Minimizing the Importance Inequality of Nodes in a Social Network Graph

Ahmad Zareie and Rizos Sakellariou
Department of Computer Science, The University of Manchester, Manchester, U.K.
ahmad.zareie@postgrad.manchester.ac.uk, rizos@manchester.ac.uk

***Abstract*—Network graphs are widely used to model a variety of real-world interactions. In such graphs, nodes do not have the same importance in the graph structure as a result of the graph's topological properties. This may have various implications concerning a network's behaviour as, for example, how different nodes operate (even a node's failure) may not have the same impact for the whole network. The differences in the structural properties of the nodes imply that each node has different importance, which, in turn, gives rise to the notion of *importance inequality* in a graph. This paper defines and addresses the problem of *importance inequality minimization*, which may be useful to achieve certain properties in a network. Given a network graph and an integer k, the problem aims to identify k edges to connect non-adjacent nodes, in a way that minimizes the importance inequality of the graph. The paper provides a formal definition of the problem and proves its NP-hardness. Then, a naive greedy method is proposed, which is enhanced by heuristics that make its use practical. Experiments using 8 real-world networks are conducted to evaluate the proposed methods in terms of effectiveness and efficiency.**

I. Introduction

Network graphs are used to model different real-life applications in order to represent the inter-connectivity between the components of a network. The structural information of a network in diverse fields such as computer systems, social friendships, transport, power supply, e-commerce, or biology can be presented by a graph. Most specifically, a graph contains a set of nodes and edges, where the nodes indicate the components and the edges represent the connectivity (interaction) between the components.

Typically, not all nodes in a graph have the same structural properties. Nodes may have different importance in various ways, e.g., controlling the flow in the graph or boosting the connectivity of the graph. In order to capture the structural properties of nodes (and hence their individual importance), different centrality measures have been proposed [1]; a centrality measure assigns a value to a node to determine the relative importance of the node in the graph. Some centrality measures, e.g., degree and h-index [2], determine the importance of a node based on local structural information, while others, e.g, betweenness [3] and closeness [4], take global structural information to characterize a node's importance.

In real-life applications, a small fraction of a graph's nodes may be greatly important and may substantially affect the properties and functionality of a graph [1]. These nodes are also known as influential nodes, key players, vital nodes or critical nodes. The behaviour of such nodes may significantly affect a network. For instance, in a friendship social network when some important users leave the network this may lead to the departure of a relative large number of other users, such as their followers [5]; or, some users may have a significantly higher spreading ability to spread news or rumours [6]. In the case of a power transmission graph, the failure of important nodes may cause a breakdown of the network [7]. In cyberattacks, it has been reported that nodes with high centrality are primarily targeted [8]. All these examples stem from a root cause, which is the *inequality* in the importance of the nodes in the graph. The question which arises is how to minimize the inequality in the importance of the nodes in a graph. This paper is dedicated to answering this question and presents a method to find a set of edges whose addition to the graph minimizes the inequality of importance.

The contributions of the paper are summarized as follows: (i) a definition of the problem of importance inequality minimization, which consists in finding a set of edges whose addition to the graph minimizes the inequality of the nodes' importance; (ii) a proof that the problem is NP-hard and cannot be approximated in polynomial time within a ratio of $(1-1/e+\epsilon)$, unless P = NP; (iii) a solution of the importance inequality minimization problem based on a greedy method and heuristics that identify specific edges to add to the graph; (iv) experiments using 8 real-world network datasets to demonstrate that the proposed methods outperform a set of baseline methods.

The rest of the paper is organised as follows. First, a brief review of the literature about node importance and edge addition is given in Section II. Section III provides the necessary terminology, problem definition and a proof of the problem's NP-hardness. The details of the proposed methods to solve the problem are described in Section IV. Evaluation results are presented in Section V and the paper is concluded in Section VI.

II. Related Work

A. Node Importance

Different centrality measures [1] have been proposed to capture node properties and determine a node's importance. These measures determine node importance in different ways. Some measures, e.g., degree and h-index [2], determine node importance according to the neighbourhood by using local, structural information of the graph. In another set of measures,

IEEE/ACM ASONAM 2022, November 10-13, 2022
978-1-6654-5661-6/22/$31.00 © 2022 IEEE

e.g., k-core decomposition [9] and k-truss decomposition [10], the importance of a node is determined on the basis of density of the subgraph in which the node is located. In some other measures, e.g., closeness [4] and betweenness [3], the shortest paths between all pairs of nodes are taken into account to determine node importance; these measures use global information on the graph structure.

The role played by shortest paths is one of the most fundamental aspects of networks from different points of view [1] in areas such as, information dissemination, outbreak of epidemics, network flows, communication delays, vehicle routing and so on. Accordingly, how often a given node is on the shortest path between all pairs of nodes can reflect the importance of this node. Betweenness centrality was proposed to characterize the importance of a node according to the ratio of the shortest paths passing through the node. Failure (removal or misbehaviour) of a node with a large betweenness centrality value may disproportionately increase the distance between different pairs of nodes; such a node's failure significantly affects the functionality of the network. In real-world applications, such a failure may degrade information flow, traffic flows, interaction between communities or create bottlenecks in a graph. Given the prominent role of nodes with a large betweenness value on the functionality of a network this measure is widely used; we also adopt it in this paper to determine node importance in a graph.

Brandes [11] suggests an algorithm, which requires time $O(n \cdot m)$ to calculate the betweenness of the nodes in a network with n nodes and m edges. To reduce the cost of computing betweenness in large networks, some ideas such as sampling [12]–[14] or bounding the length of the shortest paths (bounded betweenness) [15], [16] have been proposed to approximate node betweenness. Incremental methods [17]–[19] have also been proposed to update the betweenness of nodes in evolving networks.

B. Edge Addition

Previous research has proposed several methods that try to optimize a property of a network or a node by adding a set of edges. The authors in [20], [21] address the problem of minimizing the diameter of a network by adding a small set of edges. How to maximize the robustness of networks is studied in [22], [23]; robustness refers to the largest connected subgraph after the removal of a set of nodes. The problem of minimizing the average shortest path length is tackled in [24], [25]. The problem of maximizing the betweenness of a given node or set of nodes is discussed in [26], [27]. The authors in [28], [29] propose methods to maximize the closeness centrality of a given node. The work in [30], [31] deals with the problem of adding a set of edges to maximize the influence of a given set of nodes or susceptibility of a social network. Our paper differs from the abovementioned references as it is dedicated to the problem of minimizing the importance inequality of nodes, which may potentially reduce the reliance of a network on a small fraction of nodes.

