This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

GREASE: Graph Imbalance Reduction by Adding Sets of Edges

Yoosof Mashayekhi, Bo Kang, Jefrey Lijffijt, and Tijl De Bie

Abstract

Real-world data can often be represented as a heterogeneous network relating nodes of different types. E.g., a job market can be represented as a job seeker-skill-vacancy network. It can be relevant to consider the *imbalance* between nodes of different types, in terms of whether they are similarly connected in the network. For example, it is desirable that job seekers and vacancies are mixed well. If they are not, then there is imbalance. We propose to quantify the imbalance *between two sets of nodes* in a network as the Earth Mover's Distance between the sets. Given this quantification, we introduce `GREASE` (Graph imbalance REduction by Adding Sets of Edges), a method that selects a fixed number of unconnected node-pairs, which—if links were added between them—aims to maximally reduce the imbalance. In the job market network, `GREASE` can be used to select skills that job seekers do not yet have, but could strive to acquire, to reduce the imbalance between job seekers and vacancies. `GREASE` may also be used in other applications, such as reducing controversy between opposing sides on a polarizing topic. We evaluated `GREASE` on several datasets and find that `GREASE` outperforms baselines in reducing network imbalance.

Index Terms

Graphs and networks, Graph algorithms, Imbalance reduction, Network embedding, Representation learning

- *Y. Mashayekhi, B. Kang, J. Lijffijt, and T. De Bie are with the IDLab, Department of Electronics and Information Systems (ELIS), Ghent University, Gent 9000, Belgium.*

  E-mail: {yoosof.mashayekhi,bo.kang,jefrey.lijffijt,tijl.debie}@ugent.be

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

GREASE: Graph Imbalance Reduction by Adding Sets of Edges

1 Introduction

Graphs (or networks) are natural models for a wide range of real-world structures [1], arising from, e.g., social networks [2], biology (Protein-Protein interaction networks) [3], and linguistics (word co-occurrence networks) [4]. Network embedding provides a flexible means to solve graph analytics problems by encoding nodes as real-valued vectors, which can later be used as input feature vectors to a machine learning model [5]. Using these vector representations, machine learning methods can be applied to graph datasets to perform analysis tasks such as link prediction [6], information diffusion [7], and node classification [8].

An imbalance between two sets of nodes in terms of their connectivity in a network may be undesirable. This paper (1) studies how to quantify network imbalance and (2) proposes a method to reduce network imbalance by adding a limited number of links to the network.

Motivation. There are many networks for which it is desirable to minimize the imbalance between specific sets of nodes. Let us consider an example of a job market network with three sets of nodes: job vacancies, job seekers, and skills. Job vacancies are connected to the skills they require, and job seekers are connected to the skills they have and possibly to the job vacancies they have shown an interest in.

Imagine there are many Python developers seeking a job and few vacancies requiring Python programming, while there are many vacancies requiring Java programming and few Java developers on the market. As a result, many Java jobs would remain unfilled and Python programmers would remain unemployed. With an ever faster evolving job market, such imbalances are increasingly common and serious, harming job market efficiency and ultimately the economy. Quantifying such imbalances may provide policy makers with an objective picture of the current state of affairs.

Moreover, the ability to quantify imbalance also opens up the possibility of trying to reduce it through targeted interventions and incentives. While policy makers may not be able to influence employers to shift their requirements, they can provide courses and training material for specific worker profiles lacking sought-after skills, to shift their area of expertise and better meet the demand of the job market. In network terms, it is equivalent to adding a certain number of links connecting job seekers (let us call this the set of *source nodes*) to skills (*auxiliary nodes*), to reduce the imbalance between job seekers (*source nodes*) and job vacancies (*target nodes*).

A naive approach would be to formalize imbalance in this case as the difference in the number of job seekers having a skill and the number of jobs requiring that skill, but the problem is actually more complex. Jobs typically require a rich set of skills, hence to fill an open position for which there are no perfect candidates, it makes sense to offer a relevant course to a job seeker that is otherwise most suitable. We aim to go beyond individual recommendations and study this problem at the level of the whole network, which also means there are interactions and competing interests. Hence, the imbalance quantification is based on a type of distance between job seekers and positions (or abstractly, between any two given sets of nodes).

Other applications. Imbalance quantification and reduction could have applications in domains other than the job market, such as controversy in social networks. In social networks, there are topics that users have different opinions on such as politics, COVID-19 vaccination, sports club fans, etc. This conflict in users' opinions could result in an imbalance between users. In this case, recommending new content with a non-controversial topic to users could reduce the imbalance between them in the social network, thus potentially decreasing conflict. Another example of imbalance is in a company where employees have multiple expertise and projects also require a set of expertises. It may not be possible to allocate the employees over the projects such that all requirements are met, in which case there is an imbalance. Training courses could be offered to a well-chosen set of employees to expand their area of expertise to meet the needs of their projects. In this way, the imbalance between employees and projects would be reduced. A similar problem may also exist in universities for assigning students to supervisors for performing their theses due to a mismatch in the students' skills and the supervisors' expertises. As a solution, some students could be recommended additional courses related to the work of less overloaded supervisors, allowing them to do a thesis with those supervisors.

In the paper, we use the job market example as a running example, for clarity of the explanations. Our contributions are generic and applicable well beyond this use case. We also evaluate our method to reduce the imbalance in networks in a wider diversity of settings in the experiments.

Our approach. In a job market network, assigning job seekers to job vacancies with the lowest cost—where cost could be defined as the required training time of employees in the company, or the effort a job seeker has to make to be suited for a job—appears to be the ideal situation. Hence, informally speaking, we quantify the imbalance through Earth Mover's Distance (EMD) [9], with a model-dependent distance between the nodes in the network. The EMD corresponds to the minimal amount of work to map two sets onto each other. It is equivalent to the Wasserstein 1-distance, and as shown in Appendix A, it also corresponds to a matching problem, i.e., to fractional perfect b-matching.

More formally: We denote an undirected network by $G = (V, E)$, where V and E are the sets of nodes and links respectively. Moreover, we define two sets of nodes, namely *source nodes* $S \subset V$ (e.g. job seekers in the job market

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

network) and *target nodes* $T \subset V$ (e.g. job vacancies). S and T must be disjoint ($S \cap T = \emptyset$) and V may contain also other nodes besides those contained in S and T. There are no restrictions on the links in the network, i.e., both links within and between nodes of any sets are allowed.

We define the *imbalance* between S and T in G as the EMD between the sets S and T, where the costs of mapping each node in S to each node in T (costs in EMD) are given by a model. The mapping cost is lower for a pair of nodes from S and T if they have a higher affinity with each other in the network. More details are presented in Section 2.

Next, we define the problem of adding a limited number of links between sets S and $U \subseteq V$ in graph G, to *reduce the imbalance* between sets S and T. Adding links will change the structure of the network G and thereby affect the imbalance (i.e., the EMD). We propose a method called Graph imbalance REduction by Adding Sets of Edges (GREASE) to tackle this problem. GREASE is based on updating the parameters of some nodes in the model that provides the mapping costs, instead of updating the whole model after adding a few links, thus providing the necessary scalability. The proposed method benefits from some heuristics and assumptions that are necessary to solve the problem in a limited execution time. For the definition of imbalance, U can be any subset of V and overlap with S and T, but in our experiments U will be disjoint from both S and T. Hence, the effectivity of GREASE is only evaluated for this setting.

This paper is an **extended version** of our previous paper [10], proposing a completely new method for the problem of imbalance reduction with a major difference in imbalance quantification and problem formulation. The **main contributions** of the present paper are:

- We *define the imbalance* between two sets of nodes, S and T, in a network and propose the *measure* $\psi(\boldsymbol{D}, S, T)$ *for quantifying it*, where the costs of node-pair mappings $\boldsymbol{D}$ are given by a suitable model based on the network structure.
- We introduce the novel problem of *reducing the imbalance* in a network by adding a number of links.
- We also propose a novel generic method, *Graph imbalance REduction by Adding Sets of Edges (GREASE)*, to select a set of links in an efficient manner.
- To better understand the merits of GREASE, we propose two *baselines* (a naive and a more intelligent baseline) for the novel problem of reducing the imbalance in a network.
- GREASE is proposed as a generic method, applicable to a wide range of models that provides the costs of node-pair mappings in networks. We also develop a concrete instantiation using *conditional network embedding (CNE)* [11], a state-of-the-art network embedding method that provides the node-pair mapping costs (based on link probabilities).
- We perform several experiments to compare the performance of GREASE and baselines in reducing the imbalance in a network. The experiments show that GREASE outperforms the baselines in this task.

Outline. In Section 2, we define and quantify imbalance in networks. In Section 3, we formalize the problem of reducing network imbalance by adding links to a network. In Section 4, we introduce our method GREASE to reduce the imbalance in a network. In Section 5, we provide an experimental evaluation of GREASE. In Section 6, we discuss the related work. In Section 7, we conclude and outline avenues for future work.

2 Network Imbalance

In this section, we first review EMD. Secondly, we motivate informally through an example of what the imbalance measure should quantify. Finally, we provide the definition and quantification of imbalance in networks.

