank

- **Recap:** PageRank
  - Closed-form Solution

$$\mathbf{r} = (1-c)(\mathbf{I} - c\mathbf{A}^T)^{-1}\mathbf{e}$$

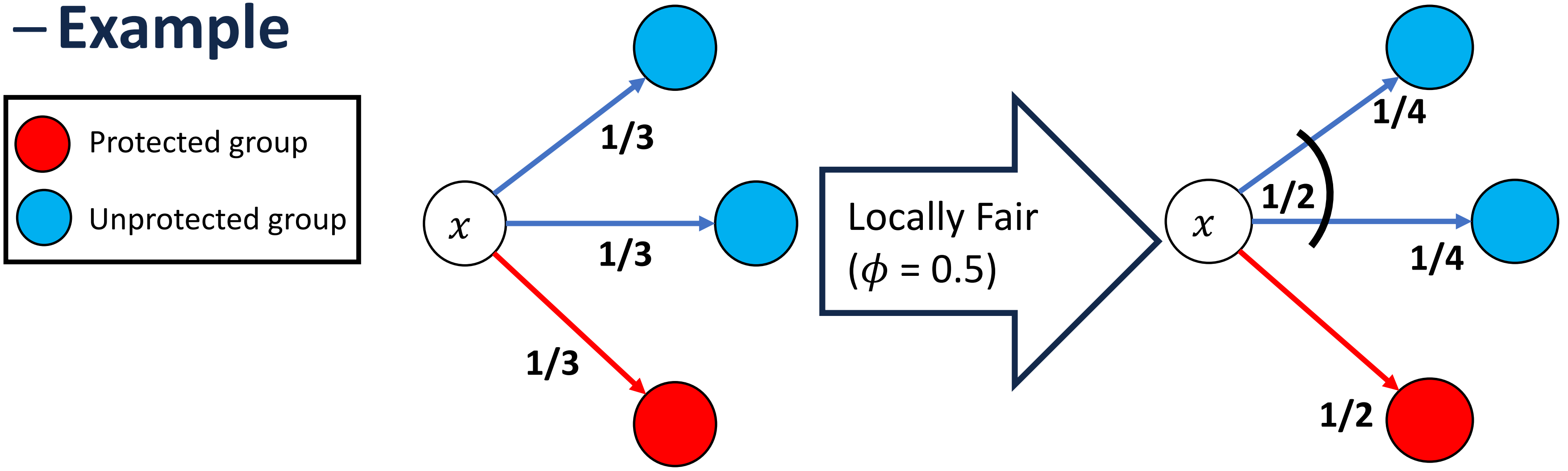
- **Parameters in PageRank**
  - **Damping factor c:** Avoiding sinks in the random walk (i.e., nodes without outgoing links)
  - **Teleportation vector $\mathbf{e}$:** Controlling the starting node where a random walker restarts
    - **Question:** Can we let the walker restart at a protected node or a node near many protected nodes?
  - **Transition matrix $\mathbf{A}$**: Controlling the next step where the walker goes to

Solution #2: Locally fair PageRank

- **Question:** Can we let the walker go to protected node more frequently?

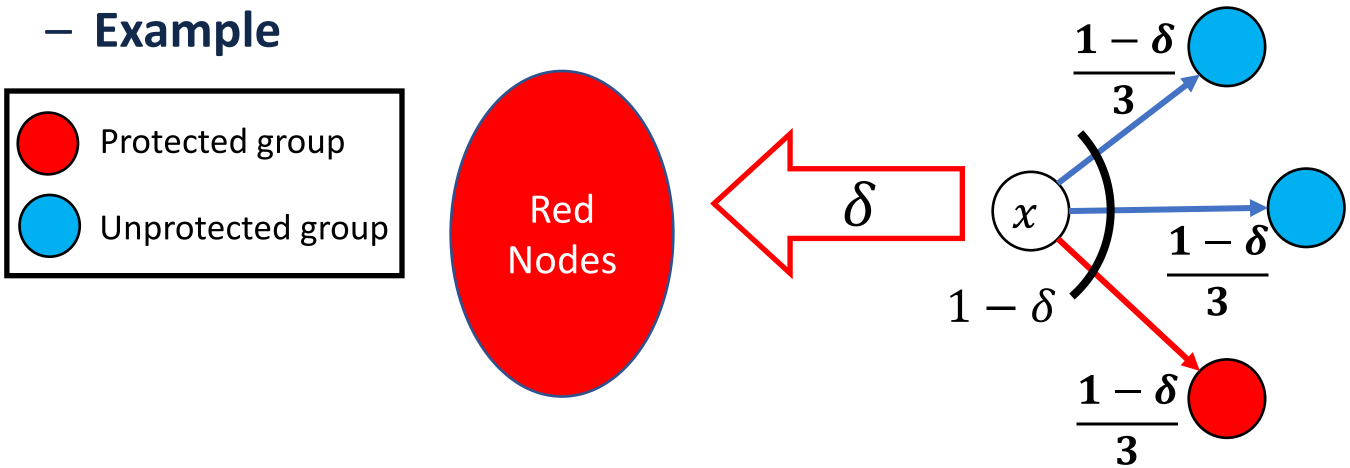
Solution #2: Locally Fair PageRank

- **Intuition:** Adjust the transition matrix A to obtain a fair random walk
- **Neighborhood locally fair PageRank**
  - **Key idea:** Jump with probability ϕ to red nodes and (1-ϕ) to blue nodes
  - **Example**



Solution #2: Locally Fair PageRank

- **Residual locally fair PageRank**
  - **Key idea:** Jump with
    - Equal probability to 1-hop neighbors
    - A residual probability δ to the other red nodes
  - **Example**

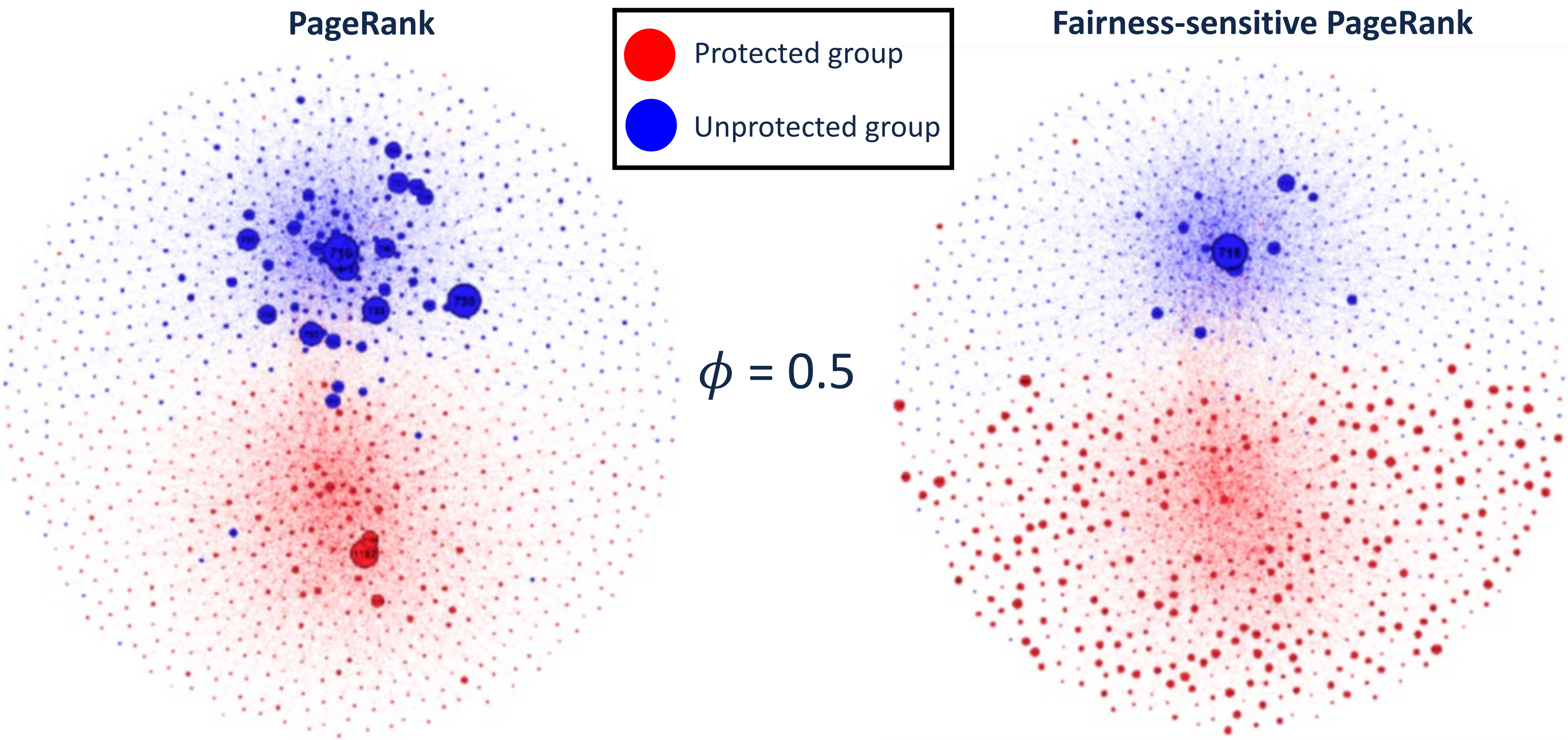


$$\delta + \frac{1-\delta}{3} = \phi$$

- **Residual allocation policies:** Neighborhood allocation, uniform allocation, proportional allocation, optimized allocation

Fairness-sensitive PageRank: Experiment

- **Observation:** The teleportation vector allocates more weight to the red nodes, especially nodes at the periphery of the network

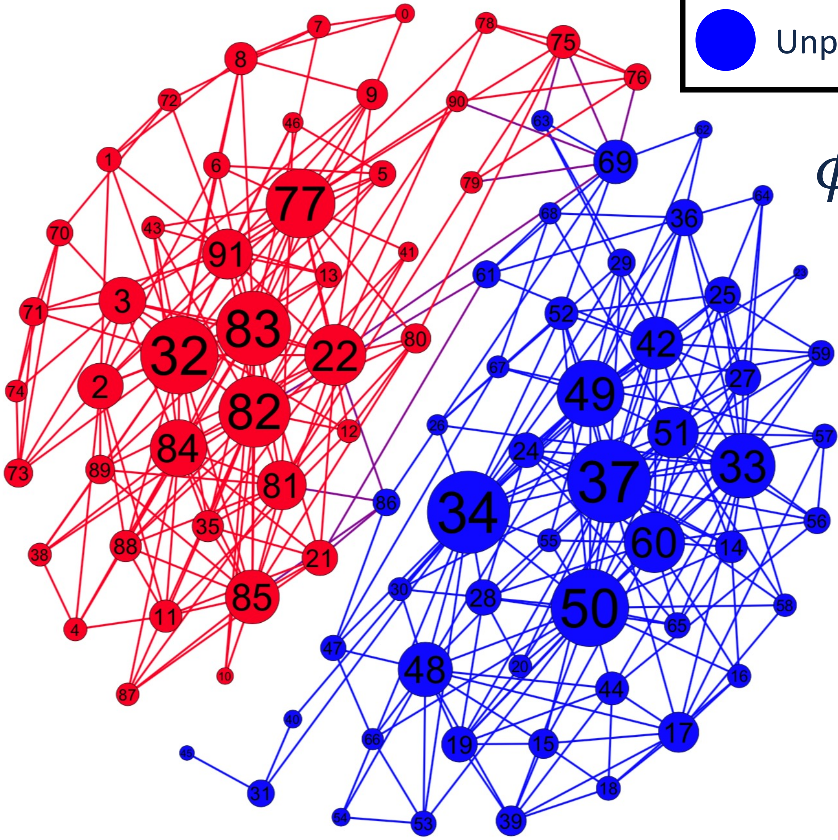


$\phi = 0.5$

Locally Fair PageRank: Experiment

- **Observation:** PageRank weight is shifted to the boundary nodes

PageRank

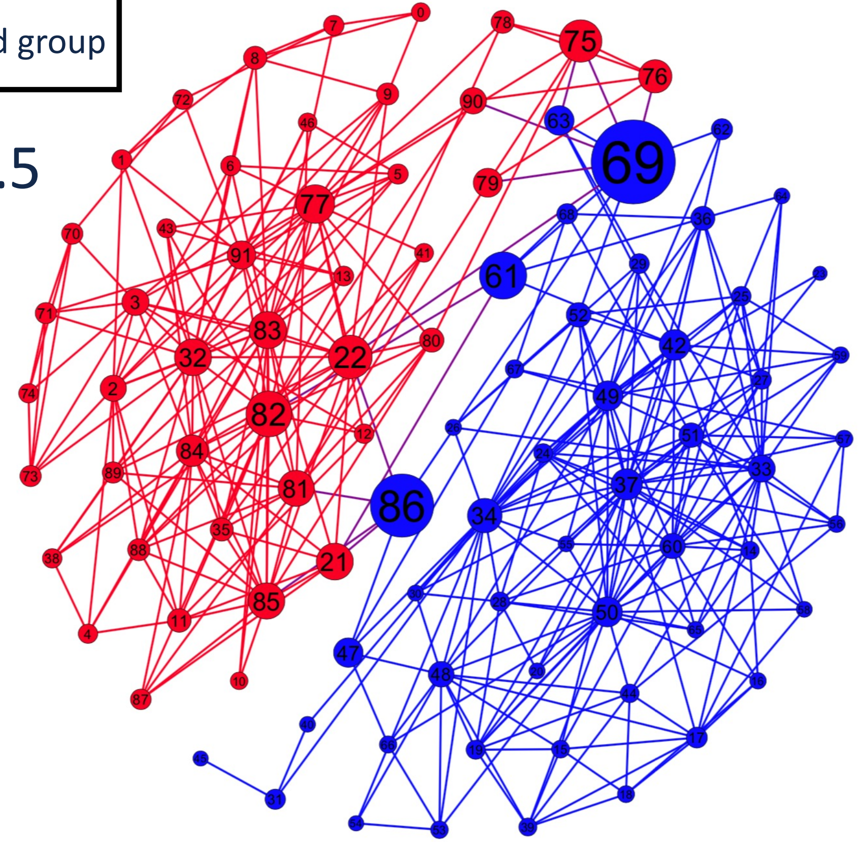


Protected group

Unprotected group

$\phi = 0.5$

Neighborhood Locally Fair PageRank



Overview of Part I

Group Fairness on Graphs

- **Fairness on Graph Ranking**
  - Fairness-aware PageRank
- **Fairness on Graph Clustering**
  - Fairness-aware spectral clustering
- **Fairness on Graph Embedding**
  - Adversarial learning-based method
  - Random walk-based method
  - Bayesian method

Preliminary: Spectral Clustering (SC)

- **Goal:** Find k clusters such that $\begin{cases} \text{maximize intra-connectivity} \\ \text{minimize inter-connectivity} \end{cases}$

- **Optimization problem**

$$\min_{\mathbf{U}} \quad \underbrace{\boxed{\mathrm{Tr}\left(\mathbf{U}^T\mathbf{L}\mathbf{U}\right)}}_{\text{Ratio cut}}$$
$$\text{s.t.} \quad \mathbf{U}^T\mathbf{U} = \mathbf{I}$$

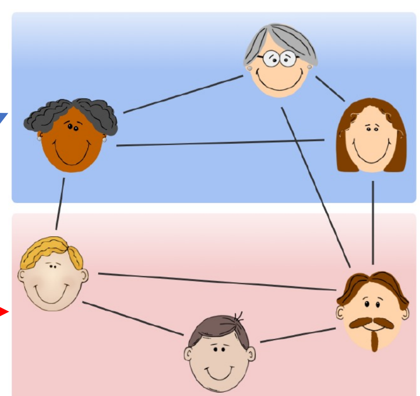
where $\mathbf{L}$ is Laplacian matrix of $\mathbf{A}$, $\mathbf{U}$ is a matrix with k orthonormal column vectors

- **Solution:** Rank-k eigen-decomposition
  - $\mathbf{U}$ = eigenvectors with k smallest eigenvalues

- **Example**

All female individuals are grouped together

All male individuals are grouped together



Highly unfair: The clustering results are correlated with gender

[1] Ng, A. Y., Jordan, M. I., & Weiss, Y.. On Spectral Clustering: Analysis and an Algorithm. NeurIPS 2002.

[2] Shi, J., & Malik, J.. Normalized Cuts and Image Segmentation. TPAMI 2000.

Fair Spectral Clustering: Fairness Measure

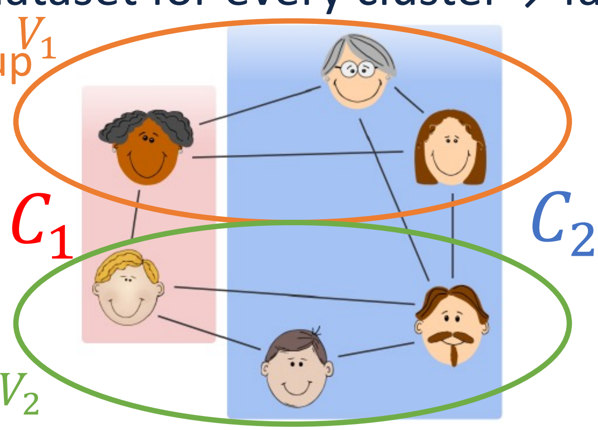
- **Intuition:** Fairness as balance among clusters
- **Given:** A node set V with
  - h demographic groups: $V = V_1 \cup V_2 \ldots \cup V_h$
  - k clusters: $V = C_1 \cup C_2 \ldots \cup C_k$
- **Define**

$$\text{balance}(C_l) = \min_{s \neq s' \in [h]} \frac{|V_s \cap C_l|}{|V_{s'} \cap C_l|} \in [0, 1], \qquad \forall l \in [1, 2, \ldots, k]$$

- **Intuition:** higher balance → each demographic group is presented with similar fractions as in the whole dataset for every cluster→ fairer
- **Example**

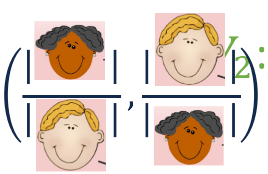
V_1: female group

V_2: male group



$\text{balance}(C_1)$

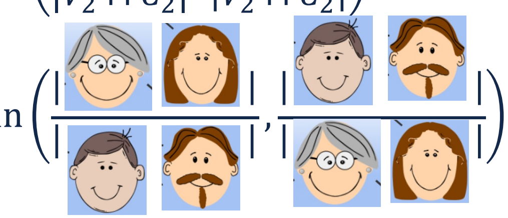
$= \min\left(\frac{|V_1 \cap C_1|}{|V_2 \cap C_1|}, \frac{|V_2 \cap C_1|}{|V_1 \cap C_1|}\right)$



$= 1$

$\text{balance}(C_2)$

$= \min\left(\frac{|V_1 \cap C_2|}{|V_2 \cap C_2|}, \frac{|V_2 \cap C_2|}{|V_2 \cap C_2|}\right)$



$= 1$

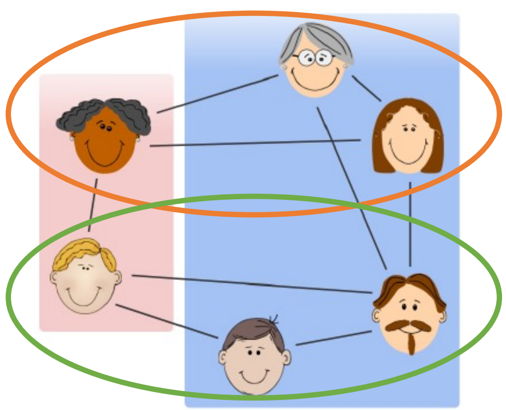
[1] Kleindessner, M., Samadi, S., Awasthi, P., & Morgenstern, J.. Guarantees for Spectral Clustering with Fairness Constraints. ICML 2019.

Fair Spectral Clustering: Solution

- **Fairness as linear constraint**
  - **Given**
    - The spectral embedding $\mathbf{U}$ of n nodes in l clusters $(C_1, \dots, C_l)$
    - h demographic groups $(V_1, \dots, V_s)$
  - **Define**
    - $\mathbf{f}^{(s)}[i] = 1$ if $i \in V_s$ and 0 otherwise
    - $\mathbf{F} =$ a matrix with $\mathbf{f}^{(s)} - \left(\frac{|V_s|}{n}\right)\mathbf{1}_n$ (s $\in [1, \dots, h-1]$) as column vectors
  - **Observation: $\mathbf{F}^T\mathbf{U} = \mathbf{0}$** $\Leftrightarrow$ balanced clusters (i.e., fair clusters)
- **Example**

V_1: female group

V_2: male group



$\mathbf{f}^{(1)}$	$\mathbf{f}^{(2)}$
1	0
1	0
1	0
0	1
0	1
0	1

Fair fraction

0.5	0.5
0.5	0.5
0.5	0.5
0.5	0.5
0.5	0.5
0.5	0.5

$\mathbf{F} =$

0.5	-0.5
0.5	-0.5
0.5	-0.5
-0.5	0.5
-0.5	0.5
-0.5	0.5

Fair Spectral Clustering: Solution

- **Optimization problem**

$$\min_{\mathbf{U}} \quad \mathrm{Tr}\left(\mathbf{U}^T\mathbf{L}\mathbf{U}\right)$$
$$\text{s.t.} \quad \mathbf{U}^T\mathbf{U} = \mathbf{I}, \boxed{\mathbf{F}^T\mathbf{U} = \mathbf{0}}$$

How to solve?

- **Solution**
  - **Observation:** $\mathbf{F}^T\mathbf{U} = \mathbf{0}$ → $\mathbf{U}$ is in the null space of $\mathbf{F}^T$
  - Define $\mathbf{Z} =$ orthonormal basis of null space of $\mathbf{F}^T$
  - Rewrite $\mathbf{U} = \mathbf{Z}\mathbf{Y}$

$$\min_{\mathbf{U}} \quad \mathrm{Tr}\left(\mathbf{Y}^T\mathbf{Z}^T\mathbf{L}\mathbf{Z}\mathbf{Y}\right)$$
$$\text{s.t.} \quad \mathbf{Y}^T\mathbf{Y} = \mathbf{I}$$

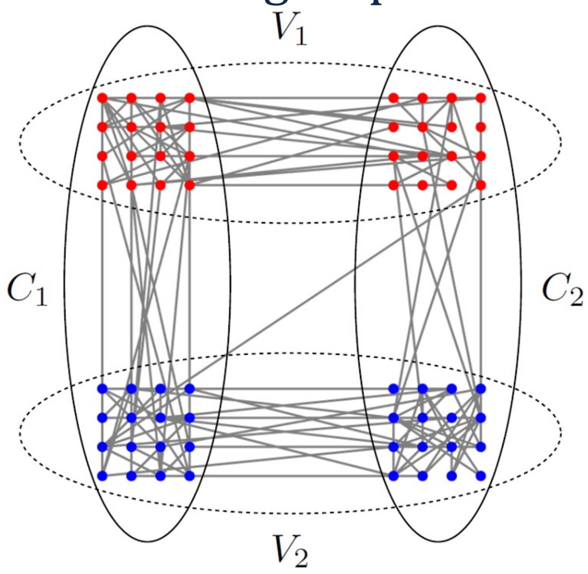
  - **Solution:** Rank-k eigen-decomposition on $\mathbf{Z}^T\mathbf{L}\mathbf{Z}$

Correctness of Fair Spectral Clustering

- **Given**
  - A random graph with nodes V by a variant of the Stochastic Block Model (SBM)
  - Edge probability between two nodes i and j

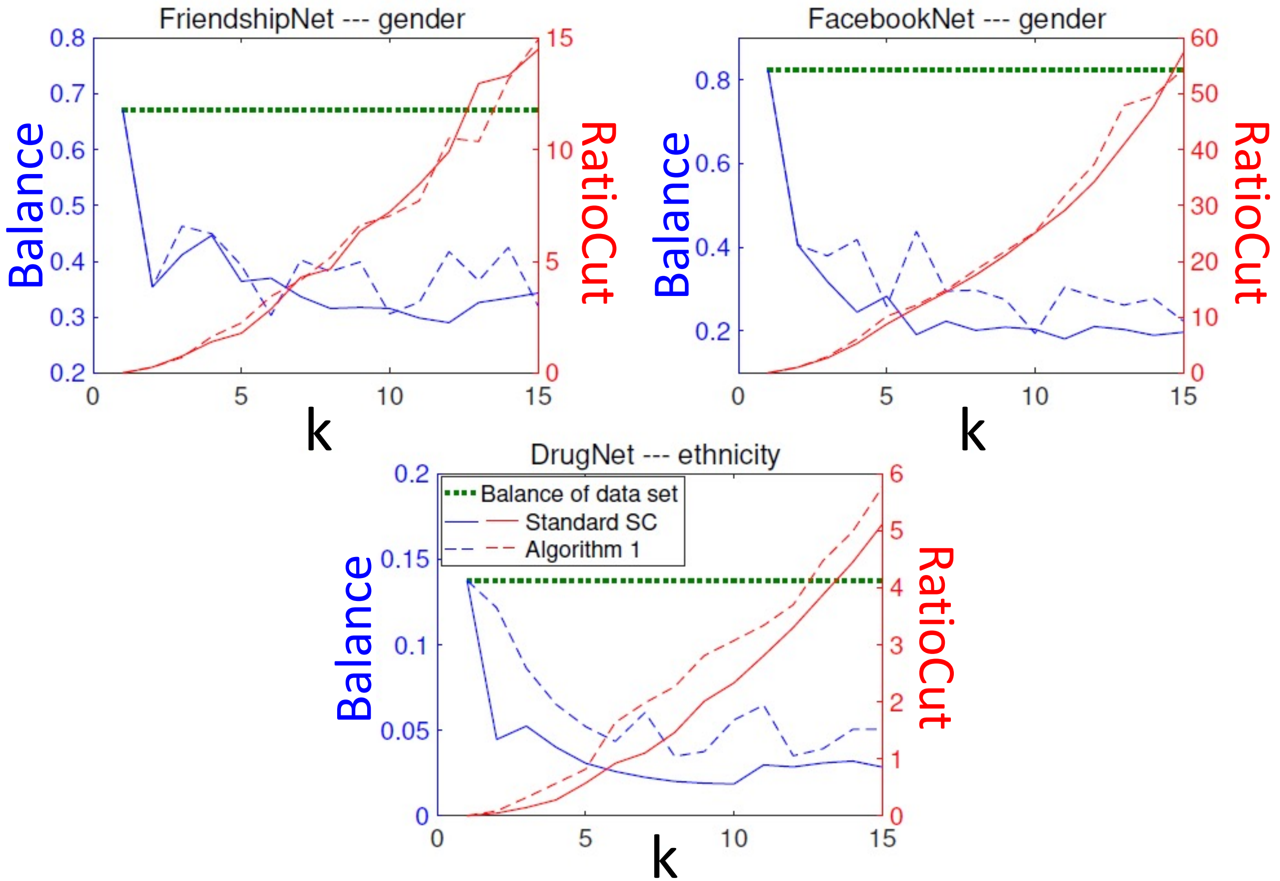
$$\mathrm{P}(i,j) = \begin{cases} a, & i \text{ and } j \text{ in same cluster and in same group} \\ b, & i \text{ and } j \text{ not in same cluster but in same group} \\ c, & i \text{ and } j \text{ in same cluster but not in same group} \\ d, & i \text{ and } j \text{ not in same cluster and not in same group} \end{cases}$$

    for some $a > b > c > d$
  - A fair ground-truth clustering $V = C_1 \cup C_2$
- **Theorem:** Fair SC recovers the ground-truth clustering $C_1 \cup C_2$ with high probability
- **Example**
  - Standard SC is likely to return $V_1 \cup V_2$



Fair Spectral Clustering: Experiment

- **Observation:** Fairer (higher balance score) with similar ratio cut values for the proposed method (Algorithm 1 in the figure)



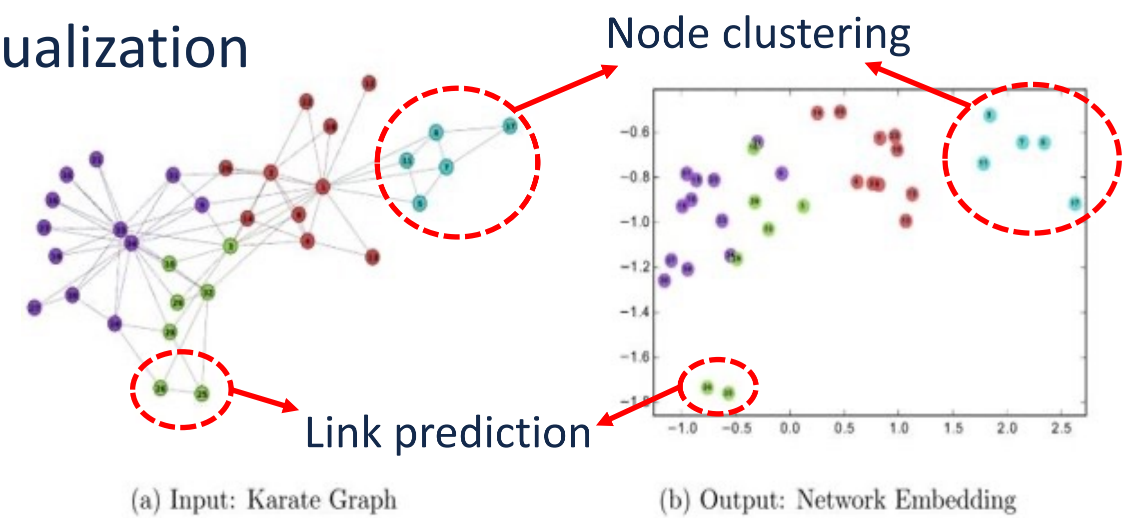
Overview of Part I

Group Fairness on Graphs

Fairness on Graph Ranking	Fairness on Graph Clustering	Fairness on Graph Embedding
• Fairness-aware PageRank	• Fairness-aware spectral clustering	• Adversarial learning-based method • Random walk-based method • Bayesian method

Preliminary: Graph Embedding

- **Motivation:** Learn low-dimensional node representations that preserve structural/attributive information
- **Applications**
  - Node classification
  - Link prediction
  - Node visualization
- **Example**



Visualization of Node Embedding

[1] Perozzi, B., Al-Rfou, R., & Skiena, S.. DeepWalk: Online Learning of Social Representations. KDD 2014.
[2] Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q.. LINE: Large-scale Information Network Embedding. WWW 2015.
[3] Tang, J., Liu, J., Zhang, M., & Mei, Q.. Visualizing Large-scale and High-dimensional Data. WWW 2016.