C. Inequality in Network Graphs

The concept of inequality in a network was first used in [32] to propose a metric to capture degree distribution. This metric has also been applied in [33] to calculate the inequality of other centrality measures, such as pagerank, closeness and betweenness. In [33], the impact of adding edges to the graph on the nodes' pagerank inequality has been investigated. The concept of centrality inequality has been used in [34] to assess degree inequality in an evolving network. Our paper differs from all these, as it defines the problem of minimizing importance inequality in a graph as an optimization problem and solves it by adding a set of edges between non-adjacent nodes. In practical terms, this could translate, for example, to adding specific friend connections in a social network.

III. Problem Definition

A network can be modelled as an undirected, unweighted graph $G(V, E)$, where V and $E \subseteq \{V \times V\}$ express the set of nodes and the set of edges, respectively. We denote the number of nodes and edges by n and m, respectively. If there is an edge between nodes v_s and v_t (neighbour or adjacent nodes) then $e_{st} \in E$; otherwise $e_{st} \notin E$. We use d_{st} to denote the length (the number of mediator edges) of the shortest path between the pair of nodes v_s and v_t, and $N_s^{(\gamma)}$ to denote the set of nodes whose shortest distance to v_s is not greater than γ, i.e., $N_s^{(\gamma)} = \{v_t | d_{st} \leq \gamma\}$.

We apply the *G*ini coefficient [35], the most commonly index used to measure inequality, to calculate the importance inequality of nodes in a graph. This index measures how far the distribution is away from a uniform distribution. The Gini coefficient is defined as follows:

$$\Phi(B) = \frac{\sum_{s=1}^{n} \sum_{t=1}^{n} |b_s - b_t|}{2n \sum_{s=1}^{n} b_s}, \tag{1}$$

where $B = \{b_1, b_2, \ldots, b_n\}$ denotes an importance vector, that is, element b_s is the importance of node v_s in the graph. The Gini coefficient is a value between 0 and 1, expressing an equal distribution for 0 (all nodes are equally important) and an unequal distribution for 1 (all nodes bar one have importance 0).

Lemma 1: If we sort nodes in ascending order of importance (that is, from the least important to the most important node), increasing the importance of the nodes whose rank is greater than $\frac{n(\Phi(B)+1)+1}{2}$ increases inequality in the network.

Proof: If the nodes are sorted in ascending order based on their importance, the Gini coefficient can be equivalently calculated by Eq. 2, where r_s indicates the rank of node v_s in the sorted list.

$$\Phi(B) = \frac{\sum_{s=1}^{n} (2r_s - n - 1) b_s}{n \sum_{s=1}^{n} b_s} \tag{2}$$

Suppose the importance of node v_s is increased by ϵ, i.e., $b_s = b_s + \epsilon$; the value of Gini coefficient changes to:

$$\begin{aligned} \Phi(B') &= \frac{\sum_{s=1}^{n} (2r_s - n - 1) b_s + \epsilon(2r_s - n - 1)}{n \sum_{s=1}^{n} b_s + n\epsilon} = \\ &= \frac{\Phi(B) n \sum_{s=1}^{n} b_s + \epsilon(2r_s - n - 1)}{n \sum_{s=1}^{n} b_s + n\epsilon} \end{aligned} \tag{3}$$

where $B' = \{b_1, b_2, \ldots, b_s + \epsilon, \cdots, b_n\}$ is the importance vector updated after increasing the importance of node v_s. We need to find out for what rank r_s the value of the Gini coefficient increases, i.e., $\Phi(B) < \Phi(B')$. Based on Eqs. 2 and 3:

$$\Phi(B) < \frac{\Phi(B) n \sum_{s=1}^{n} b_s + \epsilon(2r_s - n - 1)}{n \sum_{s=1}^{n} b_s + n\epsilon}$$
$$\Rightarrow \Phi(B) n \sum_{s=1}^{n} b_s + \Phi(B) n\epsilon < \Phi(B) n \sum_{s=1}^{n} b_s + \epsilon(2r_s - n - 1)$$
$$\Rightarrow \Phi(B) n < (2r_s - n - 1) \Rightarrow r_s > \frac{n(\Phi(B) + 1) + 1}{2} \quad (4)$$

Therefore, increasing the importance of nodes with rank greater than r_s leads to increasing the importance inequality in the graph. □

Problem Statement: Given an undirected, unweighted graph $G(V, E)$ and an integer value k, the problem of *Importance Inequality Minimization* aims to find a set $A \subset \{V \times V\} \backslash E$ of k edges whose addition to the graph minimizes the importance inequality of the graph.

As already mentioned, node importance in this paper is measured using betweenness centrality [3]. Thus, in the rest of the paper, we apply *betweenness inequality* to refer to importance inequality; accordingly, the problem of importance inequality minimization may be termed as Betweenness Inequality Minimization (BIM). Recall that the fraction of shortest paths between all pairs of nodes passing trough v_r determines v_r's betweenness centrality, denoted by b_r. Formally,

$$b_r = \sum_{v_s, v_t \in V \backslash \{v_r\}} \frac{\sigma_{srt}}{\sigma_{st}}, \quad (5)$$

where σ_{st} indicates the number of shortest paths between v_s and v_t, and σ_{srt} denotes the number of shortest paths between v_s and v_t that pass through v_r (v_r is a mediator on the paths).

Problem Hardness: BIM is NP-hard for any given k.

Proof: To prove NP-hardness of the BIM problem, we reduce a well-known NP-hard problem, the boolean satisfiability problem (3SAT) [36]. Given a boolean expression in conjunctive normal form, the 3SAT problem is defined as determining if there is a true assignment to its literals for which the expression is true. For any given instance of the 3SAT problem $C_1 \wedge C_2 \wedge \cdots \wedge C_n$, each clause $C_i (i = 1 \ldots n)$ consists of at most three literals $C_i = (x_{i1} \vee x_{i2} \vee x_{i3})$. In the theory of NP-hardness, if a restricted version of a problem is NP-hard, then the problem is also NP-hard.