2.1 Earth Mover's Distance

The Earth Mover's Distance (EMD) is a measure of the distance between two distributions and we restrict it here to the distance between two probability distributions. EMD is defined as the minimum amount of work to map the distribution of the source nodes to the distribution of the target nodes, based on given mapping costs d_{ij} for all pairs of events (here nodes of the network) $i \in S, j \in T$. Formally:

Definition 1 (Earth Mover's Distance (EMD) [9])**.** *Assume two probability distributions represented by $P_S = \{(i, w_i)|i \in S\}$ and $P_T = \{(j, w_j)|j \in T\}$, where w_i and w_j are the probabilities for the respective events. Let $\boldsymbol{D} = [d_{ij}]$ be the matrix of mapping costs between events in P_S and P_T. EMD is a linear program whose goal is to find a flow $\boldsymbol{F} = [f_{ij}]$ between two distributions P_S and P_T that minimizes the overall cost:*

$$
\begin{aligned}
D_{\text{EMD}}(\boldsymbol{D}, P_S, P_T) = \min_{\boldsymbol{F}} \quad & \sum_{i=1}^{m}\sum_{j=1}^{r} f_{ij} d_{ij} \\
\text{s.t.} \quad & f_{ij} \geq 0, \;\; 1 \leq i \leq m, \;\; 1 \leq j \leq r, \\
& \sum_{j=1}^{r} f_{ij} = w_{pi}, 1 \leq i \leq m, \qquad (1) \\
& \sum_{i=1}^{m} f_{ij} = w_{qj}, 1 \leq j \leq r, . \qquad (2)
\end{aligned}
$$

The optimal flow $\boldsymbol{F}$ is found by solving this linear optimization problem.

2.2 Definition of network imbalance

Before we define our proposed notion of imbalance formally, we explain it through the concrete example of the job market network. For this example, the imbalance is the EMD between all job seekers and job vacancies. Note that any job seeker and job can be mapped in EMD, also if they are not connected in the original graph, although the mapping cost is typically related to whether they are connected (we use costs based on a machine learning model of the network).

A link in the network between a job seeker and a job vacancy might mean that the job seeker has applied for the job or has otherwise expressed interest, not necessarily that they were employed for that job. Importantly, the absence of a link does not imply that the job seeker is a poor candidate for the job. *This property distinguishes our work from the literature on combinatorial matching problems in graphs.*

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

However, the skills to which the job seeker and job vacancy are linked, and jobs vacancies they *are* linked to, provide information on whether the job seeker is suited for the job; and the more suited, the smaller their mapping cost in EMD should be. Hence, the cost of mapping a job seeker to a job vacancy should be a function of the network structure, and adding or removing skills to that job seeker or job vacancy should influence their mapping cost. Thus, the cost could be defined using a model that provides link costs or link probabilities (so the costs of node-pair mapping could be computed based on them), based on the network structure. In this paper, we investigate using link probabilities provided by a network embedding for this, as it is a state-of-the-art approach to summarize the network structure, where proximity between node embeddings reflects the probability that both nodes *should be* linked.

Figure 1 shows a sample job market, where a Java product manager (E) and a Python developer (F) are looking for jobs. The costs of mapping job seekers to the job positions in their area of expertise is lowest and with other job positions is higher. Mapping with a low cost may correspond to a high chance of employment. The ideal case is where there are equally many job positions with low mapping costs for each of the job seekers (Figure 1a). On the other hand, Figure 1b shows an imbalanced job market where one job does not have a really suitable job seeker. If we define the EMD to map each job seeker to two job positions, then the dashed lines give an optimal solution, mapping the Java product manager (E) with the Python product management job (B), highlighting the imbalance.

2.3 Formal definition of network imbalance

Recall we denote an undirected network by $G = (V, E)$, where V is the set of $|V| = n$ nodes and $E \subseteq \binom{V}{2}$ is the set of links between nodes. For convenience, we index the set of nodes by natural numbers, i.e. $V = \{1, \ldots, n\}$. Let $\boldsymbol{A}$ denote the adjacency matrix, and a_{ij} is the element of the adjacency matrix corresponding to the link between node i and node j, i.e. $a_{ij} = 1$ iff $\{i, j\} \in E$. Function $z_{ij}(\boldsymbol{A})$ denotes the probability of having a link between nodes i and j in graph G, for some function $z_{ij} : \mathbb{R}^{n\times n} \rightarrow [0, 1]$. A suitable function z is presented below in Section 2.4.

Given a cost defined for each node-pair $\{i, j\}$, $\boldsymbol{D} = [d_{ij}]$ (e.g., based on the link probability or distance in the embedding space), we define the imbalance as the EMD between the distributions P_S and P_T defined respectively over the node sets S and T. These distributions are defined by means of probability weigths $w_i = \frac{1}{|S|}$ for nodes $i \in S$, and $w_j = \frac{1}{|T|}$ for nodes $j \in T$. We can then apply the EMD definition D_{EMD} from Section 2.1 to define the imbalance in a network, and we refer to it as $\boldsymbol{\psi}$ or $\boldsymbol{\psi}_{\text{EMD}}$.

Definition 2 (Imbalance Measure ψ_{EMD})**.** *Given a network $G = (V, E)$, two disjoint sets of nodes, namely source nodes $\emptyset \subset S \subset V$ and target nodes $\emptyset \subset T \subset V$, and the mapping cost of each pair of nodes $\boldsymbol{D} = [d_{ij}]$, we define two probability distributions $P_S = \{(i, w_i = \frac{1}{|S|}) \mid i \in S\}$ and $P_T = \{(j, w_j = \frac{1}{|T|}) \mid j \in T\}$. Using this, we define imbalance measure ψ or ψ_{EMD}:*

$$\psi_{\text{EMD}}(\boldsymbol{D}, S, T) = D_{\text{EMD}}(\boldsymbol{D}, P_S, P_T).$$

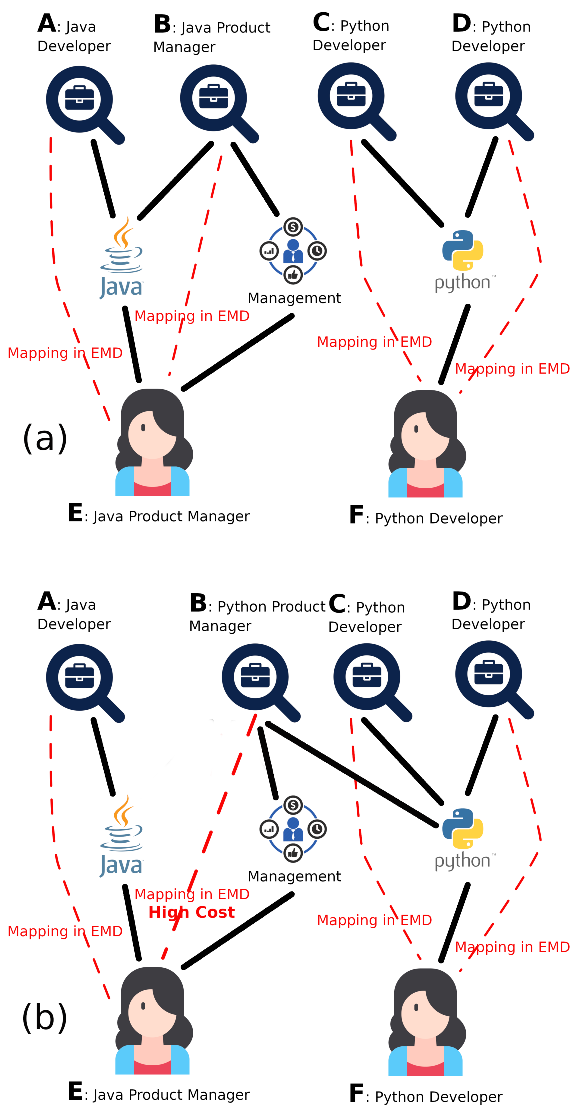


Fig. 1: A sample job market. A, B, C, and D are job positions and E and F are job seekers. E is a Java product manager and F is a Python developer. Solid lines are the links in the original graph. Dashed lines give the mapping that minimizes the EMD. (a) A and B are Java related job positions and C and D are Python related job positions. (b) A is a Java related job position and B, C, and D are Python related job positions. Note that in both cases B is about product management, whereas A, C, and D are developer positions. Since there are four job positions available and two people are seeking jobs in the example, the ideal case is (a) where there are two job positions with low mapping costs for each of the job seekers. On the other hand, in (b) there is an imbalance in the job market where one job seeker (F) can be mapped to two job positions with low costs (C and D), and the other job seeker (E) can be mapped to only one job position (A) with a low cost, while for one job (B) there is no ideal candidate available. Giving all nodes per type equal weight, the EMD maps each job seeker to a fixed number of job positions. The EMD for (b) is high, due to the high cost of mapping the Java product manager (E) to one of the Python related job positions (e.g., B). Hence, the EMD corresponds to the degree of imbalance in the job market.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

We can also define imbalance in terms of fractional perfect b-matching (see, e.g., [12]) which results in an equivalent measure. The details are described in Appendix A.

2.4 The mapping cost

The mapping cost can be freely chosen, depending on the application. Here we suggest a function based on link probabilities and provide an intuition for choosing this term.

A high link probability between two nodes means that they could be easily linked. Therefore, their mapping cost (in EMD) should be low and their mapping cost in computing the imbalance ψ should also be low. We propose using the minus log probability of links as the cost:

$$d_{ij} = -\ln(z_{ij}(\boldsymbol{A})). \tag{3}$$

In this case, the imbalance is given by $-\sum_{i\in S}\sum_{j\in T} f_{ij}\ln(z_{ij}) = -\sum_{i\in S}\sum_{j\in T}\ln(z_{ij}^{f_{ij}}) = -\ln\prod_{i\in S}\prod_{j\in T} z_{ij}^{f_{ij}}$.