Preliminary: Setup of Graph Embeddings

- **Two key components:** Edge-wise scoring function + loss function
- **Edge-wise scoring function**
  - Suppose $e = (u, v)$; $\mathbf{z}_u$ is embedding of u;
  - **Dot product:** $s(e) = s(\langle \mathbf{z}_u, \mathbf{r}, \mathbf{z}_v \rangle) = \mathbf{z}_u^T \mathbf{z}_v$
  - **TransE:** $s(e) = s(\langle \mathbf{z}_u, \mathbf{r}, \mathbf{z}_v \rangle) = -\|\mathbf{z}_u + \mathbf{r} - \mathbf{z}_v\|_2^2$
- **Edge-wise loss function**
  - Suppose e_i^- is i-th negative sample for edge e
  - **Max margin loss**

$$L_{\text{edge}}\left(s(e), s(e_1^-), \dots, s(e_m^-)\right) = \sum_{i=1}^{m} \max(1 - s(e) + s(e_i^-), 0)$$

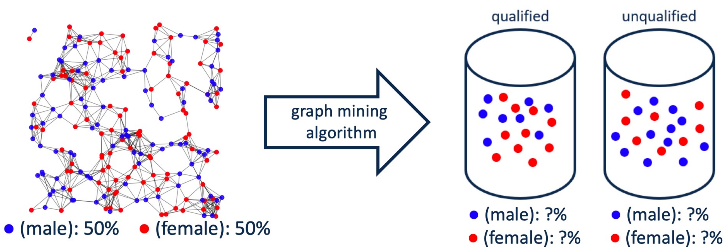
  - **Cross entropy loss**

$$L_{\text{edge}}\left(s(e), s(e_1^-), \dots, s(e_m^-)\right) = -\log\left(\sigma\left(s(e)\right)\right) - \sum_{i=1}^{m} \log\left(1 - \sigma\left(s(e_i^-)\right)\right)$$

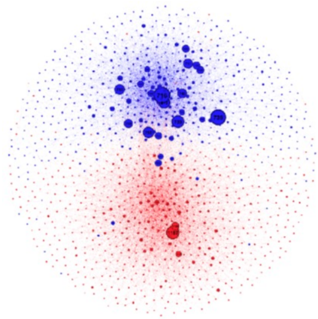
Compositional Fairness in Graph Embeddings

- **Why fairness for embeddings?**
  - Not just one classification task that considers fairness (e.g., ranking, clustering)

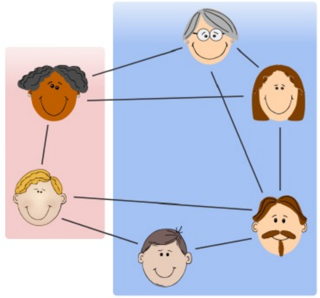
classification



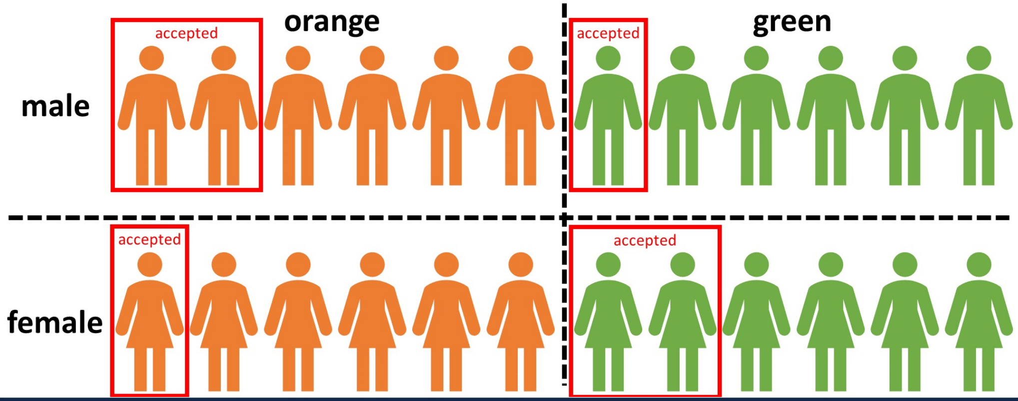
ranking



clustering



- **Why compositional fairness?**
  - **Compositional fairness:** accommodation to a combination of sensitive attributes
  - Often many possible sensitive attributes for a downstream task



- **Gender:** male vs. female
- **Race*:** orange vs. green

* We use imaginary race groups to avoid potential offenses

[1] Bose, A., & Hamilton, W.. Compositional Fairness Constraints for Graph Embeddings. ICML 2019.

Representational Invariance as Fairness

- **Intuition:** Independence between the learned embedding $\mathbf{z}$ and a sensitive attribute a

$$\mathbf{z}_u \perp a_u, \forall \text{ node } u$$

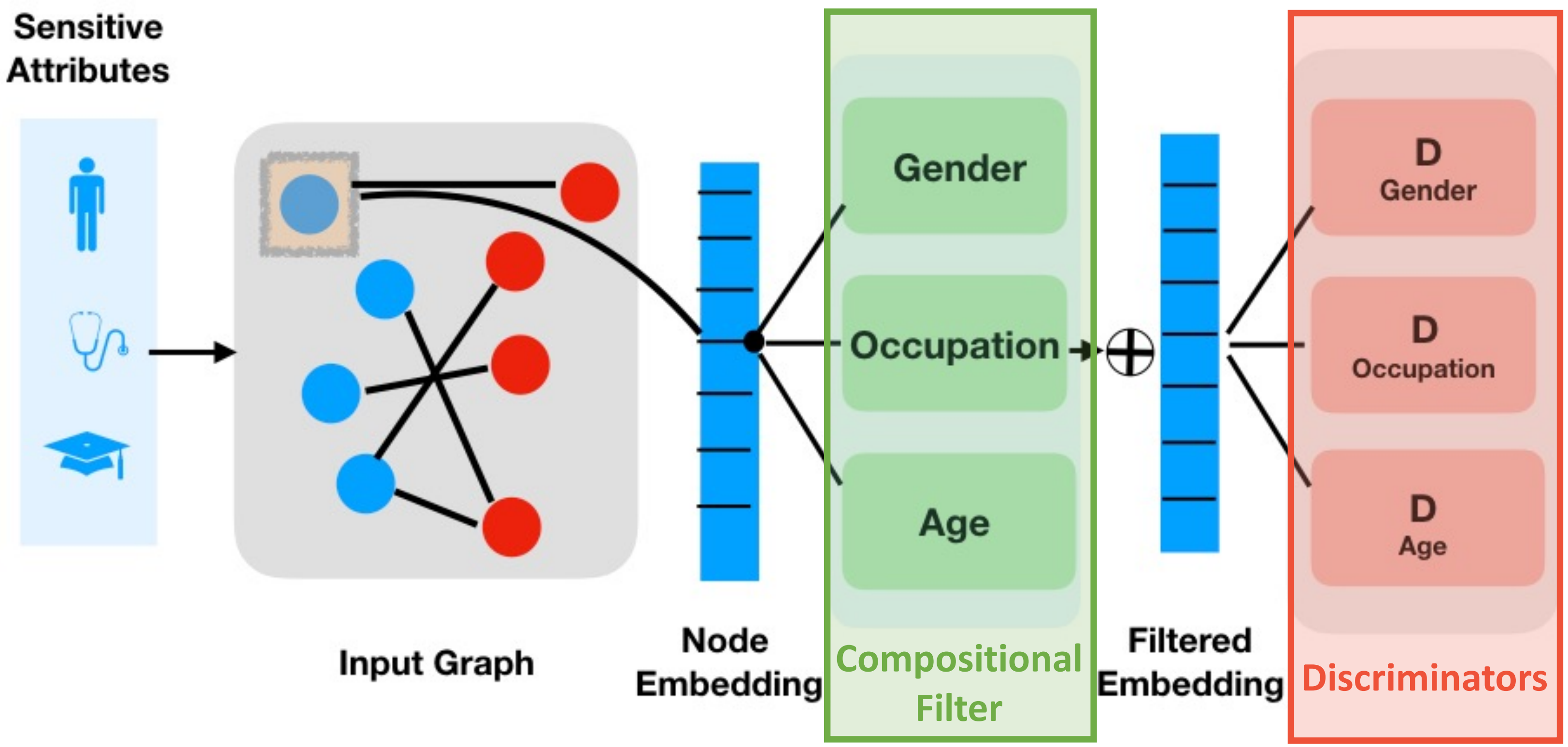
  where a_u is the sensitive value of node u
- **Formulation:** Mutual information minimization

$$I(\mathbf{z}_u, a_u) = 0, \forall \text{ node } u$$

  - Analogous to statistical parity in classification task
  - **Key idea:** Fail to predict a_u using $\mathbf{z}_u$
  - **Solution:** Adversarial learning

Compositional Fairness: Framework

- **Overview:** The proposed compositional fairness framework
- **Key components:** (1) Compositional Filter (C-ENC) and (2) Discriminators (D_k)



Key Component #1: Compositional Filter

(Also called compositional encoder, i.e., C-ENC)

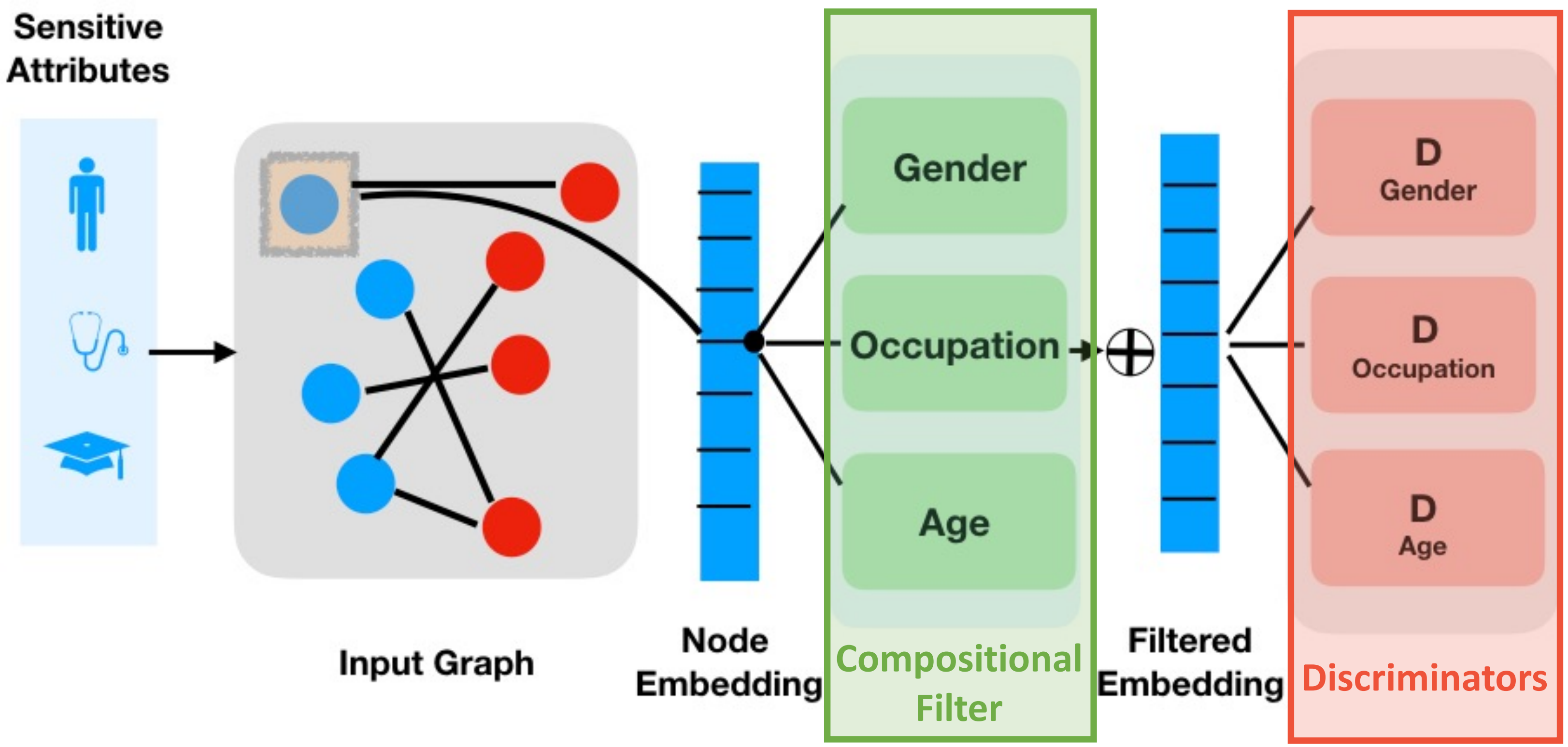
- **Goal:** Filter sensitive information from the embeddings
  - The 'filtered' embedding should be invariant to those attributes
- **Formulation**

$$\mathrm{C{-}ENC}(u, S) = \frac{1}{|S|} \sum_{k \in S} f_k\big(\mathrm{ENC}(u)\big)$$

  - Compositional filter: A collection of filters
  - Filter: Trainable function f_k (neural networks, e.g., MLP)
  - Input: Node ID u and the set of sensitive attributes S (e.g., gender, age)
  - Compositionality: Summation over all sensitive attributes

Compositional Fairness: Framework

- **Overview:** The proposed compositional fairness framework
- **Key components:** (1) Compositional filter (C-ENC) and (2) discriminators (D_k)



Key Component #2: Discriminators

- **Goal:** Predict the sensitive attribute from the 'filtered' embeddings
- **Formulation**

  $$\mathrm{D}_k\left(\mathrm{C{-}ENC}(u,S), a^k\right) = \Pr\left(a_u = a^k \middle| \mathrm{C} - \mathrm{ENC}(u,S)\right)$$

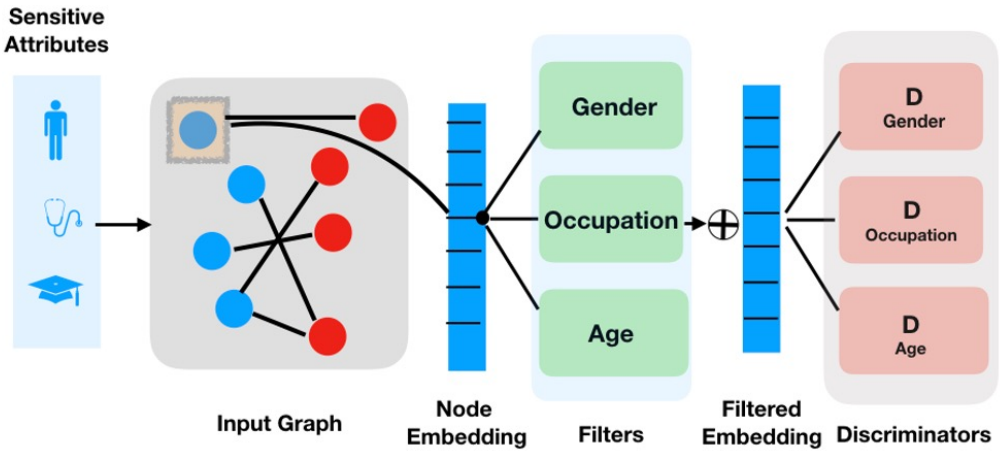
  - D_k: Discriminator for k-th sensitive attribute
  - Input: Node u's 'filtered' embedding and attribute value
  - $\Pr\left(a_u = a^k \middle| \mathrm{C} - \mathrm{ENC}(u,S)\right)$: Likelihood that node u has that attribute value

Compositional Fairness: Loss Function

- **Edge-wise objective function**

$$L(e) = L_{\text{edge}}\big(s(e), s(e_1^-), \dots, s(e_m^-)\big)$$
$$+\lambda \sum_{k \in S} \sum_{a^k \in \mathcal{A}_k} \log\Big(\mathrm{D}_k\big(\mathrm{C\text{-}ENC}(u, S), a^k\big)\Big)$$

  - L_{edge}: Edge-wise loss function for graph embedding
  - $\log\Big(\mathrm{D}_k\big(\mathrm{C\text{-}ENC}(u, S), a^k\big)\Big)$: The discriminator fails to predict sensitive attribute correctly with the 'filtered' embeddings



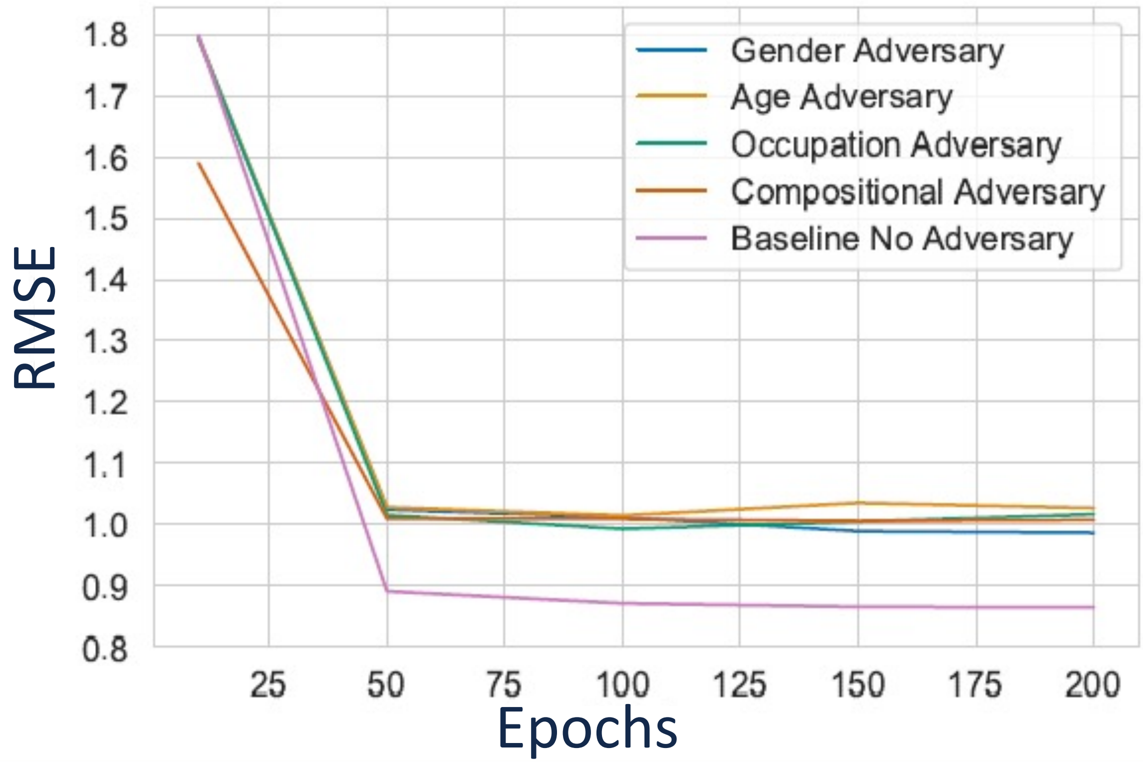
Compositional Fairness: Experiment

- **Task:** Classifying the sensitive attribute from the learned node embeddings
  - **Baseline methods:** Each adversary is a 2-layer MLP
    - Baseline (no adversary): Vanilla model train without fairness consideration
    - Independent adversary: independent adversarial model for each attribute
    - Compositional adversary: The proposed full compositional model
- **Observations**
  - Accuracy of compositional adversary is no better than majority classifier
  - Performance of compositional adversary is at the same level with independent adversaries

MovieLens1M	Baseline No Adversary	Gender Adversary	Age Adversary	Occupation Adversary	Comp. Adversary	Majority Classifier	Random Classifier	
Gender	0.712	0.532	0.541	0.551	0.511	0.5	0.5	AUC
Age	0.412	0.341	0.333	0.321	0.313	0.367	0.141	Micro F1
Occupation	0.146	0.141	0.108	0.131	0.121	0.126	0.05	

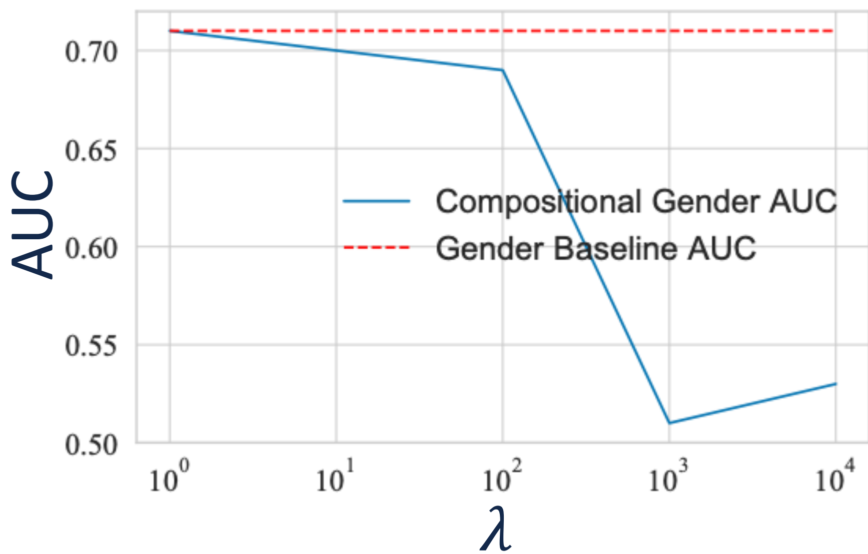
Compositional Fairness: Experiment

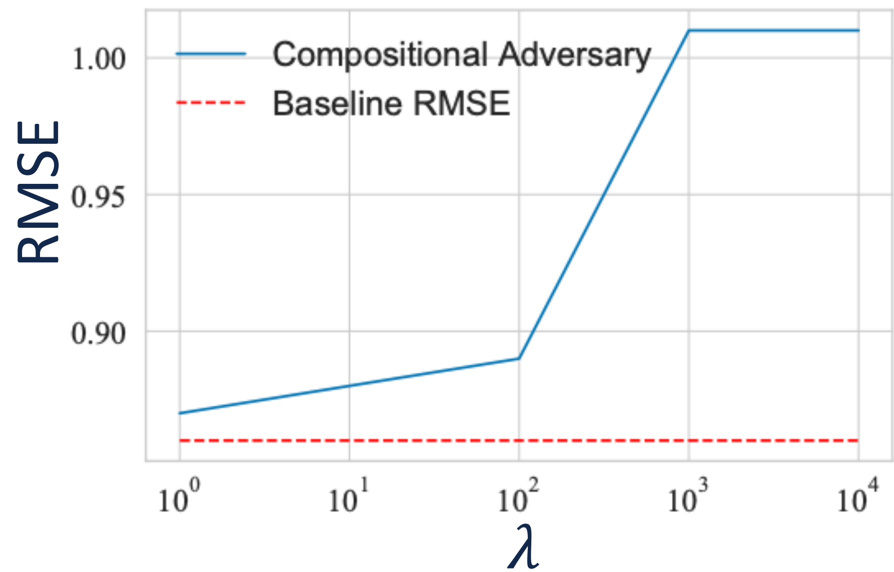
- **Task:** Recommendation
- **Observation:** There is only a small increase in root mean squared error (RMSE) compared with the vanilla model



Compositional Fairness: Experiment

- **Task:** Trade-off between fairness and recommendation quality
  - **Fairness:** Measured by regularization hyperparameter λ
  - **Recommendation quality:** Measured by AUC and RMSE
- **Observation:** The proposed method achieves a good balance between fairness and recommendation performance





Overview of Part I

Group Fairness on Graphs

Fairness on Graph Ranking

- Fairness-aware PageRank

Fairness on Graph Clustering

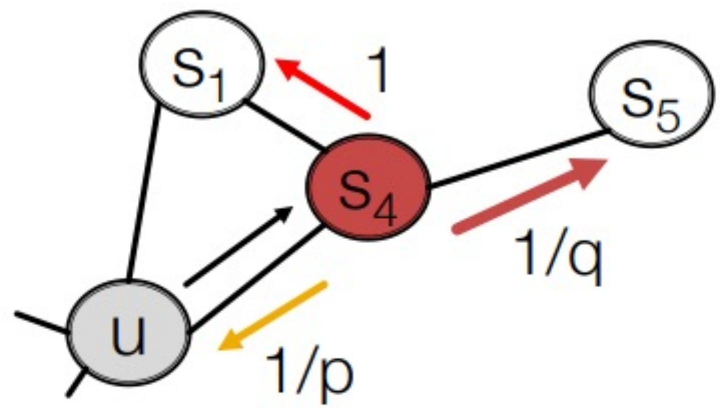
- Fairness-aware spectral clustering

Fairness on Graph Embedding

- Adversarial learning-based method
- Random walk-based method
- Bayesian method

Preliminary: node2vec

- **Goal:** Learn node embeddings that are predictive of nodes in its neighborhood
- **Key idea:** Skip-gram model with biased random walk
  - The biased random walk learns
    - Structural equivalence in BFS fashion
    - Homophily in DFS fashion
  - **Example**
    - **Return parameter** p: How fast the walk **explores** the neighborhood of the starting node
    - **In-out parameter** q: How fast the walk **leaves** the neighborhood of the starting node

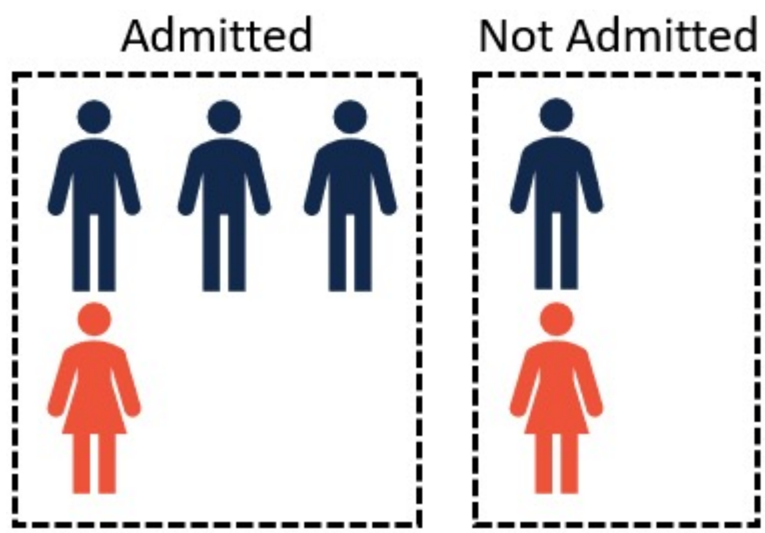


Fairness in Graph Embedding

- **Statistical parity**
  - **Given:** (1) A sensitive attribute $\mathcal{S}$; (2) multiple demographic groups $\mathcal{G}^{\mathcal{S}}$ partitioned by $\mathcal{S}$
  - **Extension to multiple groups:** Variance among the acceptance rates of each group in $\mathcal{G}^{\mathcal{S}}$

$$\text{bias}^{\text{SI}}(\mathcal{G}^{\mathcal{S}}) = \text{Var}\big(\{\text{acceptance}-\text{rate}(G^{\mathcal{S}}) \,|\, G^{\mathcal{S}} \in \mathcal{G}^{\mathcal{S}}\}\big)$$

- **Example:** A network of four  and two 
  - acceptance−rate()=3/4
  - acceptance−rate()=1/2
  - $\text{bias}^{\text{SI}} = \text{Var}\left(\left\{\frac{3}{4}, \frac{1}{2}\right\}\right) = \frac{1}{64}$



[1] Rahman, T., Surma, B., Backes, M., & Zhang, Y.. FairWalk: Towards Fair Graph Embedding. IJCAI 2019.

Fairness in Graph Embedding

- **Equality of representation - network level**
  - **Intuition:** Among all recommendations in the network, measure the bias as the variance of the number of recommendations from each demographic group
  - **Formulation**

$$\text{bias}^{\text{ER}_{\text{group}}}\left(\mathcal{G}^{\mathcal{S}}\right) = \text{Var}\left(\left\{N\left(G^{\mathcal{S}}\right) \middle| G^{\mathcal{S}} \in \mathcal{G}^{\mathcal{S}}\right\}\right)$$

- **Example:** In a social network of  and 
  - **Total recommendations**

  - $N($$) = 4$ and $N($$) = 2$
  - $\text{bias}^{\text{ER}_{\text{group}}} = \text{Var}(\{4,2\}) = 1$

Fairness in Graph Embedding

- **Equality of representation - user Level**
  - **Z-share:** Among recommendations $\rho(u)$ given to a specific user u, measure the fraction of users having sensitive value z

$$\mathrm{z{-}share}(u) = \frac{|\rho_z(u)|}{|\rho(u)|}$$

  - **Intuition:** Measure the bias as the difference between a fair fraction $\frac{1}{|\mathcal{Z}^{\mathcal{S}}|}$ and the average z-share over all users U

$$\mathrm{bias}^{\mathrm{ER_{user}}}(z) = \frac{1}{|\mathcal{Z}^{\mathcal{S}}|} - \frac{\sum_{u \in U} \mathrm{z{-}share}(u)}{|U|}$$

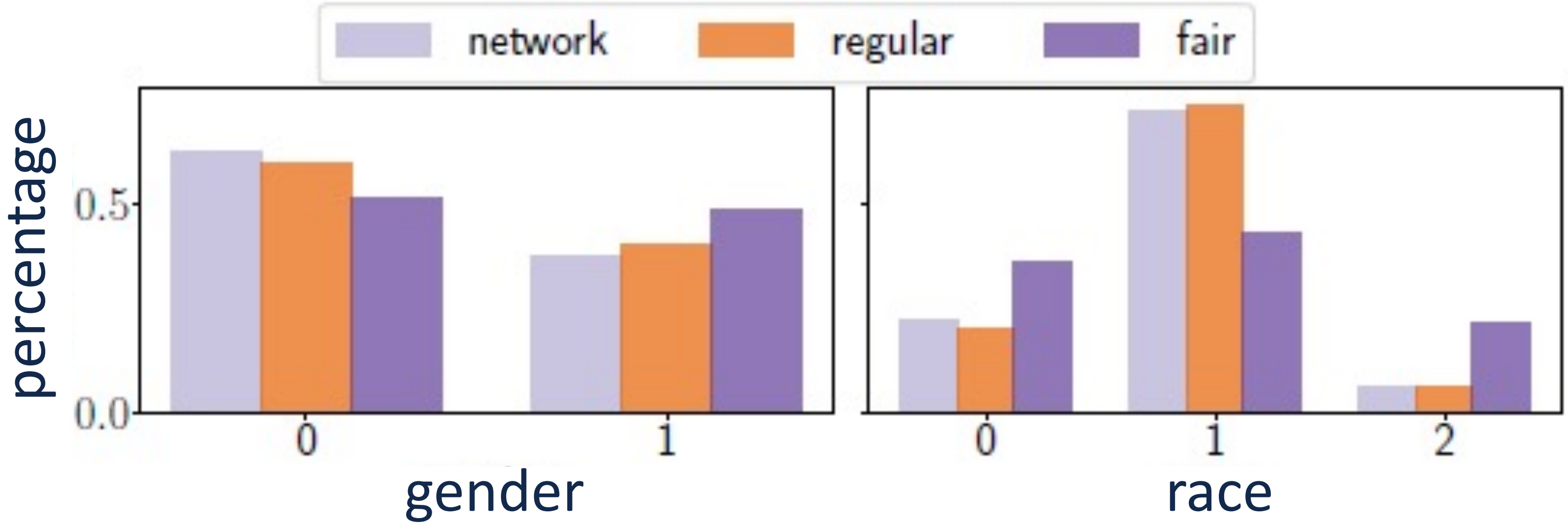
- **Example:** For *any* user u in the social network of ten  and ten 
  - $|\mathcal{Z}^{\mathcal{S}}| = |\{$  ,  $\}| = 2$ and fair fraction $\frac{1}{|\mathcal{Z}^{\mathcal{S}}|} = \frac{1}{2}$
  - The recommendations w.r.t. any user u are constant: 
  - Let $z =$  , we know $\rho_z(u) = 1$ and $\rho(u) = 3$
  - $\mathrm{z{-}share}(u) = \frac{|\rho_z(u)|}{|\rho(u)|} = \frac{1}{3}$
  - $\mathrm{bias}^{\mathrm{ER_{user}}}($  $) = \frac{1}{2} - \frac{\sum_u 1/3}{20} = \frac{1}{6}$

Fairwalk: Solution

- **Key idea:** Modify the random walk procedure in node2vec
- **Steps of Fairwalk**
  - Partition neighbors into demographic groups
  - Assign equal probability to each demographic group
  - Select a demographic group to walk to
  - Randomly select a node within the chosen demographic group

Fairwalk: Example

- **Example:** Ratio of each demographic group
  - Original network vs. regular random walk vs. fair random walk