Consider a restricted version of the BIM problem where the maximum betweenness of the nodes is 1. See, for example, the graph shown in Figure 1(a) and suppose $k = 2$. The problem can be expressed as a decision network with three layers (upper, middle and lower): see Figure 1(b). In the middle layer, each node has d (the number of its neighbours) instances, and each instance is connected to the corresponding neighbour in the upper layer. The lower layer contains all nodes in the graph and each of these nodes is connected to all nodes in the middle layer in a way that the upper and lower nodes of the middle-layer nodes are neither the same nor neighbours. The existing and non-existing edges between middle and lower layers are shown by red and dashed lines, respectively. Now, the problem changes to the selection of a set A containing k dashed edges in a way that each middle-layer node is a mediator between two nodes from upper to lower layers. This problem can be reduced to 3SAT problem as

$$(e_{13} \vee e_{15} \vee e_{14}) \wedge (e_{23} \vee e_{25} \vee e_{24}) \wedge (e_{31} \vee e_{32}) \wedge$$
$$\wedge (e_{41} \vee e_{42}) \wedge (e_{54} \vee e_{51} \vee e_{52}) \wedge (e_{53} \vee e_{51} \vee e_{52})$$

where each clause corresponds to a node instance in the middle layer of the decision network. If an edge between nodes v_s and v_t is existing or selected as a member of A then $e_{st} = 1$; otherwise $e_{st} = 0$. Note that $e_{st} = e_{ts}$ as the graph is undirected. A true assignment for this 3SAT problem is equivalent to minimizing the betweenness inequality in the graph shown in Figure 1(a) as each node is a mediator between exactly two nodes (betweenness centrality 1). As the restricted version of the problem is NP-hard, the BIM problem is NP-hard and cannot be approximated in polynomial time within a ratio of $(1 - 1/e + \epsilon)$ unless P = NP. □

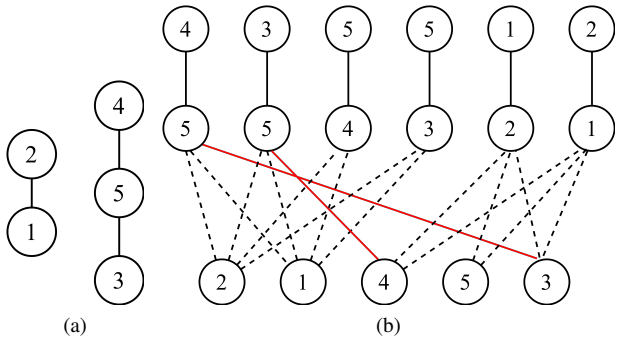


Fig. 1. (a) An example graph and (b) its decision network for BIM problem

IV. Solution

This section proposes a naive greedy method which iteratively selects edges whose addition to a graph minimizes inequality. Heuristics are proposed to speed up this greedy method.

A. A Naive Greedy Method

The pseudo code of the method is shown in Algorithm 1.

Algorithm 1: Naive greedy method

Data: Graph $G(V, E)$ and k
Result: A, a set of k edges
$A = \{\}$;
compute the betweenness of every node $v_r \in V$;
for $c = 1$ *to* k **do**
  $e_{xy} = \underset{e_{ij} \in \{V \times V\} \backslash E}{\text{argmin}} \Phi(B(e_{ij}))$;
  $A = A \cup \{e_{xy}\}$;
  update the graph as $G(V, E \cup \{e_{xy}\})$;
  update the betweenness of all nodes;
return A;

The naive greedy method selects edges to add in k iterations. In each iteration, first the impact of adding each non-existing edge on the betweenness inequality of the nodes is calculated. The vector $B(e_{ij})$ indicates the betweenness of each node if edge e_{ij} is added. The edge whose addition minimizes inequality is added to A. In the naive greedy method, we apply the method proposed by Bergamini et al. [18] to determine the impact of adding each edge on the betweenness of nodes (line 4) and update the betweenness of nodes after adding an edge (line 7). Bergamini et al. [18] propose a method to efficiently update the betweenness of nodes after adding an edge. This method first determines the pairs of nodes (affected pairs) whose shortest paths are affected after adding an edge; it then updates the betweenness of the mediator nodes of the shortest paths between affected nodes.

In Algorithm 1, computing the betweenness of all nodes (line 2) requires $O(n \cdot m)$ [11]. Calculating the impact of each non-existing edge on inequality (line 4) requires $O(n^2)$ [18]. Also, updating the betweenness of nodes after adding an edge (line 7) requires $O(n^2)$ [18]. Therefore, the time complexity of Algorithm 1 is $O(n \cdot m + k \cdot m' \cdot n^2)$, where m' denotes the number of non-existing edges in the graph. In what follows, this time complexity is improved with some heuristics.

B. Approximation of the Betweenness

The calculation of the betweenness of nodes results in high time complexity for Algorithm 1; approximating the betweenness helps to reduce the complexity associated with the computations in lines 2, 4 and 7.

The calculation of the exact betweenness in unweighted graphs requires a breadth-first algorithm to determine the shortest paths between all pairs of nodes. The idea of using limited-length paths [15], [16] can help approximate betweenness efficiently. In this idea, breadth-first search is bounded to a threshold and the paths whose length is greater than the threshold are ignored. Adopting this idea reduces the time complexity of calculating betweenness in lines 2, 4 and 7 in Algorithm 1. To implement this idea, we define *radial betweenness* to approximate the betweenness of every node v_r based on the shortest paths between pairs of nodes whose distance to v_r is less than or equal to a given radius γ. The γ-radial betweenness of node v_r, denoted by $b_r^{(\gamma)}$, is given by:

$$b_r^{(\gamma)} = \sum_{v_s, v_t \in N_r^{(\gamma)} \setminus \{v_r\}} \frac{\sigma_{srt}}{\sigma_{st}}. \quad (6)$$

Note that γ-radial betweenness is different from bounded betweenness as defined in [15], [16]. In both γ-radial and bounded betweenness, the length of the paths between a pair of nodes (v_s, v_t) is limited to $2 \cdot \gamma$. However, in γ-radial betweenness, the additional constraint is that the distance between both nodes of the pair to v_r must be less than or equal to γ. In order to calculate the γ-radial betweenness, we limit the breadth-first algorithm in the method proposed in [11], [16] to paths of length $2 \cdot \gamma$ to calculate the shortest distance and paths between pairs of nodes; the shortest distance between nodes whose distance is greater than $2 \cdot \gamma$ is assumed ∞. Then, we apply Eq. (6) to calculate γ-radial betweenness.