The intuition behind choosing this term is as follows. In the case where $|S| = |T|$ (regardless of $\boldsymbol{D}$) the EMD has an optimum where each source node is mapped to exactly one target node and vice versa, i.e., $f_{ij} \in \{0,1\}$ in $\boldsymbol{F} = [f_{ij}]$. For this case, it is easy to see that computing the imbalance would be the same as computing a matching between source nodes and target nodes with the highest probability.

3 Problem Formulation

In this section, we formulate the problem of reducing imbalance by adding links to a network.

We assume that the costs of node-pair mappings (mapping costs in EMD) depend on the structure of the network, such that they can be influenced by modifying the network. Motivated by the job market example, we propose the operation of adding links as the kind of modification that can be made. We further propose to restrict the problem to add links between the nodes from the source set S to the auxiliary set of nodes U. This is again motivated by the job market, where we can realistically add new links between skills and job seekers (by training job seekers), but not between job vacancies and skills. We introduce the following problem:

Problem 1 (Imbalance Reduction Problem)**.** *Assume given a network $G = (V, E)$, two mutually disjoint sets of nodes source nodes $\emptyset \subset S \subset V$ and target nodes $\emptyset \subset T \subset V$, a set of auxiliary nodes $\emptyset \subset U \subset V$, a set of candidate non-links $L \subseteq S \times U$, $L \cap E = \emptyset$ connecting nodes from set S with nodes from set U, and imbalance measure $\psi(\boldsymbol{D}, S, T)$. We write $\boldsymbol{D}_{G'}$ to correspond to the mapping costs of $G' = (V, E \cup \mathcal{E})$. The Imbalance Reduction Problem is to find at most k links $\mathcal{E} \subseteq L$ that most reduce the imbalance between sets S and T:*

$$\underset{\mathcal{E}\subseteq L\subseteq S\times U, \mathcal{E}\cap E=\emptyset, 0\leq|\mathcal{E}|\leq k}{\underset{\mathcal{E}}{\mathit{argmin}}} \quad \psi(\boldsymbol{D}_{G'}, S, T).$$

In the rest of the paper, we simply refer to the candidate non-links L as the candidate set L.

4 Reducing Network Imbalance: GREASE

In this section, we introduce `GREASE`, a method to solve Problem 1, i.e., how to add at most k links to a network in order to maximally reduce the network imbalance. We first provide a sketch of the solution and then discuss more details in the respective sections below.

Problem 1 amounts to finding a set of links that jointly minimize the imbalance. An exact approach may be computationally expensive when the number of node-pairs in the candidate set is large, since it may require computing the imbalance reduction for every possible set of $k' \leq k$ links. Besides the vast number of possible sets, to compute the reduction in imbalance we need to recompute the mapping costs (e.g., costs based on the link probabilities; and hence recompute the machine learning model on which those are based), and recompute the imbalance. This process of recomputing could be computationally demanding depending on the model that provides the costs of node-pair mappings, such as network embedding methods, and could be practically impossible even for a moderate number of node-pairs in the candidate set.

To formally define the exact approach, we assume there is a given model M used to define the cost of node-pair mappings, based on the network structure. We assume that the mapping costs are computed by the parameters of model M, which are learned by optimizing an objective function based on a network. For example, network embedding methods [6], [8], [13], including CNE [11] are an example of a model on which the cost of node-pair mapping could be based. Section 4.3 below introduces CNE, the model used for the mapping costs in the experiments.

Let B indicate which node-pairs in L should be connected in the network to reduce the imbalance optimally in Problem 1: $B = \{b_{iu} \in \{0,1\} \mid \{i,u\} \in L\}$, i.e., a binary indicator per candidate link in L. Since the mapping costs are based on the network structure (via model M), they also depend on B. For convenience, we write $d_{ij}(B)$ defined as $d_{ij} : \{0,1\}^{|L|} \to \mathbb{R}$ to correspond to the mapping costs $\boldsymbol{D}_{G'}$ where $G' = (V, E \cup \mathcal{E})$ with $\mathcal{E} = \{\{i,u\} \mid b_{iu} = 1\}$. Hence, we want to solve the following program:

$$\begin{aligned}
\min_{\boldsymbol{F},B} \quad & \sum_{i\in S}\sum_{j\in T} f_{ij} d_{ij}(B), \\
\text{s.t.} \quad & f_{ij} \geq 0 \quad \forall (i,j) \in S \times T, \\
& \sum_{j\in T} f_{ij} = w_i \quad \forall i \in S, \\
& \sum_{i\in S} f_{ij} = w_j \quad \forall j \in T, \\
& \sum_{\{i,u\}\in L} b_{iu} \leq k, \\
& b_{iu} \in \{0,1\}, \quad \forall \{i,u\} \in L.
\end{aligned} \tag{4}$$

Model M computes the cost d_{ij} of mapping node i to node j based on the network structure. Here, the network structure is defined by the initial adjacency matrix and the links added to the network. The function d_{ij} is usually a function that depends on model parameters which themselves are learned based on the network structure. Since different values of B change the graph structure, for each

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

possible set of values of B, function d_{ij} has to be recomputed. The model M from which d_{ij} is computed may be the result of an optimization problem (based on the graph structure). We see an example of those optimization problems in Section 4.3. For these kinds of models, Eq. (4) is thus a mixed-integer non-linear bi-level program (MINLBLP).

Most approaches to solve MINLBLP problems require some assumptions such as the convexity of the objective function [14], bounds on variables [15], [16], and having polynomial objective functions [17], which might not be true for model M. Our approach is to simplify the MINLBLP in Eq. (4) and turn it to a mixed-integer quadratic program (MIQP), which has less complexity and easier to solve.

Rather than solving Eq. (4) exactly, we use some heuristics and assumptions to arrive at a scalable method. Specifically, we precompute the changes in the mapping costs (d_{ij}) for adding each link to the network, instead of computing it for each set B. Hence, Eq. (4) is turned into a MIQP, which is easier to solve. To fully explain our approach, we first describe the required properties of model M that enables the precomputation of the mapping costs after adding links to the network, then we introduce our method, and finally, we discuss our choice of model M.

4.1 Properties of the model for the mapping costs

We assume that model M has three properties. First, we assume that model M computes the mapping costs (based on link probabilities) between nodes i and j based only on the parameters (model internal parameters) related to nodes i and j. Some examples of such models are network embedding methods [6], [8], [11].

Second, we assume that adding a link between node i and j would mostly affect parameters related to nodes i and j in model M, an assumption that has been studied and used before for network embedding methods [18]. Hence, by adding a link between nodes i and j, we could estimate the new mapping costs in the network by only updating the parameters related to the two nodes i and j. We validated this property for the models used in the experiments, the results are given in Appendix B.

Third, we assume that adding links to high degree nodes would have little impact on the parameters related to it in model M. Since the auxiliary nodes usually have high degrees (we show that in the dataset statistics), this assumption allows us to only update the parameters related to node i after adding a link between source node i and auxiliary node j. We investigated this property for the models used in the experiments, the results are given in Appendix C.

The result of the properties is that adding a link between a source node i and auxiliary node u would mostly affect the parameters related to node i and hence, only the mapping costs of node i to other target nodes need to be updated. Based on these properties, d_{ij} only depends on the values in B that correspond to links connected to source node i, instead of all values in B.

It goes without saying that the last two properties are not perfectly satisfied in many models, but the changes in the parameters of other nodes may be limited. We expect that the results of GREASE are better if the properties described are satisfied more strongly. Hence, the use of the proposed approach is limited to the models that the properties described are satisfied. We investigate those properties for a model that we use in this paper in Section 4.3.

4.2 GREASE

In order to find at most k links from the candidate set L to add to the network to reduce the imbalance optimally, our approach is to precompute the mapping costs based on adding each individual node-pair in the candidate set to the network and include it in the imbalance quantification. To find the mapping costs after adding a link between source node i and auxiliary node u, we only update the parameters of node i in model M, in accordance with the properties in Section 4.1. Since we assume the cost of a node-pair mapping can be computed only based on the parameters of those two nodes, we only have to update the costs of mapping the source node i to all target nodes. The value d_{ij}^{iu} denotes the cost of mapping node i to j after adding a link between nodes i and u, which can be precomputed.

We further increase scalability by performing an incremental update of the parameters related to node i after adding a link between nodes i and u (and ignore the small changes in other parameters, as discussed in Section 4.1). As we only update the parameters related to node i, the mapping costs can be computed with almost no additional cost. We denote the cost of mapping the source node i to the target node j based on the initial adjacency matrix as d_{ij}. As a result, values d_{ij}^{iu} and d_{ij} appear as constants.