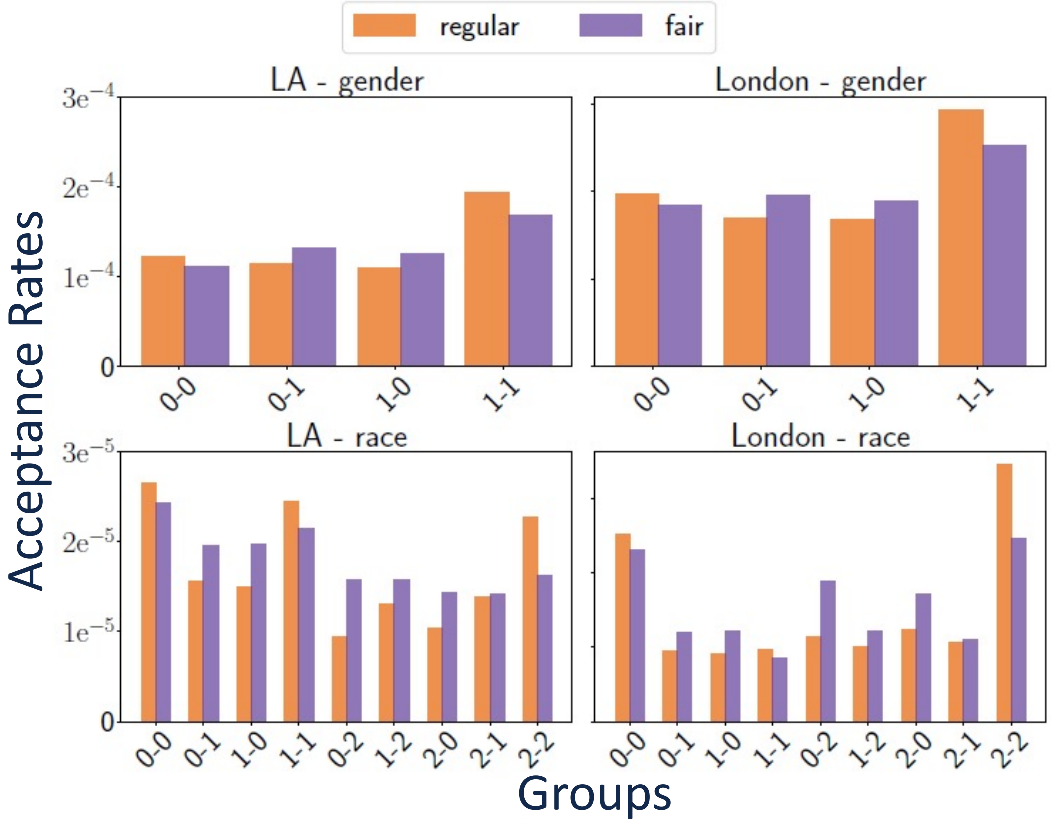


Fairwalk vs. Existing Works

- **Fairwalk vs. node2vec**
  - **Node2vec:** skip-gram model + walk sequences by original random walk
  - **Fairwalk:** skip-gram model + walk sequences by fair random walk
- **Fairwalk vs. fairness-aware PageRank**
  - **Fairness-aware PageRank:** The minority group should have a certain proportion of PageRank probability mass
  - **Fairwalk:** All demographic group have the same random walk transition probability mass

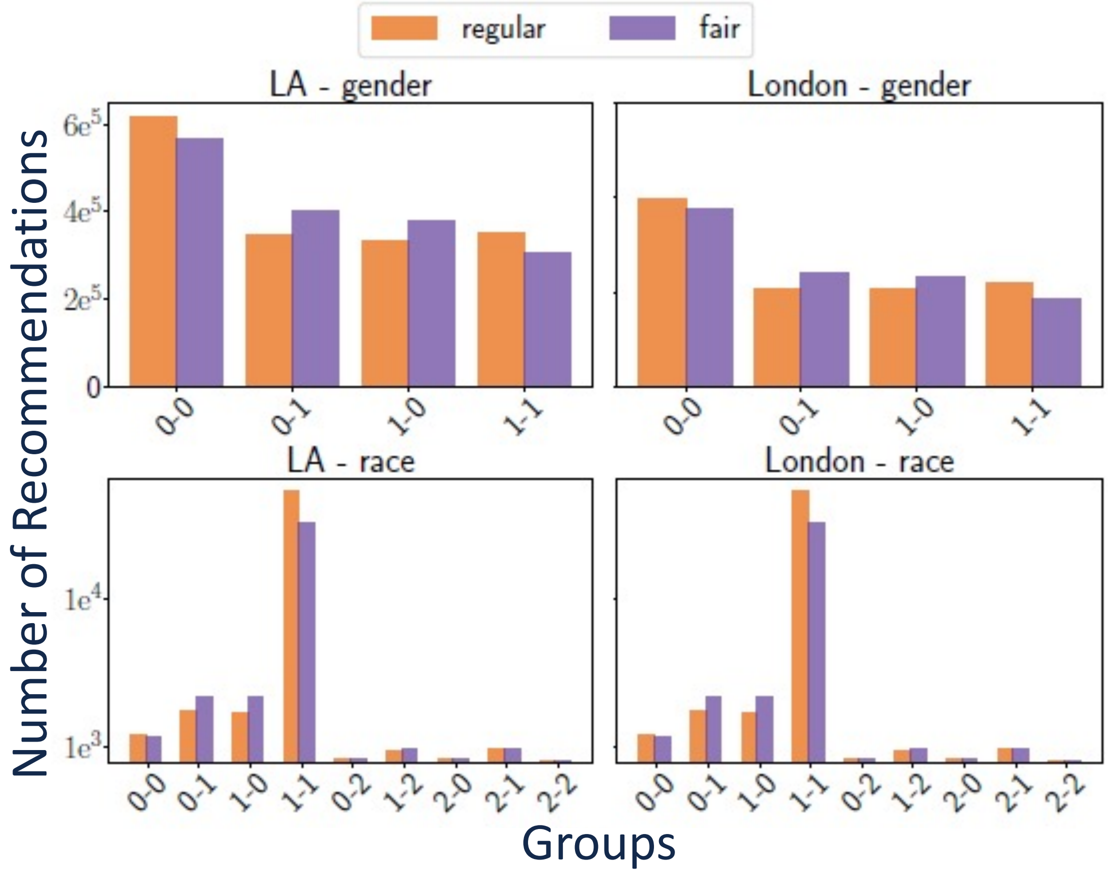
Fairwalk: Statistical Parity

- **Observations**
  - Fairwalk achieves a more balanced acceptance rates among groups
  - Fairwalk increases the fraction of cross-group recommendations



Fairwalk: Network-level Equality of Representation

- **Observation:** Fairwalk increases the number of recommendation for underrepresented groups



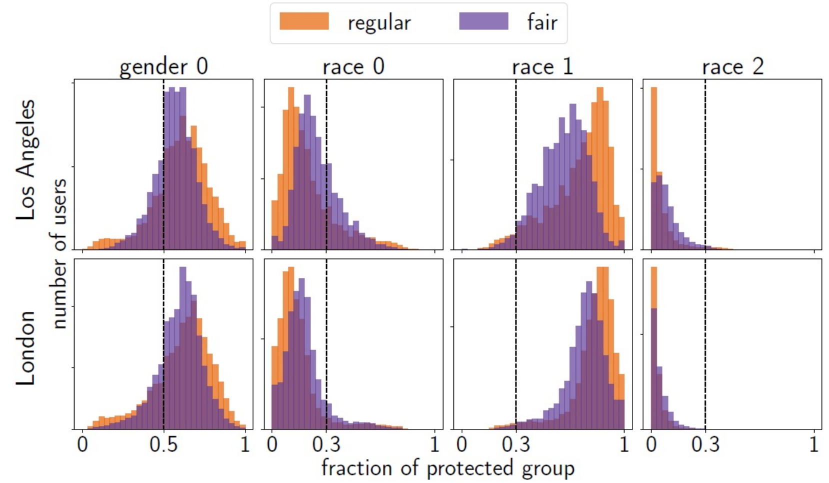
Fairwalk: User-level Equality of Representation

- **Observations**
  - Fairwalk decreases the user-level bias
  - Z-share distribution of Fairwalk leans towards the fair fraction

Bias for User-level Equality of Representation

		gender		race		
		0	1	0	1	2
LA	network	0.104	0.104	0.117	0.392	0.275
LA	node2vec	0.103	0.103	0.115	0.387	0.272
LA	fairwalk	0.068	0.068	0.054	0.288	0.234
London	network	0.097	0.097	0.183	0.481	0.298
London	node2vec	0.112	0.112	0.176	0.474	0.298
London	fairwalk	0.095	0.095	0.135	0.417	0.282

Z-share Distribution



Overview of Part I

Group Fairness on Graphs

Fairness on Graph Ranking

- Fairness-aware PageRank

Fairness on Graph Clustering

- Fairness-aware spectral clustering

Fairness on Graph Embedding

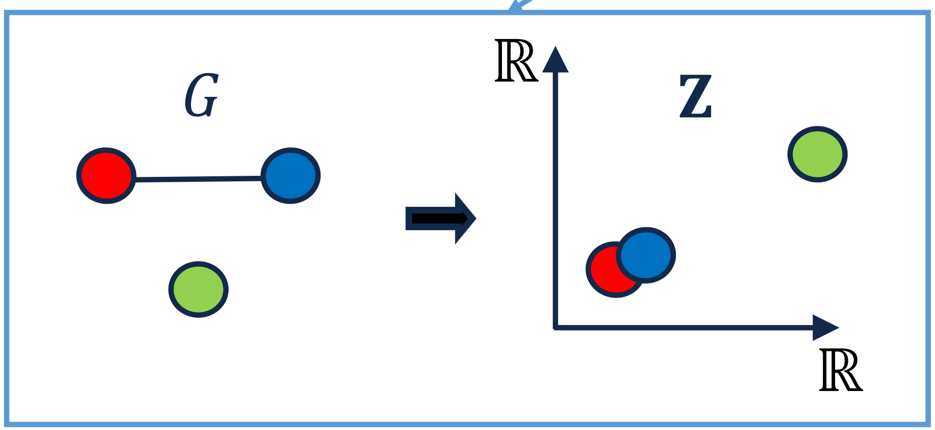
- Adversarial learning-based method
- Random walk-based method
- Bayesian method

Preliminary: Conditional Network Embedding

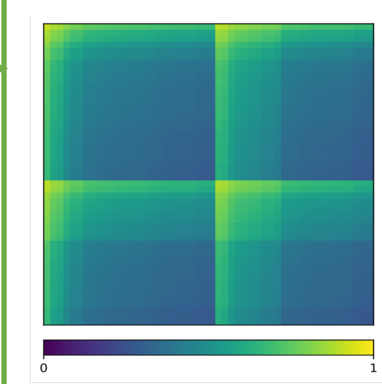
- **Goal:** Find an embedding $\mathbf{Z}$ using maximum likelihood estimation of $\mathrm{P}(G|\mathbf{Z})$ with the empirical graph G
- **Key idea:** Bayes rule

$$\mathrm{P}(G|\mathbf{Z}) = \frac{\boxed{\mathrm{P}(\mathbf{Z}|G)} \cdot \boxed{\mathrm{P}(G)}}{\boxed{\mathrm{P}(\mathbf{Z})}}$$

Likelihood: $\mathrm{P}(\mathbf{Z}|G)$



Prior: $P(G)$



Marginalization: $P(\boldsymbol{Z})$

$$P(\mathbf{Z}) = \sum_{G} P(\mathbf{Z}|G)P(G)$$

[1] Kang, B., Lijffijt, J., & De Bie, T.. Conditional Network Embeddings. ICLR 2019.

Preliminary: Conditional Network Embedding

- **Key idea:** Modeling degree distribution into prior
  - **Given**
    - (1) **A degree distribution:** p_{deg}
    - (2) **A random graph:** $G = (V, E)$ with node set V, edge set E
    - (3) **Degree constraint:** degree distribution of G is $p_{\text{deg}}(G) = p_{\text{deg}}$
  - **Find:** A maximum entropy distribution $p(G)$ that satisfies degree constraint
  - **Result:** $\mathrm{P}(G) = \prod_{(u,v)\in E} p_{uv} \prod_{(u,v)\notin E} (1 - \boxed{p_{uv}})$

Probability that node u and node v are connected

An instance of G

	0	1	0	0	0	0	0	0
Node u	1	0	1	1	1	1	0	1
	0	1	0	0	1	0	0	1
	0	1	0	0	0	0	1	0
	0	1	1	0	0	0	0	0
	0	1	0	0	0	0	0	0
	0	0	0	1	0	0	0	0
	0	1	1	0	0	0	0	0

Degree distribution p_{deg}

1
6
3
2
2
1
1
2

Maximize entropy of G

s.t. **degree constraint**

For every u

$\sum_v p_{uv}$ = degree of u in p_{deg}

Prior $\mathrm{P}(G)$

.0	.6	.1	.1	.1	.0	.0	.1
.6	.9	.9	.8	.8	.6	.6	.8
.1	.9	.6	.4	.4	.1	.1	.4
.1	.8	.4	.2	.2	.1	.1	.2
.1	.8	.4	.2	.2	.1	.1	.2
.0	.6	.1	.1	.1	.0	.0	.1
.0	.6	.1	.1	.1	.0	.0	.1
.1	.8	.4	.2	.2	.1	.1	.2

DeBayes: Fairness Measures

- **Tasks**
  - **Task #1:** Fair network embedding

    ↓ Downstream task

  - **Task #2:** Link prediction
  - **Goal:** Debias the embeddings in order to debias the link prediction
- **Fairness measures**
  - **Low-level fairness:** Fairness for network embedding
  - **High-level fairness:** Fairness for link prediction

[1] Buyl, M., & De Bie, T.. DeBayes: A Bayesian Method for Debiasing Network Embeddings. ICML 2020.

DeBayes: Fairness Measures

- **Fairness for network embedding:** Representation bias (RB)

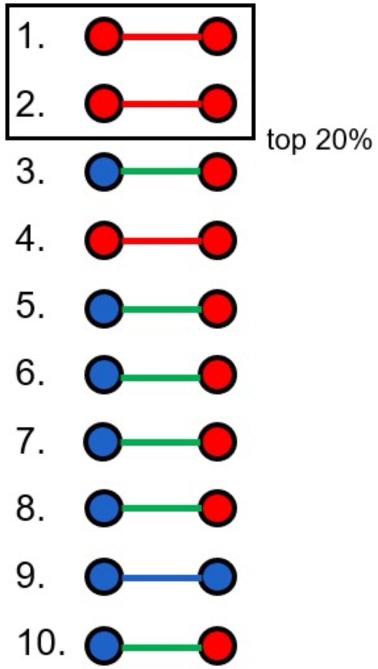
$$\mathrm{RB} = \sum_{s \in S} \frac{1}{|V_s|} \mathrm{AUC}(\{\mathrm{P}(A(v)|\mathbf{z}_v)|\forall v \in V_s\})$$

  - s: A sensitive attribute value
  - $A(v)$: Node v's sensitive attribute value
  - V_s: The set of nodes with sensitive value s, i.e., $\{v|A(v) = s\}$
  - $\mathrm{P}(A(v)|\mathbf{z}_v)$: Probability of predicting sensitive attribute value $A(v)$ of node v using its embedding $\mathbf{z}_v$
  - $\frac{1}{|V_s|}$: Weighted average, weighted by size of demographic group
  - $\mathrm{AUC}(\cdots)$: One-vs.-rest AUC
- **Intuition**
  - Fair embedding should not infer the ground-truth sensitive attribute value → low $\mathrm{P}(A(v)|\mathbf{z}_v)$
  - low $\mathrm{P}(A(v)|\mathbf{z}_v)$ → low true positive rate → low AUC

[1] Buyl, M., & De Bie, T.. DeBayes: A Bayesian Method for Debiasing Network Embeddings. ICML 2020.

DeBayes: Fairness Measures

- **Fairness for link prediction**
  - **Goal:** Main concern should be fairness in downstream task
  - **Statistical Parity:** Equal acceptance rate
  - **Equal Opportunity:** Equal true positive rate
  - **Accuracy Rate Parity:** Balance in top k% edge predictions
- **Example:** Accuracy rate parity
  - 10 edge predictions in total
  - Top 2 (i.e., top 20%) predictions in consideration
  - **Highly biased:** Both are red-red links



[1] Buyl, M., & De Bie, T.. DeBayes: A Bayesian Method for Debiasing Network Embeddings. ICML 2020.

DeBayes: Key Idea

- **Two types of prior**
  - **The biased prior:** a prior with information about sensitive attribute
  - **The oblivious prior:** a prior without information about sensitive attribute
    - E.g., The prior used in conditional network embedding
- **Key idea:** Debias embedding by modeling bias in prior
  - Learn embeddings with the biased prior
  - Evaluate embeddings with the oblivious prior
- **Question:** How to find the biased prior?

DeBayes: The Biased Prior

- **Given**
  - **(1) Random graph:** G (e.g., an instance as shown below)
  - **(2) Sensitive attribute:** A = {blue, red}
  - **(3) Fairness-aware distribution:** Blue/red degree distributions $p_{\deg}^{(\text{blue})}$, $p_{\deg}^{(\text{red})}$
  - **(4) Fairness-aware degree constraint:** $p_{\deg}^{(\text{blue})}(G) = p_{\deg}^{(\text{blue})}$ and $p_{\deg}^{(\text{red})}(G) = p_{\deg}^{(\text{red})}$
- **Find:** Maximum entropy distribution $p(G|A)$ that satisfies the fairness-aware degree constraint

An instance of G

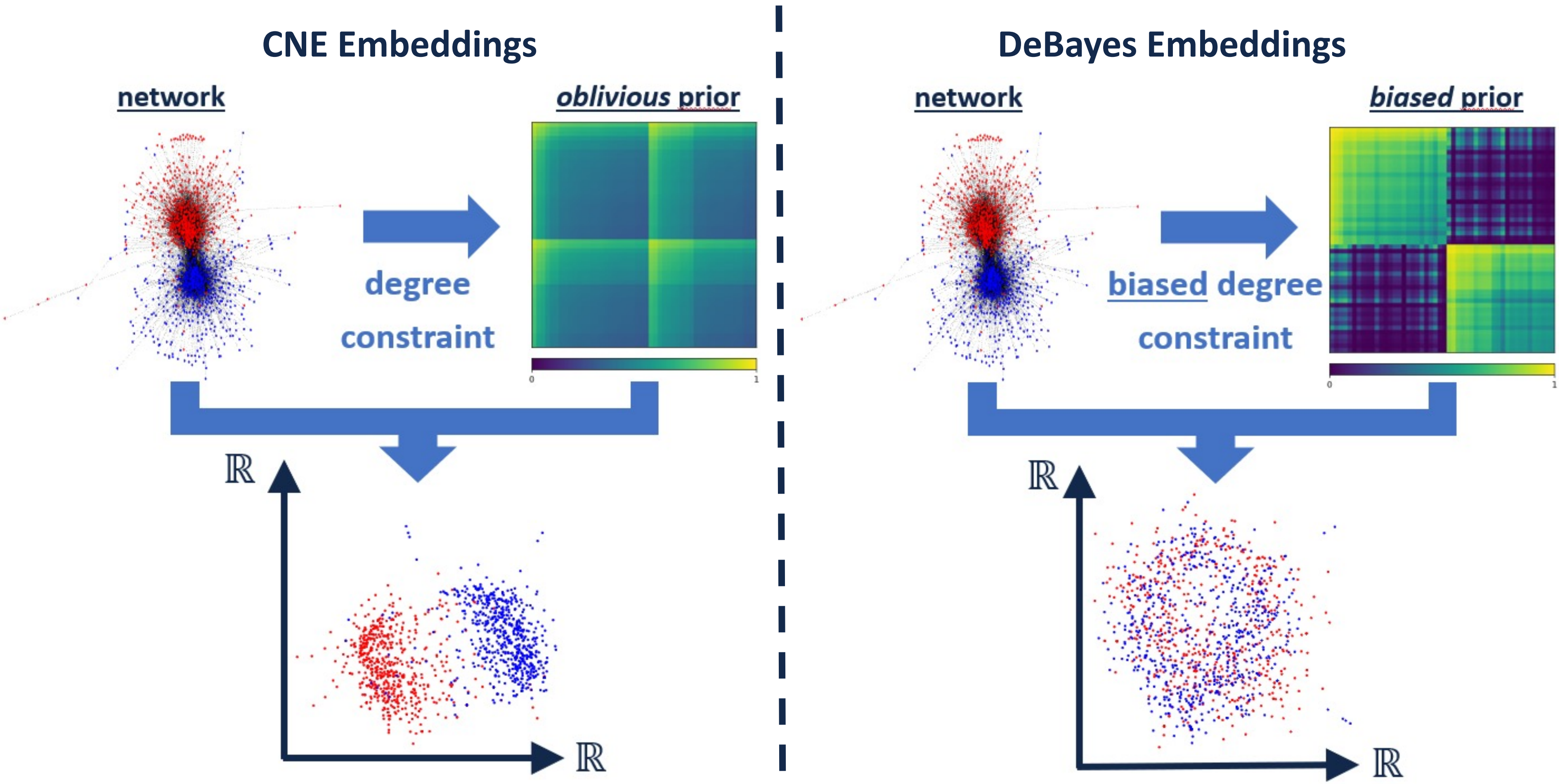
	blue	blue	blue	blue	blue	red	red	red
blue	0	1	0	0	0	0	0	0
Node u (blue)	1	0	1	1	1	0	0	1
blue	0	1	0	0	0	0	0	1
blue	0	1	0	0	1	0	1	0
blue	0	1	0	1	0	0	0	0
red	0	0	0	0	0	0	1	1
red	0	0	0	1	0	1	0	0
red	0	1	1	0	0	1	0	0

$p_{\deg}^{(\text{blue})}$	$p_{\deg}^{(\text{red})}$
1	0
4	1
1	1
2	1
2	0
0	2
1	1
2	1

Maximize entropy of G given A s.t. **fairness–aware degree constraints**

- Blue degree distribution $p_{\deg}^{(\text{blue})}$ for node u in G
- Red degree distribution $p_{\deg}^{(\text{red})}$ for node u in G

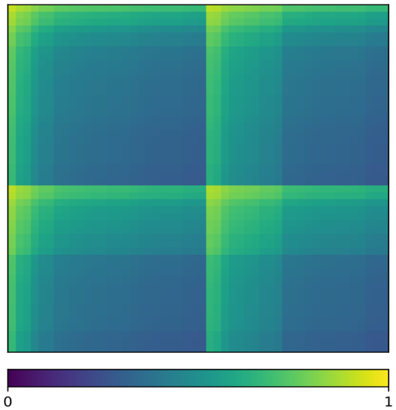
DeBayes vs. Conditional Network Embedding (CNE)



Observation: Embeddings are less biased toward the node color

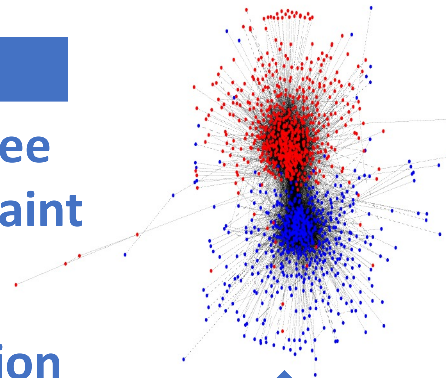
DeBayes: Workflow

$P(G|\mathbf{Z}, A)$



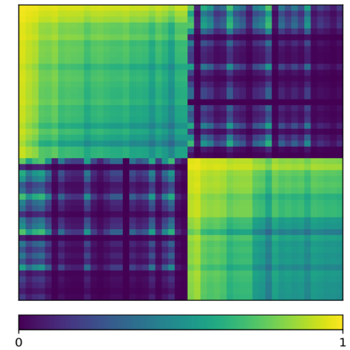
Degree constraint

G



Biased degree constraint

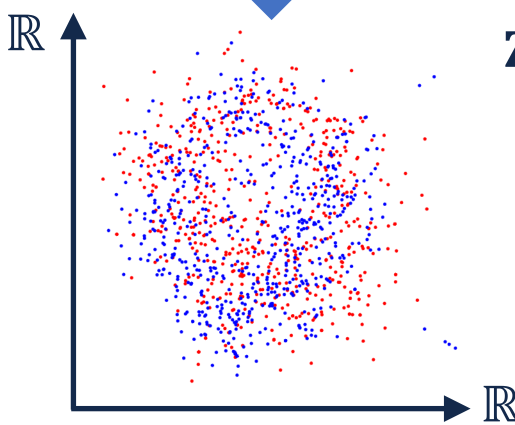
$P(G|A)$



Evaluation (link prediction)

Training

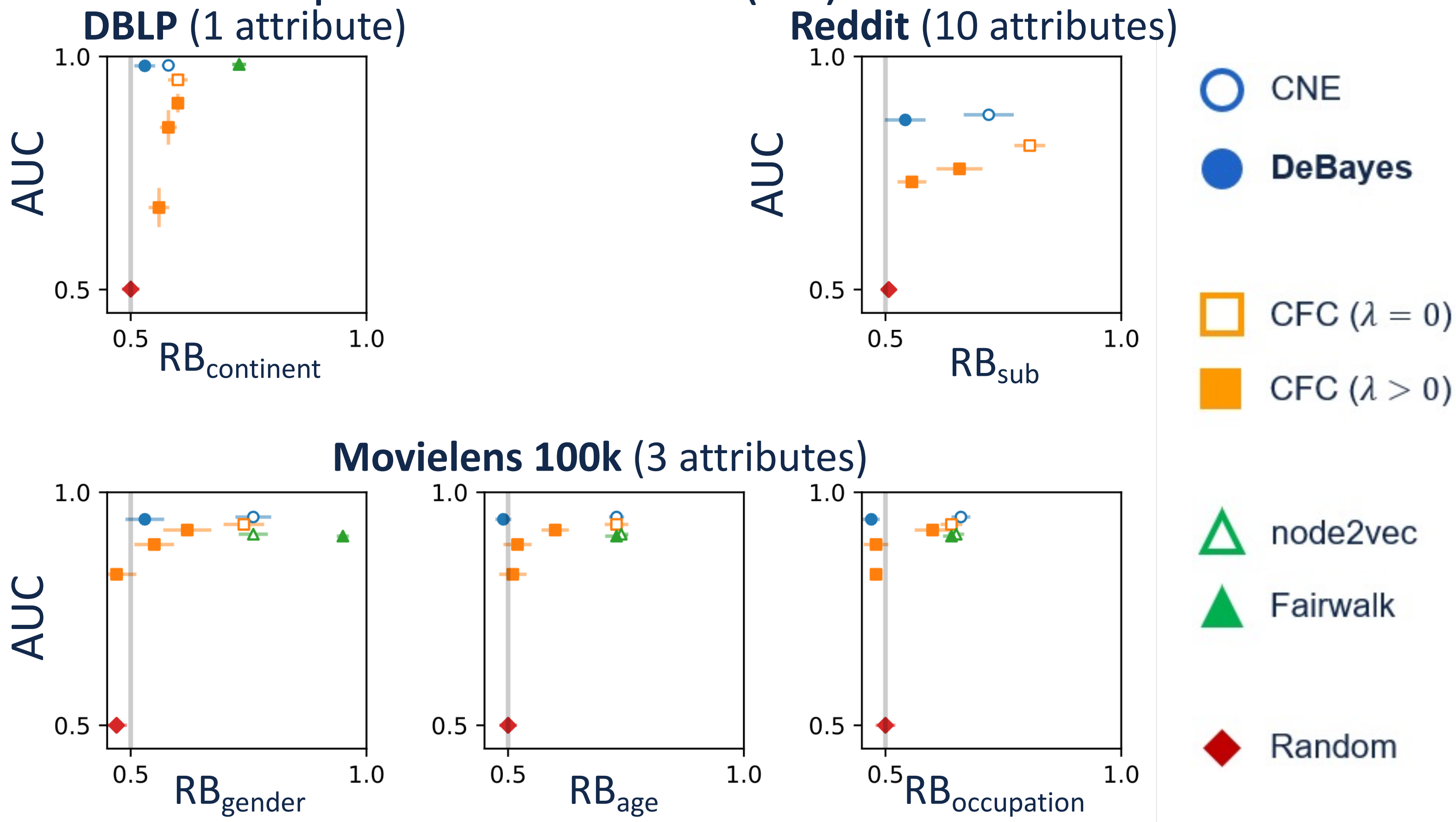
$$P(G|\mathbf{Z}_{deb}) = \frac{P(\mathbf{Z}_{deb}|G)P(G)}{P(\mathbf{Z}_{deb})}$$



$$\begin{aligned}\mathbf{Z}_{\text{deb}} &= \operatorname*{argmax}_{\mathbf{Z}} P(G|\mathbf{Z}, A) \\ &= \operatorname*{argmax}_{\mathbf{Z}} \frac{P(\mathbf{Z}|G, A)P(G|A)}{P(\mathbf{Z}|A)} \\ &= \operatorname*{argmax}_{\mathbf{Z}} \frac{P(\mathbf{Z}|G)P(G|A)}{P(\mathbf{Z}|A)}\end{aligned}$$

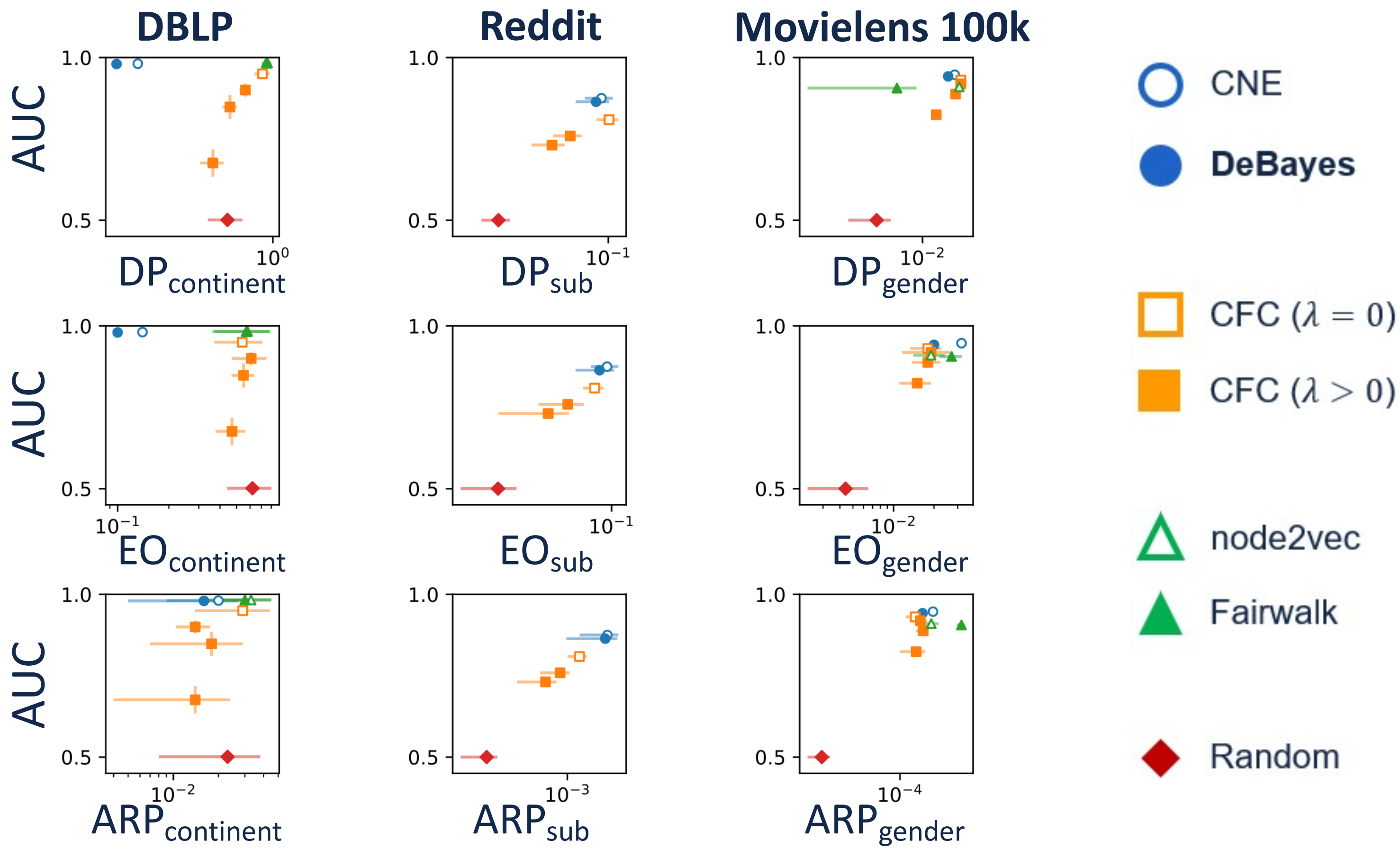
Fairness for Network Embedding: Experiment