Approximating the betweenness of nodes using γ-radial betweenness limits the computation within a distance γ in line 7 of Algorithm 1. When updating the γ-radial betweenness of nodes after adding an edge, using the proposed method in [18], a pair (v_s, v_t) is affected if there is a change in the shortest paths between v_s and v_t and the length of these paths is not greater than $2 \cdot \gamma$. Then, the γ-radial betweenness of every node, which is a mediator between the affected pairs (v_s, v_t) and the node's distance to v_s and v_t is not greater than γ, must be updated.

C. Approximation of the Impact of Edge Addition

The calculation of the impact of adding an edge e_{ij} on the betweenness of all nodes to determine the effect of the edge addition (on the inequality) has considerable time complexity (line 4 of Algorithm 1). Approximating the effect of this addition, by considering its impact just on the betweenness of the incident nodes v_i and v_j, reduces the complexity of the naive greedy method. We propose Algorithm 3 to approximate the effect of adding an edge e_{ij} on the inequality of a network based on the impact of this addition on the γ-radial betweenness of the incident nodes, v_i and v_j. The impact on the γ-radial betweenness of nodes v_i and v_j is denoted by Imp_{ij} and Imp_{ji}, respectively. As paths with a length greater than $2 \cdot \gamma$ are ignored in γ-radial betweenness, this is taken into account in line 5 of Algorithm 3.

Adding edge e_{ij} may increase the γ-radial betweenness of v_i and v_j on the paths between every pair $\{(v_s, v_t) | v_s \in N_i^{(\gamma)}, v_t \in N_j^{(\gamma)}\}$; thus all these pairs should be assessed to determine the impact of adding edge e_{ij} on the γ-radial betweenness of v_i and v_j. The function I in Algorithm 2 calculates the impact of adding e_{ij} on the γ-radial betweenness of v_i on the paths between a pair (v_s, v_t). Then, Algorithm 3 calls function I of Algorithm 2 to assess all pairs $\{(v_s, v_t) | v_s \in N_i^{(\gamma)}, v_t \in N_j^{(\gamma)}\}$ and determine the impact of adding e_{ij} on the γ-radial betweenness of nodes v_i and v_j.

Algorithm 2: Function $I(G, \gamma, v_i, v_j, v_s, v_t)$

Data: Graph $G(V, E)$, γ, v_i, v_j, v_s and v_t
Result: r, the impact of adding e_{ij} on the γ-radial betweenness of v_i on the paths between pair (v_s, v_t)

1 $r = 0$;
2 **if** $(d_{st} > d_{si} + 1 + d_{jt})$ **then**
3 $\quad r = 1$;
4 $\quad$ **if** $(\sigma_{st} > 0)$ **then**
5 $\quad\quad r = r - (\sigma_{sit})/(\sigma_{st})$;
6 **else if** $(d_{st} = d_{si} + 1 + d_{jt})$ **then**
7 $\quad r = (\sigma_{si} \cdot \sigma_{jt})/(\sigma_{st} + \sigma_{si} \cdot \sigma_{jt})$;
8 $\quad$ **if** $(\sigma_{st} > 0)$ **then**
9 $\quad\quad r = r - \sigma_{sit}/\sigma_{st} + \sigma_{sit}/(\sigma_{st} + \sigma_{si} \cdot \sigma_{jt})$;

D. Updating the Impact of Edge Addition

Line 4 of Algorithm 1 calculates the impact of adding every non-existing edge to select an edge whose addition minimizes the inequality. However, this impact for every edge can be

Algorithm 3: Approximate the impact of adding e_{ij}

Data: Graph $G(V, E)$, γ, v_i and v_j
Result: Imp_{ij} and Imp_{ji}
1 $Imp_{ij} = 0$, $Imp_{ji} = 0$;
2 **foreach** $\{v_s \in N_i^{(\gamma)}\}$ **do**
3 **foreach** $\{v_t \in N_j^{(\gamma)}\}$ **do**
4 **if** $((v_s\neg = v_i) \vee (v_t\neg = v_j))$ **then**
5 **if** $(d_{si} + 1 + d_{jt} <= 2 \cdot \gamma)$ **then**
6 $Imp_{ij} = Imp_{ij} + I(G, \gamma, v_i, v_j, v_s, v_t)$;
7 $Imp_{ji} = Imp_{ji} + I(G, \gamma, v_j, v_i, v_t, v_s)$;

calculated once and then stored, outside the *for* loop of the algorithm, not in every iteration. This means that, in each iteration, only the impact of edges which are affected by the addition of a new edge needs to be updated. This becomes particularly useful as, using the heuristics for approximating betweenness and the impact of edge addition proposed in previous sections, only a small set of local edges are affected in each iteration. Thus, Algorithm 4 is proposed to update, if needed, the impact of affected edges in each iteration. The impact of adding edge e_{ij} on the γ-radial betweenness of nodes v_i and v_j is denoted by Imp_{ij} and Imp_{ji}, respectively.

In order to update the impact of local edges after adding a new edge e_{xy}, first the set of affected nodes is determined. Affected nodes are the nodes whose shortest path to v_x or v_y changes after edge e_{xy} is added. Such a change means that a new path with a length shorter than or equal to a previously existing shortest path is created.

As the paths between an affected node and the other nodes of the graph may change after adding e_{xy}, the impact of a set of edges in the neighbourhood of an affected node may also change. Thus, we define a set of affected edge-nodes, AEN, as in Eq. (7), where $\mathscr{d}_{ij}$ indicates the distance between v_i and v_j after adding e_{xy} to the graph and AN is the set of affected nodes.

$$AEN = \{(e_{ij}, v_r) | (v_r \in AN) \wedge (\mathscr{d}_{ri} \leq \gamma) \vee (\mathscr{d}_{rj} \leq \gamma)\} \quad (7)$$

An affected edge-node (e_{ij}, v_r) indicates an edge e_{ij} whose impact changes because it is in within a distance γ of an affected node v_r. Given the affected edge-node (e_{ij}, v_r) after adding edge e_{xy}, the γ-radial betweenness of nodes v_i and v_j on the paths between node v_r and all other nodes $\{v_t | (d_{jt} \leq \gamma) \vee (d_{it} \leq \gamma)\}$ must be assessed to update the impact of e_{ij}. Algorithm 4 shows pseudocode to update the impact Imp_{ij} and Imp_{ji} of an affected edge-node (e_{ij}, v_r) in graph $\mathscr{G}$ (i.e., after adding e_{xy} to G). In the algorithm, variables in cursive style refer to the graph after adding edge e_{xy}.