Note that if we would add multiple links to a source node i, we would have to precompute the mapping costs of node i to all target nodes for each possible set of $k' \leq k$ links in L that are connected to node i. Let $L_i \subseteq L$ denote the set of links in L that are connected to i. The number of computations is $\sum_{i \in S} \sum_{0 \leq k' \leq k} \binom{|L_i|}{k'}$, which is infeasible to compute. To avoid this problem, we add at most one link to each source node (this results in finding a sub-optimal solution). This approach results in a mixed-integer quadratic program (MIQP) to find at most k links to add to the network in order to reduce the imbalance optimally. The minimization MIQP is as follows:

$$
\begin{aligned}
\min_{\boldsymbol{F}, B} \quad & \sum_{i \in S} \sum_{j \in T} f_{ij} \left(\left(1 - \sum_{\{i,u\} \in L_i} b_{iu} \right) d_{ij} + \right. \\
& \qquad \qquad \left. \sum_{\{i,u\} \in L_i} b_{iu} d_{ij}^{iu} \right), \\
\text{s.t.} \quad & f_{ij} \geq 0 \quad \forall (i,j) \in S \times T, \\
& \sum_{j \in T} f_{ij} = w_i \quad \forall i \in S, & (5) \\
& \sum_{i \in S} f_{ij} = w_j \quad \forall j \in T, & (6) \\
& \sum_{\{i,u\} \in L_i} b_{iu} \leq 1 \quad \forall i \in S, \\
& \sum_{\{i,u\} \in L} b_{iu} \leq k, \\
& b_{iu} \in \{0,1\}, \quad \forall \{i,u\} \in L. & (7)
\end{aligned}
$$

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

B gives the links that should be added to the network to maximally reduce the imbalance. We call this method ***Graph imbalance REduction by Adding Sets of Edges (`GREASE`)***.

4.2.1 Speed up solving the MIQP

The MIQP in Eq. (7) has $|L|$ binary variables b_{ij} for each link $\{i,j\}$ in the candidate set L and $|S| \cdot |T|$ real variables for the mapping variables $\boldsymbol{F}$. In one of the datasets used in the experiments (VDAB), there are 21,794,110 real variables $\boldsymbol{F}$ in Eq. (7). Solving the MIQP with a large number of variables is time consuming. We propose an approach to reduce the number of variables, to improve the performance.

The mapping variables $\boldsymbol{F}$ are used to map the nodes in S to nodes in T. If mapping node $i \in S$ to node $j \in T$ has a high cost compared to others (i.e., d_{ij} is high), f_{ij} would probably be zero and not participate in the solution of EMD. Hence, for each node $i \in S$ (or $j \in T$), we only keep a r ratio of mapping variables f_{ij}, $j \in T$ (or $i \in S$) with lowest costs d_{ij}. The value e_{ij} shows if f_{ij} is removed or not (if e_{ij} is zero, f_{ij} is removed. Otherwise, e_{ij} is one). Removing some of the mapping variables can make Eq. (7) infeasible due to the constraints in Eq. (5) and Eq. (6). To solve this issue, we introduce artificial variables $Y = \{y_i : i \in S \cup T\}$ in constraints in Eq. (5) and Eq. (6), and we also add them to the objective function with a large coefficient Υ to force them to be as small as possible. The modified MIQP is:

$$\begin{aligned}
\min_{\boldsymbol{F},B,Y} \quad & \sum_{i\in S}\sum_{j\in T} e_{ij} f_{ij} \left(\left(1 - \sum_{\{i,u\}\in L_i} b_{iu} \right) d_{ij} + \right. \\
& \left. \sum_{\{i,u\}\in L_i} b_{iu} d_{ij}^{iu} \right) + \Upsilon \left(\sum_{i\in S} y_i + \sum_{j\in T} y_j \right), \\
\text{s.t.} \quad & f_{ij} \geq 0 \quad \forall (i,j) \in S \times T, \\
& y_i + \sum_{j\in T} e_{ij} f_{ij} = w_i, \quad y_i \geq 0, \quad \forall i \in S, \\
& y_j + \sum_{i\in S} e_{ij} f_{ij} = w_j, \quad y_j \geq 0, \quad \forall j \in T, \\
& \sum_{\{i,u\}\in L_i} b_{iu} \leq 1 \quad \forall i \in S, \\
& \sum_{\{i,u\}\in L} b_{iu} \leq k, \\
& b_{iu} \in \{0,1\}, \quad \forall \{i,u\} \in L. \qquad (8)
\end{aligned}$$

Using Eq. (8) might worsen the results compared to Eq. (7) in situations where node i will be mapped to node j despite relatively high mapping cost d_{ij}. In this case, maybe adding a link to the source node i would reduce the mapping cost d_{ij}, which will not be possible if variable f_{ij} is removed. We investigate the effect of ratio r on the imbalance reduction in Section 5.4.3.

4.3 Choice of the model providing the mapping costs

While mapping costs based on link probabilities could be computed in several ways, network embeddings arguably offer a natural way to define them by finding an embedding of the network. Network embedding methods encode each node $i \in V$ as a d-dimensional real vector $\boldsymbol{x}_i \in \mathbb{R}^d$. For convenience, the embeddings may be aggregated in a matrix $\boldsymbol{X} = (\boldsymbol{x}_1, ..., \boldsymbol{x}_n)' \in \mathbb{R}^{n \times d}$.

Our approach is to use Conditional Network Embedding (CNE; [11]) as the model to provide the link probabilities and thus the mapping costs. We use CNE for four reasons. First, CNE computes the link probabilities between two nodes only based on the embeddings of the two nodes (first property). Second, computing partial embedding after adding a link to the network is possible in CNE, as we can only update the embedding of the node of interest instead of the whole network (second property). We also investigate the how much the second and third properties hold in CNE in the experiments in Appendix B and Appendix C, respectively. Third, re-embedding the network starting from an initial embedding is easily done with CNE, greatly speeding up our proposed method. And fourth, CNE was shown to have state of the art performance on downstream tasks [11].

The objective of CNE is to find the embedding $\boldsymbol{X}$ that maximizes the likelihood of observing graph G. The optimization task is formulated as a Maximum Likelihood Estimation (MLE) problem, $argmax_{\boldsymbol{x}} P(G \mid \boldsymbol{X})$ (Eq. (6) in [11]). The optimal embeddings are computed using a block stochastic gradient descent approach. The gradient of the likelihood function with respect to the embeddings of any node i could also be computed for that purpose. We then only update the embedding of a particular node i as the partial embedding approach.

In summary, the link probability between any two nodes i and j in CNE depends only on the embeddings of nodes i and j. Moreover, we can update the embeddings of only specific nodes in CNE after adding a link to the network. In Section C, we show that the third property in Section 4.1 (which is about the embedding stability of high degree nodes) is also true to some extent for CNE. Hence, CNE is a suitable model for our approach.

5 Experimental Evaluation

In this section, we describe the experimental evaluation of `GREASE`. We investigate four research questions **Q1**: How does `GREASE` perform in reducing the imbalance ψ compared to the baselines? **Q2**: How does `GREASE` perform in terms of execution time compared to the baselines? **Q3**: How sensitive is `GREASE` to the changes in mapping variables ratio r? **Q4**: How does `GREASE` perform in reducing network imbalance with a reduced solver time limit?

We also investigate two other research questions in the appendix. **Q5**: How the embeddings of nodes change after adding a link to the network, based on their distance (in the graph, i.e., the number of links in a shortest path connecting them) from the nodes of the added link (Appendix B)? **Q6**: How stable are the embeddings of high degree nodes compared to others after adding links to them (Appendix C)?

We first discuss the datasets, baselines, and settings. Next, we present the result of each experiment. The source code for repeating the experiments and the preprocessed public datasets are available here [1].

1. https://github.com/aida-ugent/GREASE

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

5.1 Datasets

We evaluated the methods using the following datasets:

VDAB [2]: VDAB is the employment service of Flanders in Belgium. It provides a platform for job seekers to find jobs. The dataset contains a suitably anonymized sample of the applications made by job seekers to available job vacancies from April 2020 until October 2020. We construct the job market network with three sets of nodes: job seekers, job vacancies, and skills. Job vacancies are connected to job seekers that have applied for them and to the skills they require. Our goal is to reduce the imbalance between job seekers (source nodes) and job vacancies (target nodes) by adding links connecting job seekers with skills. This could be seen as teaching a group of job seekers some skills, in a way that balances the job market network.

Twitter [3]: Twitter is one of the most popular microblogging services in the world. The dataset contains the social graph and tweets of Manchester United and Manchester City fans between January 1 and September 22 in 2021. We construct the network with three sets of nodes: Manchester United, Manchester City, and hashtags. Users are connected to their friends (reciprocal follow relationships) and the hashtags of their tweets. We crawled the followers of fan club pages on Twitter to find the fans of each club. We also removed the hashtags related to the clubs. Our goal is to reduce the imbalance between United fans (source nodes or target nodes) and City fans (target nodes or source nodes) by adding new links connecting users with hashtags (auxiliary nodes). It is more like recommending tweets with specific hashtags to the users to increase their interest in that hashtag. In the experiments, we call this dataset **Twitter-UC** if United fans are the source nodes and City fans are the target nodes. Otherwise, we call the dataset **Twitter-CU**.

Weibo [19]: Weibo is the most popular Chinese microblogging service. The dataset contains tweets of the users and the topic distribution of each tweet. We construct the network with three sets of nodes: male users, female users, and topics. Users are connected to their friends (reciprocal follow relationships) and the top topics of their tweets. To find the top topics for each tweet, we first sort the topic probabilities (relevance of the topic for the tweet) in descending order. Next, we select the top topics until the difference between the probabilities of two consecutive topics is greater than a very small threshold (1e-6). We select a sample from the dataset by only considering tweets for the first year of the data. Our goal is to reduce the imbalance between males (source nodes or target nodes) and females (target nodes or source nodes) by adding new links connecting males/females with topics (auxiliary nodes). It is more like recommending tweets with specific topics to the users to increase their interest in that topic. In the experiments, we call this dataset **Weibo-mf** if males are the source nodes and females are the target nodes. Otherwise, we call the dataset **Weibo-fm**.