- **Observation:** DeBayes achieves the best trade-off between link prediction AUC and representation bias (RB)

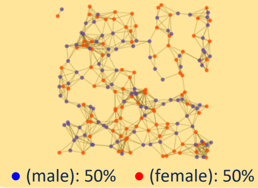


Fairness for Link Prediction: Experiment

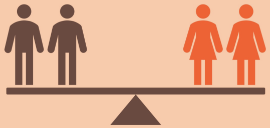
- **Observation:** DeBayes achieves the best trade-off between link prediction AUC and statistical parity (DP), equal opportunity (EO) and accuracy rate parity (ARP)



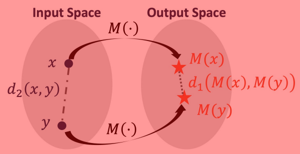
Roadmap



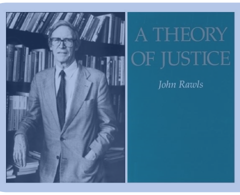
Introduction



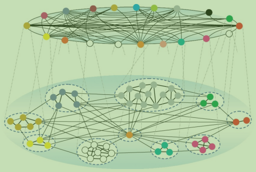
Part I: Group Fairness on Graphs



Part II: Individual Fairness on Graphs



Part III: Other Fairness Definitions on Graphs



Part IV: Future Research Directions

Overview of Part II

Individual Fairness on Graphs

Optimization-based Approach

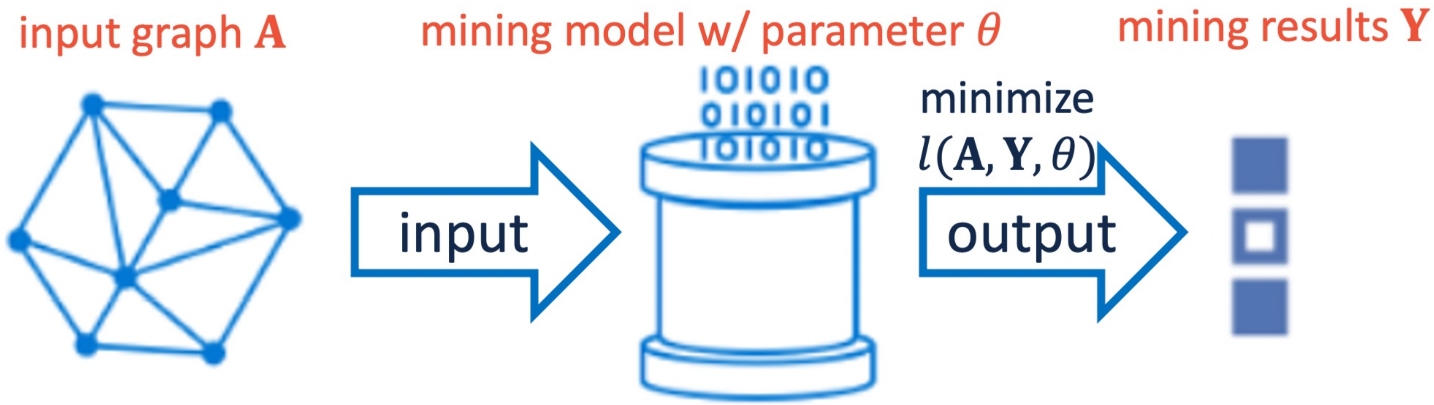
- Laplacian regularization-based individual fairness

Ranking-based Approach

- Ranking-based individual fairness

Preliminary: Graph Mining Pipeline

- **Graph mining: An optimization perspective**



– **Input:**
  - Input graph $\mathbf{A}$
  - Model parameters θ

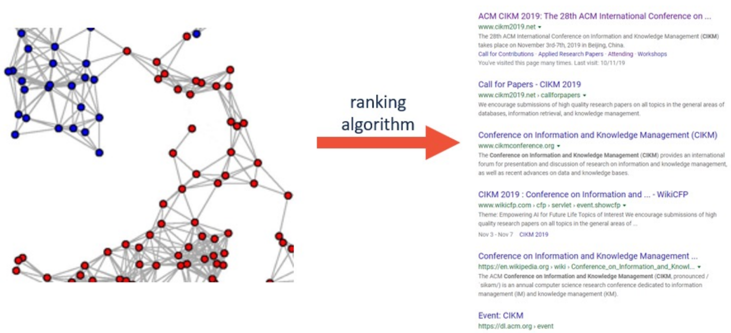
Minimize task-specific loss function $l(\mathbf{A}, \mathbf{Y}, \theta)$

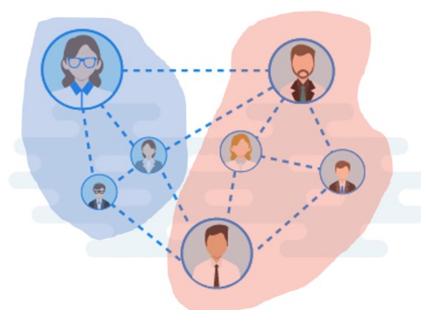
– **Output:** Mining results $\mathbf{Y}$
  - **Examples:** Ranking vectors, class probabilities, embedding

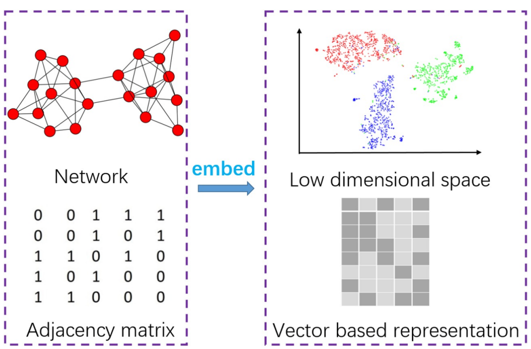
Preliminary: Classic Graph Mining Algorithms

Examples of Classic Graph Mining Algorithm

Mining Task	Task Specific Loss Function $l()$	Mining Result Y^*	Parameters
PageRank	$\min_{\mathbf{r}} c\mathbf{r}'(\mathbf{I}-\mathbf{A})\mathbf{r} + (1-c)\lVert\mathbf{r}-\mathbf{e}\rVert_F^2$	PageRank vector $\mathbf{r}$	damping factor c teleportation vector $\mathbf{e}$
Spectral Clustering	$\min_{\mathbf{U}} \mathrm{Tr}\,(\mathbf{U}'\mathbf{L}\mathbf{U})$ $\mathrm{s.t.}\quad \mathbf{U}'\mathbf{U}=\mathbf{I}$	eigenvectors $\mathbf{U}$	# clusters k
LINE (1st)	$\min_{\mathbf{X}} \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{A}[i,j]\left(\log g(-\mathbf{X}[j,:]\mathbf{X}[i,:]')\right)$ $+b\mathbb{E}_{j'\sim P_n}[\log g(-\mathbf{X}[j',:]\mathbf{X}[i,:]')]$	embedding matrix $\mathbf{X}$	embedding dimension d # negative samples b

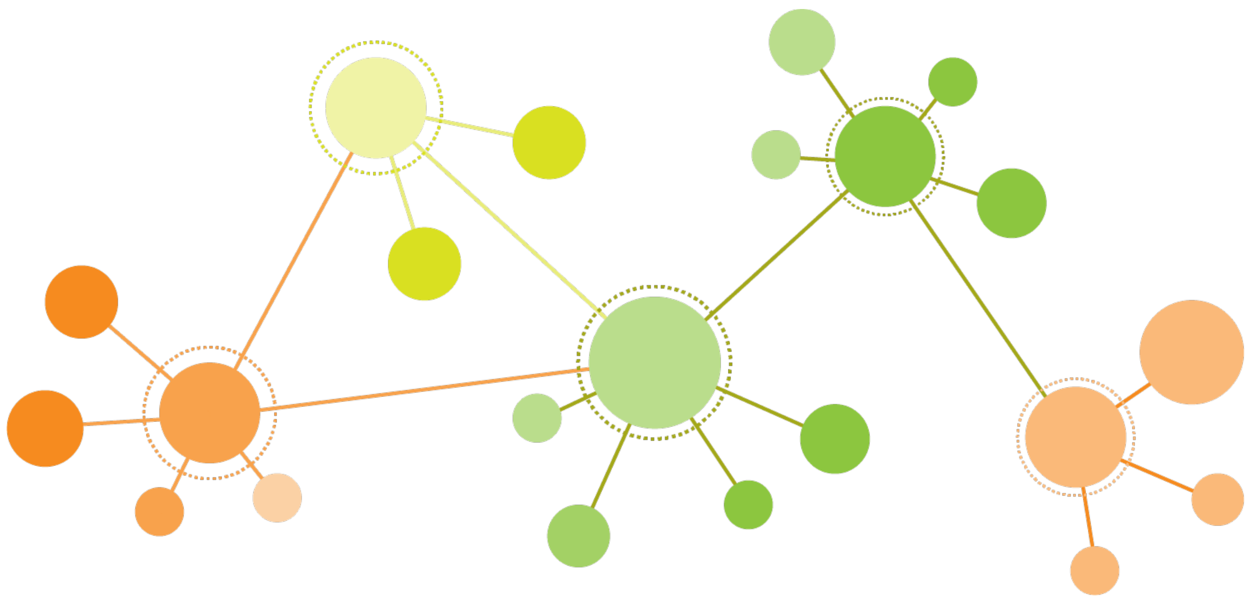






InFoRM: Individual Fairness on GRaph Mining

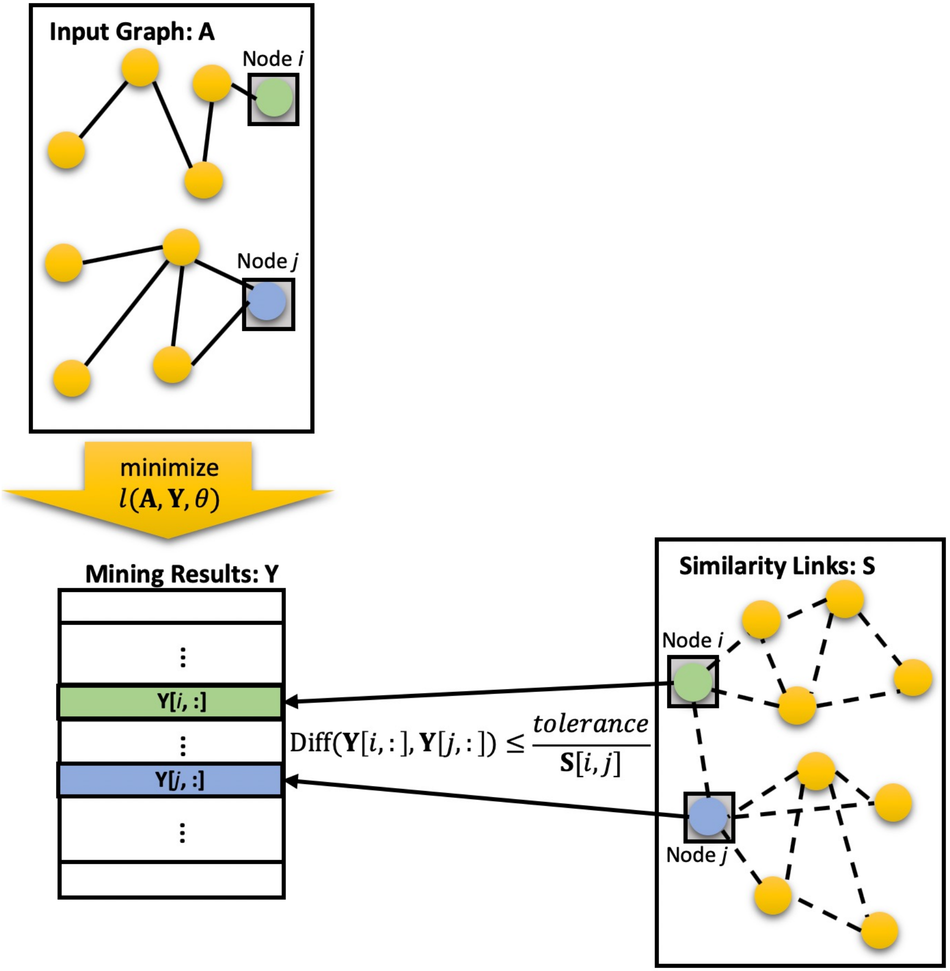
- **Research questions**
  - **Q1. Measures:** How to quantitatively measure individual bias?
  - **Q2. Algorithms:** How to enforce individual fairness?
  - **Q3. Cost:** What is the cost of individual fairness?



[1] Kang, J., He, J., Maciejewski, R., & Tong, H.. InFoRM: Individual Fairness on Graph Mining. KDD 2020.

Problem Definition: InFoRM Measures

- **Questions**
  - How to determine if the mining results are fair?
  - How to quantitatively measure the overall bias?
- **Input**
  - Node-node similarity matrix $\mathbf{S}$
    - Non-negative, symmetric
  - Graph mining algorithm $l(\mathbf{A}, \mathbf{Y}, \theta)$
    - Loss function $l(\cdot)$
    - Additional set of parameters θ
  - Fairness tolerance parameter ϵ
- **Output**
  - Binary decision on whether the mining result is fair
  - Individual bias measure $\mathrm{Bias}(\mathbf{Y}, \mathbf{S})$

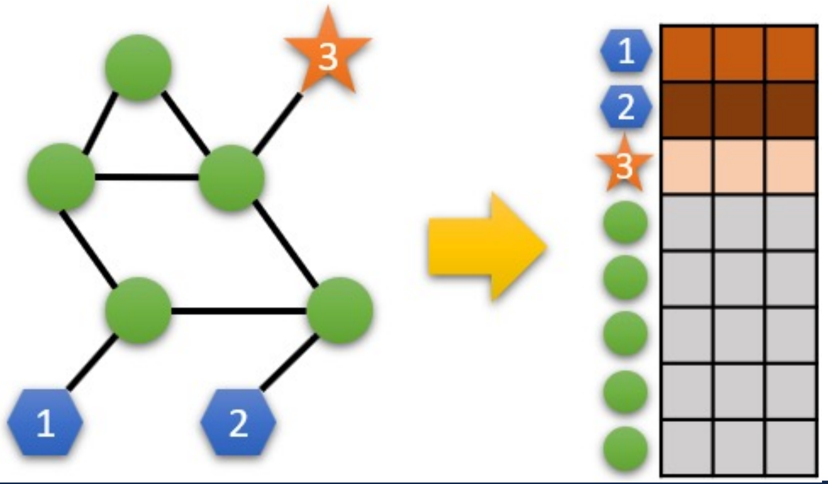


Measuring Individual Bias: Formulation

- **Principle:** Similar nodes → similar mining results
- **Mathematical formulation**

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

  - **Intuition:** If $\mathbf{S}[i,j]$ is high, $\frac{\epsilon}{\mathbf{S}[i,j]}$ is small → push $\mathbf{Y}[i,:]$ and $\mathbf{Y}[j,:]$ to be more similar
  - **Observation:** Inequality should hold for *every* pairs of nodes i and j
    - **Problem:** Too restrictive to be fulfilled
- **Relaxed criteria:** $\sum_{i=1}^{n}\sum_{j=1}^{n}\|\mathbf{Y}[i,:] - \mathbf{Y}[j,:]\|_F^2\mathbf{S}[i,j] = 2\mathrm{Tr}(\mathbf{Y}'\mathbf{L_S}\mathbf{Y}) \leq m\epsilon = \delta$



Measuring Individual Bias: Solution

- **InFoRM (Individual Fairness on GRaph Mining)**
  - **Given:** (1) A graph mining result $\mathbf{Y}$; (2) a symmetric similarity matrix $\boldsymbol{S}$; and (3) a constant fairness tolerance δ
  - $\mathbf{Y}$ is individually fair w.r.t. $\boldsymbol{S}$ if it satisfies

$$\mathrm{Tr}(\mathbf{Y}'\mathbf{L}_{\mathbf{S}}\mathbf{Y}) \leq \frac{\delta}{2}$$

  - Overall individual bias is $\mathrm{Bias}(\mathbf{Y}, \mathbf{S}) = \mathrm{Tr}(\mathbf{Y}'\mathbf{L}_{\mathbf{S}}\mathbf{Y})$

[1] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., & Zemel, R.. Fairness through Awareness. ITCS 2012.

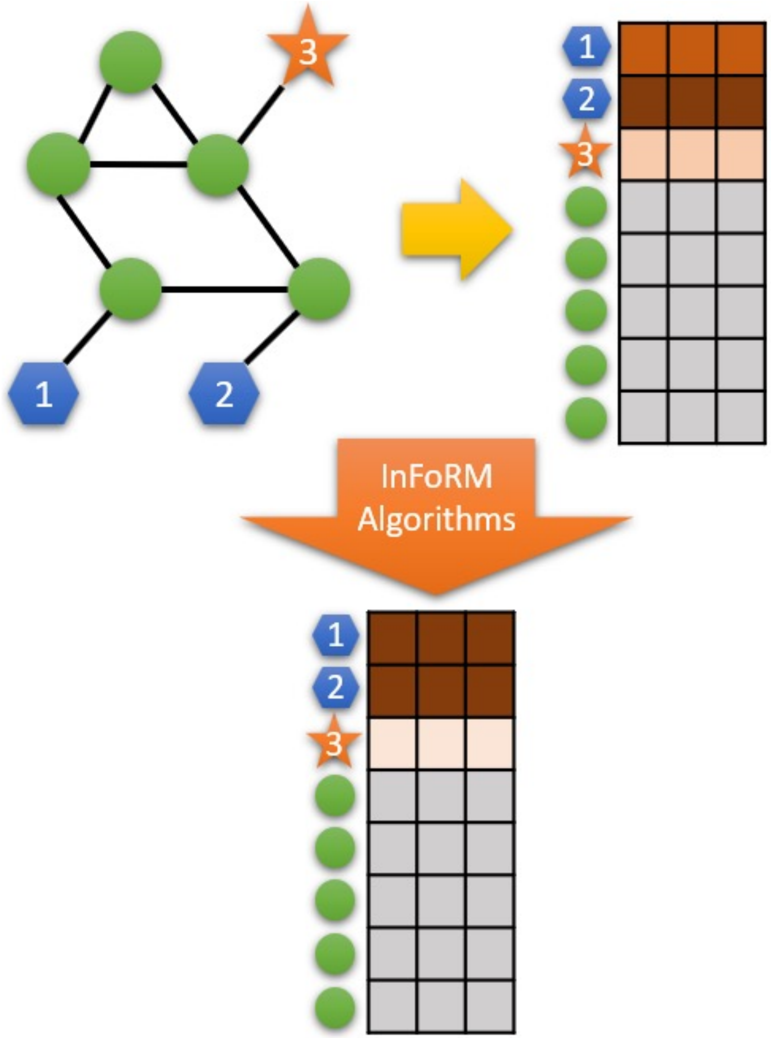
Lipschitz Property of Individual Fairness

- **Connection to Lipschitz Property**
  - $(\boldsymbol{D_1}, \boldsymbol{D_2})$**-Lipschitz property:** A function f is (D_1, D_2)-Lipschitz if it satisfies
$$D_1\big(f(i), f(j)\big) \leq L D_2(i, j), \forall (x, y)$$
    - L is Lipschitz constant
  - InFoRM naturally satisfies (D_1, D_2)-Lipschitz property as long as
    - $f(i) = \mathbf{Y}[i, :]$
    - $D_1\big(f(i), f(j)\big) = \|\mathbf{Y}[i, :] - \mathbf{Y}[j, :]\|_F^2$, $D_2(i, j) = \frac{1}{\mathbf{S}[i,j]}$
  - Lipschitz constant of InFoRM is ϵ

[1] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., & Zemel, R.. Fairness through Awareness. ITCS 2012.

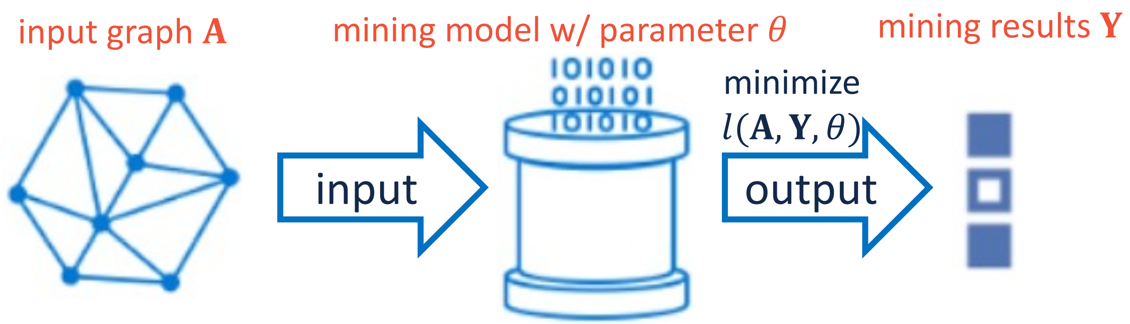
Problem Definition: InFoRM Algorithms

- **Question:** How to mitigate the bias of the mining results?
- **Input**
  - Node-node similarity matrix $\mathbf{S}$
  - Graph mining algorithm $l(\mathbf{A}, \mathbf{Y}, \theta)$
  - Individual bias measure $\mathrm{Bias}(\mathbf{Y}, \mathbf{S})$
    - Defined in the previous problem (InFoRM Measures)
- **Output:** Revised mining result $\mathbf{Y}^*$ that minimizes
  - Task-specific loss function $l(\mathbf{A}, \mathbf{Y}, \theta)$
  - Individual bias measure $\mathrm{Bias}(\mathbf{Y}, \mathbf{S})$



Mitigating Individual Bias: How To

- **Graph mining pipeline**



- **Observation:** Bias can be introduced/amplified in each component
  - **Solution:** Bias can be mitigated in each part
- **Algorithmic frameworks**
  - Debiasing the input graph
  - Debiasing the mining model
  - Debiasing the mining results

mutually complementary

Debiasing the Input Graph

- **Goal:** Bias mitigation via a pre-processing strategy
- **Intuition:** Learn a new topology of graph $\widetilde{\mathbf{A}}$ such that
  - $\widetilde{\mathbf{A}}$ is as similar to the original graph $\mathbf{A}$ as possible
  - Bias of mining results on $\widetilde{\mathbf{A}}$ is minimized
- **Optimization problem**

consistency in graph topology

$$\min_{\mathbf{Y}} \quad J = \left\|\widetilde{\mathbf{A}} - \mathbf{A}\right\|_F^2 + \alpha \mathrm{Tr}(\mathbf{Y}^T \mathbf{L}_{\mathbf{S}} \mathbf{Y})$$

bias measure

$$\text{s.t.} \quad \mathbf{Y} = \mathrm{argmin}_{\mathbf{Y}} \; l(\widetilde{\mathbf{A}}, \mathbf{Y}, \theta)$$

- **Challenge:** Bi-level optimization
  - **Solution:** Exploration of KKT conditions

[1] Kang, J., & Tong, H.. N2N: Network Derivative Mining. CIKM 2019.
[2] Mei, S., & Zhu, X.. Using Machine Teaching to Identify Optimal Training-Set Attacks on Machine Learners. AAAI 2015.

Debiasing the Input Graph

- Considering the KKT conditions,

$$\min_{\mathbf{Y}} \quad J = \left\|\widetilde{\mathbf{A}} - \mathbf{A}\right\|_F^2 + \alpha \mathrm{Tr}(\mathbf{Y}^T \mathbf{L}_{\mathbf{S}} \mathbf{Y})$$

$$\mathrm{s.t.} \quad \partial_{\mathbf{Y}} l\left(\widetilde{\mathbf{A}}, \mathbf{Y}, \theta\right) = 0$$

- **Proposed method**
  - (1) Fix $\widetilde{\mathbf{A}}$ ($\widetilde{\mathbf{A}} = \mathbf{A}$ at initialization), find $\mathbf{Y}$ using current $\widetilde{\mathbf{A}}$
  - (2) Fix $\mathbf{Y}$, update $\widetilde{\mathbf{A}}$ by gradient descent
  - (3) Iterate between (1) and (2)
- **Problem:** How to compute the gradient w.r.t. $\widetilde{\mathbf{A}}$?

Debiasing the Input Graph

Key component to calculate

- **Computing gradient w.r.t. $\widetilde{\mathbf{A}}$**

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

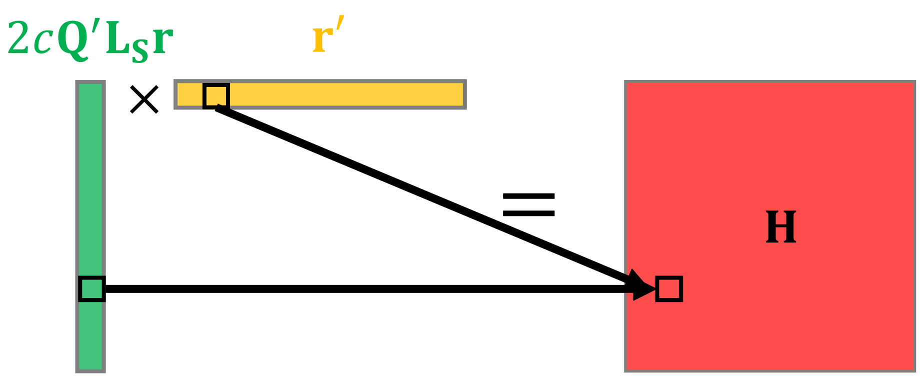
$$\frac{\mathrm{d}J}{\mathrm{d}\widetilde{\mathbf{A}}} = \begin{cases} \frac{\partial J}{\partial \widetilde{\mathbf{A}}} + (\frac{\partial J}{\partial \widetilde{\mathbf{A}}})' - \mathrm{diag}\left(\frac{\partial J}{\partial \widetilde{\mathbf{A}}}\right), & \text{if undirected} \\ \frac{\partial J}{\partial \widetilde{\mathbf{A}}}, & \text{if directed} \end{cases}$$

  - $\widetilde{\mathbf{Y}}$ satisfies $\partial_{\mathbf{Y}} l\left(\widetilde{\mathbf{A}}, \mathbf{Y}, \theta\right) = 0$
  - $\mathbf{H} = \left[\mathrm{Tr}\left(2\widetilde{\mathbf{Y}}\mathbf{L}_{\mathbf{S}}\frac{\partial \widetilde{\mathbf{Y}}}{\partial \widetilde{\mathbf{A}}[i,j]}\right)\right]$ is a matrix with $\mathbf{H}[i,j] = \mathrm{Tr}\left(2\widetilde{\mathbf{Y}}\mathbf{L}_{\mathbf{S}}\frac{\partial \widetilde{\mathbf{Y}}}{\partial \widetilde{\mathbf{A}}[i,j]}\right)$

- **Question:** How to efficiently calculate $\mathbf{H}$?