In lines 1-4 of Algorithm 4, p and q are used to define which node (v_i or v_j) is closer to the affected node v_r in graph G; the symbols $\mathscr{p}$ and $\mathscr{q}$ are used for the same purpose in graph $\mathscr{G}$. If the added edge e_{xy} does not affect neither the shortest paths between v_r and $v_{\mathscr{p}}$ nor the shortest paths between v_r and all the nodes in neighborhood $v_{\mathscr{q}}$ (i.e., $v_t \in N_{\mathscr{q}}^{(\gamma)}$), adding e_{xy} does not make changes in the impact of e_{ij} on the γ-radial betweenness of v_i and v_j. This situation is assessed by $condition1$ in line 5. Otherwise, we should first reduce the impact of adding e_{ij} on the γ-radial betweenness of v_i and v_j on the paths between v_r and all other nodes $v_t \in N_q^{(\gamma)}$ in graph G, i.e., the graph before adding e_{xy} (lines 6-11). Then, we update the impact of adding e_{ij} (on the γ-radial betweenness of v_i and v_j) by adding the impact related to the effect of adding e_{xy} (lines 12-17). Note that if adding edge e_{xy} does not change the shortest paths between v_r and node $v_t \in N_{\mathscr{q}}^{(\gamma)}$, then adding e_{ij} does not affect the γ-radial betweenness of v_i and v_j on the paths between v_r and v_t; $condition2$ is used to assess this situation.

Algorithm 4: Update the impact of adding e_{ij}

Data: Graph $G(V, E)$, $\mathscr{G}(V, E \cup e_{xy})$, γ and (e_{ij}, v_r)
Result: Updated Imp_{ij} and Imp_{ji}
1 $p = \underset{v_i, v_j}{\text{argmin}}(d_{ri}, d_{rj})$;
2 $q = \underset{v_i, v_j}{\text{argmax}}(d_{ri}, d_{rj})$;
3 $\mathscr{p} = \underset{v_i, v_j}{\text{argmin}}(\mathscr{d}_{ri}, \mathscr{d}_{rj})$;
4 $\mathscr{q} = \underset{v_i, v_j}{\text{argmax}}(\mathscr{d}_{ri}, \mathscr{d}_{rj})$;
5 **if** $(condition1)$ **then**
6 **if** $(d_{ri}\neg = d_{rj})$ **then**
7 **foreach** $\{v_t \in N_q^{(\gamma)}\}$ **do**
8 **if** $(condition2 \wedge ((v_r\neg = v_p) \vee (v_t\neg = v_q)))$ **then**
9 **if** $(d_{rp} + 1 + d_{qt} \leq 2 \cdot \gamma)$ **then**
10 $Imp_{pq} = Imp_{pq} - I(G, \gamma, v_p, v_q, v_r, v_t)$;
11 $Imp_{qp} = Imp_{qp} - I(G, \gamma, v_q, v_p, v_t, v_r)$;
12 **if** $(\mathscr{d}_{ri}\neg = \mathscr{d}_{rj})$ **then**
13 **foreach** $\{v_t \in \mathscr{N}_{\mathscr{q}}^{(\gamma)}\}$ **do**
14 **if** $(condition2 \wedge ((v_r\neg = v_p) \vee (v_t\neg = v_q)))$ **then**
15 **if** $(\mathscr{d}_{r\mathscr{p}} + 1 + \mathscr{d}_{\mathscr{q}t} \leq 2 \cdot \gamma)$ **then**
16 $Imp_{\mathscr{p}\mathscr{q}} = Imp_{\mathscr{p}\mathscr{q}} + I(\mathscr{G}, \gamma, v_{\mathscr{p}}, v_{\mathscr{q}}, v_r, v_t)$;
17 $Imp_{\mathscr{q}\mathscr{p}} = Imp_{\mathscr{q}\mathscr{p}} + I(\mathscr{G}, \gamma, v_{\mathscr{q}}, v_{\mathscr{p}}, v_t, v_r)$;

E. Further Efficiency Improvements

In Algorithm 1, all non-existing edges are considered as candidate edges for addition. The complexity of the algorithm can be further improved by ignoring edges whose addition may increase or have only a minor impact in decreasing inequality. Recall *Lemma 1*, which proves that increasing the γ-radial betweenness of some nodes increases the inequality in the network; this suggests that edges connected to such nodes can be ignored. To do so, the γ-radial betweenness of each node is calculated (using Eq. (6)) and nodes are sorted based on their γ-radial betweenness in ascending order. Then, the value of r_s is calculated as $\lceil \frac{n(\Phi(B^{(\gamma)})+1)+1}{2} \rceil$ (see Eq. (1) for $\Phi(B)$); the edges connected to the nodes whose rank is greater than r_s are removed from the candidate edge set. This process can be repeated in each iteration to remove from the candidate edge set those edges whose addition increases inequality.

Moreover, some edges with no (or minor) effect on minimizing inequality could be removed from the candidate edge set as well. To determine the impact of addition of each edge e_{ij} on the γ-radial betweenness of its incident nodes (Imp_{ij} and Imp_{ji}), we use Algorithm 3. Then, the importance of every node $v_r \in V$ (the γ-radial betweenness of node) under

the impact of adding the edge is calculated using Eq. (8) that determines the importance vector $B^{(\gamma)(e_{ij})}$.

$$b_r^{(\gamma)}(e_{ij}) = \begin{cases} b_r^{(\gamma)} & \text{if } (r\neg = i) \wedge (r\neg = j) \\ b_r^{(\gamma)} + Imp_{ij} & \text{if } r = i \\ b_r^{(\gamma)} + Imp_{ji} & \text{if } r = j \end{cases} \quad (8)$$

Eq. (1) is used to calculate the inequality of the network based on the importance vector $B^{(\gamma)}(e_{ij})$ (i.e., $\Phi(B_r^{(\gamma)}(e_{ij}))$); this helps assess the impact of adding any edge e_{ij} on the inequality of the network. Then, a number (μ) of edges that have the largest effect on minimizing inequality are kept as candidate edges; the value of μ may be experimentally determined.

F. Putting it Together

All the above allow us to propose an enhanced greedy method that reduces the complexity of the naive greedy algorithm; this is shown in Algorithm 5.