Movielens [20]: MovieLens is a web-based movie recommender system. The dataset contains 100000 user ratings on movies. We construct the network with three sets of nodes: users, movies, and movie genres. There is a link between each user and movie for each rating. We also connect each movie to its genres. Our goal is to reduce the imbalance between movies (source nodes or target nodes) and users (target nodes or source nodes) by adding new links connecting movies with genres (auxiliary nodes).

Weibo2 [19]: Weibo2 is the same as Weibo dataset, except that the auxiliary nodes are connected to each other if they have at least three common neighbors. Identical to the Weibo dataset, we evaluate the methods on **Weibo2-mf** and **Weibo2-fm** datasets depending on whether males or females are considered as the source nodes.

We only used the largest connected component in each network. Table 1 shows the main statistics of each of the networks. The average degrees of auxiliary nodes are higher than the whole network average degrees, which supports the required properties of the model providing the mapping costs in Section 4.1.

5.2 Baselines

As mentioned earlier, we are the first to introduce the concept of imbalance in a network in the way described in Section 2.2 and to reduce it by adding links to the network. However, there exist other methods that try to add links to the networks to make them more cohesive. We consider two of those methods [21], [22] for comparison.

The work in [21] minimizes the average shortest path in a network by adding links. the authors in [22] compute controversy between two sets of nodes using the random walks starting from one set, and ending in the same or the other set. The main difference between the imbalance and controversy is that the amount of links between nodes in the same set has a major effect on the controversy, which is not necessarily the case in computing the imbalance. Moreover, we compute the costs of node-pair mappings based on the link probabilities, while they consider the actual links in the network to compute the controversy.

We also designed a random method that ensures a reduction in the imbalance after adding links and a simple random method for comparison, see below.

In summary, the following methods will be evaluated as baselines:

`GraB` [10]: 'Graph Balancing' is a method to reduce the imbalance in a network by adding links to the network. It estimates if adding a link would reduce the imbalance locally for a node, and proposes a greedy algorithm to consider the imbalance reduction globally. It also adds links in batches to increase the robustness of the recommended links. `GraB` selects b links connecting source nodes with auxiliary nodes for each batch based on the local imbalance reduction estimation. To select k links to add to the network, `GraB` runs $\frac{k}{b}$ iterations. It also filters the results in a post-hoc filtering phase to ensure the selected links improve the local imbalance measure.

ROV [22]: 'Recommend opposing view' adds links to the network to reduce controversy. In this work, k links between high degree nodes in sets S and T that reduce controversy the most, are added using a greedy algorithm. We adopt this method for our problem setting by adding links between sets S and U using the same method.

SSW [21]: 'Shortcuts for a smaller world' adds links to the network to reduce the average shortest path length. In

2. https://www.vdab.be/
3. https://www.twitter.com/

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

TABLE 1: Main statistics of the networks used for evaluation.

Dataset	VDAB	Twitter-UC	Twitter-CU	Weibo-mf	Weibo-fm	Movielens	Weibo2-mf	Weibo2-fm
Nodes	9539	6751	6751	1364	1364	2644	1364	1364
Source nodes	4531	4939	1724	822	442	1682	822	442
Target nodes	4810	1724	4939	442	822	943	442	822
Auxiliary nodes	198	88	88	100	100	19	100	100
Links	48451	220498	220498	14308	14308	102893	14873	14873
Average degree	12	65	65	20	20	77	21	21
Auxiliary nodes average degree	154	138	138	34	34	152	39	39

this work, k links are added to the network using different strategies. We employ the greedy strategy since it has the best performance [21].

S-Random: The 'simple random' baseline selects k random links connecting source nodes with auxiliary nodes.

G-Random: The 'greedy random' baseline selects a link at each iteration, adds it to the network, re-embeds the network, and computes the network imbalance. If the imbalance measure is reduced, the link is added to the result set. It continues the iterations for a given amount of time.

5.3 Experimental settings

We compare methods in terms of the imbalance ψ (Definition 2) or imbalance reduction $-\Delta\psi$. We conduct the experiments on CNE with dimensions 2 and 4 with a combination of block and degree prior (see [11]). The names of the datasets are combined with d2 or d4, which show the embedding dimensions.

For the experiments, we design a strategy for the candidate set of links L to be added to the network. Adding a link to a source node means that the source node has to form a connection with an auxiliary node. Hence, the candidate set should have a high link probability, to enhance forming the connection in real life. E.g., a job seeker could acquire skills with less effort if it has already high link probabilities with them. Hence, we consider the top n skills with the highest link probabilities for each source node. We call this strategy as *top n links per source node (TnLS)*. In the experiments, we set n to one (T1LS) and five (T5LS).

In the experiments, the methods are evaluated on several datasets. We set k to 5% of the size of the source nodes for all experiments. G-Random and `GREASE` can add up to k links (zero to k links) to each dataset. However, other methods do not allow adding flexible number of links to the networks. Hence, other baselines add exactly k links to the networks.

We solve the MIQP in Eq. (8) using Gurobi v9.1 in `GREASE`. We limit the time of solving the MIQP to 10 minutes using Gurobi in all experiments. We also tune r from values $\{0.2, 0.4, 1.0\}$ for experiment in Section 5.4.1.

In the experiment in Section 5.4.1, we use the author's implementation of computing the controversy in a network between two sets and used their default hyper-parameters, which are used in ROV. We used 10% of the high degree source nodes and 20 high degree auxiliary nodes for the candidate selection in ROV. We also use the author's implementation of SSW. SSW does not require setting any particular hyper-parameter. For I-Random, we limit the execution time to one hour. Moreover, we average the results for S-Random and G-Random over 3 repetitions to smoothen out random fluctuations. For `GraB`, we added links in one batch and also in several batches of 25 links in Movielens and Weibo datasets, and added links in one batch for VDAB and Twitter datasets, since adding links in several batches resulted in high execution time.

5.4 Experimental results

In this section, we present the results of the experiments to investigate the research questions **Q1**, **Q2**, **Q3**, **Q4**, and **Q5**.

5.4.1 Baseline comparison

Here we compare the methods in terms of ψ on all datasets. We report ψ on the main graph as well (**Q1**). Table 2 and Table 3 show the result of this experiment with the candidate set strategies T1LS and T5LS respectively.

`GREASE` reduces ψ over the main graph and outperforms the other methods in most of the datasets. Moreover, `GREASE` performs better in reducing ψ for T1LS than T5LS compared to the baselines. It also performs better for T1LS than T5LS in some datasets compared to itself, although the candidate set in T5LS includes the node-pairs in T1LS. This is due to two reasons. First, the MIQP in T5LS is bigger than in T1LS. Since we limited the amount of time to solve the MIQP in Eq. (8), it is possible `GREASE` performs worse for T5LS with the same amount of solver time limit. Second, despite all baselines, `GREASE` only adds at most one link to each source node. Hence, in T5LS, the baselines have the opportunity to add several links to a source node that could reduce the imbalance a great amount, which is not possible in `GREASE`.

`GraB` and SSW also outperform the other baselines in most of the datasets. On the other hand, G-Random reduces the imbalance by a small amount. It even performs worse than S-Random in some of the datasets. The reason is that G-Random computes the embedding for each link addition, and only adds the link if the imbalance reduces. Since computing the embedding is time consuming itself, G-Random does not add many links in one hour, which is the maximum time we set for it to select the links.

The other methods S-Random and ROV do not perform particularly well (in some datasets they increase the imbalance, and the amount of imbalance reduction in other datasets is less than `GREASE`), since they are random or designed for a different purpose and objective function. We did not report the result of ROV on VDAB and Twitter datasets because it did not finish in a reasonable amount of time (several hours).

Moreover, we also analyze the number of links that are added to the datasets. The baselines add k links (the maximum number of links) to each dataset (except for G-Random, which adds up to k links to the datasets). Although

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

TABLE 2: ψ after adding links to each network for candidate set strategy T1LS. The best performance per dataset is highlighted in bold.

Dataset	Main Graph	S-Random	G-Random	SSW	ROV	GraB	GREASE
VDAB-d2	6.2858	6.2859	6.2858	6.2857		6.2851	**6.2846**
Twitter-UC-d2	15.5578	15.5489	15.5576	15.5207		15.5179	**15.4917**
Twitter-CU-d2	15.5578	15.5099	15.5572	15.4328		15.4441	**15.4173**
Weibo-mf-d2	10.4774	10.4768	10.4756	10.4774	10.4775	10.4724	**10.4708**
Weibo-fm-d2	10.4774	10.4767	10.4747	10.4752	10.4796	10.4704	**10.4677**
Movielens-d2	3.7369	3.7363	3.7363	3.7355	3.7368	**3.7297**	3.7313
Weibo2-mf-d2	10.4788	10.4778	10.47566	0.4788	10.4780	10.4725	**10.4712**
Weibo2-fm-d2	10.4788	10.4771	10.4753	10.4733	10.4788	10.4714	**10.4671**
VDAB-d4	6.3096	6.3097	6.3096	6.3092		6.3089	**6.3082**
Twitter-UC-d4	15.4756	15.4586	15.4754	15.4193		15.4393	**15.3856**
Twitter-CU-d4	15.4756	15.4344	15.4749	15.3459		15.3669	**15.3261**
Weibo-mf-d4	10.4959	10.4958	10.4945	10.4944	10.4964	10.4931	**10.4910**
Weibo-fm-d4	10.4959	10.4953	10.4941	10.4934	10.4963	10.4920	**10.4867**
Movielens-d4	3.8063	3.8047	3.8060	3.8038	0.8067	3.8047	**3.8022**
Weibo2-mf-d4	10.4966	10.4966	10.4948	10.4967	10.4965	10.4920	**10.4912**
Weibo2-fm-d4	10.4966	10.4959	10.4938	10.4923	10.4964	10.4924	**10.4902**

TABLE 3: ψ after adding links to each network for candidate set strategy T5LS. The best performance per dataset is highlighted in bold.