Instantiation #1: PageRank

- **Goal:** Efficiently calculate $\mathbf{H}$ for PageRank
- **Mining results $\mathbf{Y}$: $\mathbf{r} = (1 - c)\mathbf{Qe}$**
- **Partial derivatives $\mathbf{H}$: $\mathbf{H} = 2c\mathbf{Q}'\mathbf{L}_\mathbf{S}\mathbf{r}\mathbf{r}'$**
- **Remarks: $\mathbf{Q} = (\mathbf{I} - c\mathbf{A})^{-1}$**
- **Time Complexity**
  - Straightforward: $O(n^3)$
  - Ours: $O(m_1 + m_2 + n)$
    - $m_\mathbf{A}$: number of edges in $\mathbf{A}$
    - $m_\mathbf{S}$: number of edges in $\boldsymbol{S}$
    - n: number of nodes



Instantiation #2: Spectral Clustering

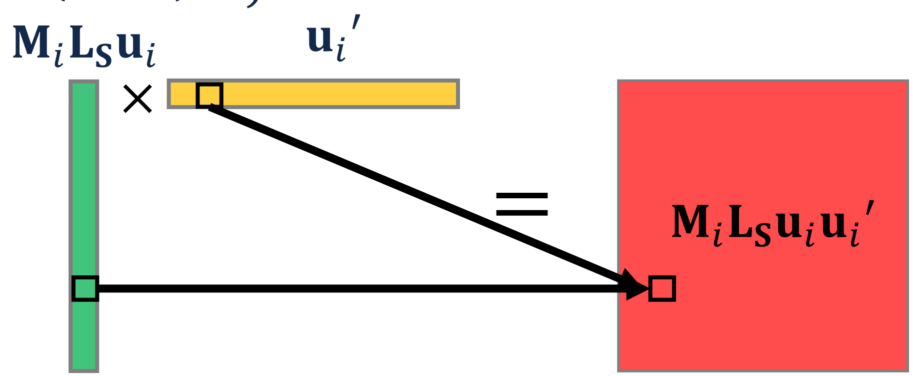
- **Goal:** Efficiently calculate $\mathbf{H}$ for spectral clustering
- **Mining results $\mathbf{Y}$: $\mathbf{U}$** $=$ eigenvectors with k smallest eigenvalues

Low-rank

- **Partial derivatives $\mathbf{H}$: $\mathbf{H}$** $= 2\sum_{i=1}^{k}\left(\boxed{\mathrm{diag}(\mathbf{M}_i\mathbf{L}_\mathbf{S}\mathbf{u}_i\mathbf{u}_i{}')\mathbf{1}_{n\times \mathrm{n}}} - \boxed{\mathbf{M}_i\mathbf{L}_\mathbf{S}\mathbf{u}_i\mathbf{u}_i{}'}\right)$

Vectorize $\mathrm{diag}(\mathbf{M}_i\mathbf{L}_\mathbf{S}\mathbf{u}_i\mathbf{u}_i{}')$ and stack it n times

- **Remarks**
  - $(\lambda_i, \mathbf{u}_i) = i$-th smallest eigenpair
  - $\mathbf{M}_i = (\lambda_i\mathbf{I} - \mathbf{L}_\mathbf{A})^+$
- **Time complexity**
  - Straightforward: $O(k^2(m+n) + k^3 n + kn^3)$
  - Ours: $O\big((k+r)(m_1+n) + k(m_2+n) + (k+r)^2 n\big)$
    - k: number of smallest eigenvalues
    - r: number of largest eigenvalues
    - m_1: number of edges in $\mathbf{A}$
    - m_2: number of edges in $\mathbf{S}$
    - n: number of nodes



Instantiation #3: LINE (1st)

- **Goal:** Efficiently calculate $\mathbf{H}$ for LINE (1st)
- **Mining results $\mathbf{Y}$**: $\mathbf{Y}[i,:]\mathbf{Y}[j,:]' = \log \frac{T(\widetilde{\mathbf{A}}[i,j]+\widetilde{\mathbf{A}}[j,i])}{d_i d_j^{3/4} + d_i^{3/4} d_j} - \log b$
  - $d_i =$ outdegree of node i, $T = \sum_{i=1}^{n} d_i^{3/4}$ and $b =$ number of negative samples
- **Partial derivatives $\mathbf{H}$: $\mathbf{H} =$** $\boxed{2f\left(\widetilde{\mathbf{A}} + \widetilde{\mathbf{A}}'\right) \circ \mathbf{L_S}} - \boxed{2\mathrm{diag}(\mathbf{BL_S})\mathbf{1}_{n\times n}}$

Element-wise in-place calculation

Vectorize $\mathrm{diag}(\mathbf{BL_S})$ and stack it n times

- **Remarks**
  - $f(\cdot)$ calculates Hadamard inverse, $\circ$ calculates Hadamard product
  - $\mathbf{B} = \frac{3}{4} f\left(\mathbf{d}^{5/4}\left(\mathbf{d}^{-1/4}\right)' + \boxed{\mathbf{d}\mathbf{1}_{1\times n}}\right) + f\left(\mathbf{d}^{3/4}\left(\mathbf{d}^{1/4}\right)' + \boxed{\mathbf{d}\mathbf{1}_{1\times n}}\right)$ with $\mathbf{d}^x[i] = d_i^x$

Stack $\mathbf{d}$ n times

- **Time complexity**
  - Straightforward: $O(n^3)$
  - Ours: $O(m_1 + m_2 + n)$
    - m_1: number of edges in $\mathbf{A}$
    - m_2: number of edges in $\mathbf{S}$
    - n: number of nodes

Debiasing the Mining Model

- **Goal:** Bias mitigation during model optimization
- **Intuition:** Optimizing a regularized objective such that
  - Task-specific loss function is minimized
  - Bias of mining results as regularization penalty is minimized
- **Optimization problem**

task-specific loss function

$$\min_{\mathbf{Y}} \quad J = l(\mathbf{A}, \mathbf{Y}, \theta) + \alpha \mathrm{Tr}(\mathbf{Y}'\mathbf{L}_{\mathbf{S}}\mathbf{Y})$$

bias measure

- **Solution**
  - **General:** Solve by (stochastic) gradient descent $\frac{\partial J}{\partial \mathbf{Y}} = \frac{\partial l(\mathbf{A},\mathbf{Y},\theta)}{\partial \mathbf{Y}} + 2\alpha \mathbf{L}_{\mathbf{S}}\mathbf{Y}$
  - **Task-specific:** Solve by specific algorithm designed for the graph mining problem
- **Advantage**
  - Linear time complexity incurred in computing the gradient

Debiasing the Mining Model: Instantiations

- **PageRank**
  - **Objective function:** $\min_{\mathbf{r}} c\mathbf{r}'(\mathbf{I}-\mathbf{A})\mathbf{r} + (1-c)\|\mathbf{r}-\mathbf{e}\|_F^2 + \alpha\mathbf{r}'\mathbf{L_S}\mathbf{r}$
  - **Solution:** $\mathbf{r}^* = c\left(\mathbf{A} - \frac{\alpha}{c}\mathbf{L_S}\right)\mathbf{r}^* + (1-c)\mathbf{e}$
    - PageRank on new transition matrix $\mathbf{A} - \frac{\alpha}{c}\mathbf{L_S}$
    - If $\mathbf{L_S} = \mathbf{I} - \mathbf{S}$, then $\mathbf{r}^* = \left(\frac{c}{1+\alpha}\mathbf{A} + \frac{\alpha}{1+\alpha}\mathbf{S}\right)\mathbf{r}^* + \frac{1-c}{1+\alpha}\mathbf{e}$
- **Spectral clustering**
  - **Objective function:** $\min_{\mathbf{U}} \mathrm{Tr}(\mathbf{U}'\mathbf{L_A}\mathbf{U}) + \alpha\mathrm{Tr}(\mathbf{U}'\mathbf{L_S}\mathbf{U}) = \mathrm{Tr}(\mathbf{U}'\mathbf{L}_{\mathbf{A}+\alpha\mathbf{S}}\mathbf{U})$
  - **Solution:** $\mathbf{U}^*$ = eigenvectors of $\mathbf{L}_{\mathbf{A}+\alpha\mathbf{S}}$ with k smallest eigenvalues
    - Spectral clustering on an augmented graph $\mathbf{A} + \alpha\mathbf{S}$
- **LINE (1st)**
  - **Objective function**

    $$\max_{\mathbf{x}_i,\mathbf{x}_j} \log g(\mathbf{x}_j\mathbf{x}_i') + b\mathbb{E}_{j'\in P_n}\left[\log g\left(-\mathbf{x}_{j'}\mathbf{x}_i'\right)\right] - \alpha\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_F^2\mathbf{S}[i,j] \quad \forall i,j = 1,\dots,n$$
  - **Solution:** Stochastic gradient descent

Debiasing the Mining Results

- **Goal:** Bias mitigation via a post-processing strategy
- **Intuition:** No access to either the input graph or the graph mining model
- **Optimization problem**

consistency of mining results, convex

$$\min_{\mathbf{Y}} \quad J = \|\mathbf{Y} - \bar{\mathbf{Y}}\|_F^2 + \alpha \mathrm{Tr}(\mathbf{Y}'\mathbf{L_S}\mathbf{Y})$$

bias measure, convex

  - $\bar{\mathbf{Y}}$ is the vanilla mining results
- **Solution:** $(\mathbf{I} + \alpha\mathbf{S})\mathbf{Y}^* = \bar{\mathbf{Y}}$
  - Convex loss function as long as $\alpha \geq 0 \rightarrow$ global optima by $\frac{\partial J}{\partial \mathbf{Y}} = 0$
  - Solve by conjugate gradient (or other linear system solvers)
- **Advantages**
  - No knowledge needed on the input graph
  - Model-agnostic

Problem Definition: InFoRM Cost

- **Question:** How to quantitatively characterize the cost of individual fairness?
- **Input**
  - Vanilla mining result $\bar{\mathbf{Y}}$
  - Debiased mining result $\mathbf{Y}^*$
    - Learned by the previous problem (InFoRM Algorithms)
- **Output:** An upper bound of $\|\bar{\mathbf{Y}} - \mathbf{Y}^*\|_F$
- **Debiasing methods**
  - Debiasing the input graph
  - Debiasing the mining model

  depend on specific graph topology/mining model
  - Debiasing the mining results

  main focus

Cost of Debiasing the Mining Results

- **Given**
  - A graph with n nodes and adjacency matrix $\mathbf{A}$
  - A node-node similarity matrix $\mathbf{S}$
  - Vanilla mining results $\bar{\mathbf{Y}}$
  - Debiased mining results $\mathbf{Y}^* = (\mathbf{I} + \alpha\mathbf{S})^{-1}\bar{\mathbf{Y}}$
- If $\|\mathbf{S} - \mathbf{A}\|_F = \Delta$, we have

$$\|\bar{\mathbf{Y}} - \mathbf{Y}^*\|_F \leq 2\alpha\sqrt{n}\left(\Delta + \sqrt{rank(\mathbf{A})}\sigma_{\max}(\mathbf{A})\right)\|\bar{\mathbf{Y}}\|_F$$

- **Observation:** The cost of debiasing the mining results depends on
  - The number of nodes n (i.e., size of the input graph)
  - The difference Δ between $\mathbf{A}$ and $\mathbf{S}$
  - The rank of $\mathbf{A}$ ⟶ could be small due to low-rank structures in real-world graphs
  - The largest singular value of $\mathbf{A}$ ⟶ could be small if $\mathbf{A}$ is normalized

InFoRM: Experiment

- **Graph mining task:** PageRank
- **Observation:** Effective in mitigating bias while preserving the performance of the vanilla algorithm with relatively small changes to the original mining results
  - Similar observations for spectral clustering and LINE (1st)

Debiasing the Input Graph												
Datasets	Jaccard Index						Cosine Similarity					
	Diff	KL	Prec@50	NDCG@50	Reduce	Time	Diff	KL	Prec@50	NDCG@50	Reduce	Time
Twitch	0.109	5.37×10^{-4}	1.000	1.000	24.7%	564.9	0.299	5.41×10^{-3}	0.860	0.899	62.9%	649.3
PPI	0.185	1.90×10^{-3}	0.920	0.944	43.4%	584.4	0.328	8.07×10^{-3}	0.780	0.838	68.7%	636.8

Debiasing the Mining Model												
Datasets	Jaccard Index						Cosine Similarity					
	Diff	KL	Prec@50	NDCG@50	Reduce	Time	Diff	KL	Prec@50	NDCG@50	Reduce	Time
Twitch	0.182	4.97×10^{-3}	0.940	0.958	62.0%	16.18	0.315	1.05×10^{-2}	0.940	0.957	73.9%	12.73
PPI	0.211	4.78×10^{-3}	0.920	0.942	50.8%	10.76	0.280	9.56×10^{-3}	0.900	0.928	67.5%	10.50

Debiasing the Mining Results												
Datasets	Jaccard Index						Cosine Similarity					
	Diff	KL	Prec@50	NDCG@50	Reduce	Time	Diff	KL	Prec@50	NDCG@50	Reduce	Time
Twitch	0.035	9.75×10^{-4}	0.980	0.986	33.9%	0.033	0.101	5.84×10^{-3}	0.940	0.958	44.6%	0.024
PPI	0.045	1.22×10^{-3}	0.940	0.958	27.0%	0.020	0.112	6.97×10^{-3}	0.940	0.958	45.0%	0.019

Overview of Part II

Individual Fairness on Graphs

Optimization-based Approach

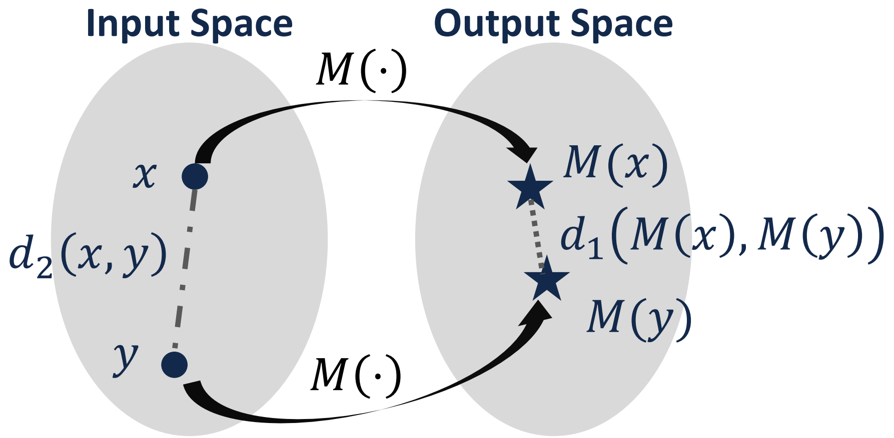
- Laplacian regularization-based individual fairness

Ranking-based Approach

- Ranking-based individual fairness

Individual Fairness on GNNs

- **Goal:** Debias a graph neural network (GNN) to ensure its output is individually fair
- **Key challenge:** Distance calibration



[1] Dong, Y., Kang, J., Tong, H., & Li, J.. Individual Fairness for Graph Neural Networks: A Ranking based Approach. KDD 2021.

Key Challenge: Distance calibration

- **Existing formulation:** Lipschitz condition (used in InFoRM)

$$d_1\big(M(x),M(y)\big) \le L d_2(x,y)$$

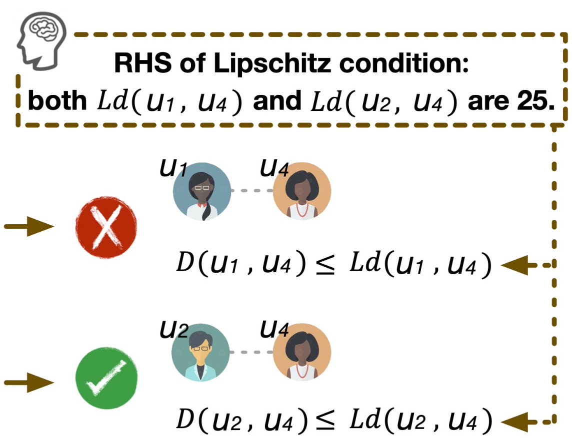
distance metric in output space

distance metric in input space

- **Limitation:** Direct distance comparison fails to calibrate the differences between different individuals
- **Example**

	u_1	u_2	u_3	u_4	u_5	u_6
u_1	0	90	85	30	95	90
u_2	90	0	1	20	3	2
u_3	85	1	0	70	20	2
u_4	30	20	70	0	50	50
u_5	95	3	20	50	0	5
u_6	90	2	2	50	5	0

(a) Outcome distance matrix from distance metric D



(b) Lipschitz condition judgement based on human knowledge

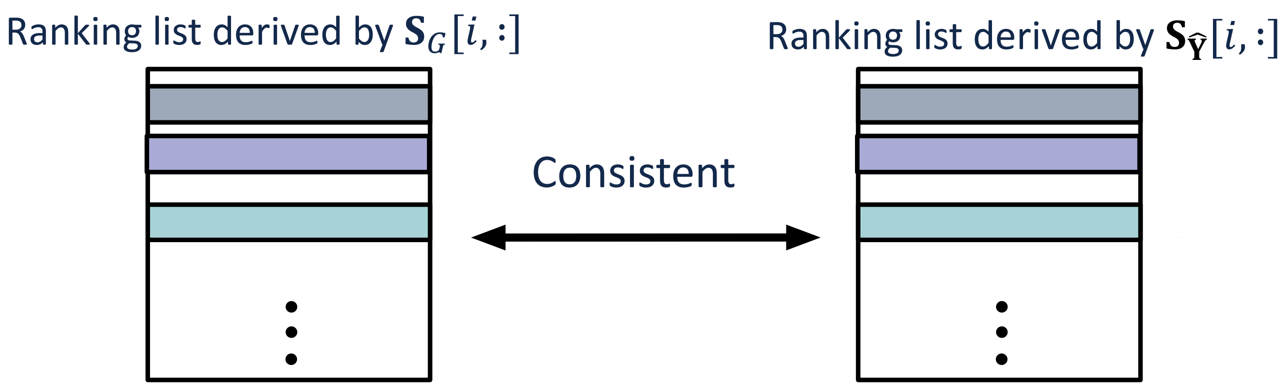
- **Question:** Can we achieve fairness with natural calibration across individuals?

REDRESS: Ranking basEd InDividual FaiRnESS

- **Ranking-based individual fairness**
  - **Given:** (1) The pairwise node similarity matrix $\mathbf{S}_G$ of the input graph G; (2) the pairwise similarity matrix $\mathbf{S}_{\hat{\mathbf{Y}}}$ of the GNN output $\hat{\mathbf{Y}}$
  - $\hat{\mathbf{Y}}$ is individually fair if, for each node i, it satisfies that

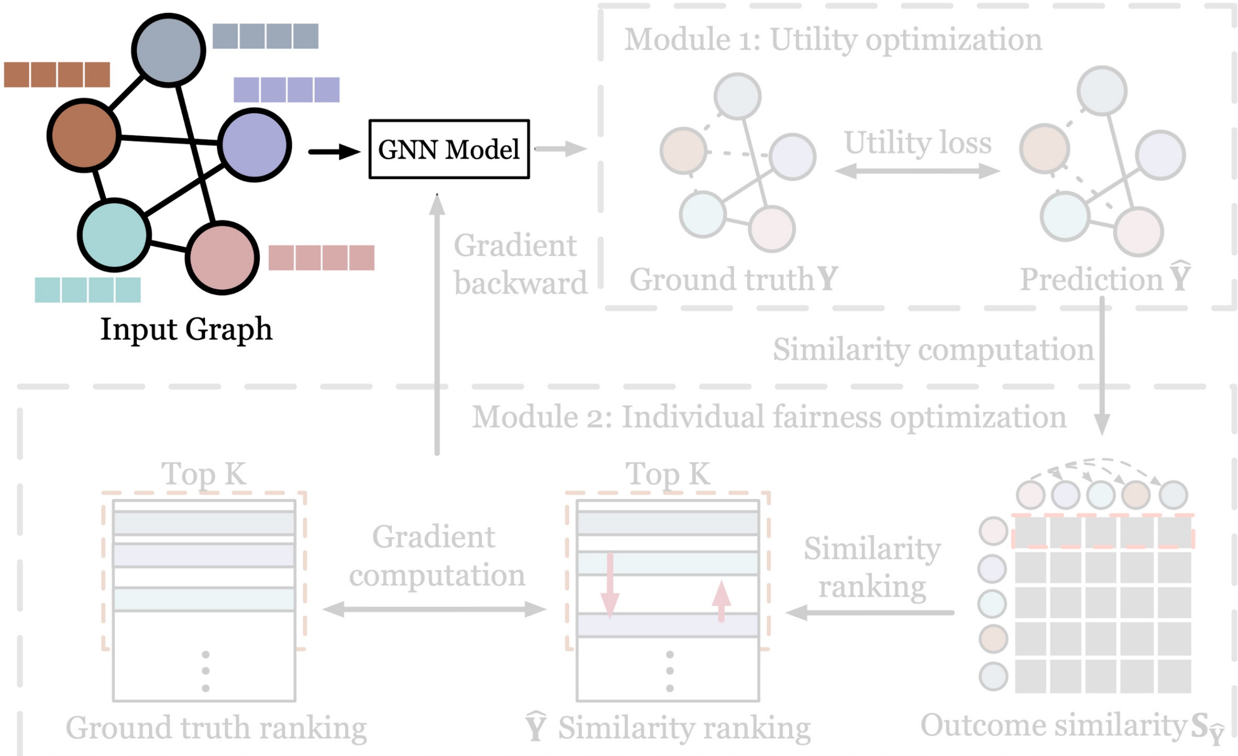
    ranking list derived by $\mathbf{S}_G[i,:]$ = ranking list derived by $\mathbf{S}_{\hat{\mathbf{Y}}}[i,:]$

- **Example**



- **Advantage:** Naturally calibrate across individuals
  - No direct distance comparison

REDRESS: Framework



- **GNN backbone model**
  - Learn node representations
- **Utility maximization**
  - Minimize the downstream task-specific loss
- **Individual fairness optimization**
  - Enforce ranking-based individual fairness

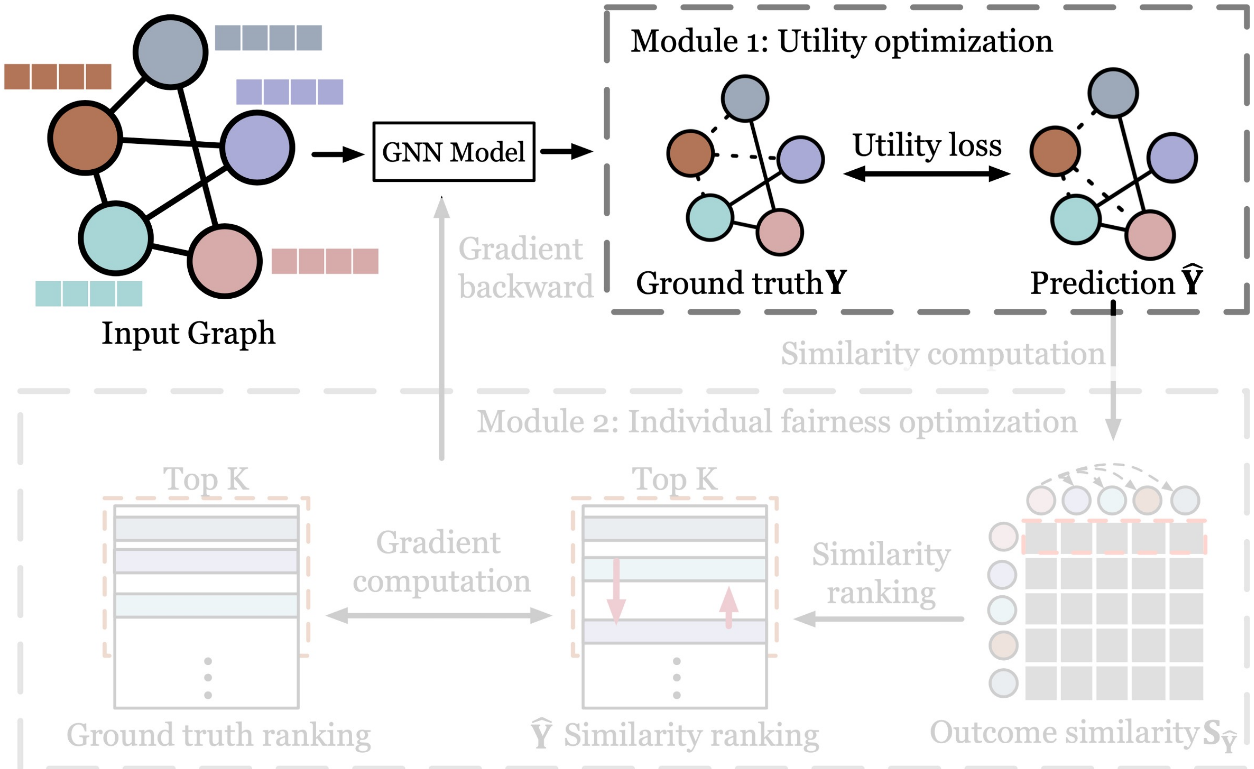
REDRESS: GNN Backbone Model

- **Goal:** Learn node representations by a GNN
- **Formulation:** l-th GNN Layer

$$h_v^{(l+1)} = \sigma\left(\mathrm{AGG}\left(\left\{h_u^{(l)}: u \in \mathcal{N}(v) \cup \{v\}\right\}\right)\right)$$

  - $h_v^{(l)}$: Embedding of node v at l-th layer
  - $\mathrm{AGG}(\cdot)$: Information aggregation function (e.g., mean, weighted sum)
  - $\sigma(\cdot)$: Activation function (e.g., ReLU)
  - $\mathcal{N}(v)$: Neighborhood set of node v
- **Advantage:** No restriction on the GNN architecture
  - REDRESS works on any GNN model

REDRESS: Framework



- GNN backbone model
  - Learn node representations
- **Utility maximization**
  - Minimize the downstream task-specific loss
- Individual fairness optimization
  - Enforce ranking-based individual fairness

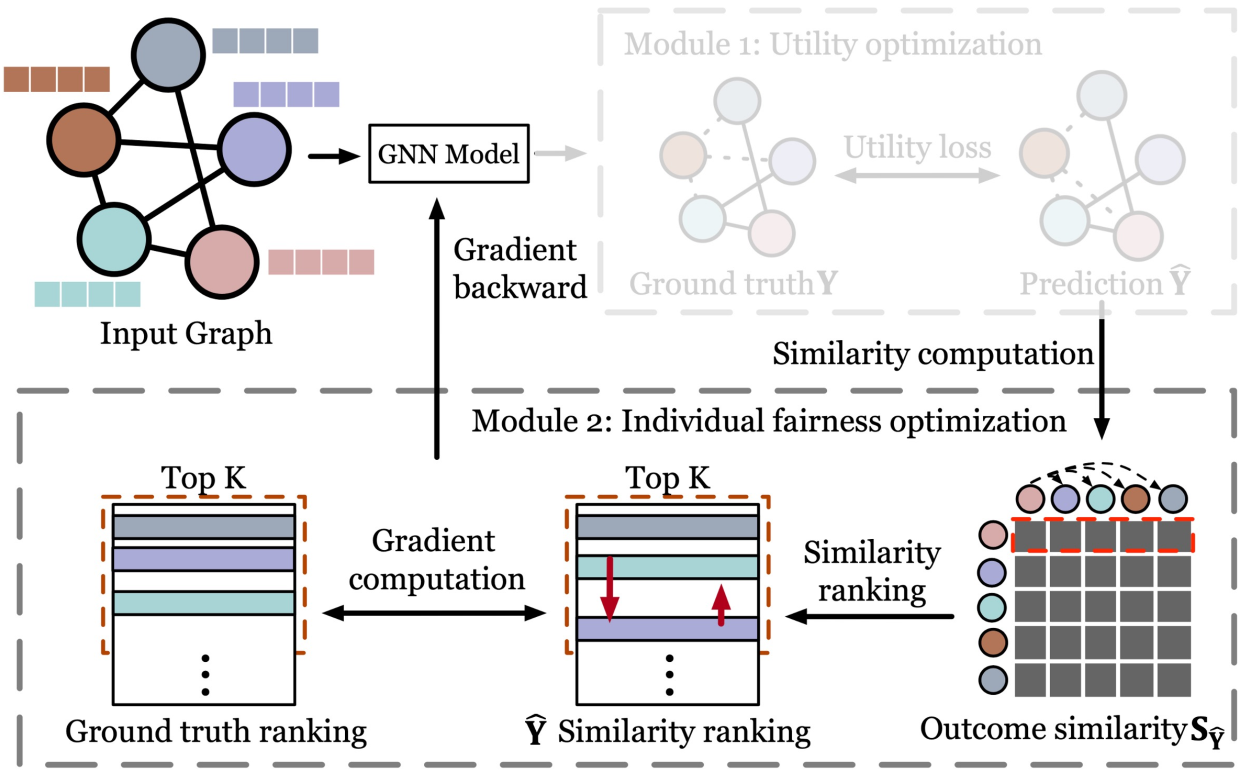
REDRESS: Utility Maximization

- **Goal:** Minimize the downstream task-specific loss function
- **Choice of loss function:** Cross-entropy loss

$$L_{\text{utility}} = -\sum_{(i,j)\in\mathcal{T}} \mathbf{Y}[i,j] \log \widehat{\mathbf{Y}}[i,j]$$

  - $\mathbf{Y}[i,j]$: i-th row and j-th column in ground truth $\mathbf{Y}$
  - $\hat{\mathbf{Y}}[i,j]$: i-th row and j-th column in GNN predictions $\hat{\mathbf{Y}}$
  - $\mathcal{T}$: A set of tuples
    - **Node classification:** $\mathcal{T}$ is a set of (node, class) tuples
    - **Link prediction:** $\mathcal{T}$ is a set of (node, node) tuples

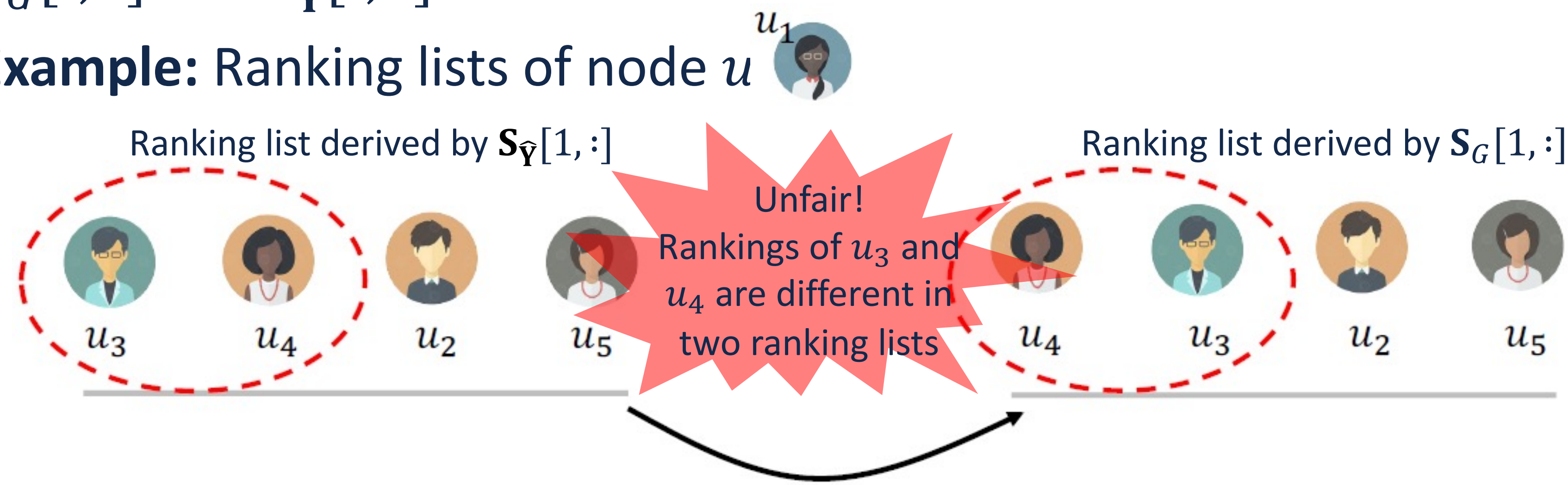
REDRESS: Framework



- GNN backbone model
  - Learn node representations
- Utility maximization
  - Minimize the downstream task-specific loss
- **Individual fairness optimization**
  - Enforce ranking-based individual fairness

REDRESS: Individual Fairness Optimization

- **Given:** (1) Pairwise node similarity matrix $\boldsymbol{S}_G$ of input graph G and (2) pairwise similarity matrix $\mathbf{S}_{\hat{\mathbf{Y}}}$ of GNN output $\hat{\mathbf{Y}}$
- **Goal:** For each node i, ensure that the ranking lists derived from $\boldsymbol{S}_G[i,:]$ and $\mathbf{S}_{\hat{\mathbf{Y}}}[i,:]$ are similar
- **Example:** Ranking lists of node u



- **Problem:** Ranking is a non-differentiable operation

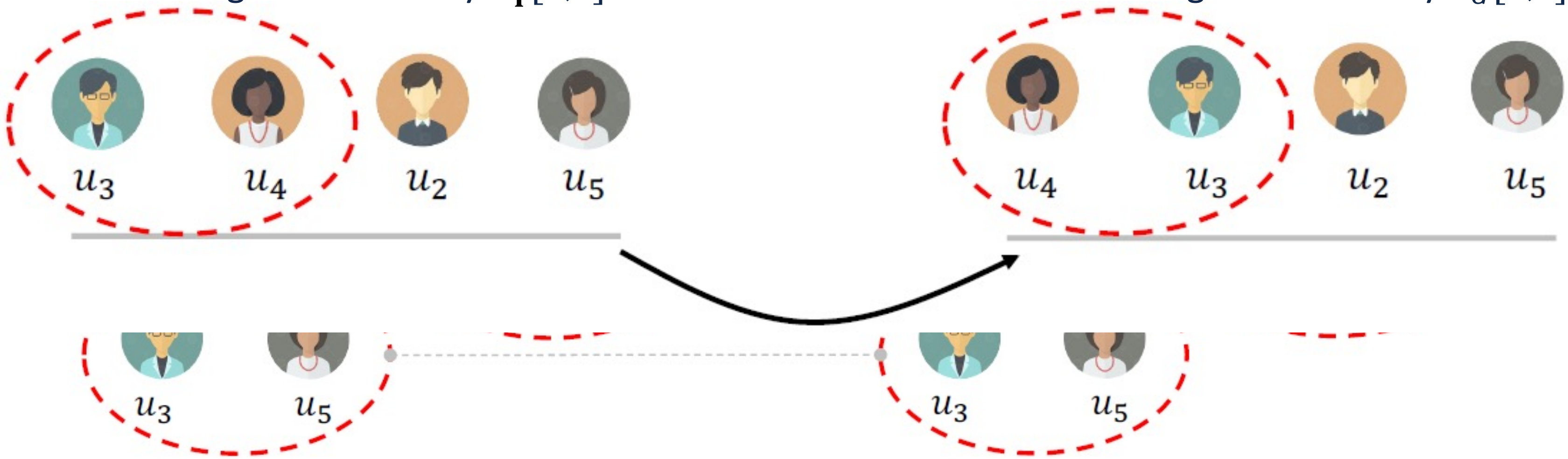
$\rightarrow$ loss on the ranking lists will be non-differentiable