In line 2, the γ-radial betweenness of every node is computed to initialize the importance vector B^{γ}. In line 3, the candidate set C is initialized with all non-existing edges. The inequality of the original network is calculated in line 4. Applying the heuristics in Section IV-E, a number of candidate edges are removed in lines 5-8. The algorithm continues in k iterations. In each iteration, a candidate edge e_{xy} whose addition to the network minimizes inequality is determined in line 10 (where $B^{(\gamma)}(e_{ij})$ is calculated using Eq. (8)) and added to the set A in line 11; then, by adding e_{xy} to the graph G, the graph $\mathscr{G}$ is determined in line 12 and the γ-radial betweenness vector is updated in line 13. After adding an edge to the graph, the set of affected edge-nodes is determined in line 14. Then, in line 15, the affected edge-node set is used to update the impact of adding each candidate edge in $\mathscr{G}$. In line 16, a set of candidate edges are removed (using the first heuristic in Section IV-E). Finally, the graph is updated by adding e_{xy}.

Algorithm 5: Enhanced greedy method

```
   Data: Graph G(V, E), k, γ and μ
   Result: A, a set of k edges
1  A = {};
2  B^(γ) = γ-radial betweenness of every node v_r ∈ V;
3  C = {e_ij | v_i¬ = v_j ∧ e_ij ∈ {V × V}\E};
4  compute the γ-radial betweenness inequality in G;
5  remove the set of connected edges to top-ranked nodes from C;
6  compute the Imp_ij and Imp_ji ∀e_ij ∈ C (Algorithm 3);
7  compute the effect of adding every edge e_ij ∈ C as Φ(B^(γ)(e_ij));
8  keep μ number of the edges in C with largest effects;
9  for c = 1 to k do
10     e_xy = argmin_{e_ij ∈ C} Φ(B^(γ)(e_ij));
11     A = A ∪ {e_xy};
12     set 𝒢 = (V, ℰ) where ℰ = E ∪ {e_xy};
13     update vector B^(γ), as e_xy is added;
14     determine the set of affected edge-node using Eq. (7);
15     update the impact of candidate edges using the affected
         edge-node set (Algorithm 4);
16     remove the set of connected to top-ranked nodes from C;
17     update the graph as G(V, E ∪ {e_xy});
18 return A;
```

TABLE I
PROPERTIES OF THE NETWORKS USED IN THE EVALUATION

Name	Node size	Edge size	Diameter	Density	BI
Karate (KRT)	34	78	5	0.1390	0.7992
Dolphins (DLN)	62	159	8	0.0841	0.6141
Infectious (INF)	113	2,197	3	0.3472	0.5594
Jazz (JZZ)	198	2,742	6	0.1406	0.7217
UsAir (UAR)	332	2,126	6	0.0387	0.9277
Power (PWR)	662	906	25	0.0041	0.7263
Email (EML)	1,133	5,451	8	0.0085	0.7432
Euroroad (ERD)	1,174	1,417	62	0.0021	0.7758

Algorithm 5 initially requires $O(n \cdot \Gamma)$ to compute the γ-radial betweenness and $O(|C| \cdot \Gamma^2)$ to compute the effect of adding every candidate edge on the inequality of network, where Γ denotes the average number of nodes within a distance γ for all nodes. In each iteration, the γ-radial betweenness of nodes in the neighbourhood of an added edge e_{xy} must be updated which requires $O(\Gamma^2)$; also, the impact of adding every edge in the neighbourhood of edge e_{xy} must be updated, which requires $O(\Gamma^2)$. In total, the time complexity of the enhanced greedy method is $O(n \cdot \Gamma + |C| \cdot \Gamma^2 + k \cdot 2 \cdot \Gamma^2)$.

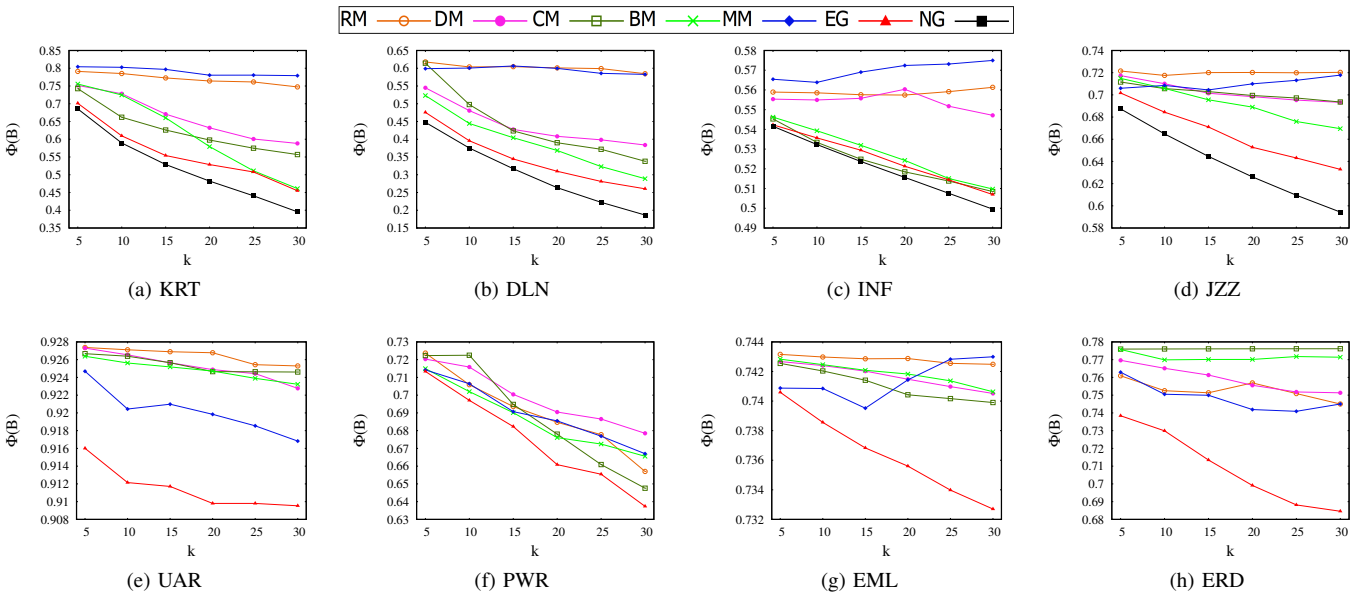
V. EVALUATION

To evaluate the two proposed methods, the naive greedy method (NG) (Algorithm 1) and the enhanced greedy method (EG) (Algorithm 5), a set of 8 real-world networks from different domains (taken from the KONECT Project [1] and the Network Repository [2]) and with varying properties has been used. The properties of these networks are reported in Table I. Column BI shows the betweenness inequality in the original network (the network before adding any edges for the purpose of minimizing inequality).