Dataset	Main Graph	S-Random	G-Random	SSW	ROV	GraB	GREASE
VDAB-d2	6.2858	6.2860	6.2858	6.2866		6.2851	**6.2846**
Twitter-UC-d2	15.5578	15.5421	15.5575	15.4986		15.4967	**15.4939**
Twitter-CU-d2	15.5578	15.5184	15.5560	15.4403		**15.4248**	15.4291
Weibo-mf-d2	10.4774	10.4775	10.4748	10.4770	10.4782	10.4696	**10.4688**
Weibo-fm-d2	10.4774	10.4771	10.4746	10.4794	10.4771	10.4633	**10.4624**
Movielens-d2	3.7369	3.7364	3.7366	3.7330	3.7373	**3.7251**	3.7294
Weibo2-mf-d2	10.4788	10.4787	10.4763	10.4790	10.4775	10.4714	**10.4703**
Weibo2-fm-d2	10.4788	10.4767	10.4752	10.4707	10.4787	10.4669	**10.4633**
VDAB-d4	6.3096	6.3099	6.3096	6.3100		6.3089	**6.3082**
Twitter-UC-d4	15.4756	15.4623	15.4756	15.3865		15.4333	**15.3817**
Twitter-CU-d4	15.4756	15.4395	15.4750	15.3563		15.3625	**15.3525**
Weibo-mf-d4	10.4959	10.4966	10.4943	10.4956	10.4974	10.4927	**10.4879**
Weibo-fm-d4	10.4959	10.4950	10.4943	10.4922	10.4962	10.4922	**10.4839**
Movielens-d4	3.8063	3.8048	3.8060	3.8011	3.8064	**3.7987**	3.8011
Weibo2-mf-d4	10.4966	10.4966	10.4941	10.4977	10.4963	10.4926	**10.4911**
Weibo2-fm-d4	10.4966	10.4960	10.4940	10.4919	10.4964	10.4845	**10.4830**

GREASE allows for adding zero to k links, in practice it adds the maximum number of links to all datasets in our experiments. Hence, all methods (except for G-Random) add the same number of links to the datasets.

5.4.2 *Execution time comparison*

In this experiment, we compare the methods in terms of the execution time (**Q2**) with the same settings as experiment 5.4.1. Figure 2a and 2b show the execution time in seconds (log-scale) for all methods, including the time for hyper-parameter tuning for candidate set strategy T1LS and T5LS respectively.

GREASE, GraB, ROV, and G-Random have the highest execution time among all methods.

GraB has a high execution time due to the number of hyper-parameters to be tuned, the link selection step, and also re-embedding after adding each batch. ROV has a high execution time due to the time needed for computing the controversy after adding each candidate link to the graph. We did not report ROV execution time for VDAB and Twitter datasets because they did not finish in a reasonable amount of time (several hours). We set one hour as the maximum execution time for G-Random.

Although GREASE has a high execution time, it still is faster than some baselines (faster than ROV in all datasets, and lower or equal execution time compared to G-Random and GraB in some of the datasets). We set the maximum time for the solver, Gurobi, to 10 minutes in GREASE. The total execution time also includes some pre-processing time to design the MIQP in Eq. (8). We also tune the mapping variables ratio r from 3 values.

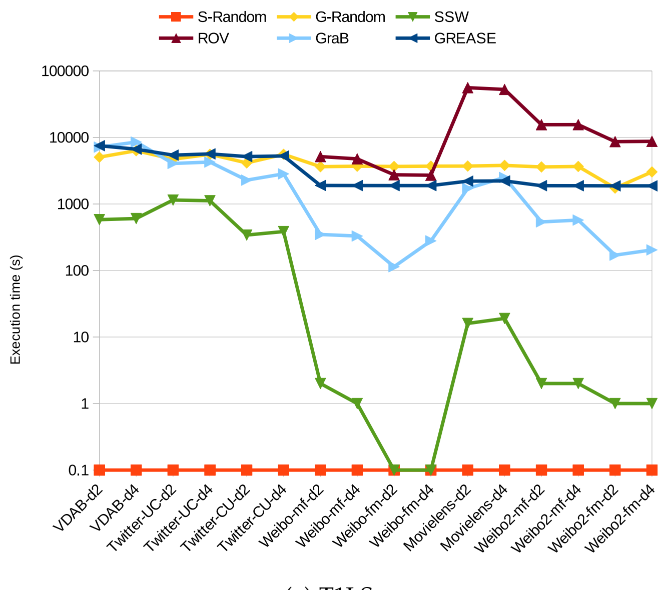
5.4.3 *Sensitivity to the mapping variables ratio r*

In this section, we test if removing some of the mapping variables in Section 4.2.1 improves the performance in reducing the imbalance in networks. We evaluate the sensitivity of GREASE to the mapping variables ratio r (**Q3**) in terms of the amount of imbalance reduction, $-\Delta\psi$. Higher values of $-\Delta\psi$ mean a greater reduction in the imbalance. To investigate **Q3**, we let Gurobi run for 10 minutes for each dataset. Figure 3 shows $-\Delta\psi$ against mapping variables ratio r after adding links to each dataset.

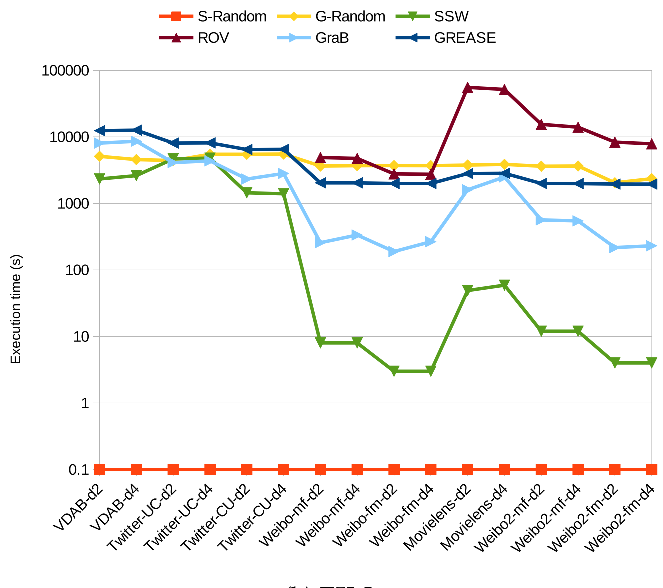
In general, it appears that very small values of r hinder the optimization, except for the large VDAB network. The reason is that for large datasets, the MIQP in Eq. (8) is large and complex, and difficult to solve. Hence, in the limited

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478



(a) T1LS



(b) T5LS

Fig. 2: Execution time (log-scale) of methods for experiment 5.4.1 on each evaluated network for candidate set strategies T1LS and T5LS. Since the y-axis is in log-scale, 0.1 is used to represent all values between zero and 0.1.

amount of time for the solver, reducing the number of mapping variables would make it simpler and easier to solve. On the other hand, we observe that for the other datasets values 0.1 and 0.2 of r result in keeping only a small amount of mapping variables and hence, a poor performance. The results are mostly consistent across different dimensions and different candidate set strategies.

Since the imbalance reduction in `GREASE` improves in some datasets with a value of r other than one, we conclude that the technique described in Section 4.2.1 is effective in some situations.

5.4.4 *Sensitivity to solving time*

In this section, we analyze the sensitivity of `GREASE` to the solver time limit (**Q4**) in terms of the amount of imbalance reduction, $-\Delta\psi$. Higher values of $-\Delta\psi$ mean a greater reduction in the imbalance.

To investigate **Q4**, we let Gurobi run for 10 minutes for each dataset and evaluate the results after every minute. Figure 4 shows $-\Delta\psi$ against solving time after adding links to each dataset. Here we only include the result for the mapping variables ratio r with the highest imbalance reduction after 10 minutes. The complete results of this experiment for different values of mapping variables ratio r are presented in Appendix D.

The amount of reduction in imbalance $-\Delta\psi$ generally improves by increasing the solving time, as expected. However, in some cases increasing the solving time worsens the performance in reducing the imbalance. This would be due to the estimation of computing the parameters in the model after adding a link (partial embedding in CNE) and reducing the number of mapping variables. Both techniques in computing the new mapping costs with partial embedding and also reducing mapping variables with high costs sometimes have errors that could cause inconsistent results. The results are mostly consistent across different dimensions and different candidate set strategies.

6 Related Work

Imbalance in the workforce has been studied in various systems [23], [24]. However, our work differs from these studies. Previous studies are mostly domain-specific and they analyze the supply and demand based on domain-specific features such as educational training program length, retirement, and salary. Previous studies in this area also lack a global measure to quantify the imbalance between two different entities. In contrast to this, we tackle the problem from a graph analysis approach. We propose a quantification of the imbalance in the network and a method to reduce the imbalance by adding links to the network.