REDRESS: Individual Fairness Optimization

- **Solution**
  - Consider the relative ranking orders of every node pair in $\boldsymbol{S}_G$ and $\mathbf{S}_{\hat{\mathbf{Y}}}$
  - Ensure that every node pair's relative orders are consistent across $\boldsymbol{S}_G$ and $\mathbf{S}_{\hat{\mathbf{Y}}}$
- **Example:** Ranking lists of node u_1 

Ranking list derived by $\mathbf{S}_{\hat{\mathbf{Y}}}[1,:]$

Ranking list derived by $\boldsymbol{S}_G[1,:]$



REDRESS: Individual Fairness Optimization

- How to calculate relative ranking order
  - **Key idea:** Relative ranking order of u and v = Probability that u ranks higher than v
    - Inspired by learning-to-rank
  - **Input space:** Pairwise node similarity matrix $\mathbf{S}_G$ of graph G

$$P_{uv}(i) = \frac{1}{2}\left(1 + T_{uv}(i)\right) \qquad T_{uv}(i) = \begin{cases} 1 & u \text{ ranks higher than } v \\ 0 & u \text{ and } v \text{ has the same rank} \\ -1 & v \text{ ranks higher than } u \end{cases}$$

  - **Output space:** Pairwise similarity matrix $\mathbf{S}_{\hat{\mathbf{Y}}}$ of GNN output $\hat{\mathbf{Y}}$

$$\hat{P}_{uv}(i) = \frac{1}{1 + e^{-\alpha\left(\mathbf{S}_{\hat{\mathbf{Y}}}[i,u] - \mathbf{S}_{\hat{\mathbf{Y}}}[i,v]\right)}}$$

    where α is a constant scalar
  - **Fairness loss for a node pair**

$$L_{uv}(i) = -P_{uv}(i) \log \hat{P}_{uv}(i) - \left(1 - P_{uv}(i)\right) \log\left(1 - \hat{P}_{uv}(i)\right)$$

REDRESS: Individual Fairness Optimization

- **Solution:** Focus on top-k similar nodes for each node i in $\mathbf{S}_{\hat{\mathbf{Y}}}$
  - **Individual fairness:** Similar outcomes for similar individuals
  - Define $z_{@k}$ = similarity metric for two top-k ranking lists (e.g., NDCG@k)

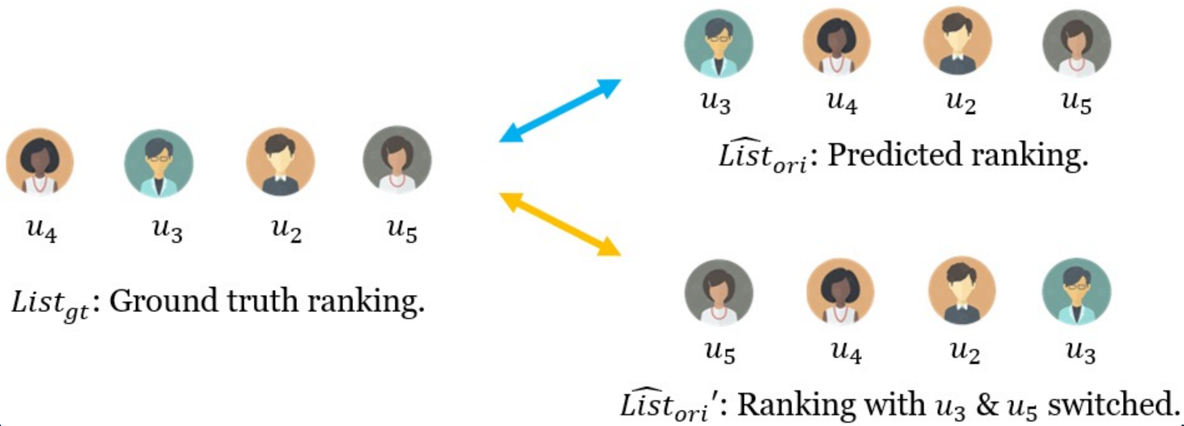
$$L_{\text{fairness}} = \boxed{\sum_i \sum_u \sum_v} L_{uv}(i) |\Delta z_{@k}|_{u,v}$$

$O(nk^2)$ time complexity

where $|\Delta z_{@k}|_{u,v}$ = absolute value changes in $z_{@k}$ if nodes u and v are swapped

- **Intuition of $|\Delta z_{@k}|_{u,v}$**
  - High $|\Delta z_{@k}|_{u,v} \rightarrow u$ and v are dissimilar $\rightarrow$ more penalty if ranked wrong
- **Example**

$$|\Delta z_{@k}|_{3,5} = |z_{@k}(List_{gt}, \widehat{List}_{ori}) - z_{@k}(List_{gt}, \widehat{List}_{ori}{}')|$$



REDRESS: Total Loss Functions

- **Utility loss**

$$L_{\text{utility}} = -\sum_{(i,j)\in\mathcal{T}} \mathbf{Y}[i,j] \log \hat{\mathbf{Y}}[i,j]$$

- **Fairness loss**

$$L_{uv}(i) = -P_{uv}(i) \log \hat{P}_{uv}(i) - \left(1 - P_{uv}(i)\right) \log\left(1 - \hat{P}_{uv}(i)\right)$$

$$L_{\text{fairness}} = \sum_{i}\sum_{u}\sum_{v} L_{uv}(i)\,|\Delta z_{@k}|_{u,v}$$

- **Total loss**

$$L = L_{\text{utility}} + \gamma L_{\text{fairness}}$$

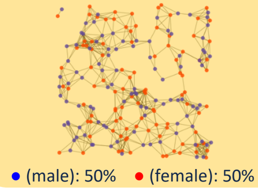
where γ is the regularization hyperparameter

REDRESS: Experiment

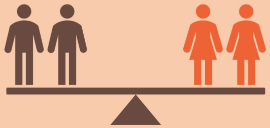
- **Observations for node classification**
  - **Comparable** performance on model utility compared with the best ones
  - **Best** performance on the ranking-based individual fairness
- Similar observations for link prediction

CS	GCN	Vanilla	90.59 ± 0.3 (−)	50.84 ± 1.2 (−)	**90.59 ± 0.3 (−)**	18.29 ± 0.8 (−)
		InFoRM	88.66 ± 1.1 (−2.13%)	53.38 ± 1.6 (+5.00%)	87.55 ± 0.9 (−3.36%)	19.18 ± 0.9 (+4.87%)
		PFR	87.51 ± 0.7 (−3.40%)	37.12 ± 0.9 (−27.0%)	86.16 ± 0.2 (−4.89%)	11.98 ± 1.3 (−34.5%)
		REDRESS (Ours)	**90.70 ± 0.2** (+0.12%)	**55.01 ± 1.9** (+8.20%)	89.16 ± 0.3 (−1.58%)	**21.28 ± 0.3** (+16.4%)
	SGC	Vanilla	87.48 ± 0.8 (−)	74.00 ± 0.1 (−)	87.48 ± 0.8 (−)	32.36 ± 0.3 (−)
		InFoRM	88.07 ± 0.1 (+0.67%)	74.29 ± 0.1 (+0.39%)	88.65 ± 0.4 (+1.34%)	32.37 ± 0.4 (+0.03%)
		PFR	88.31 ± 0.1 (+0.94%)	48.40 ± 0.1 (−34.6%)	84.34 ± 0.3 (−3.59%)	28.87 ± 0.9 (−10.8%)
		REDRESS (Ours)	**90.01 ± 0.2** (+2.89%)	**76.60 ± 0.1** (+3.51%)	**89.35 ± 0.1** (+2.14%)	**34.24 ± 0.2** (+5.81%)

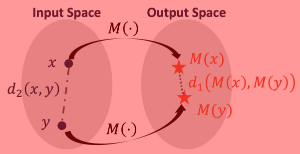
Roadmap



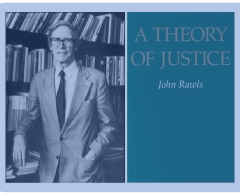
Introduction



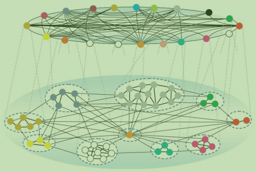
Part I: Group Fairness on Graphs



Part II: Individual Fairness on Graphs



Part III: Other Fairness Definitions on Graphs



Part IV: Future Research Directions

Overview of Part III

Other Fairness Definitions on Graphs

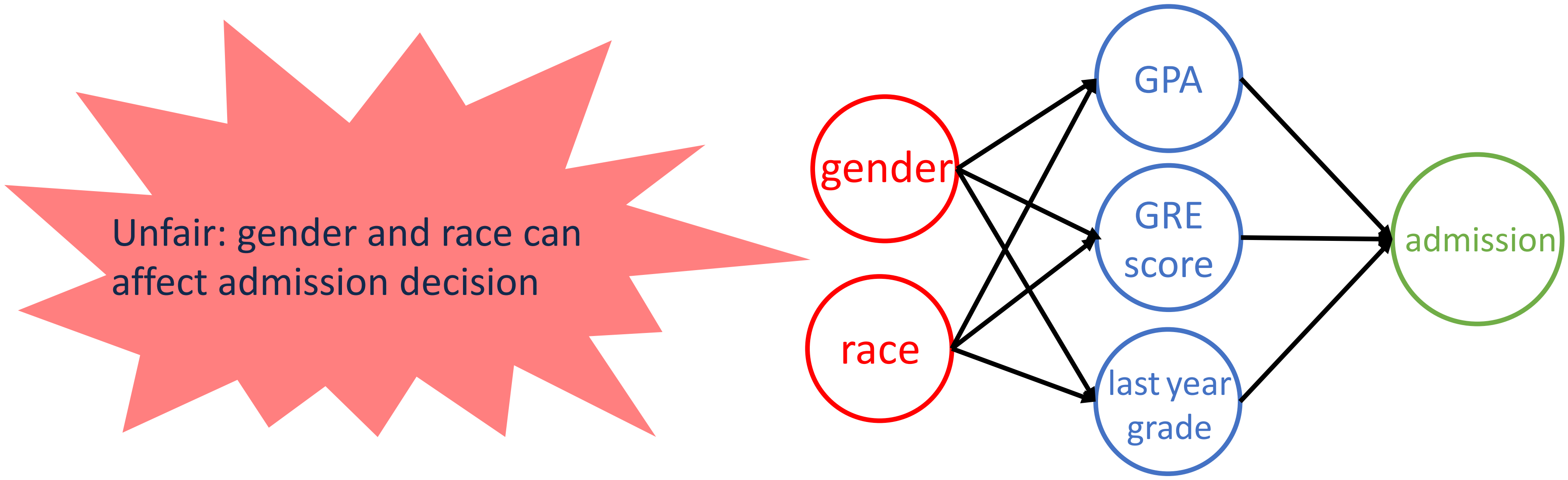
- Counterfactual Fairness
  - Fairness on graph embedding
- Degree-related Fairness
  - Fairness on graph neural networks
- Rawlsian Difference Principle
  - Fairness on graph covering

Recap: Counterfactual Fairness

- **Definition:** Same outcomes for 'different versions' of the same candidate

$$\Pr\left(\hat{y}_{s=s_1} = c | s = s_1, x = \mathbf{x}\right) = \Pr\left(\hat{y}_{s=s_2} = c | s = s_2, x = \mathbf{x}\right)$$

  - $\Pr\left(\hat{y}_{s=s_1} = c | s = s_1, x = \mathbf{x}\right)$: version 1 of $\mathbf{x}$ with sensitive demographic s_1
  - $\Pr\left(\hat{y}_{s=s_2} = c | s = s_2, x = \mathbf{x}\right)$: version 2 of $\mathbf{x}$ with sensitive demographic s_2
- **Example:** Causal graph of graduate college admission



Preliminary: Stability

- **Definition:** Perturbations on the input data should not affect the output too much
- **Mathematical formulation:** Lipschitz condition

$$d_1\big(M(x), M(\tilde{x})\big) \leq L d_2(x, \tilde{x})$$

  - M: A mapping from input to output
  - d_1: Distance metric for output
  - d_2: Distance metric for input
  - L: Lipschitz constant
  - $\tilde{x}$: Perturbed version of original input data x

Counterfactual Fairness vs. Stability

- **Given**
  - A graph with binary adjacency matrix $\mathbf{A}$
  - A node u with feature vector $\mathbf{x}_u$
    - Information vector of node u: $\mathbf{b}_u = [\mathbf{x}_u; \mathbf{A}[u,:]]$
  - Perturbed version $\tilde{u}$ of node u with information vector $\tilde{\mathbf{b}}_u$
    - Perturbation(s) on $\mathbf{x}_u$ or $\mathbf{A}[u,:]$
  - Counterfactual version $\tilde{u}^s$ of node u
    - Modification on the value of sensitive attribute s in $\mathbf{x}_u$
  - An encoder function $\mathrm{ENC}(\cdot)$ that learns the embedding $\mathrm{ENC}(u)$ of node u
- **Counterfactual fairness**

$$\mathrm{ENC}(u) = \mathrm{ENC}(\tilde{u}^s)$$

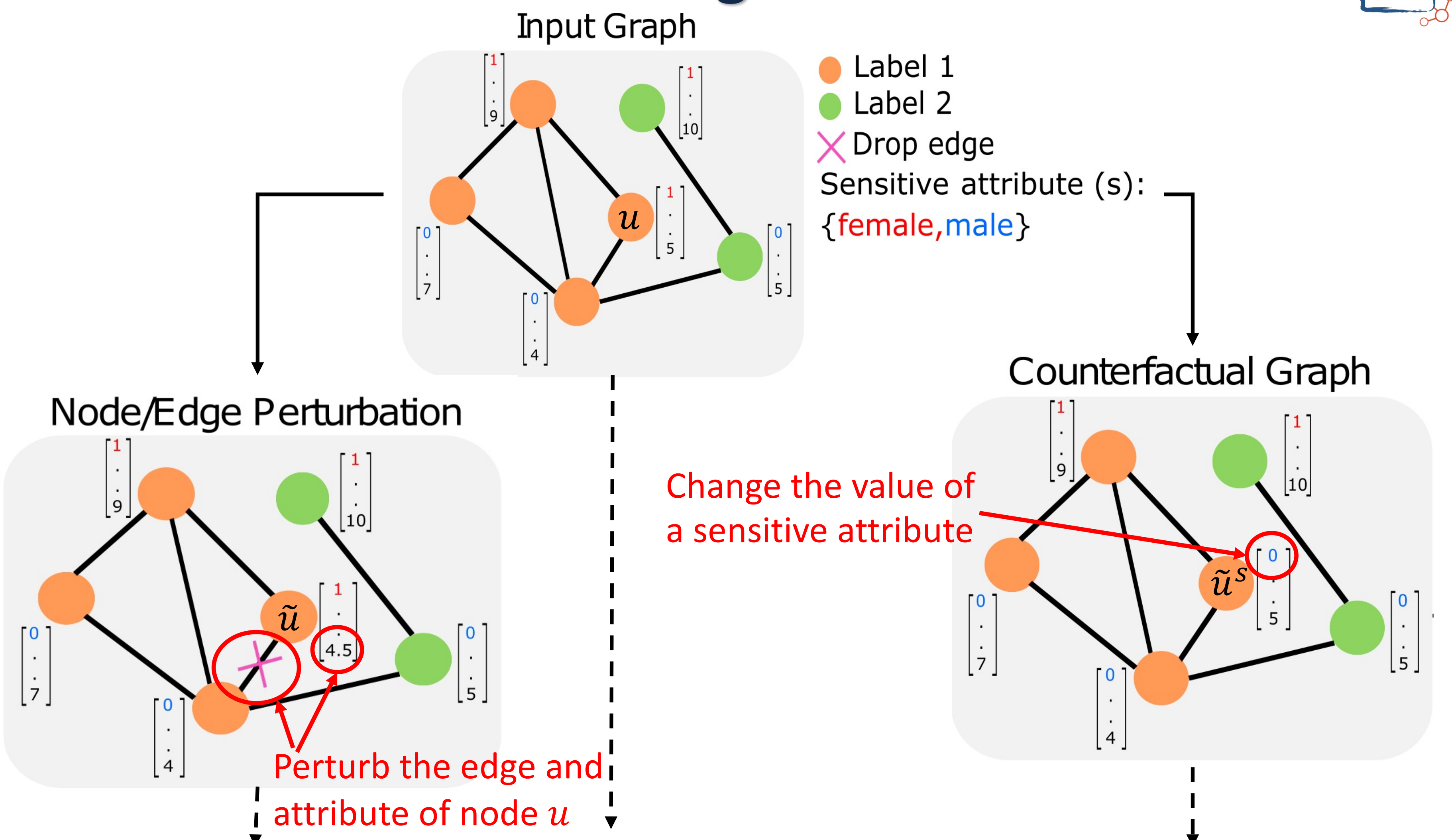
- **Stability**

$$\|\mathrm{ENC}(u) - \mathrm{ENC}(\tilde{u})\|_p \leq L\left\|\tilde{\mathbf{b}}_u - \mathbf{b}_u\right\|_p$$

- **Question:** Can we learn node embedding that is both counterfactually fair and stable?

[1] Agarwal, C., Lakkaraju, H., & Zitnik, M.. Towards a Unified Framework for Fair and Stable Graph Representation Learning. UAI 2021.

NIFTY: Contrastive Learning-based Framework



maximize similarity among embeddings of u, $\tilde{u}$, $\tilde{u}^s$

NIFTY Layer and NIFTY Encoder

- **Given**
  - $\mathbf{h}_u^{(k)}$: Representation of node u at k-th layer
  - $\mathcal{N}(u)$: Neighborhood of node u
  - $\mathbf{W}_a^{(k)}$: Self-attention weight matrix at k-th layer
  - $\widetilde{\mathbf{W}}_a^{(k)} = \frac{\mathbf{W}_a^{(k)}}{\left\|\mathbf{W}_a^{(k)}\right\|_p}$: Lipschitz-normalization on $\mathbf{W}_a^{(k)}$
    - $\left\|\mathbf{W}_a^{(k)}\right\|_p$: Spectral norm of $\mathbf{W}_a^{(k)}$
  - $\mathbf{W}_n^{(k)}$: Weight matrix associated with the neighbors of node u
- The k-th NIFTY layer learns node representation by

$$\mathbf{h}_u^{(k)} = \sigma\left(\widetilde{\mathbf{W}}_a^{(k-1)}\mathbf{h}_u^{(k-1)} + \mathbf{W}_n^{(k-1)} \sum_{v\in\mathcal{N}(u)} \mathbf{h}_v^{(k-1)}\right)$$

- NIFTY Encoder $\mathrm{ENC}(\cdot)$= a stack of K NIFTY layers

NIFTY: Similarity Loss

- **Goal:** Maximize similarity among embeddings of u, $\tilde{u}$, $\tilde{u}^s$
- **Augmented graph:** Either (1) edge/attribute perturbed graph or (2) counterfactual graph with modification on the value of sensitive attribute
- **Formulation**

$$L_s(u, \tilde{u}^{\text{aug}}) = \frac{D\left(\text{FC}(\mathbf{z}_u), \text{SG}\left(\mathbf{z}_u^{\text{aug}}\right)\right) + D\left(\text{FC}\left(\mathbf{z}_u^{\text{aug}}\right), \text{SG}(\mathbf{z}_u)\right)}{2}$$

  - $D(\cdot,\cdot)$: Cosine distance
  - $\tilde{u}^{\text{aug}}$: Counterpart of node u in the augmented graph
  - $\mathbf{z}_u, \mathbf{z}_u^{\text{aug}}$: Representation of nodes u and $\tilde{u}^{\text{aug}}$ learned by NIFTY encoder
  - $\text{FC}(\cdot)$: A fully-connected layer to transform and align embeddings
  - $\text{SG}(\cdot)$: Stop-grad operator, stop calculating the gradient with respect to its input

- **Intuition:** Minimize L_s $\begin{cases} \text{FC}(\mathbf{z}_u) \text{ and } \mathbf{z}_u^{\text{aug}} \text{ are similar} \\ \text{FC}\left(\mathbf{z}_u^{\text{aug}}\right) \text{ and } \mathbf{z}_u \text{ are similar} \end{cases}$

NIFTY: Total Loss

- **Total loss**

  $$L = (1-\lambda)L_c + \lambda(\mathbb{E}_u[L_s(u,\tilde{u})] + \mathbb{E}_u[L_s(u,\tilde{u}^s)])$$

  - λ: Regularization hyperparameter
  - L_c: Task-specific loss
    - E.g., cross-entropy loss for node classification
  - $\mathbb{E}_u[L_s(u,\tilde{u})]$: Similarity loss of original graph and the edge/attribute perturbed graph
  - $\mathbb{E}_u[L_s(u,\tilde{u}^s)]$: Similarity loss of original graph and the counterfactual graph
- **Intuition:** Jointly minimize
  - The task-specific loss
  - Distance among embeddings of u, $\tilde{u}$ and $\tilde{u}^s$, for each node u

NIFTY Stability

- **Given**
  - A K-layer NIFTY encoder $\mathrm{ENC}(\cdot)$
    - Self-attention weight matrix at k-th layer $\widetilde{\mathbf{W}}_a^{(k)}$
  - A binary-valued sensitive attribute s
  - A node u with information vector $\mathbf{b}_u$
  - Perturbed version $\tilde{u}$ of node u with information vector $\tilde{\mathbf{b}}_u$
- NIFTY learns stable node embedding

$$\|\mathrm{ENC}(u) - \mathrm{ENC}(\tilde{u})\|_p \leq \prod_{k=1}^{K} \left\|\widetilde{\mathbf{W}}_a^{(k)}\right\|_p \|\mathbf{b}_u - \tilde{\mathbf{b}}_u\|_p$$

- **Remarks**
  - Lipschitz constant = $\prod_{k=1}^{K} \left\|\widetilde{\mathbf{W}}_a^{(k)}\right\|_p$
  - Normalized $\widetilde{\mathbf{W}}_a^{(k)}$ → small Lipschitz constant → stable $\mathrm{ENC}(u)$

NIFTY Counterfactual Fairness

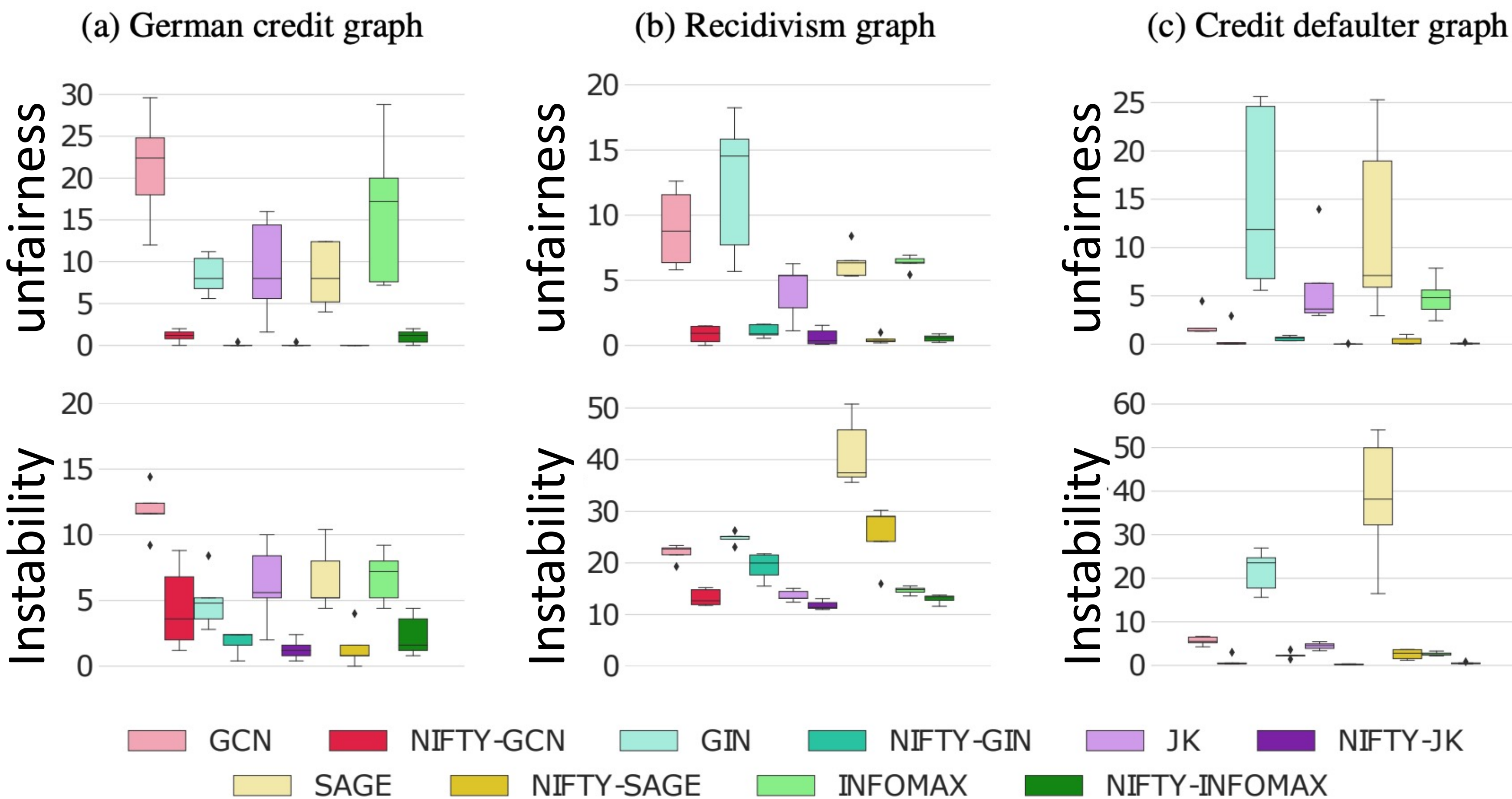
- **Given**
  - A K-layer NIFTY encoder $\mathrm{ENC}(\cdot)$
    - Self-attention weight matrix at k-th layer $\widetilde{\mathbf{W}}_a^{(k)}$
  - A binary-valued sensitive attribute s
  - A node u with its counterfactual version $\tilde{u}^s$ by flipping the value of s
- NIFTY is counterfactually fair with the unfairness upper bounded as follows

$$\|\mathrm{ENC}(u) - \mathrm{ENC}(\tilde{u}^s)\|_p \leq \prod_{k=1}^{K} \left\|\widetilde{\mathbf{W}}_a^{(k)}\right\|_p$$

- **Remarks**
  - Upper bounded counterfactual unfairness (i.e., $\|\mathrm{ENC}(u) - \mathrm{ENC}(\tilde{u}^s)\|_p$)
  - Normalized $\widetilde{\mathbf{W}}_a^{(k)}$ → counterfactually fair $\mathrm{ENC}(u)$

NIFTY: Experiment

- **Observation:** NIFTY improves both fairness and stability



Overview of Part III

Other Fairness Definitions on Graphs

Counterfactual Fairness	Degree-related Fairness	Rawlsian Difference Principle
• Fairness on graph embedding	• Fairness on graph neural networks	• Fairness on graph covering

Preliminary: Graph Convolutional Network (GCN)

- **Key Idea:** Learn representations by aggregating information from the neighbors
- **GCN:** A stack of graph convolution layers
  - The l-th graph convolution layer with σ activation

Weight parameters

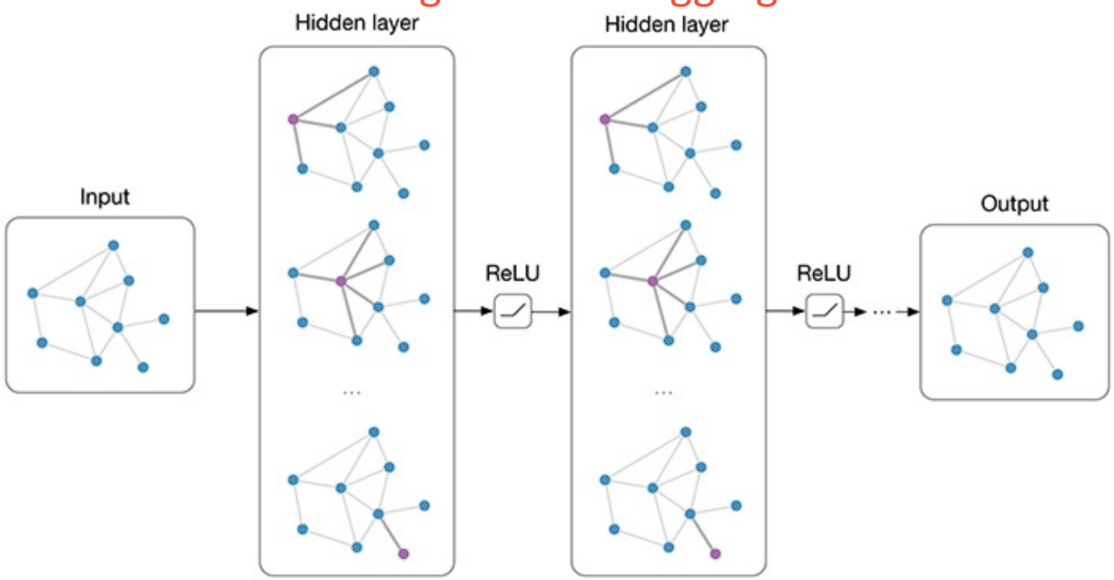
$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{j\in\mathcal{N}_i\cup\{i\}} a_{ij}\mathbf{W}^{(l)}\mathbf{h}_j^{(l)}\right)$$

Node representations of nodes i and j

Edge weight between nodes i and j

Neighborhood aggregation

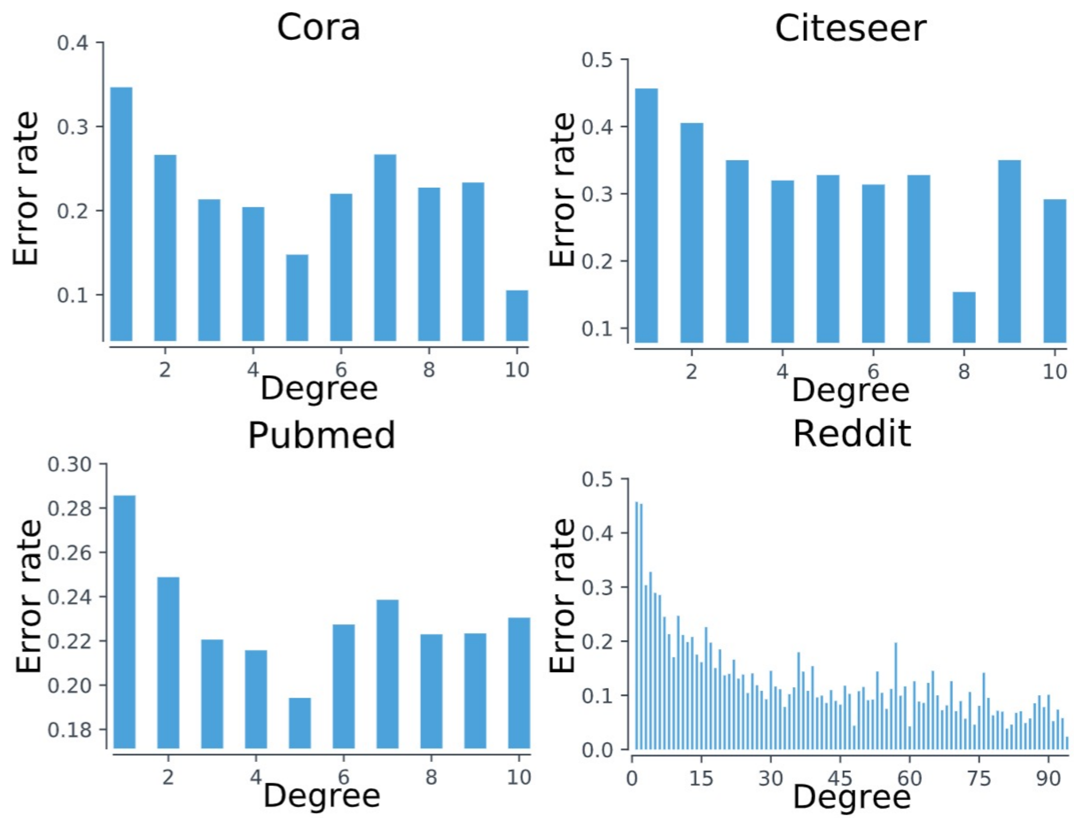
- **Example**



[1] Kipf, T. N., & Welling, M. (2016). Semi-Supervised Classification with Graph Convolutional Networks. ICLR 2017.