As there is no other method for the problem of inequality minimization to compare against, we consider five baseline methods, which select k edges iteratively using a simple strategy as described next. (i) Random method (RM): randomly selects a non-existing edge; (ii) Degree method (DM): selects the non-existing edge that connects the node with minimum degree to a non-adjacent node with minimum degree and updates the degree of nodes; (iii) Closeness method (CM): selects the non-existing edge that connects the node with minimum closeness to a non-adjacent node with minimum closeness and updates the closeness of all nodes; (iv) Betweenness method (BM): selects the non-existing edge that connects the node with minimum betweenness to a non-adjacent node with minimum betweenness and updates the betweenness of all nodes; and (v) Min-Max method (MM): randomly selects a non-existing edge that connects a node with low betweenness to a node with high betweenness and updates the betweenness of all nodes (this strategy is applied in [33] to assess the impact of adding edges on the centrality inequality in a network).

According to [16], limiting the length of the shortest paths to 4 offers a good approximation of betweenness. Thus, we set the radius γ to 2 (to take into account paths up to a length of

[1] konect.cc
[2] networkrepository.com



(a) KRT (b) DLN (c) INF (d) JZZ

(e) UAR (f) PWR (g) EML (h) ERD

Fig. 2. Betweenness inequality achieved by different methods

4) during the experiments. Experimentally, we also determine that $\mu = k \times 50$ is a good choice for this value in the enhanced greedy method (EG, see Algorithm 5).

Two different experiments are designed. In the first experiment, the performance of all methods in terms of minimizing inequality (as measured by the Gini coefficient) is evaluated. In the second experiment, the running time of all methods is compared. Note that the naive greedy method (NG) was included only for the networks where it produced results after it was left running for 3 days; these were the 4 smallest networks (KRT, DLN, INF and JZZ). Experiments were run on an Intel Xeon 3.6GHz processor with 32GB memory.

A. Performance

In the first experiment, a set A containing k edges (varying from 5 to 30 in steps of 5) is selected and added to the network to minimize the betweenness inequality. The betweenness inequality of the network, after adding the set A that is selected by each method, is calculated using the Gini coefficient (Eq. 1). The results are shown in Figure 2. It can be seen that NG outperforms all other methods for all values of k in the four smallest networks; this is expected as NG examines the impact of adding each edge on all paths and all nodes. The proposed enhanced greedy method, EG, is the second best method with the exception of the INF network where CM outperforms EG for the values $k = 10, 15, 20$. The reason is that the diameter of INF is very small (diameter=3). Adding an edge may affect the shortest paths between other pairs; considering just incident nodes to approximate the impact of edge addition (the heuristic in Section IV-C) is not too efficient. However, the strategy applied by CM is generally not helpful; EG outperforms CM for greater values of k (25 and 30) in INF. Generally, EG is the best method after NG to minimize inequality, suggesting that the heuristics proposed in this paper are appropriate and efficient.

TABLE II
RUNNING TIME (SECONDS) OF THE 7 METHODS

Network	RM	DM	CM	BM	MM	EG	NG
KRT	0.0097	0.1727	0.3398	0.7074	0.6791	0.5134	93.9557
DLN	0.0054	0.0812	1.1569	1.5213	1.4604	0.8496	641.3909
INF	0.0031	0.0039	8.6982	9.8493	9.4553	2.4162	21656.0000
JZZ	0.0022	0.0042	18.0151	19.8132	19.0207	10.8313	36056.0000
UAR	0.0020	0.0045	37.2991	40.4743	38.8554	26.0793	> 250000
PWR	0.0014	0.0062	112.1185	125.3042	120.2920	8.8432	> 250000
EML	0.0020	0.0137	433.9534	500.0766	480.0735	244.2891	> 250000
ERD	0.0020	0.0066	286.1584	347.7974	333.8855	26.5921	> 250000

B. Running Time

In the second experiment, the proposed methods are compared to the baseline methods in terms of running time. For this purpose, a set containing 15 edges is selected by each method. Each method is executed 100 times on each network and the average running time obtained by each of the methods on each network is summarized. The obtained results in this experiment are shown in Table II. As mentioned, in four of the networks NG failed to complete in less than 3 days.

As can be seen, the running time of NG is significantly higher even in small networks, as expected. EG has a running time, which is comparable to the other baseline methods. RM and DM are the fastest methods but these are simple methods that, as seen in the first experiment, do not perform well. EG is faster than CM, BM and MM, especially in networks with a large diameter (see for example PWR and ERD). This is because CM (or BM and MM) needs to update the closeness (betweenness) of the nodes in each iteration, while EG approximates the betweenness of nodes. Overall, the heuristics suggested by EG help reduce the complexity of calculations with reasonable running time, as shown in Table II, and performance, as demonstrated in Figure 2.

VI. Conclusion

This paper deals with the problem of minimizing importance inequality in a network. The problem was formally modelled and a naive greedy method, enhanced by heuristics, was proposed to tackle this problem through the identification of a set of edges whose addition to the network minimizes the network's importance inequality. Experimental evaluation demonstrated the effectiveness and efficiency of the proposed enhanced method compared to the baseline.

Future work can explore a number of directions. First, the problem can be investigated for networks with different properties such as directed, weighted or dynamic networks. The problem can also be investigated using (other than betweenness) measures of importance, such as closeness, k-core and h-index. Second, extensions of the problem allowing removal of edges (not only addition) in the network can be considered. Third, the definition of inequality can be broadened to take into account not only importance but non-structural properties of the network too. From a general point of view, we believe that the problem studied in this paper highlights interesting properties of network design/manipulation that may have useful implications for the wider analysis and understanding of social networks.

References

[1] L. Lü, D. Chen, X.-L. Ren, Q.-M. Zhang, Y.-C. Zhang, and T. Zhou, "Vital nodes identification in complex networks," *Physics Reports*, vol. 650, pp. 1–63, 2016.

[2] J. E. Hirsch, "An index to quantify an individual's scientific research output," *Proceedings of the National Academy of Sciences*, vol. 102, no. 46, pp. 16 569–16 572, 2005.

[3] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, vol. 40, pp. 35–41, 1977.

[4] ——, "Centrality in social networks conceptual clarification," *Social Networks*, vol. 1, no. 3, pp. 215–239, 1978.

[5] R. Laishram, A. E. Sariyüce, T. Eliassi-Rad, A. Pinar, and S. Soundarajan, "Measuring and improving the core resilience of networks," in *Proceedings of the 2018 World Wide Web Conference*. ACM, 2018, p. 609–618.