Another line of research focuses on matching problems [25], [26]. Finding the cost of the fractional perfect b-matching [12] with minimum cost in bipartite networks is related to our work (Appendix A). In this problem, the goal is to find a matching between two sets of nodes in a network with minimum total cost. We define the imbalance as the cost of the minimum cost fractional perfect b-matching on a new bipartite network created from the original network (see Appendix A for more details). Our work differs from the studies focusing on matching since we do not address the computational problem of how to find the matching. We only use the cost of the matching to quantify the imbalance. Moreover, we add links to the network (not directly between the two sets of nodes of interest), to change the cost of links between the two sets of nodes, and hence, reduce the imbalance in the network.

Fairness in node embeddings is studied in various research papers [27], [28]. These studies try to learn an unbiased network embedding. The similarity between learning an unbiased embedding and reducing the imbalance in a network embedding is that they both try to have a mix between nodes with different attributes or different types

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

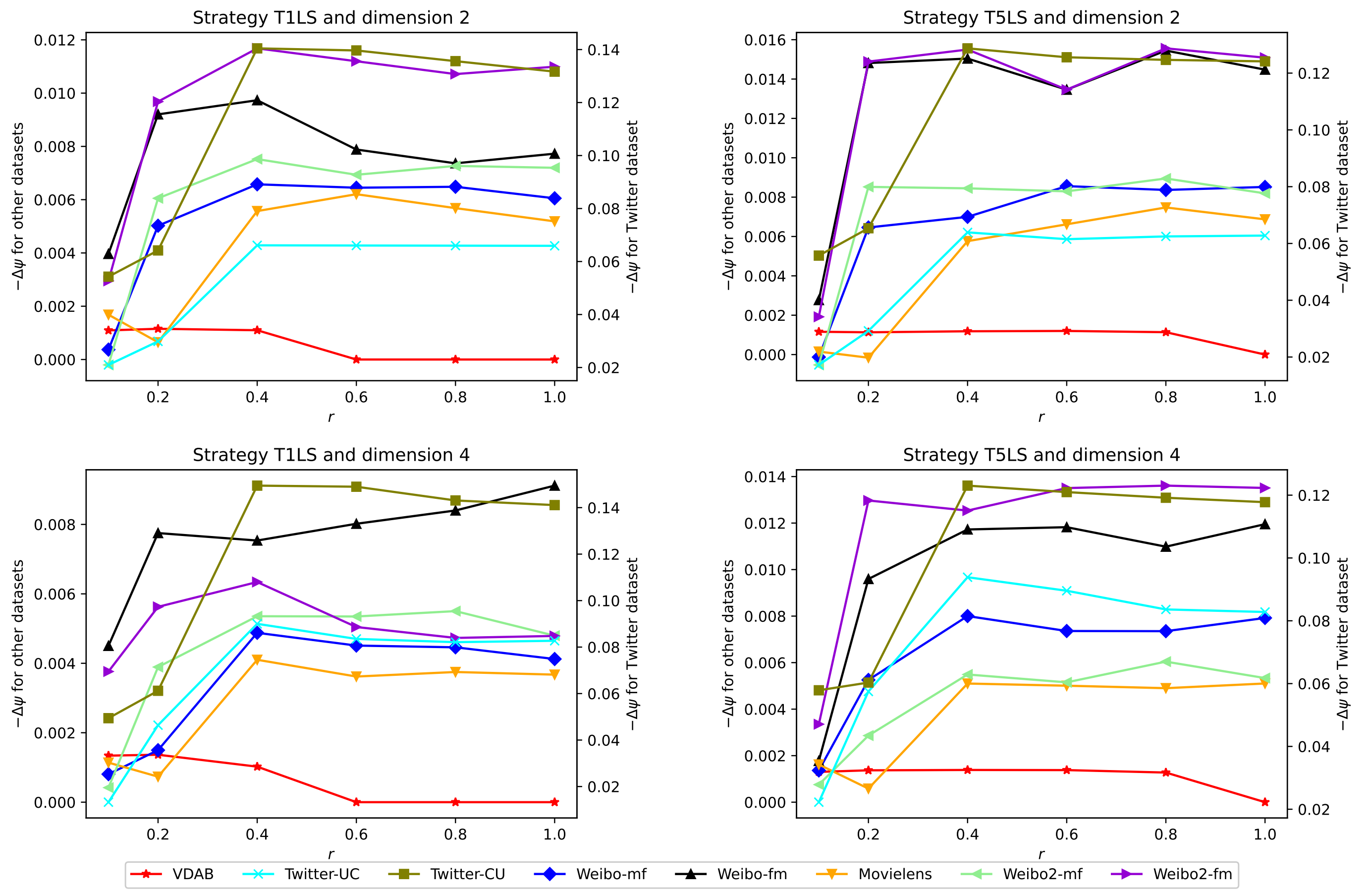


Fig. 3: Amount of imbalance reduction $-\Delta\psi$ against matching variables ratio r for dimensions 2 and 4 and candidate link strategy T1LS and T5LS. Note a different scale is used for the Twitter datasets, hence there is a double y-axis.

(job seeker and job vacancy, female and male, etc.). The main difference is that debiasing methods learn an unbiased embedding based on the original network, while we modify the network in order to make it more balanced. A second difference is in the quantification of the imbalance or unfairness. In our setting, we intend to bring two sets of nodes closer, to reduce the imbalance, while the goal to reduce unfairness is that the two sets of nodes cannot be separated, which is not important in our case.

There are also several papers aiming to add links to a network to modify its structure. Some of the research focuses on adding links to a network to make it more cohesive, where cohesiveness is quantified using network properties such as shortest paths [21], [29], diameter [30], information unfairness [31], controversy [22] and structural bias [32]. The work in [22] is most related to our work since they add links to the network to reduce the controversy between two sets of nodes. However, the difference between our works is that we consider a different measure to compute the imbalance and use a different approach to optimize it.

7 Conclusions and Future Work

We defined and quantified the concept of imbalance between two sets of nodes in a network, and introduced the novel problem of reducing that imbalance by introducing new links. We proposed `GREASE` to tackle this problem by solving a mixed-integer quadratic program. To ensure the efficiency of solving this problem, several reasonable assumptions as well as effective heuristics were made. We presented experiments applying `GREASE` together with CNE as the network embedding method to various networks. The experimental results indicate that `GREASE` outperforms baselines for reducing imbalance in a network embedding.

In future work, we plan to investigate the benefits of a new link for individual nodes (e.g., improving the access to a target set of nodes) instead of just for the global balance of the network, as well as other problem settings such as reducing the imbalance in a network by *removing* a specific number of links from the network (e.g., changing job contents and required skills, making jobs more accessible), or both adding and removing links at the same time.

Moreover, other types of networks, such as attributed networks, can be investigated. More specifically, in order to know whether `GREASE` could be used on such networks, it has to be investigated to what degree link probability models (such as network embedding models) for attributed networks satisfy the requirements for `GREASE`.

Acknowledgments

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

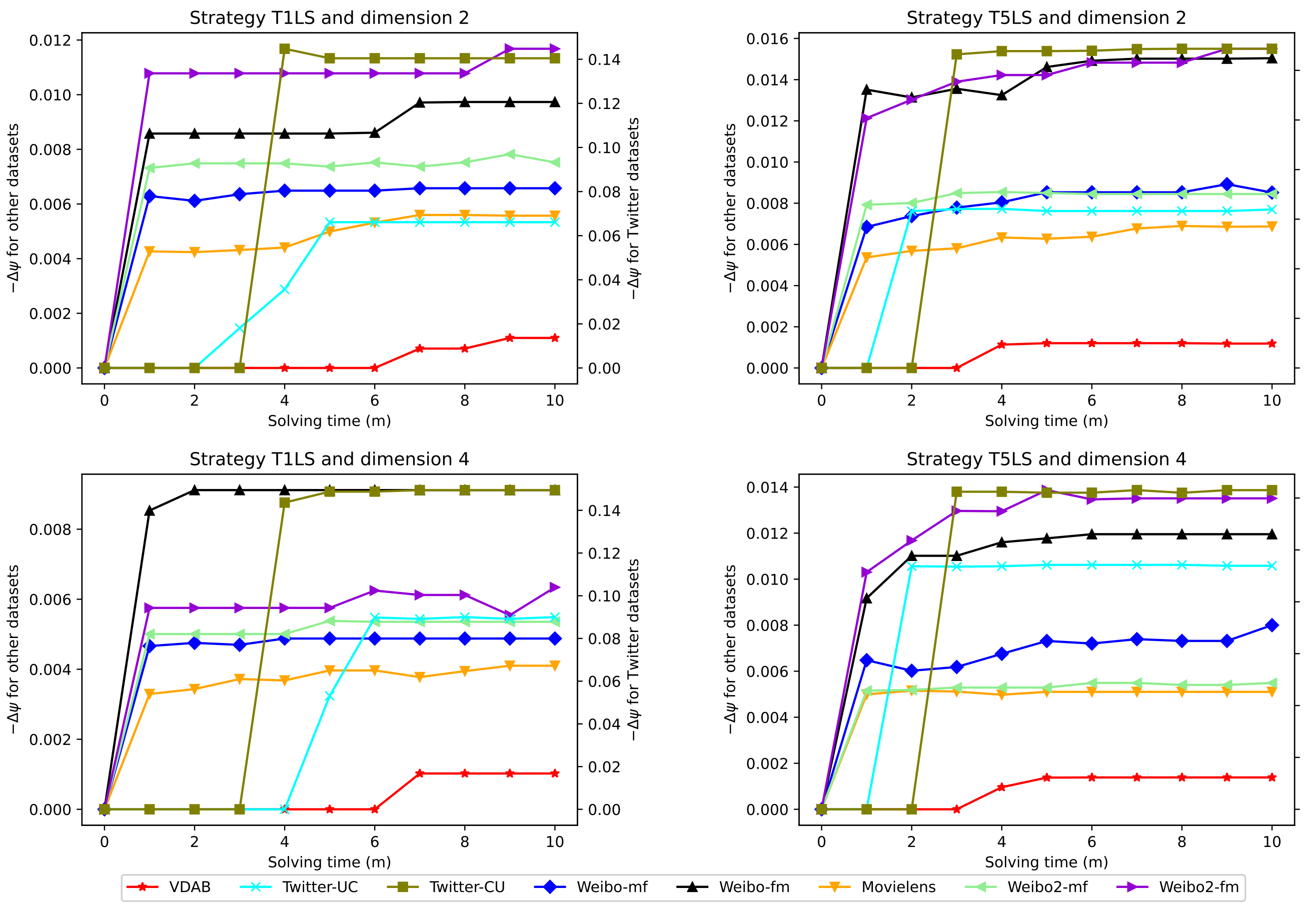


Fig. 4: Amount of imbalance reduction $-\Delta\psi$ against solving time for mapping variables ratio r with the highest imbalance reduction r for dimensions 2 and 4 and candidate link strategy T1LS and T5LS. Note a different scale is used for the Twitter datasets, hence there is a double y-axis.