GCN Analysis: Error Rate Distribution

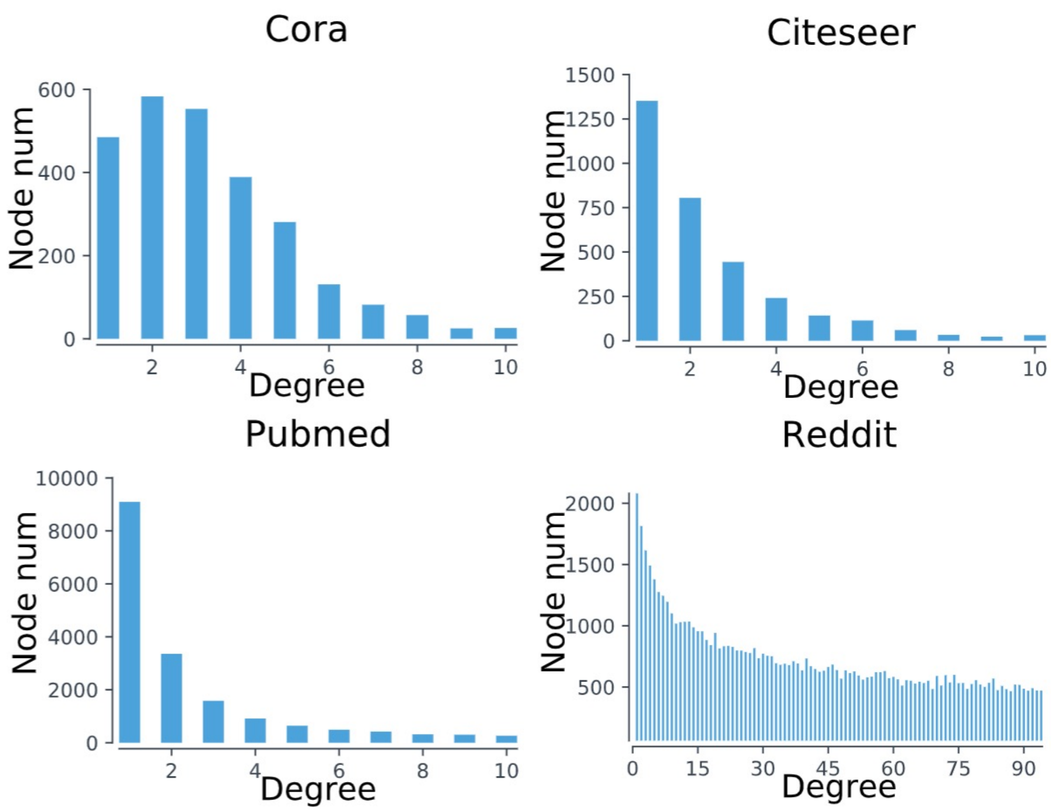
- **Observation:** Low-degree nodes get higher error rate
- **Question:** Why should we concern about low-degree nodes?



[1] Tang, X., Yao, H., Sun, Y., Wang, Y., Tang, J., Aggarwal, C., ... & Wang, S.. Investigating and Mitigating Degree-Related Biases in Graph Convolutional Networks. CIKM 2020.

Degree Distribution of Real-World Graphs

- **Observation:** Degree distribution is long-tailed
  – Low-degree nodes are the majority in the graph



Why GCN Fails

- **Key steps in GCN training**
  - Learn node representations by message passing
  - Train the model parameters by backpropagation
- **Question #1:** Does GCN fail because of the message passing schema?
  - **Hypothesis #1:** High-degree nodes have higher influence to affect the training of GCN on other nodes
- **Question #2:** Does GCN fail during the backpropagation?
  - Only information of labeled nodes can be backpropagated to its neighbors
  - **Hypothesis #2:** High-degree nodes are more likely to connect with labeled nodes

Cause #1: Influence of High-Degree Nodes

- **Given**
  - A set of labeled nodes $\mathcal{V}_{\text{labeled}}$
  - An L-layer GCN with $\mathbf{W}^{(L)}$ as the weight of L-th layer
  - Two nodes i and k whose degrees are d_i and d_k
    - $\mathbf{x}_i$ and $\mathbf{x}_k$ as their corresponding input node features
    - $\mathbf{h}_i^{(L)}$ and $\mathbf{h}_k^{(L)}$ as the output embeddings learned by GCN
- **Influence of node $\boldsymbol{i}$ to node $\boldsymbol{k}$**

$$\mathbb{E}\left[\partial \mathbf{h}_i^{(L)} / \partial \mathbf{x}_k\right] \propto \sqrt{d_i d_k}\mathbf{W}^{(L)}$$

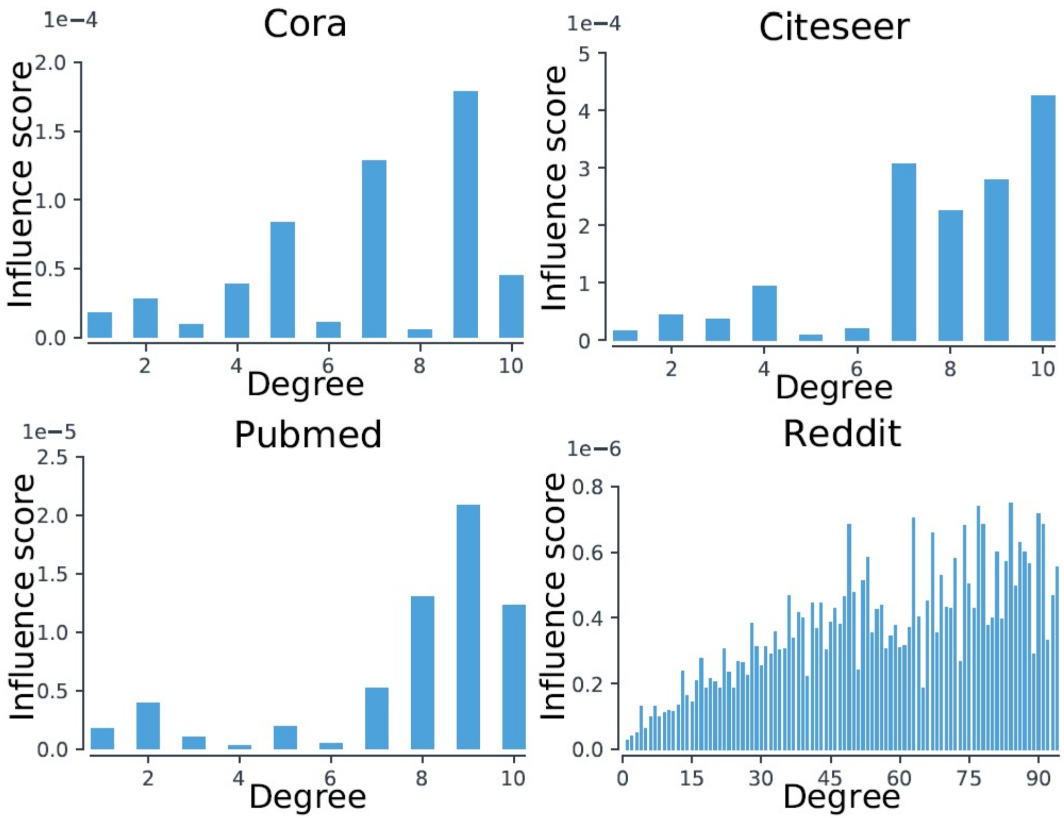
- **Influence of node $\boldsymbol{i}$ on GCN training**

$$S(i) = \sum_{k \in \mathcal{V}_{\text{labeled}}} \left\| \mathbb{E}\left[\partial \mathbf{h}_i^{(L)} / \partial \mathbf{x}_k\right] \right\| \propto \sqrt{d_i} \left\| \mathbf{W}^{(L)} \right\| \sum_{k \in \mathcal{V}_{\text{labeled}}} \sqrt{d_k}$$

- **Remark**
  - For two nodes i and j, if $d_i > d_j$, then $S(i) > S(j)$
    → Node with higher degree will have higher influence on GCN training
- **Question:** How to mitigate the impact of $\sqrt{d_i}$?

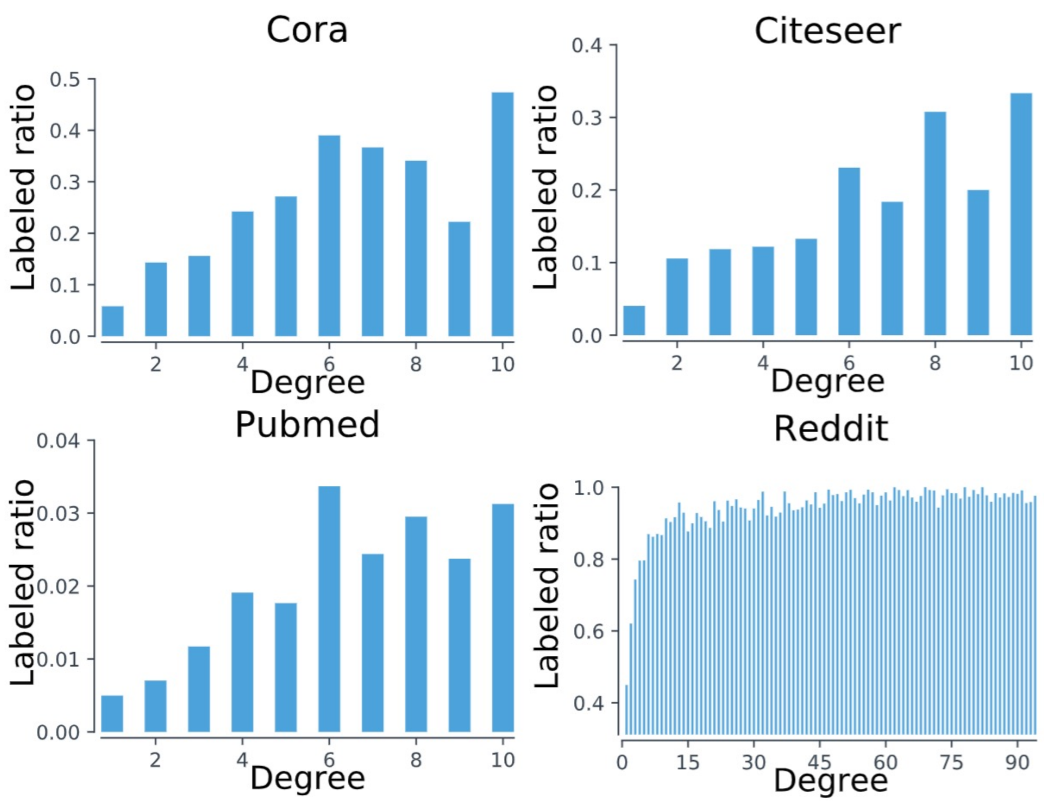
Cause #1: Influence of High-Degree Nodes

- **Goal:** Visualize the influence score $S(\cdot)$ for each node
- **Observation:** High-degree nodes have higher influence score



Cause #2: Ratio of Labeled Neighbors

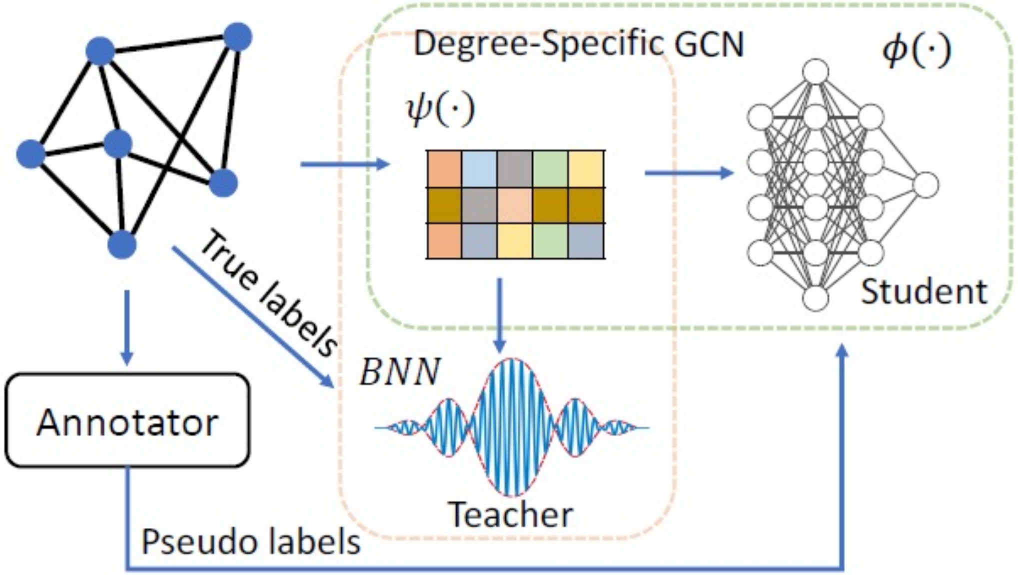
- **Observation:** High-degree nodes are more likely to have labeled neighbors



- **Question:** How to ensure that low-degree nodes receive enough training signals?

SL-DSGCN: Framework

- **Strategy:** Pre-training + fine-tuning
- **Pre-training**
  - Mitigate the impact of node degree by degree-specific GCN
  - Pre-train (1) an annotator through label propagation and (2) a Bayesian neural network (BNN) with true labels for further use in fine-tuning stage



Degree-Specific GCN

- **Degree-specific GCN:** Two components
  - A stack of degree-specific graph convolution layer for embedding learning
  - A fully-connected layer for node classification
- **Given:** The settings of GCN in the l-th layer and
  - d_i: The degree of node i
  - $\mathcal{N}_i$: The neighborhood of node i
  - $\mathbf{W}_{d_j}^{(l)}$: The degree-specific weight w.r.t. degree of node j
- **Degree-specific graph convolution layer**

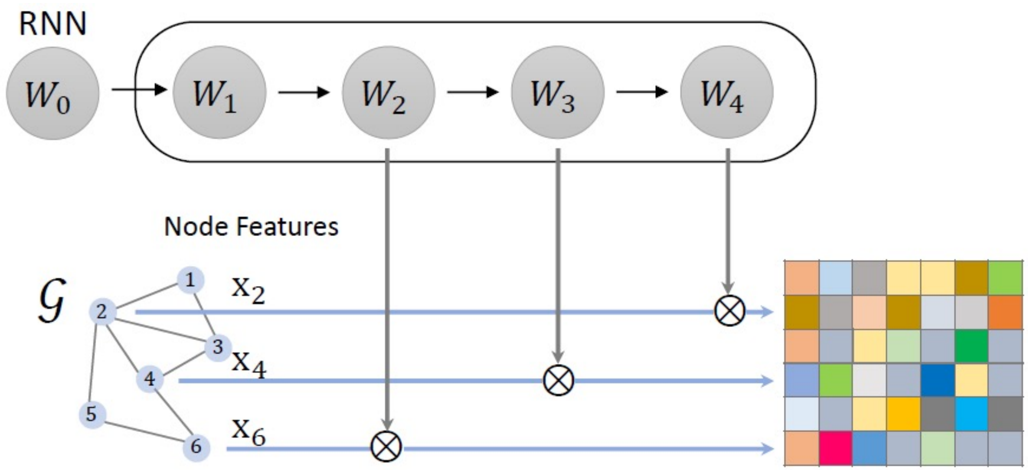
$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}_i \cup \{i\}} a_{ij} \left(\mathbf{W}^{(l)} + \mathbf{W}_{d_j}^{(l)}\right) \mathbf{h}_j^{(l)}\right)$$

- **Question:** How to generate the degree-specific weight?

Degree-Specific Weight

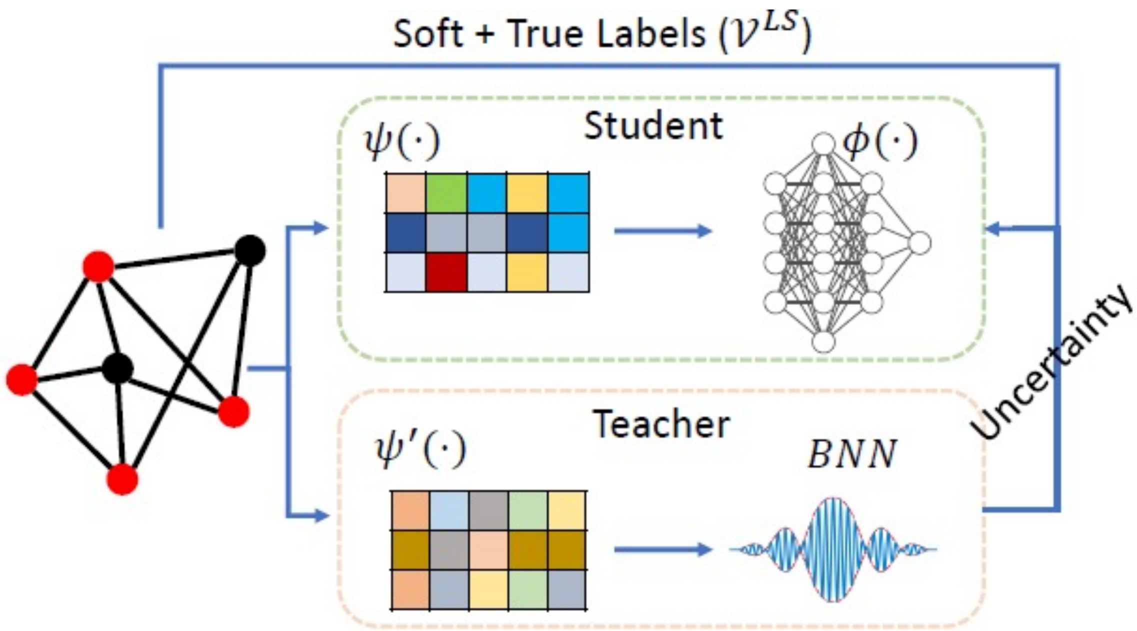
- **Method:** Generate degree-specific weight by a recurrent neural network (RNN)
  - **Hypothesis:** Existence of the complex relations among nodes with different degrees
- **Given:** (1) A RNN and (2) $\mathbf{W}_k^{(l)}$ = degree-specific weight of degree k at l-th layer
- We have

$$\mathbf{W}_{k+1}^{(l)} = \mathrm{RNN}\left(\mathbf{W}_k^{(l)}\right)$$



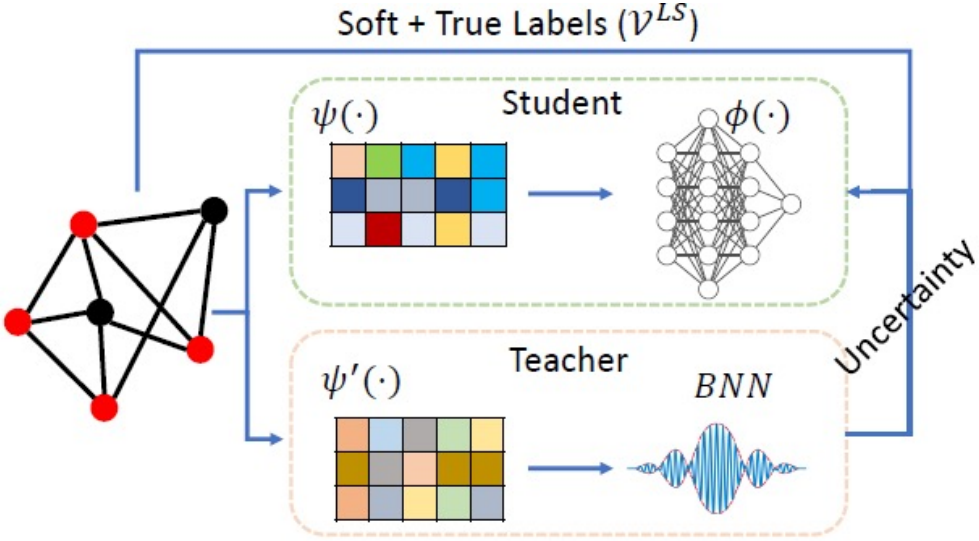
SL-DSGCN: Framework

- **Strategy:** Pre-training + fine-tuning
- **Fine-tuning**
  - Provide pseudo training signals to low-degree nodes for self-supervision



Fine-Tuning with Self-Supervision

- **Student network:** Degree-specific GCN
- **Teacher network:** BNN
  - Provide additional softly-labeled set for self-supervision in student network
    - **Softly-labeled set:** nodes labeled identically by the annotator and the BNN
  - Exponentially decay the learning rate of labeled and softly-labeled nodes by uncertainty score
    - Higher uncertainty score → smaller learning rate



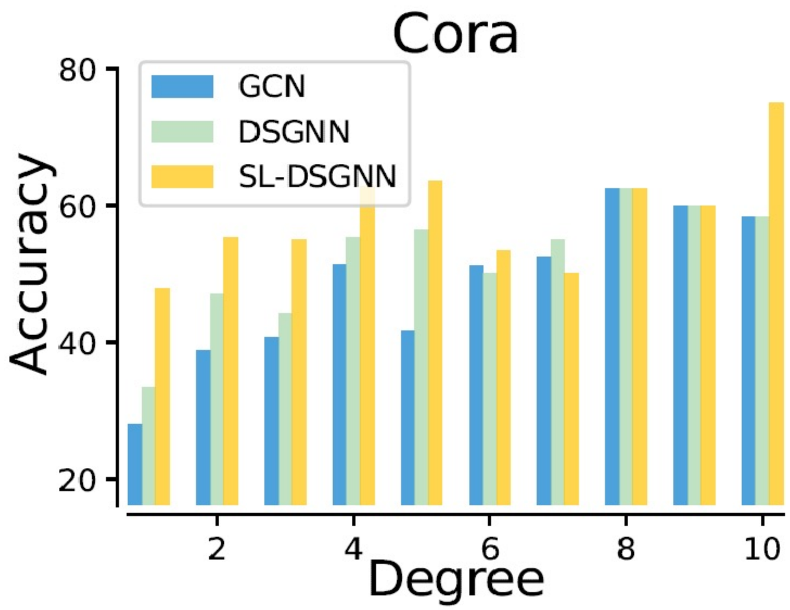
SL-DSGCN: Experiment

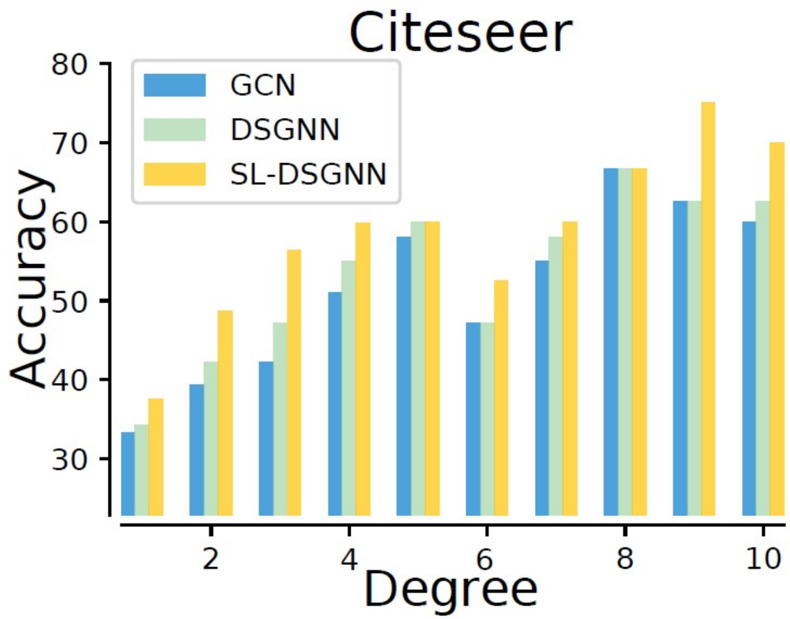
- **Observations**
  - Increased label rate implies higher classification accuracy
  - Self-supervision provides useful information (i.e., high accuracy when the label rate is low)
  - SL-DSGCN outperforms all baseline methods

Dataset	Cora					Citeseer					PubMed		
Label Rate	0.5%	1%	2%	3%	4%	0.5%	1%	2%	3%	4%	0.03%	0.06%	0.09%
LP	29.05	38.63	53.26	70.31	73.47	32.10	40.08	42.83	45.32	49.01	39.01	48.7	56.73
ParWalks	37.01	41.40	50.84	58.24	63.78	19.66	23.70	29.17	35.61	42.65	35.15	40.27	51.33
GCN	35.89	46.00	60.00	71.15	75.68	34.50	43.94	54.42	56.22	58.71	47.97	56.68	63.26
DEMO-Net	33.56	40.05	61.18	72.80	77.11	36.18	43.35	53.38	56.5	59.85	48.15	57.24	62.95
Self-Train	43.83	52.45	63.36	70.62	77.37	42.60	46.79	52.92	58.37	60.42	57.67	61.84	64.73
Co-Train	40.99	52.08	64.27	73.04	75.86	40.98	56.51	52.40	57.86	62.83	53.15	59.63	65.50
Union	45.86	53.59	64.86	73.28	77.41	45.82	54.38	55.98	60.41	59.84	58.77	60.61	67.57
Interesction	33.38	49.26	62.58	70.64	77.74	36.23	55.80	56.11	58.74	62.96	59.70	60.21	63.97
M3S	50.28	58.74	68.04	75.09	78.80	48.96	53.25	58.34	61.95	63.03	59.31	65.25	70.75
SL-DSGCN	**53.58**	**61.36**	**70.31**	**80.15**	**81.05**	**54.07**	**56.68**	**59.93**	**62.20**	**64.45**	**61.15**	**65.68**	**71.78**

SL-DSGCN: Experiment

- **Observations:** Degree-wise classification accuracy
  - SL-DSGCN > DSGNN > GCN for all degrees





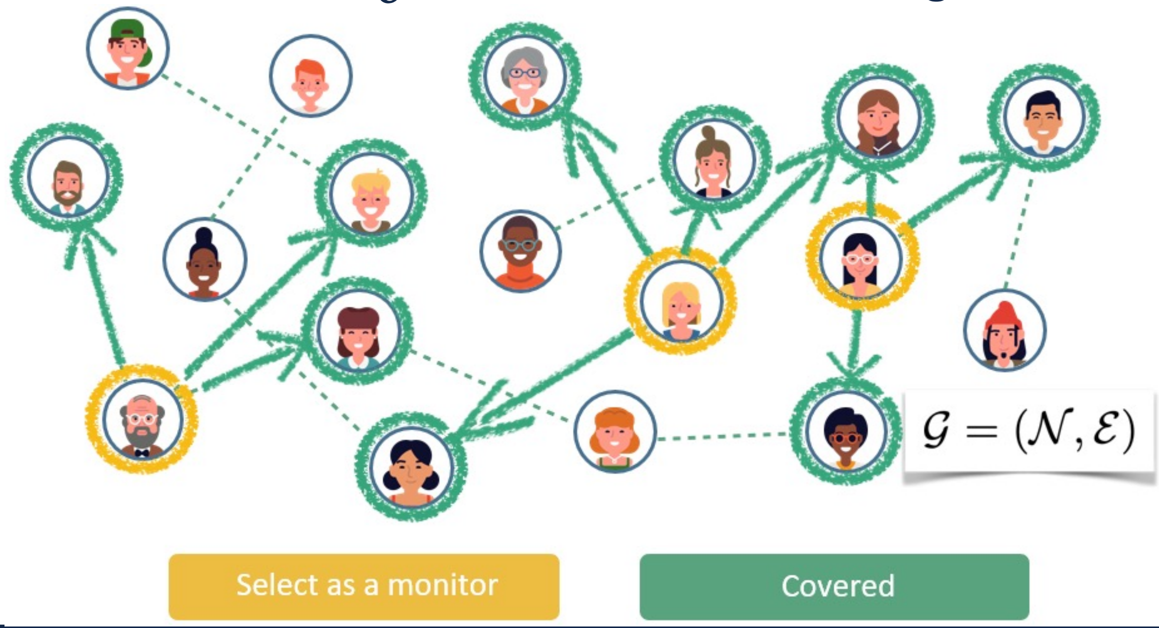
Overview of Part III

Other Fairness Definitions on Graphs

- Counterfactual Fairness
  - Fairness on graph embedding
- Degree-related Fairness
  - Fairness on graph neural networks
- Rawlsian Difference Principle
  - Fairness on graph covering

Preliminary: Graph Covering

- **Definitions**
  - **A monitor:** A node selected by the graph covering algorithm
  - **A covered node:** A neighbor of the monitor
  - **Coverage:** The total number of covered nodes
- **Given:** (1) A graph $\mathcal{G}$; (2) An integer budget I
- **Find:** A subset of I nodes in $\mathcal{G}$ to maximize the coverage
- **Example**



[1] Kratochv, J., Proskurowski, A., & Telle, J.. Complexity of Graph Covering Problems. NJC 1998.

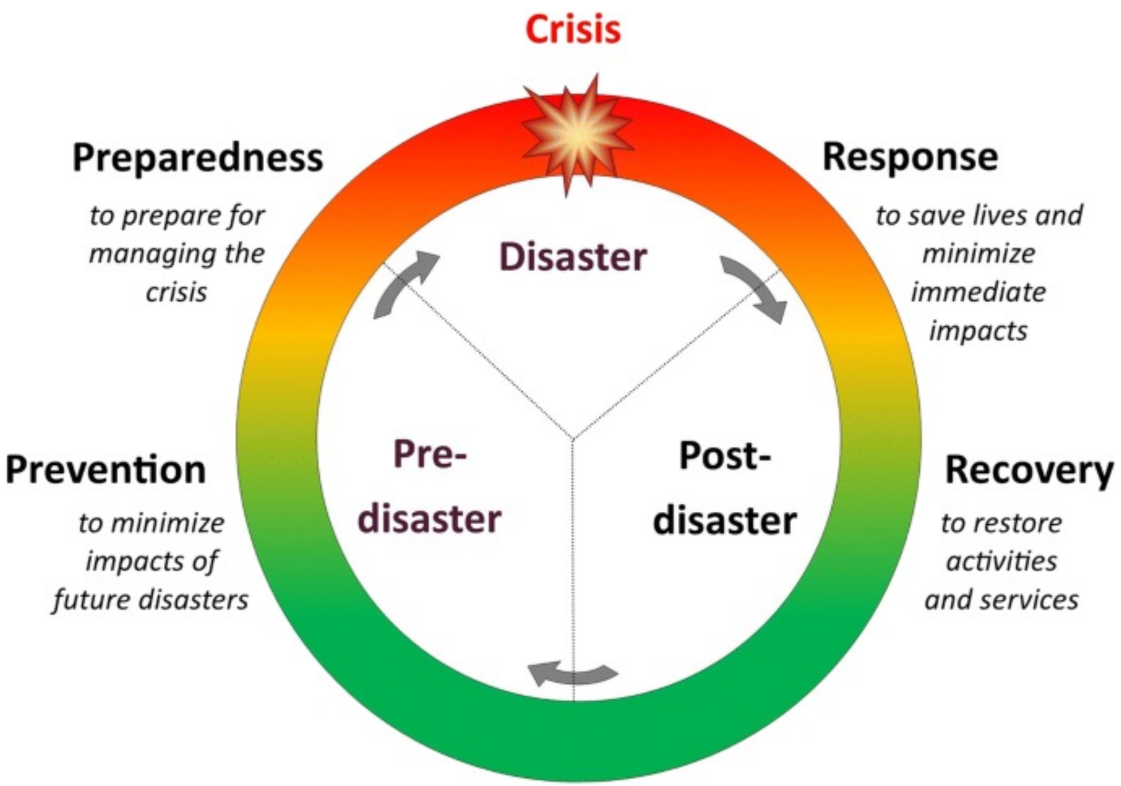
Preliminary: Applications of Graph Covering

- **Suicide prevention**
  - The monitors will identify the warning signs of suicide among their covered nodes

[1] Isaac, M., Elias, B., Katz, L. Y., Belik, S. L., Deane, F. P., Enns, M. W., ... & Swampy Cree Suicide Prevention Team. Gatekeeper Training as a Preventative Intervention for Suicide: A Systematic Review. CJP 2009.