[6] A. Zareie and R. Sakellariou, "Minimizing the spread of misinformation in online social networks: A survey," *Journal of Network and Computer Applications*, vol. 186, p. 103094, 2021.

[7] S. Wang, W. Lv, J. Zhang, S. Luan, C. Chen, and X. Gu, "Method of power network critical nodes identification and robustness enhancement based on a cooperative framework," *Reliability Engineering & System Safety*, vol. 207, p. 107313, 2021.

[8] S. Freitas, D. Yang, S. Kumar, H. Tong, and D. H. Chau, "Graph vulnerability and robustness: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[9] D. W. Matula and L. L. Beck, "Smallest-last ordering and clustering and graph coloring algorithms," *Journal of the ACM*, vol. 30, no. 3, p. 417–427, Jul. 1983.

[10] J. Cohen, "Trusses: Cohesive subgraphs for social network analysis," National Security Agency, Tech. Rep., 2008.

[11] U. Brandes, "A faster algorithm for betweenness centrality," *The Journal of Mathematical Sociology*, vol. 25, no. 2, pp. 163–177, 2001.

[12] D. A. Bader, S. Kintali, K. Madduri, and M. Mihail, "Approximating betweenness centrality," in *Algorithms and Models for the Web-Graph*. Springer Berlin Heidelberg, 2007, pp. 124–137.

[13] M. Borassi and E. Natale, "Kadabra is an adaptive algorithm for betweenness via random approximation," *ACM Journal of Experimental Algorithmics*, vol. 24, Feb. 2019.

[14] M. Haghir Chehreghani, A. Bifet, and T. Abdessalem, "Adaptive algorithms for estimating betweenness and k-path centralities," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. ACM, 2019, p. 1231–1240.

[15] M. Everett and S. P. Borgatti, "Ego network betweenness," *Social Networks*, vol. 27, no. 1, pp. 31–38, 2005.

[16] J. Pfeffer and K. M. Carley, "K-centralities: Local approximations of global measures based on shortest paths," in *Proceedings of the 21st International Conference on World Wide Web*. ACM, 2012, p. 1043–1050.

[17] M. Kas, M. Wachs, K. M. Carley, and L. R. Carley, "Incremental algorithm for updating betweenness centrality in dynamically growing networks," in *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, 2013, p. 33–40.

[18] E. Bergamini, H. Meyerhenke, M. Ortmann, and A. Slobbe, "Faster Betweenness Centrality Updates in Evolving Networks," in *16th International Symposium on Experimental Algorithms (SEA 2017)*, C. S. Iliopoulos, S. P. Pissis, S. J. Puglisi, and R. Raman, Eds., vol. 75. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017, pp. 23:1–23:16.

[19] F. Jamour, S. Skiadopoulos, and P. Kalnis, "Parallel algorithm for incremental betweenness centrality on large graphs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 3, pp. 659–672, 2018.

[20] E. D. Demaine and M. Zadimoghaddam, "Minimizing the diameter of a network using shortcut edges," in *Algorithm Theory*, H. Kaplan, Ed. Springer Berlin Heidelberg, 2010, pp. 420–431.

[21] F. Frati, S. Gaspers, J. Gudmundsson, and L. Mathieson, "Augmenting graphs to minimize the diameter," *Algorithmica*, vol. 72, no. 4, pp. 995–1010, 2015.

[22] P. Cui, P. Zhu, K. Wang, P. Xun, and Z. Xia, "Enhancing robustness of interdependent network by adding connectivity and dependence links," *Physica A: Statistical Mechanics and its Applications*, vol. 497, pp. 185–197, 2018.

[23] X. Ji, B. Wang, D. Liu, G. Chen, F. Tang, D. Wei, and L. Tu, "Improving interdependent networks robustness by adding connectivity links," *Physica A: Statistical Mechanics and its Applications*, vol. 444, pp. 9–19, 2016.

[24] M. Papagelis, "Refining social graph connectivity via shortcut edge addition," *ACM Transactions on Knowledge Discovery from Data*, vol. 10, no. 2, Oct. 2015.

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

[26] E. Bergamini, P. Crescenzi, G. D'angelo, H. Meyerhenke, L. Severini, and Y. Velaj, "Improving the betweenness centrality of a node by adding links," *ACM Journal of Experimental Algorithmics*, vol. 23, Aug. 2018.

[27] S. Medya, A. Silva, A. Singh, P. Basu, and A. Swami, "Group centrality maximization via network design," in *Proceedings of the 2018 SIAM International Conference on Data Mining (SDM)*. SIAM, 2018, pp. 126–134.

[28] N. Parotsidis, E. Pitoura, and P. Tsaparas, "Centrality-aware link recommendations," in *Proceedings of the 9th ACM International Conference on Web Search and Data Mining*. ACM, 2016, p. 503–512.

[29] P. Crescenzi, G. D'angelo, L. Severini, and Y. Velaj, "Greedily improving our own closeness centrality in a network," *ACM Transactions on Knowledge Discovery from Data*, vol. 11, no. 1, Jul. 2016.

[30] L. Yu, G. Li, and L. Yuan, "Maximizing boosted influence spread with edge addition in online social networks," *ACM/IMS Transactions on Data Science*, vol. 1, no. 2, May 2020.

[31] G. D'Angelo, L. Severini, and Y. Velaj, "Recommending links through influence maximization," *Theoretical Computer Science*, vol. 764, pp. 30–41, 2019.

[32] J. Kunegis and J. Preusse, "Fairness on the web: Alternatives to the power law," in *Proceedings of the 4th Annual ACM Web Science Conference*. ACM, 2012, p. 175–184.

[33] H. Kwak, J. Kim, Y. Lim, S.-K. Han, and K. Jung, "Centrality fairness: Measuring and analyzing structural inequality of online social network," *Journal of Internet Technology*, vol. 18, no. 7, pp. 1515–1524, 2017.

[34] M. R. Barnes, V. Nicosia, and R. G. Clegg, "Measuring equality and hierarchical mobility on abstract complex networks," *arXiv preprint arXiv:2205.14091*, 2022.

[35] C. Gini, "Measurement of inequality of incomes," *The Economic Journal*, vol. 31, no. 121, pp. 124–126, 1921.

[36] S. A. Cook, "The complexity of theorem-proving procedures," in *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing*. ACM, 1971, p. 151–158.