(ERC Grant Agreement no. 615517), and under the European Union's Horizon 2020 research and innovation programme (ERC Grant Agreement no. 963924), from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme, and from the FWO (project no. G091017N, G0F9816N, 3G042220). Part of the experiments were conducted on pseudonimized HR data generously provided by VDAB.

References

[1] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1616–1637, 2018.

[2] L. C. Freeman, "Visualizing social networks," *Journal of social structure*, vol. 1, no. 1, p. 4, 2000.

[3] A. Theocharidis, S. Van Dongen, A. J. Enright, and T. C. Freeman, "Network visualization and analysis of gene expression data using biolayout express 3d," *Nature protocols*, vol. 4, no. 10, p. 1535, 2009.

[4] R. F. I. Cancho and R. V. Solé, "The small world of human language," *Proceedings of the Royal Society of London. Series B: Biological Sciences*, vol. 268, no. 1482, pp. 2261–2265, 2001.

[5] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge-Based Systems*, vol. 151, pp. 78–94, 2018.

[6] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.

[7] S. Gao, H. Pang, P. Gallinari, J. Guo, and N. Kato, "A novel embedding method for information diffusion prediction in social network big data," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 2097–2105, 2017.

[8] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th international conference on world wide web*, 2015, pp. 1067–1077.

[9] Y. Rubner, C. Tomasi, and L. J. Guibas, "A metric for distributions with applications to image databases," in *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*. IEEE, 1998, pp. 59–66.

[10] Y. Mashayekhi, B. Kang, J. Lijffijt, and T. De Bie, "Quantifying and reducing imbalance in networks," in *RECSYS IN HR 2021*, vol. 2967. CEUR, 2021.

[11] B. Kang, J. Lijffijt, and T. De Bie, "Conditional network embeddings," in *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[12] R. E. Behrend, "Fractional perfect b-matching polytopes i: General theory," *Linear Algebra and its Applications*, vol. 439, no. 12, pp. 3822–3858, 2013.

[13] P. Jiao, X. Guo, X. Jing, D. He, H. Wu, S. Pan, M. Gong, and W. Wang, "Temporal network embedding for link prediction via VAE joint attention mechanism," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 12, pp. 7400–7413, 2021.

[14] T. A. Edmunds and J. F. Bard, "An algorithm for the mixed-integer nonlinear bilevel programming problem," *Annals of Operations Research*, vol. 34, no. 1, pp. 149–162, 1992.

[15] P.-M. Kleniati and C. S. Adjiman, "A generalization of the branch-and-sandwich algorithm: from continuous to mixed-integer nonlinear bilevel problems," *Computers & Chemical Engineering*, vol. 72, pp. 373–386, 2015.

[16] A. Mitsos, "Global solution of nonlinear mixed-integer bilevel

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2023.3304478

programs," *Journal of Global Optimization*, vol. 47, no. 4, pp. 557–582, 2010.

[17] L. F. Domínguez and E. N. Pistikopoulos, "Multiparametric programming based algorithms for pure integer and mixed-integer bilevel programming problems," *Computers & Chemical Engineering*, vol. 34, no. 12, pp. 2097–2106, 2010.

[18] Kang, Bo and Lijffijt, Jefrey and De Bie, Tijl, "Explanations for network embedding-based link predictions," in *MACHINE LEARNING AND PRINCIPLES AND PRACTICE OF KNOWLEDGE DISCOVERY IN DATABASES, ECML PKDD 2021, PT I*, vol. 1524, 2021, pp. 473–488. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-93736-2_36

[19] J. Zhang, B. Liu, J. Tang, T. Chen, and J. Li, "Social influence locality for modeling retweeting behaviors," in *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[20] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *Acm transactions on interactive intelligent systems (tiis)*, vol. 5, no. 4, pp. 1–19, 2015.

[21] N. Parotsidis, E. Pitoura, and P. Tsaparas, "Selecting shortcuts for a smaller world," in *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM, 2015, pp. 28–36.

[22] K. Garimella, G. De Francisci Morales, A. Gionis, and M. Mathioudakis, "Reducing controversy by connecting opposing views," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 2017, pp. 81–90.

[23] P. Zurn, M. R. Dal Poz, B. Stilwell, and O. Adams, "Imbalance in the health workforce," *Human resources for health*, vol. 2, no. 1, pp. 1–12, 2004.

[24] G. Willis, A. Woodward, and S. Cave, "Robust workforce planning for the english medical workforce," in *Conference Proceedings, The 31st International Conference of the System Dynamics Society*, 2013.

[25] V. Kolmogorov, "Blossom v: a new implementation of a minimum cost perfect matching algorithm," *Mathematical Programming Computation*, vol. 1, no. 1, pp. 43–67, 2009.

[26] L. Ramshaw and R. E. Tarjan, "On minimum-cost assignments in unbalanced bipartite graphs," *HP Labs, Palo Alto, CA, USA, Tech. Rep. HPL-2012-40R1*, 2012.

[27] M. Buyl and T. De Bie, "Debayes: a bayesian method for debiasing network embeddings," in *Proceedings of the 37th International Conference on Machine Learning*, 2020, pp. 1220–1229.

[28] A. Bose and W. Hamilton, "Compositional fairness constraints for graph embeddings," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 715–724.

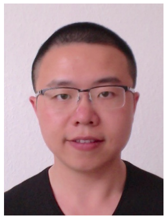
[29] M. Papagelis, F. Bonchi, and A. Gionis, "Suggesting ghost edges for a smaller world," in *Proceedings of the 20th ACM international conference on Information and knowledge management*, 2011, pp. 2305–2308.

[30] E. D. Demaine and M. Zadimoghaddam, "Minimizing the diameter of a network using shortcut edges," in *Scandinavian Workshop on Algorithm Theory*. Springer, 2010, pp. 420–431.

[31] Z. S. Jalali, W. Wang, M. Kim, H. Raghavan, and S. Soundarajan, "On the information unfairness of social networks," in *Proceedings of the 2020 SIAM International Conference on Data Mining*. SIAM, 2020, pp. 613–521.

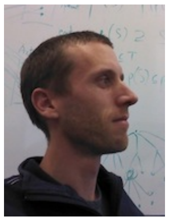
[32] S. Haddadan, C. Menghini, M. Riondato, and E. Upfal, "Repbublik: Reducing polarized bubble radius with link insertions," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 2021, pp. 139–147.

Yoosof Mashayekhi Yoosof Mashayekhi is a Ph.D. student at the IDLab, Ghent University, Belgium. He holds an MSc degree in Computer Engineering from Amirkabir University of Technology, Tehran, Iran. His primary interests are data mining and machine learning.



Bo Kang Bo Kang is a postdoctoral researcher at the IDLab, Ghent University, Belgium. He holds a Ph.D. degree in Computer Science Engineering from Ghent University, Belgium. His primary interests are data mining and machine learning, and more specifically representation learning and dimensionality reduction. He has a website at http://users.ugent.be/~bkang/.

Jefrey Lijffijt Jefrey Lijffijt is Assistant Professor of Data Science, Knowledge Discovery, and Visual Analytics at IDLab, Ghent University, Belgium. He graduated with distinction as a Doctor of Science in Technology in 2013 at Aalto University, was a Research Fellow at the University of Bristol, and an FWO [Pegasus]^2 Marie Skłodowska-Curie Fellow at Ghent University. His main research interests are theory and practice of statistical modeling, knowledge discovery, data visualization, and interaction with data. He has a website at http://users.ugent.be/~jlijffij/.



Tijl De Bie Tijl De Bie is currently a Full Professor at Ghent University, Belgium. Before moving to Ghent, he was a Reader at the University of Bristol, where he was appointed Lecturer (Assistant Professor) in January 2007. Before that, he was a postdoctoral researcher at the KU Leuven (Belgium) and the University of Southampton. He completed his Ph.D. in machine learning and advanced optimization techniques in 2005 at KU Leuven. During his Ph.D. he also spent a combined total of about 1 year as a visiting research scholar in U.C. Berkeley and U.C. Davis. He is currently most actively interested in the formalization of subjective interestingness in exploratory data mining, and in the use of machine learning and data mining for job market applications as well as for web and social media mining.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/