Preliminary: Applications of Graph Covering

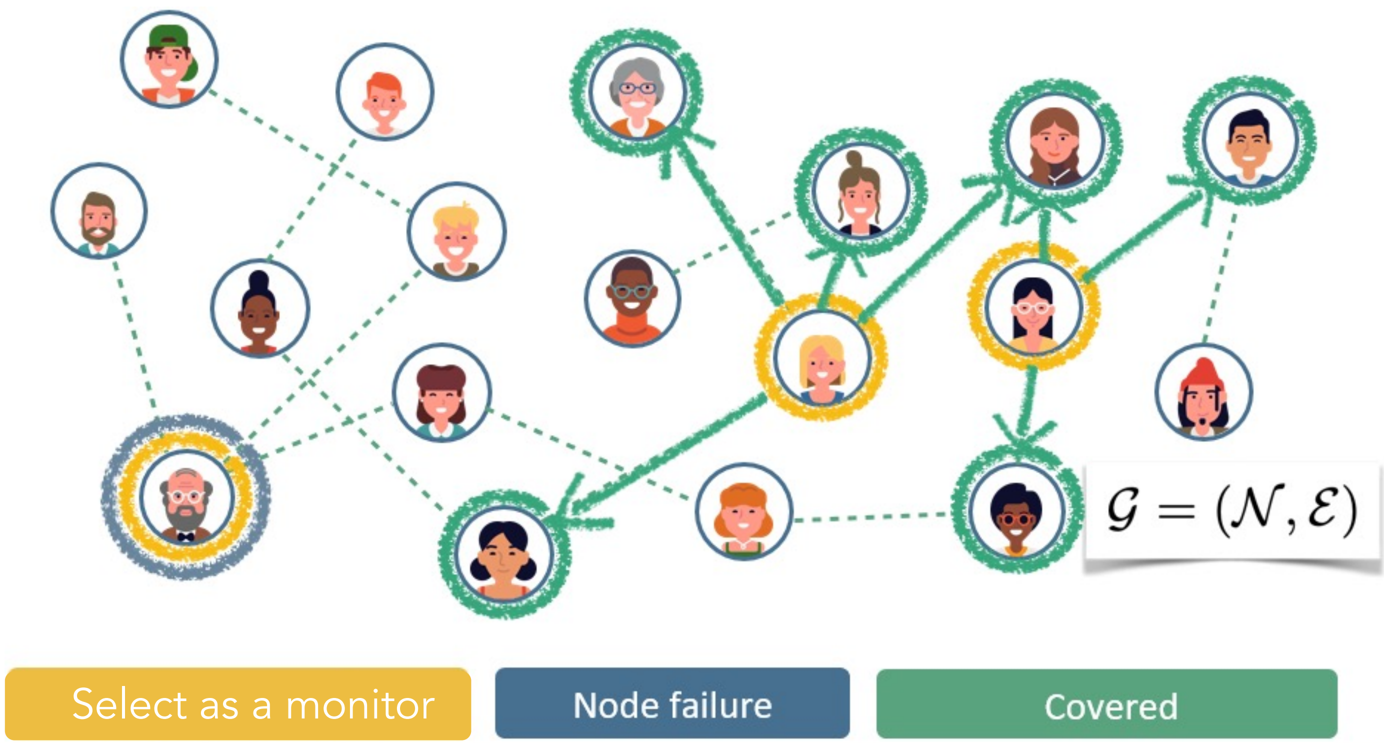
- **Disaster risk management**
  - The monitors will watch out their covered nodes in the case of natural disasters



[1] Ab Ahmad, R., Amin, Z. A. M., Abdullah, C. H., & Ngajam, S. Z.. Public Awareness and Education Programme for Landslide Management and Evaluation Using a Social Research Approach to Determining "Acceptable Risk" and "Tolerable Risk" in Landslide Risk Areas in Malaysia. WLF4 2017.

Robust Graph Covering

- **Key difference:** Some monitors may fail
  - If a monitor fails, its neighbors are not covered
- **Given:** (1) A graph $\mathcal{G}$; (2) Two integer budgets J and I ($J < I$)
- **Find:** A subset of I nodes in $\mathcal{G}$ to maximize the worst-case coverage when any J nodes fail
- **Example**



[1] Tzoumas, V., Gatsis, K., Jadbabaie, A., & Pappas, G. J.. Resilient Monotone Submodular Function Maximization. CDC 2017.

Robust Graph Covering: Formulation

- **Given**
  - A graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with n nodes in total
  - A binary vector $\mathbf{x}$ of length n, $\mathbf{x}[i] = 1$ if the i-th node is selected as monitor
    - The candidate binary vector $\mathbf{x}$ is chosen from a set $\mathcal{X}$
  - A binary vector $\boldsymbol{\xi}$ of length n, $\boldsymbol{\xi}[i] = 1$ if the i-th node does not fail
    - The candidate binary vector $\boldsymbol{\xi}$ is chosen from a set Ξ
  - A binary vector $\mathbf{y}$ of length n, $\mathbf{y}_{\mathbf{x},\boldsymbol{\xi}}[i] = 1$ if the i-th node is covered
- **Define:** The coverage $F_{\mathcal{G}}(\mathbf{x}, \boldsymbol{\xi}) = \mathbf{1}^T \mathbf{y}_{\mathbf{x},\boldsymbol{\xi}}$ where $\mathbf{1}$ with all 1s
- **Mathematical formulation**

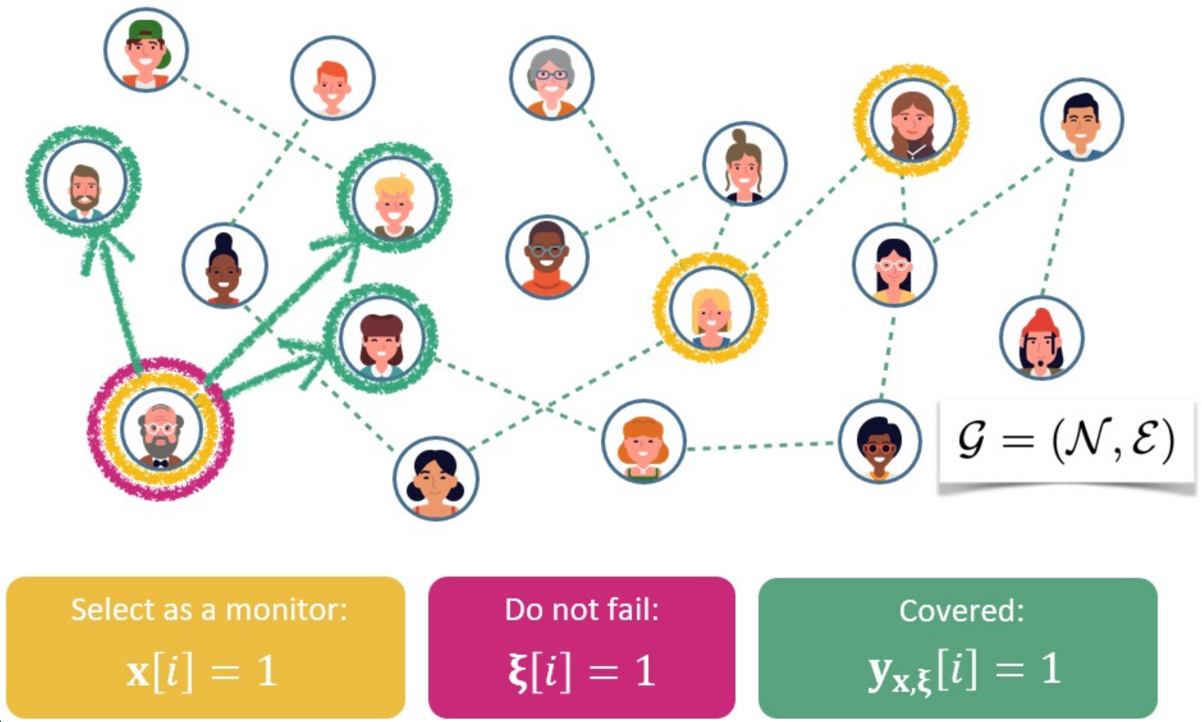
$$\max_{\mathbf{x} \in \mathcal{X}} \min_{\boldsymbol{\xi} \in \Xi} \quad F_{\mathcal{G}}(\mathbf{x}, \boldsymbol{\xi})$$

Maximize its coverage

Find the worst-case

Robust Graph Covering: Example

- **Monitor:** Node in yellow
- **Covered nodes:** Neighbors of the nodes in both yellow and red
- **Coverage:** The number of nodes in green
- **Goal:** Maximize the number of green nodes



Unfairness in Robust Covering

- **Observation:** Coverage by racial group varies in the network
  - Existing algorithms are biased against the race of an individual

Network	Size	Percentage Covered by Racial Group				
		White	Black	Hisp.	Mixed	Other
SPY1	95	70	**36**	-	86	95
SPY2	117	78	-	**42**	76	67
SPY3	118	88	-	**33**	95	69
MFP1	165	96	77	69	73	**28**
MFP2	182	**44**	85	70	77	72

- **Question:** Can we ensure that different racial groups have similar coverages?

[1] Tzoumas, V., Gatsis, K., Jadbabaie, A., & Pappas, G. J.. Resilient Monotone Submodular Function Maximization. CDC 2017.

Fairness in Robust Graph Covering

- **Fairness definition**
  - Rawlsian difference principle to maximize the utility of the worst-off groups
- **Example**
  - **Sensitive attribute:** Race
  - We need to maximize the utilities of Hispanic people in SPY2 and SPY3

Network	Size	Percentage Covered by Racial Group				
		White	Black	Hisp.	Mixed	Other
SPY1	95	70	**36**	-	86	95
SPY2	117	78	-	**42**	76	67
SPY3	118	88	-	**33**	95	69
MFP1	165	96	77	69	73	**28**
MFP2	182	**44**	85	70	77	72

[1] Rahmattalabi, A., Vayanos, P., Fulginiti, A., Rice, E., Wilder, B., Yadav, A., & Tambe, M.. Exploring Algorithmic Fairness in Robust Graph Covering Problems. NeurIPS 2019.

RCFair: Robust Graph Covering with Fairness Constraint

- **Given:** The settings of robust graph covering and
  - A set of sensitive attribute value $\mathcal{C}$, e.g., $\mathcal{C} = \{\text{male}, \text{female}\}$ for gender
  - The demographic groups $\mathcal{N} = \cup_{c \in \mathcal{C}} \mathcal{N}_c$ defined by $\mathcal{C}$
- **Define:** The group-specific coverage $F_{\mathcal{G},c}(\mathbf{x}, \boldsymbol{\xi}) = \sum_{i \in \mathcal{N}_c} \mathbf{y}_{\mathbf{x},\boldsymbol{\xi}}[i]$
- **Mathematical formulation**

$$\max_{\mathbf{x} \in \mathcal{X}} \min_{\boldsymbol{\xi} \in \Xi} \quad F_{\mathcal{G}}(\mathbf{x}, \boldsymbol{\xi}) = \sum_{c \in \mathcal{C}} F_{\mathcal{G},c}(\mathbf{x}, \boldsymbol{\xi})$$

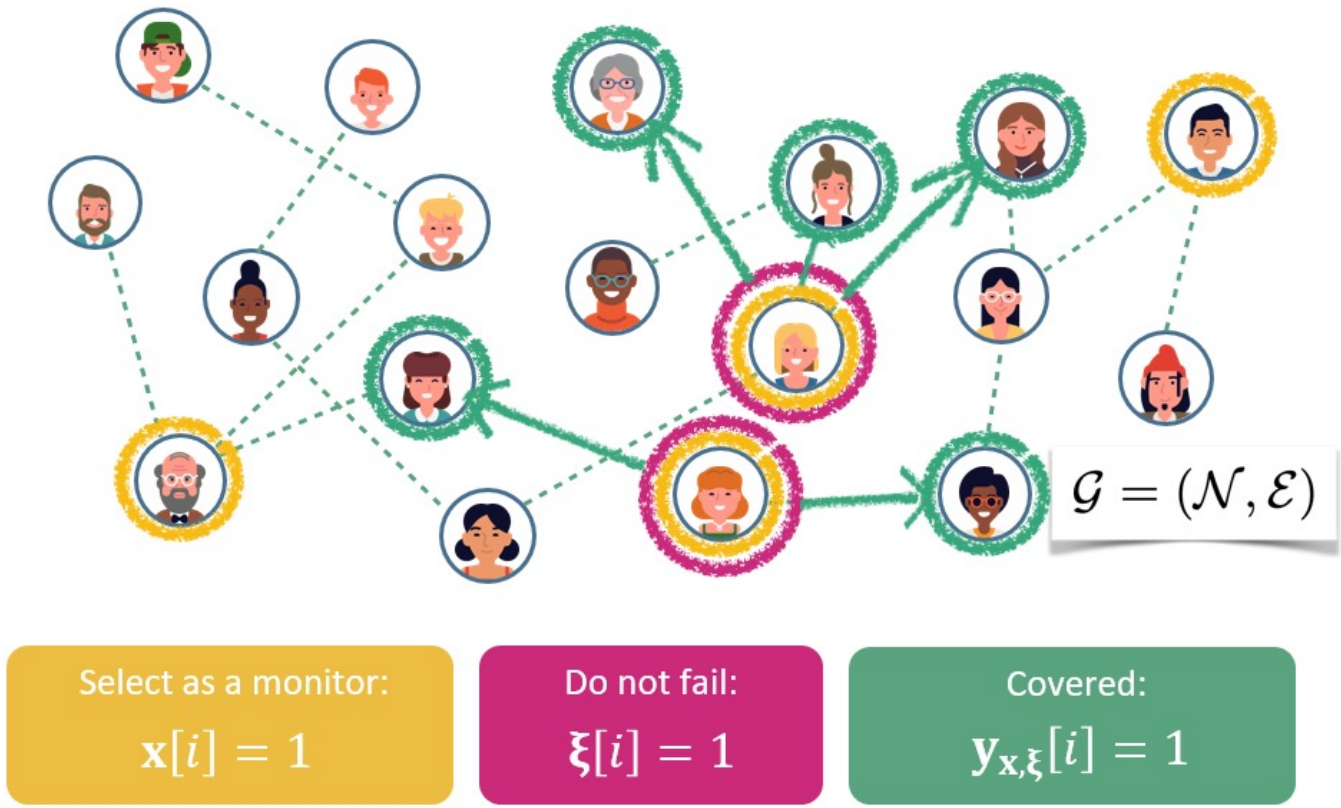
$$\text{s.t.} \quad F_{\mathcal{G},c}(\mathbf{x}, \boldsymbol{\xi}) \geq W|\mathcal{N}_c| \quad \forall c \in \mathcal{C}, \forall \boldsymbol{\xi} \in \Xi$$

where $W \in [0,1]$ is a constant

- **Intuition of fairness constraint**
  - At least W fraction of nodes from each group should be covered

RCFair: Example

- **Example**
  - 15 people with lighter skin, 3 people with darker skin
  - $W = 1/3$, at least 1/3 of people in each group should be covered



RCFair: Hardness

- **Formulation**

$$\max_{\mathbf{x}\in\mathcal{X}} \min_{\boldsymbol{\xi}\in\Xi} \quad F_{\mathcal{G}}(\mathbf{x},\boldsymbol{\xi}) = \sum_{c\in\mathcal{C}} F_{\mathcal{G},c}(\mathbf{x},\boldsymbol{\xi})$$
$$\text{s.t.} \quad F_{\mathcal{G},c}(\mathbf{x},\boldsymbol{\xi}) \geq W|\mathcal{N}_c| \quad \forall c\in\mathcal{C}, \forall\boldsymbol{\xi}\in\Xi$$

- **Challenges**
  - Discontinuity of $F_{\mathcal{G}}(\mathbf{x},\boldsymbol{\xi})$
  - NP hard combinatorial problem
- **Question:** How to solve the problem?

RCFair: Problem Reformulation

- **Key idea:** Reformulation on the objective function

$$F_{\mathcal{G}}(\mathbf{x}, \boldsymbol{\xi}) = \max_{\mathbf{y}} \left\{ \sum_{i \in \mathcal{N}} \mathbf{y}[i] : \mathbf{y}[i] \leq \sum_{j \in \delta_i} \mathbf{x}[i]\boldsymbol{\xi}[j], \forall i \in \mathcal{N} \right\}$$

  - δ_i: neighborhood of node i
- **Equivalence of reformulation**
  - $\mathbf{y}[i] = 1$ if and only if node i is covered
  - $\mathbf{y}[i] = 0$ when no neighbor of node i is the non-'fail' monitor
- **Question:** How to reformulate the RCFair problem?

RCFair: Problem Reformulation

- **Problem reformulation**
  - Reformulation on both $F_{\mathcal{G},c}(\mathbf{x}, \boldsymbol{\xi})$ and $F_{\mathcal{G},c}(\mathbf{x}, \boldsymbol{\xi})$

$$\max_{\mathbf{x}\in\mathcal{X}} \min_{\boldsymbol{\xi}\in\Xi} \max_{\mathbf{y}\in\mathcal{Y}} \quad \left\{ \sum_{i\in\mathcal{N}} \mathbf{y}[i] : \mathbf{y}[i] \leq \sum_{j\in\delta_i} \mathbf{x}[i]\boldsymbol{\xi}[j], \forall i \in \mathcal{N} \right\}$$

$$\text{s.t.} \quad \mathcal{Y} = \left\{ \mathbf{y}: \sum_{i\in\mathcal{N}_c} \mathbf{y}[i] \geq W|\mathcal{N}_c|, \forall i \in \mathcal{N}_c \right\}$$

- **Challenge:** Max-min-max problem
  - How to solve?

RCFair: K-Adaptability Approximation

- **Key steps**
  - Find K candidate solutions that
    - Achieve highest coverage
    - Satisfy the fairness constraint
    - Without considering node failure
  - Select the best solution within the candidates when considering node failure
- **Question:** How to find K candidates efficiently?
- **Solution:** Derive the equivalence to Mixed-Interger Linear Programming (MILP)
  - Apply Bender's Decomposition to solve the MILP

[1] Hanasusanto, G. A., Kuhn, D., & Wiesemann, W.. K-Adaptability in Two-Stage Robust Binary Programming. OPRE 2015.
[2] Rahmattalabi, A., Vayanos, P., & Tambe, M.. A Robust Optimization Approach to Designing Near-Optimal Strategies for Constant-Sum Monitoring Games. GameSec 2018.

RCFair: Price of Fairness

- **Intuition:** Incorporating fairness constraint comes at a price
  - Lead to suboptimal solution to take care of the fairness
- **Question:** What is the cost of ensuring fairness?
- **Definition:** Price of Fairness (PoF)
  - $\mathrm{OPT}(\mathcal{G}, I, J)$: Optimal coverage without fairness constraint
  - $\mathrm{OPT}^{\mathrm{fair}}(\mathcal{G}, I, J)$: Optimal coverage with fairness constraint

$$\mathrm{PoF}(\mathcal{G}, I, J) = 1 - \frac{\mathrm{OPT}^{\mathrm{fair}}(\mathcal{G}, I, J)}{\mathrm{OPT}(\mathcal{G}, I, J)}$$

- **Intuition of PoF**
  - High PoF → few nodes are covered in fair solution

RCFair: Price of Fairness in Real Networks

- **Given:** An arbitrary number $\epsilon > 0$
- There exists
  - A budget I
  - A network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with $\mathcal{N} \geq \frac{4}{\epsilon} + 3$
- Such that

$$\mathrm{PoF}(\mathcal{G}, I, 0) \geq 1 - \epsilon$$

- **Remark:** RCFair without node failure can be arbitrarily bad in real networks

RCFair: Price of Fairness in Random Graphs

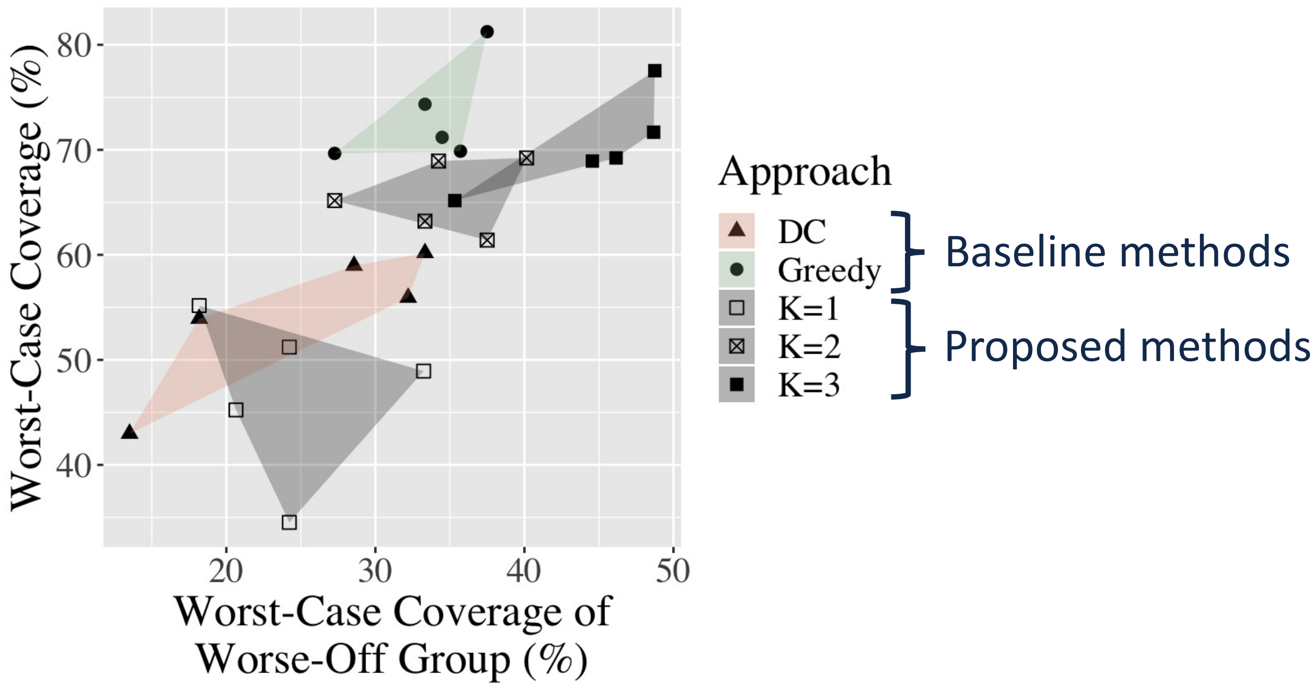
- **Random graph model:** Stochastic Block Model (SBM)
- **Expected PoF:** $\overline{\text{PoF}}(I,J) = 1 - \mathbb{E}_{\mathcal{G}\sim\text{SBM}}\left[\frac{\text{OPT}^{\text{fair}}(\mathcal{G},I,J)}{\text{OPT}(\mathcal{G},I,J)}\right]$
- **Conditions**
  - Certain assumption on the edge probability in SBM model
  - Budget $I = O(\log|\mathcal{N}|)$
- **Expected PoF in SBM**

$$\overline{\text{PoF}}(I,J) = 1 - \frac{\eta\sum_{c\in\mathcal{C}}|\mathcal{N}_c| + J\sum_{c\in\mathcal{C}\setminus\{C\}} d(c)}{(I-J)d(|\mathcal{C}|)} - o(1)$$

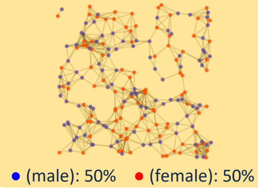
  - $d(c)$ is a term related to $|\mathcal{N}_c|$
  - η is related to $|\mathcal{C}|$ and $|\mathcal{N}_c|$ for each $c \in \mathcal{C}$
- **Remark:** Expected PoF in SBM model changes with the relative size of each community determined by the sensitive attribute

RCFair: Experiment

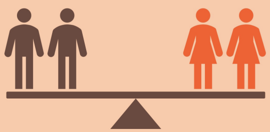
- **Shaded area:** Convex hull of the associated algorithm
- **Observations**
  - **Utility:** The proposed method (K=3) have similar worst-case coverage with state-of-the-art
  - **Fairness:** The proposed method has the best worst-case coverage of the worst-off group



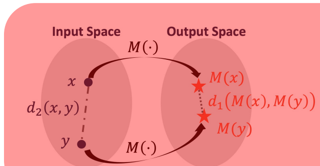
Roadmap



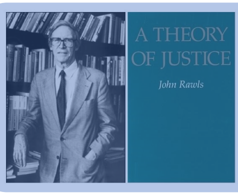
Introduction



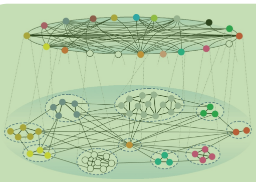
Part I: Group Fairness on Graphs



Part II: Individual Fairness on Graphs



Part III: Other Fairness Definitions on Graphs



Part IV: Future Research Directions

Fairness on Dynamic Graphs

- **Motivation:** Networks are dynamically changing over time
- **Trivial solution:** Re-run the fair graph mining algorithm from scratch at each timestamp
- **Limitations**
  - Time-consuming to re-train the mining model
  - Fail to capture the dynamic information in ensuring fairness
- **Questions**
  - How to efficiently update the mining results and ensure the fairness at each timestamp?
  - How to characterize the impact of dynamics over the bias measure?

Fairness on Multi-Networks

- **Motivation:** Real-world networks are often multi-sourced

Social Networks

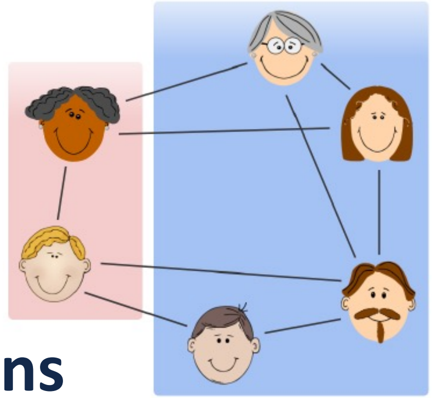
Transaction Networks

- **Trivial solution:** Flatten the multi-network to a single network and ensure fairness on the flattened single network
- **Limitations**
  - May introduce noise due to different distributions of different networks
  - Fail to characterize the impact of cross-network links in ensuring fairness
- **Question**
  - How to ensure the fairness of mining results across multiple networks?
  - How to understand the implication of ensuring fairness on one network over the bias of another network?

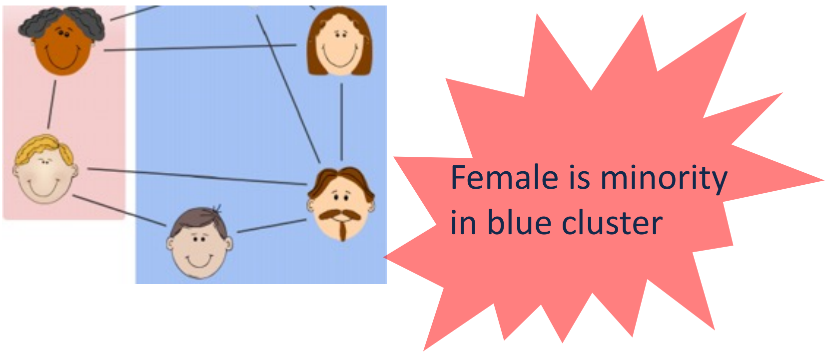
Multi-Resolution Fairness on Graphs

- **Motivation:** Fairness on the entire graph may not imply the fairness on a subgraph
- **Example**

Fair clustering in graph level



Unfair clustering in subgraph level



- **Questions**
  – Can we ensure multi-resolution fairness on graph mining?
    - The mining results are fair across multiple resolution (e.g., graph-level, subgraph-level, node-level)?
  – How to characterize the relationship between the hierarchical structure of the graph and the multi-resolution fairness constraint?

Takeaways

- **Background knowledge and related problems**
- **Group fairness on graphs**
  - Ranking, clustering, embedding
- **Individual fairness on graphs**
  - Laplacian regularization-based approach, ranking-based approach
- **Other fairness definitions on graphs**
  - Counterfactual fairness, degree-related fairness, Rawlsian difference principle
- **Future research directions**

Acknowledgements

- Part of the slides are credited to
  - Panayiotis Tsaparas (University of Ioannina)
  - Matthäus Kleindessner (Amazon)
  - Avishek Joey Bose (McGill University)
  - Maarten Buyl (Ghent University)
  - Yushun Dong (University of Virginia)
  - Chirag Agarwal (Harvard University)
  - Xianfeng Tang (Amazon)
  - Aida Rahmattalabi (University of Southern California)
- If you want to re-use these slides, please contact the original authors directly.

References

- Tsioutsiouliklis, S., Pitoura, E., Tsaparas, P., Kleftakis, I., & Mamoulis, N.. Fairness-Aware PageRank. WWW 2021.
- Kleindessner, M., Samadi, S., Awasthi, P., & Morgenstern, J.. Guarantees for Spectral Clustering with Fairness Constraints. ICML 2019.
- Bose, A., & Hamilton, W.. Compositional Fairness Constraints for Graph Embeddings. ICML 2019.
- Rahman, T., Surma, B., Backes, M., & Zhang, Y.. FairWalk: Towards Fair Graph Embedding. IJCAI 2019.
- Buyl, M., & De Bie, T.. DeBayes: A Bayesian Method for Debiasing Network Embeddings. ICML 2020.
- Kang, J., He, J., Maciejewski, R., & Tong, H.. InFoRM: Individual Fairness on Graph Mining. KDD 2020.
- Dong, Y., Kang, J., Tong, H., & Li, J.. Individual Fairness for Graph Neural Networks: A Ranking based Approach. KDD 2021
- Agarwal, C., Lakkaraju, H., & Zitnik, M.. Towards a Unified Framework for Fair and Stable Graph Representation Learning. UAI 2021.
- Tang, X., Yao, H., Sun, Y., Wang, Y., Tang, J., Aggarwal, C., ... & Wang, S.. Investigating and Mitigating Degree-Related Biases in Graph Convolutional Networks. CIKM 2020.
- Rahmattalabi, A., Vayanos, P., Fulginiti, A., Rice, E., Wilder, B., Yadav, A., & Tambe, M.. Exploring Algorithmic Fairness in Robust Graph Covering Problems. NeurIPS 2019.